TECHNISCHE
UNIVERSITÄT
DARMSTADT

ENABLING EFFICIENT, ROBUST, AND SCALABLE WIRELESS MULTI-HOP NETWORKS: A CROSS-LAYER APPROACH EXPLOITING COOPERATIVE DIVERSITY

Am Fachbereich Informatik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertationsschrift

von

ADRIAN CARLOS LOCH NAVARRO, M.SC.

Geboren am 8. September 1987 in Madrid, Spanien

Erstreferent: Prof. Dr.-Ing. Matthias Hollick
Korreferent: Prof. Jon Crowcroft (Ph.D.)
Korreferent: Dr. rer. nat. Joerg Widmer

Tag der Einreichung: 5. Februar 2015
Tag der Disputation: 20. März 2015

Darmstadt, 2015
Hochschulkennziffer D17

ABSTRACT

The practical performance in terms of throughput, robustness, and scalability of traditional Wireless Multihop Networks (WMNs) is limited. The key problem is that such networks do not allow for advanced physical layers, which typically require (a) spatial diversity via multiple antennas, (b) timely Channel State Information (CSI) feedback, and (c) a central instance that coordinates nodes. We propose Corridor-based Routing to address these issues. Our approach widens traditional hop-by-hop paths to span multiple nodes at each hop, and thus provide spatial diversity. As a result, at each hop, a group of transmitters cooperates at the physical layer to forward data to a group of receivers. We call two subsequent groups of nodes a stage. Since all nodes participating in data forwarding at a certain hop are part of the same fully connected stage, corridors only require one-hop CSI feedback. Further, each stage operates independently. Thus, Corridor-based Routing does not require a network-wide central instance, and is scalable. We design a protocol that builds end-to-end corridors. As expected, this incurs more overhead than finding a traditional WMN path. However, if the resulting corridor provides throughput gains, the overhead compensates after a certain number of transmitted packets.

We adapt two physical layers to the aforementioned stage topology, namely, Orthogonal Frequency-Division Multiple Access (OFDMA), and Interference Alignment (IA). In OFDMA, we allocate each subchannel to a link of the current stage which provides good channel conditions. As a result, we avoid deep fades, which enables OFDMA to transmit data robustly in scenarios in which traditional schemes cannot operate. Moreover, it achieves higher throughputs than such schemes. To minimize the transmission time at each stage, we present an allocation mechanism that takes into account both the CSI, and the amount of data that each transmitter needs to transmit. Further, we address practical issues and implement our scheme on software-defined radios. We achieve roughly 30% average throughput gain compared to a WMN not using corridors. We analyze OFDMA in theory, simulation, and practice. Our results match in all three domains.

Further, we design a physical layer for corridor stages based on IA in the frequency domain. Our practical experiments show that IA often performs poorly because the decoding process augments noise. We find that the augmentation factor depends only on the channel coefficients of the subchannels that IA uses. We design a mechanism to determine which transmitters should transmit to which receivers on which subchannels to minimize noise. Since the number of possible combinations is very large, we use heuristics that reduce the search space significantly. Based on this design, we present the first practical frequency IA system. Our results show that our approach avoids noise augmentation efficiently, and thus operates robustly. We observe that IA is most suitable for stages with specific CSI and traffic conditions. In such scenarios, the throughput gain compared to a WMN not using corridors is 25% on average, and 150% in the best case.

Finally, we design a decision engine which estimates the performance of both OFDMA and IA for a given stage, and chooses the one which achieves the highest throughput. We evaluate corridors with up to five stages, and achieve roughly 20% average throughput gain. We conclude that switching among physical layers to adapt to the particular CSI and traffic conditions of each stage is crucial for efficient and robust operation.

ZUSAMMENFASSUNG

Der Durchsatz, die Robustheit, und die Skalierbarkeit von klassischen drahtlosen multi-hop Netzen ist in der Praxis beschränkt. Das grundlegende Problem ist, dass solche Netze fortschrittliche Verfahren auf der Bitübertragungsschicht nicht unterstützen. Diese Verfahren erfordern in der Regel (a) räumliche Diversität mit Hilfe mehrerer Antennen, (b) aktualisierte Kanalinformation am Sender, und (c) eine zentrale Instanz zur Koordinierung der Netzwerkknoten. Im Rahmen dieser Arbeit wird Korridor-basiertes Routing vorgeschlagen um diese Limitierungen in drahtlosen multi-hop Netzen zu beheben. Dieser Ansatz weitet klassische multi-hop Pfade auf, um mehrere Knoten in jedem Hop zu erfassen, und somit räumliche Diversität zu erreichen. Dadurch kooperiert bei jedem Hop eine Gruppe von Sendern auf der Bitübertragungsschicht, um Daten an eine Gruppe von Empfängern weiterzuleiten. Zwei aufeinanderfolgende Knotengruppen formen eine Korridoretappe. Da alle Knoten, die an der Weiterleitung von Daten in einem Hop teilnehmen, zur selben vollvermaschten Etappe gehören, ist die Rückmeldung von Kanalinformation nur über einen Hop notwendig. Zusätzlich ist Korridor-basiertes Routing skalierbar, da jede Etappe unabhängig betrieben wird, und somit keine netzwerkweite zentrale Koordinierungsinstanz benötigt wird. Durch die Etablierung von Ende-zu-Ende Korridoren entstehen im Vergleich zur Etablierung von klassischen Pfaden in drahtlosen multi-hop Netzen, wie erwartet, zusätzliche Kosten in Form von Kontrollnachrichten. Wenn der daraus entstehende Korridor jedoch Durchsatzgewinne ermöglicht, gleichen sich die Etablierungskosten nach einer gewissen Anzahl an Paketübertragungen aus.

Im Rahmen dieser Arbeit werden zwei Techniken auf der Bitübertragungsschicht für die obengenannte Systemarchitektur erforscht. Es handelt sich dabei um Orthogonal Frequency-Division Multiple Access (OFDMA) und Interference Alignment (IA). In Etappen, die OFDMA einsetzen, wird jeder Subkanal einem Link der Etappe zugeordnet, der für diesen Subkanal eine gute Kanalqualität aufweist. Somit kann OFDMA in Szenarien, in denen klassische Weiterleitungsmechanismen auf der Bitübertragungsschicht nicht betriebsfähig sind, Daten robust übertragen. Zudem kann ein höherer Durchsatz als bei solchen klassischen Ansätzen erzielt werden. Um die Etappenübertragungszeit zu minimieren, wird ein Allokationsverfahren vorgeschlagen, das sowohl die Kanalinformation als auch die Menge an Daten, die jeder Sender in einer Etappe übertragen möchte, berücksichtigt. Ferner werden die praktischen Herausforderungen, die der Betrieb einer Etappe auf Basis von OFDMA stellt, behandelt, und eine Implementierung auf Software Defined Radios realisiert. Der durchschnittliche Durchsatzgewinn liegt bei 30% im Vergleich zu klassischen drahtlosen multi-hop Netzen. OFDMA wird in Theorie, Simulation, und Praxis analysiert. Die Ergebnisse in allen drei Domänen stimmen überein.

Weiterhin wird IA in der Frequenz als Bitübertragungsschicht für Korridoretappen untersucht. Praktische Experimente zeigen, dass IA oft nur einen geringen Durchsatz aufgrund von einer Verstärkung des Rauschens im Zuge des Dekodierungsprozesses erreicht. Es wird festgestellt, dass der Verstärkungsfaktor nur von den Kanalkoeffizienten der Subkanäle abhängt, die für IA verwendet werden. Auf Basis dieser Erkenntnis wird ein Mechanismus vorgeschlagen, der bestimmt, welche Senderknoten mit welchen Empfängerknoten innerhalb einer Etappe über welche Subkanäle Daten austauschen sollten, um den Verstärkungsfaktor zu minimieren. Da die Anzahl an möglichen Kombinationen sehr umfangreich ist, werden Heuristiken entworfen, die den Suchraum erheblich reduzieren. Auf diesem Mechanismus aufbauend wird das erste praktikable Übertragungssystem entwickelt, dass IA in der Frequenz nutzt. Die Messergebnisse zeigen, dass die obengenannte Verstärkung des Rauschens effizient vermieden, und somit ein robuster Betrieb erzielt werden kann. Es wird auch deutlich, dass IA nur für Korridoretappen mit bestimmten Kanalzuständen und Datenaufkommen an den Sendern geeignet ist. Im Vergleich zu klassischen multi-hop Netzen beträgt in solchen Fällen der durchschnittliche Durchsatzgewinn 25% und der maximale Durchsatzgewinn 150%.

Abschließend wird ein Entscheidungsmechanismus entwickelt, der für eine bestimmte Etappe die Technik auf der Bitübertragungsschicht auswählt, die den höchsten Durchsatz erreicht. Es werden Korridore mit bis zu fünf Etappen untersucht; diese erzielen einen durchschnittlichen Durchsatzgewinn von 20%. Insgesamt erweist sich, dass die Fähigkeit von Korridoren die beste Technik auf der Bitübertragungsschicht auszuwählen, um sich so an die Gegebenheiten der einzelnen Etappen anpassen zu können, entscheidend ist, um einen effizienten und robusten Betrieb zu ermöglichen.

ACKNOWLEDGMENTS

A section on acknowledgments is inevitably incomplete. Many people have supported me over the last four years in a myriad of different ways. Most importantly, each and every of their contributions has been crucial to make this thesis possible. Acknowledging each of them as they deserve would render this section longer than the remainder of this work. An attempt to summarize the acknowledgments to a few paragraphs would not live up to reality. It would be partial, unfair, and disappointing for everyone who has supported me. Ultimately, it would result in an extensive list of names which at some point would become meaningless. Hence, I decide in favor of condensing all my gratitude in a single but hopefully more significant final sentence.

To all of you—thank you :)

Adrian

CONTENTS

I INTRODUCTION 1

1 INTRODUCTION 3
- 1.1 Motivation . . . 3
- 1.2 Challenges and Goals . . . 4
- 1.3 Contributions . . . 7
  - 1.3.1 Architecture and Operation of Corridor-based Routing . . . 7
  - 1.3.2 Stage Mechanism based on OFDMA . . . 7
  - 1.3.3 Stage Mechanism based on IA . . . 8
- 1.4 Outline . . . 8

2 RELATED WORK AND BACKGROUND 11
- 2.1 PHY-aware Network Layer . . . 12
  - 2.1.1 Traditional WMN Schemes . . . 12
  - 2.1.2 Advanced Routing Schemes . . . 15
- 2.2 Cross-Layer Approaches . . . 17
  - 2.2.1 Advanced Link Layer . . . 17
  - 2.2.2 Cross-Layer Information . . . 20
  - 2.2.3 Network Coding . . . 23
  - 2.2.4 Cooperative Diversity . . . 26
- 2.3 PHY-interactive Schemes . . . 28
  - 2.3.1 Orthogonal Frequency-Division Multiple Access . . . 28
  - 2.3.2 Interference Alignment . . . 32
  - 2.3.3 Software-defined Radios . . . 35
- 2.4 Summary . . . 36

II CORRIDOR-BASED ROUTING 37

3 ARCHITECTURE 39
- 3.1 Corridor Concept . . . 39
  - 3.1.1 Corridor Characteristics . . . 40
  - 3.1.2 Corridor Trade-offs . . . 42
- 3.2 Corridor Operation . . . 43
  - 3.2.1 Protocol Phases . . . 44
  - 3.2.2 Corridor Construction Protocol . . . 47
- 3.3 Summary . . . 57

4 SINGLE LINK STAGE MECHANISM 59
- 4.1 Single Link Design . . . 60
  - 4.1.1 Profile . . . 60
  - 4.1.2 Operation . . . 61
  - 4.1.3 Corridor Protocol Phases . . . 63
- 4.2 Implementation . . . 63
  - 4.2.1 Network Layer . . . 63
  - 4.2.2 Link Layer . . . 64
  - 4.2.3 Physical Layer . . . 65

- 4.3 Evaluation . . . 68
  - 4.3.1 Scenario . . . 68
  - 4.3.2 Results . . . 68
- 4.4 Summary . . . 73

5 OFDMA STAGE MECHANISM 75
- 5.1 Fundamental Gains . . . 76
  - 5.1.1 System Model . . . 76
  - 5.1.2 Analysis . . . 77
  - 5.1.3 Discussion . . . 82
- 5.2 OFDMA Design . . . 83
  - 5.2.1 Profile . . . 83
  - 5.2.2 Practical Issues . . . 84
  - 5.2.3 Resource Allocation . . . 88
  - 5.2.4 Operation . . . 92
  - 5.2.5 Corridor Protocol Phases . . . 94
- 5.3 Implementation . . . 95
  - 5.3.1 Network Layer . . . 95
  - 5.3.2 Link Layer . . . 95
  - 5.3.3 Physical Layer . . . 96
- 5.4 Evaluation . . . 98
  - 5.4.1 Scenarios . . . 98
  - 5.4.2 Results . . . 99
  - 5.4.3 Discussion . . . 104
- 5.5 Summary . . . 104

6 INTERFERENCE ALIGNMENT STAGE MECHANISM 107
- 6.1 Enabling Frequency IA . . . 108
  - 6.1.1 Operation of IA in the Frequency Domain . . . 108
  - 6.1.2 Resource Diversity . . . 111
  - 6.1.3 Selection Algorithms . . . 115
  - 6.1.4 Implementation . . . 117
  - 6.1.5 Evaluation . . . 120
- 6.2 IA Design . . . 128
  - 6.2.1 Profile . . . 128
  - 6.2.2 Resource Allocation . . . 129
  - 6.2.3 Operation . . . 131
  - 6.2.4 Corridor Protocol Phases . . . 132
- 6.3 Implementation . . . 132
  - 6.3.1 Physical Layer . . . 133
  - 6.3.2 Practical Issues . . . 134
- 6.4 Evaluation . . . 134
  - 6.4.1 Scenarios . . . 135
  - 6.4.2 Results . . . 135
  - 6.4.3 Discussion . . . 140
- 6.5 Summary . . . 141

7 SYSTEM EVALUATION 143
- 7.1 WARP Drive . . . 143
  - 7.1.1 Architecture . . . 144

7.1.2 Benchmarks 147
7.2 Evaluation 148
7.2.1 Scenario 149
7.2.2 Results 149
7.3 Summary 155

III DISCUSSION 157
8 DISCUSSION AND OUTLOOK 159
8.1 Discussion 159
8.1.1 Throughput Gains 159
8.1.2 Performance Estimation 161
8.1.3 Resource Selection 162
8.1.4 Stage Compatibility 163
8.1.5 Corridor Length and Width 164
8.2 Outlook 165
8.2.1 Stage Mechanisms 165
8.2.2 Corridor Construction 167
8.2.3 Summary 169
9 CONCLUSIONS 171

BIBLIOGRAPHY 173

IV APPENDIX 187
A WARP DRIVE PARAMETERS 189
B CURRICULUM VITÆ 191
C AUTHOR'S PUBLICATIONS 193
D ERKLÄRUNG LAUT §9 DER PROMOTIONSORDNUNG 197

LIST OF FIGURES

Figure 1.1 Challenges and Goals 5
Figure 1.2 Corridor Intuition 7
Figure 1.3 Thesis Structure 9
Figure 2.1 Overview on Related Work 12
Figure 2.2 Cross-Layer Approaches 20
Figure 2.3 Multi-Hop Signal-Level Network Coding 25
Figure 2.4 Blind Interference Alignment Scenario 34
Figure 3.1 Corridor Notation 39
Figure 3.2 Urban Environment Example 40
Figure 3.3 Corridor-based Routing Protocol 44
Figure 3.4 Stage Construction and Reorganization 48
Figure 3.5 Corridor Construction Control Messages 51
Figure 3.6 Practical Wireless Multi-hop Network Testbed 53
Figure 3.7 Corridor Construction Overhead 54
Figure 3.8 Degree of Construction 55
Figure 3.9 Practical Corridor Construction Results 57
Figure 4.1 Single Link Modes 59
Figure 4.2 Single Link Frame Format 61
Figure 4.3 Packet Segmentation 64
Figure 4.4 Transmission Time Estimation 66
Figure 4.5 Impact of Inaccurate Batch Number Estimation 67
Figure 4.6 Single Link Throughput 69
Figure 4.7 Corridor Length Impact on Single Link 70
Figure 4.8 Single Link Failed Transmissions 71
Figure 4.9 Single Link per Stage Failures 72
Figure 5.1 OFDMA Allocation Intuition 75
Figure 5.2 OFDMA Gain Intuition 76
Figure 5.3 OFDMA Capacity in Simulation 79
Figure 5.4 OFDMA BER/SER in Simulation 80
Figure 5.5 OFDMA Single Stage Setup 81
Figure 5.6 OFDMA CSI Feedback 84
Figure 5.7 CSI-based Allocation 90
Figure 5.8 CSI-based and Traffic-based Allocation 91
Figure 5.9 OFDMA Frame Format 92
Figure 5.10 OFDMA Scheduling Example 93
Figure 5.11 OFDMA Scheduling Algorithm 94
Figure 5.12 OFDMA Quantization Noise 96
Figure 5.13 Predefined Corridor Testbed 99
Figure 5.14 OFDMA PHY Gain 100
Figure 5.15 OFDMA System Gain in Simulation 101
Figure 5.16 OFDMA System Gain in Practice 102
Figure 6.1 IA Intuition 107

Figure 6.2 Frequency IA Example 109
Figure 6.3 IA Frame Format 111
Figure 6.4 IA Single-Hop Node Pairings 113
Figure 6.5 IA CSI Feedback 118
Figure 6.6 Practical Single-Hop IA Setup 122
Figure 6.7 Single-Hop IA Simulation Results 122
Figure 6.8 Plain IA Performance 124
Figure 6.9 IA Node Selection 124
Figure 6.10 IA Time Dependence 125
Figure 6.11 IA Mechanism Selection 126
Figure 6.12 IA External Interference 126
Figure 6.13 IA in an NLOS Setup 127
Figure 6.14 IA with Adaptive Modulation 127
Figure 6.15 IA Stage Mechanism Node Pairings 130
Figure 6.16 IA PHY Transmission 133
Figure 6.17 IA Multi-Hop Scenarios 135
Figure 6.18 IA Stage Throughput Gain 136
Figure 6.19 IA Gain over OFDMA 137
Figure 6.20 IA Operability 137
Figure 6.21 IA on Consecutive Stages 138
Figure 6.22 IA vs. OFDMA in Multiple Scenarios 139
Figure 6.23 IA Operability in Multiple Scenarios 139
Figure 6.24 IA and OFDMA Characteristics 140
Figure 7.1 Decision Engine 143
Figure 7.2 WARP Drive Design 145
Figure 7.3 WARP Drive Operation and Performance 147
Figure 7.4 Stage Mechanism Selection 150
Figure 7.5 SNR Gain Study 152
Figure 7.6 Scenario Gain Study 153
Figure 7.7 Decision Engine Performance 155
Figure 8.1 Overview on Future Work. 166
Figure 8.2 Intersection and Overlapping of Corridors. 168

LIST OF TABLES

Table 2.1 Overview on Related Work 36
Table 3.1 Corridor Construction Control Packets 50
Table 3.2 Corridor Construction Scenarios 52
Table 4.1 Single Link Profile 60
Table 4.2 Strider Parameters 65
Table 5.1 OFDMA Single Stage Experiment Parameters 81
Table 5.2 OFDMA in Theory, Simulation, and Practice 82
Table 5.3 OFDMA Practical Error Rate Improvement 82

Table 5.4 OFDMA Profile . . . 83
Table 5.5 CSI Feedback Compared to OFDMA BER . . . 85
Table 6.1 Selection Algorithms . . . 117
Table 6.2 IA with Increasing Number of Nodes . . . 123
Table 6.3 IA Profile . . . 128
Table 8.1 Discussion Overview . . . 160
Table A.1 WARP Drive Parameters . . . 189

LIST OF ALGORITHMS

1 Single-Path Construction Algorithm . . . 48
2 Stage Construction Algorithm . . . 49

3 Best-out-of-n^2 Allocation Mechanism . . . 77
4 Fair CSI-based Allocation Mechanism . . . 89

ACRONYMS

ADC	Analog-to-Digital Converter
AGC	Automatic Gain Control
ANC	Analog Network Coding
AODV	Ad hoc On-Demand Distance Vector Routing
AP	Access Point
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BPS	Bits Per Symbol
BPSK	Binary Phase-Shift Keying
CDF	Cumulative Distribution Function
CFO	Carrier Frequency Offset
CP	Cyclic Prefix
CRC	Cyclic Redundancy Check
CSI	Channel State Information
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance

CTS	Clear-to-Send
DoC	Degree of Construction
DOF	Degree-of-Freedom
DSR	Dynamic Source Routing
ETX	Expected Transmission Count
EVM	Error Vector Magnitude
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
GPP	General-Purpose Processor
IA	Interference Alignment
IoT	Internet of Things
ISI	Inter-Symbol Interference
LNK	Link Layer
LOS	Line-Of-Sight
LQI	Link Quality Indicator
MCS	Modulation and Coding Scheme
MIM	Message in Message
MIMO	Multiple-Input Multiple-Output
MS	Mobile Station
NAV	Network Allocation Vector
NET	Network Layer
NLOS	Non-Line-Of-Sight
NNC	Natural Network Coding
OFDM	Orthogonal Frequency-Division Multiplexing
OFDMA	Orthogonal Frequency-Division Multiple Access
OLSR	Optimized Link State Routing
PDF	Probability Density Function
PHY	Physical Layer
PNC	Physical-layer Network Coding
PPR	Partial Packet Recovery

PS	Per-Subchannel Scheduling
QAM	Quadrature Amplitude Modulation
RTS	Request-to-Send
SDR	Software-defined Radio
SER	Symbol Error Rate
SIC	Successive Interference Cancellation
SINR	Signal-to-Interference-plus-Noise Ratio
SISO	Single-Input Single-Output
SL	Single Link
SNR	Signal-to-Noise Ratio
STO	Symbol Time Offset
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
TTL	Time To Live
WARP	Wireless Open-Access Research Platform
WB	Wideband Scheduling
WMN	Wireless Multi-hop Network

PREVIOUSLY PUBLISHED MATERIAL

This thesis includes material previously published in peer-reviewed publications. In accordance with the regulations of the Computer Science department at TU Darmstadt, we list below the chapters which include verbatim fragments from these publications.

Chapter 2

- Section 2.3.3 revises the section on related work of "WARP Drive — Accelerating Wireless Multi-hop Cross-layer Experimentation on SDRs" by Adrian Loch, Matthias Schulz, and Matthias Hollick. In *Proceedings of the 3rd ACM Workshop of Software Radio Implementation Forum (SRIF 2014), co-located with ACM SIGCOMM 2014*, 2014.

Chapter 3

- Sections 3.1 and 3.2 revise "Corridor-based Routing: Opening Doors to PHY-Layer Advances for Wireless Multihop Networks" by Adrian Loch, Matthias Hollick, Alexander Kuehne, and Anja Klein. In *Proceedings of the 15th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2014)*, 2014.

- Section 3.2.2 revises "Building Cross-Layer Corridors in Wireless Multihop Networks" by Adrian Loch, Pablo Quesada, Matthias Hollick, Alexander Kuehne, and Anja Klein. In *Proceedings of the 11th IEEE International Conference on Mobile Ad hoc and Sensor Systems (MASS 2014)*, 2014.

Chapter 4

- Section 4.2 revises "OFDMA for Wireless Multihop Networks: From Theory to Practice" by Adrian Loch, Matthias Hollick, Alexander Kuehne, and Anja Klein. Submitted to *Pervasive and Mobile Computing*.

Chapter 5

- Section 5.1 revises "Practical OFDMA in Wireless Networks with Multiple Transmitter-Receiver Pairs" by Adrian Loch, Robin Klose, Matthias Hollick, and Alexander Kuehne. In *Proceedings of the 14th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2013)*, 2013.

- Section 5.2.2.1 revises "CSI Feedback in OFDMA Wireless Networks with Multiple Transmitter-Receiver Pairs" by Adrian Loch, Matthias Hollick, Thomas Nitsche, Joerg Widmer, Alexander Kuehne, and Anja Klein. In *Proceedings of the 14th IEEE International Workshop on Signal Processing Advances for Wireless Communications (SPAWC 2013)*, 2013.

- Sections 5.2.2.3 and 5.2.2.4 revise "A Rapid Prototyping Framework for Practical OFDMA Systems using Software Defined Radios" by Robin Klose, Adrian Loch, and Matthias Hollick. In *Proceedings of the 2013 IEEE International Conference on Sensing, Communication, and Networking (SECON 2013)*, 2013.

- Sections 5.2.3.2, 5.2.4.3, 5.4.2.2, and 5.4.3 revise "OFDMA for Wireless Multihop Networks: From Theory to Practice" by Adrian Loch, Matthias Hollick, Alexander Kuehne, and Anja Klein. Submitted to *Pervasive and Mobile Computing*.

- Section 5.4.2.1 revises "Practical OFDMA for Corridor-based Routing in Wireless Multihop Networks" by Adrian Loch, Matthias Hollick, Alexander Kuehne, and Anja Klein. In *Proceedings of the 39th IEEE Conference on Local Computer Networks (LCN 2014)*, 2014.

CHAPTER 6

- Section 6.1 revises "Practical Interference Alignment in the Frequency Domain for OFDM-based Wireless Access Networks" by Adrian Loch, Matthias Hollick, Thomas Nitsche, Joerg Widmer, Alexander Kuehne, and Anja Klein. In *Proceedings of the 15th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2014)*, 2014.

CHAPTER 7

- Section 7.1 revises "WARP Drive — Accelerating Wireless Multi-hop Cross-layer Experimentation on SDRs" by Adrian Loch, Matthias Schulz, and Matthias Hollick. In *Proceedings of the 3rd ACM Workshop of Software Radio Implementation Forum (SRIF 2014), co-located with ACM SIGCOMM 2014*, 2014.

Part I

INTRODUCTION

1

INTRODUCTION

1.1 MOTIVATION

Wireless Multi-hop Networks (WMNs) have not made it into everyday life despite significant efforts in this research area. The reasons are twofold. First, single-hop systems such as LTE and 802.11 cover the needs of today's wireless devices—their centralized design allows for a much lower complexity compared to WMNs. Second, the performance of WMNs is often significantly lower than for single-hop systems. Essentially, the latter benefit from recent advances at the Physical Layer (PHY) particularly designed for single-hop communication to keep up with increasing wireless demands.

However, the vision behind the Internet of Things (IoT) challenges the wireless single-hop paradigm and thus the first of the aforementioned reasons. The main problem is scalability—single-hop systems typically mitigate the increase of wireless devices by deploying more base stations, but in practice the density of such base stations is limited. For instance, deploying and wiring cells significantly smaller than a room might become infeasible in existing buildings. Outdoor environments are even more challenging, as such a dense cell deployment would require wiring the landscape. WMNs address this issue by allowing for wireless data forwarding. Many existing IoT systems raising from highly successful crowdfunded projects into commercial products have identified the need for such networks and are thus based on a multi-hop design. For example, the smart light bulb system LIFX[1] uses a multi-hop network to interconnect all light bulbs in a home. The original project raised more than ten times its initial goal on a crowdfunding platform, showing the popularity of the IoT. Similarly, the smart plug system Zuli[2] is also based on WMNs, and emerged from a successful crowdfunding campaign, too. Further platforms designed for the IoT and based on WMNs include Flutter[3] and Pinoccio[4]. In other words, the IoT is pushing WMNs into commercial products since single-hop systems are not well-suited for such a scenario. Moreover, WMNs are not only becoming present in independent, crowdfunded projects but also in widespread technology. For instance, recent versions of Apple iOS natively support mesh networking. As a result, applications such as Firechat[5], which allows for multi-hop instant messaging, have emerged.

Although the performance in terms of throughput of state-of-the-art WMNs is limited, it covers the needs of the aforementioned examples—all of them generate only small amounts of traffic, such as sensor readings, actuator instructions, or text messages. However, further applications such as multi-hop sharing of Internet connections, or multi-hop video streaming among multimedia devices require significantly more bandwidth. This directly relates to the second reason for WMNs not being part of everyday life—as opposed to single-hop systems, WMNs use simple PHYs that limit their practical performance [41]. Advanced PHY techniques typically exploit spatial diversity. That

---

[1] `http://www.lifx.co/`

[2] `http://www.zuli.io/`

[3] `http://www.flutterwireless.com/`

[4] `http://www.pinocc.io/`

[5] `http://www.opengarden.com/firechat`

is, they exploit that the Channel State Information (CSI) of each link in a network is random and linearly independent from the CSI of all other links in the network with high probability. This allows for techniques such as Orthogonal Frequency-Division Multiple Access (OFDMA), Interference Alignment (IA) [20], and Multiple-Input Multiple-Output (MIMO). For instance, OFDMA enables a fine-granular allocation of resources to multiple links. Since links have random and different CSI, the probability of at least one link having good channel conditions on each fine-granular resource is high (c.f. Chapter 5). Further, IA exploits the linear independence of link CSI to align interfering signals into the same subspace, while desired signals lie in orthogonal subspaces. Hence, IA can decode the desired signals by combining aligned unknowns in an undetermined system of equations (c.f. Chapter 6). In contrast to opportunistic approaches that adapt the Network Layer (NET) to the random behavior of the PHY, such as MIXIT [67], the aforementioned techniques directly operate at the PHY and require CSI at the transmitters.

WMNs inherently provide for spatial diversity as many links are available. Still, state-of-the-art WMNs do not use advanced PHYs since such techniques pose significant challenges in a decentralized scenario. In particular, all devices participating in a transmission using an advanced PHY typically need to (a) be coordinated regarding resource allocation, (b) be synchronized in both time as well as frequency, and (c) share timely CSI. In single-hop systems, often a central controller solves these issues. For instance, a base station or an access point can decide on resource allocations, provide tight synchronization using preambles, and collect CSI efficiently using the available direct links. In contrast, coordinating multiple equal peer nodes in a WMN is hard. In particular, nodes must achieve a consensus regarding the allocation of PHY resources before the characteristics of the PHY change. Similarly, the system must provide CSI exchange and node synchronization over multiple hops before both become outdated. The larger the network, the more challenging the aforementioned issues since existing approaches do not scale. The key problem is that the NET and the PHY operate at different timescales. Essentially, channel fluctuations at the PHY occur at a speed orders of magnitude larger than the operation of the NET. Hence, using advanced PHYs in a multi-hop environment is typically infeasible. As a result, traditional WMNs stick to basic PHYs inspired by early 802.11 standards. Unsurprisingly, the performance of such WMNs is significantly worse than modern versions of single-hop 802.11—similarly to how a horse-drawn car is slower than a motor car.

In this work, we address the aforementioned issues in order to enable state-of-the-art PHYs for WMNs. To this end, we follow a divide and conquer approach. Essentially, we enable fully connected groups of nodes to use advanced PHYs techniques—within such groups these techniques become feasible, similarly to single-hop systems. We then allow the groups to cooperate in order to forward packets over multi-hop routes. Our goal is to improve the practical performance of WMNs. Our motivation is to provide mechanisms for a wireless network connecting everything and everyone, everywhere.

1.2 CHALLENGES AND GOALS

Section 1.1 identifies the limited performance of practical WMNs as one of their main problems. Based on this, in Figure 1.1 we derive the sub-problems that we need to solve to address this issue. Next, we identify for each sub-problem a general approach on how to deal with it. We formulate the result as a sub-challenge, and a corresponding sub-goal. Finally, we infer the main goal of this thesis based on the individual sub-goals.

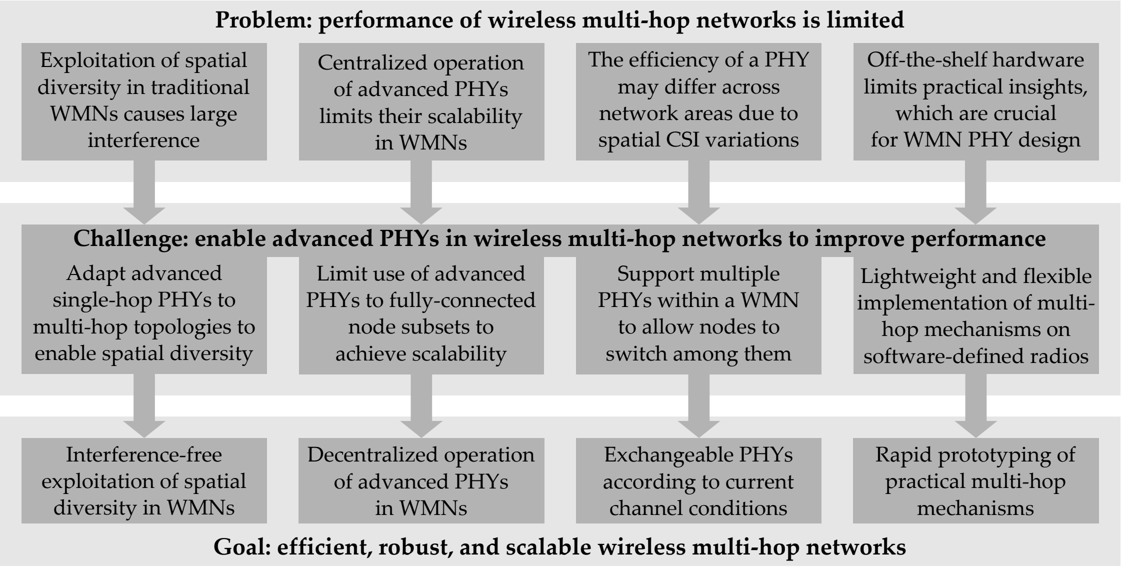

Problem: performance of wireless multi-hop networks is limited
Exploitation of spatial diversity in traditional WMNs causes large interference
Centralized operation of advanced PHYs limits their scalability in WMNs
The efficiency of a PHY may differ across network areas due to spatial CSI variations
Off-the-shelf hardware limits practical insights, which are crucial for WMN PHY design
Challenge: enable advanced PHYs in wireless multi-hop networks to improve performance
Adapt advanced single-hop PHYs to multi-hop topologies to enable spatial diversity
Limit use of advanced PHYs to fully-connected node subsets to achieve scalability
Support multiple PHYs within a WMN to allow nodes to switch among them
Lightweight and flexible implementation of multi-hop mechanisms on software-defined radios
Interference-free exploitation of spatial diversity in WMNs
Decentralized operation of advanced PHYs in WMNs
Exchangeable PHYs according to current channel conditions
Rapid prototyping of practical multi-hop mechanisms
Goal: efficient, robust, and scalable wireless multi-hop networks

Figure 1.1: Challenges and goals of this thesis.

1.2.1 *Spatial Diversity*

In order to tackle the limited performance of WMNs, prior work aims at exploiting spatial diversity using multi-path mechanisms [110, 124, 84]. Essentially, such mechanisms send packets to the destination along multiple different paths. In theory, the delivery ratio improves since the probability that at least one of the paths is successful becomes larger the more paths we use. However, the problem stems from the resulting effects at the PHY since the nodes on each of the paths are often close to each other and thus compete for the channel. As a result of node contention, performance in practical scenarios typically breaks down, and the delivery ratio may be even lower than without multi-paths.

Still, state-of-the-art single-hop PHY techniques show that spatial diversity can provide large performance gains in terms of throughput as well as robustness. Hence, the challenge is to enable such techniques in a multi-hop environment. The key difference between exploiting spatial diversity using multi-paths at the NET or using the aforementioned techniques at the PHY is that in the former case nodes *compete* at the PHY while in the latter they *cooperate*. In other words, while simultaneous transmissions obstruct the operation of multi-paths, advanced PHYs intentionally cause them. Our goal is to use such an approach to achieve interference-free exploitation of spatial diversity in WMNs.

1.2.2 *Scalability*

Multi-hop scenarios pose a significant problem to state-of-the-art PHYs. In particular, nodes transmitting simultaneously often need up-to-date CSI of all links involved in the transmission in order to compute, for instance, precoding vectors or resource allocations. As highlighted in Section 1.1, CSI is typically outdated by the time it is forwarded beyond one hop. Moreover, the larger the network, the more complex it becomes to determine which nodes shall participate in a certain transmission to optimize the resulting performance. Also, the more nodes participate, the more difficult it is to provide timely CSI to all of them. In other words, state-of-the-art PHYs do not scale.

To tackle the aforementioned scalability issues, we aim at using state-of-the-art PHYs only in fully connected subsets of the network. As a result, (a) CSI does not need to be forwarded over multiple hops, and (b) the number of participating nodes is limited. Hence, the challenge is to design mechanisms that form such fully connected subsets of nodes of a suitable size for the PHY in use. This raises a number of research questions such as the trade-off between the subset size and the achievable performance—the larger the subsets, the more spatial diversity but also the more effort needed to distribute CSI. All in all, our goal is to enable the decentralized operation of advanced PHYs in WMNs.

1.2.3 *Real-World Environments*

In Section 1.1, we highlight a number of advanced PHY techniques (OFDMA, IA, and MIMO). However, this is just a selection—recent work proposes a plethora of such techniques. The most suitable technique for a certain WMN strongly depends on the channel conditions, as well as the network topology and the ongoing data flows in the network. The problem in real-world environments is that these characteristics are usually different in different parts of a network. For instance, one part of a network may be indoors and thus feature high node density, strong links, and high frequency selectivity. In contrast, a different part may be outdoors, featuring few nodes, weak links, and less selective channels. Thus, a PHY beneficial for the indoor part may be unsuitable for outdoors, and vice versa.

The challenge is to allow WMNs to use different PHYs in different fully connected node subsets (c.f. Section 1.2.2). To this end, we need to design mechanisms that enable nodes of one subset to adapt the incoming data to the PHY of the next subset. For instance, the amount of received data at each node influences the allocation of PHY resources in the next subset. Additionally, nodes in a certain area may switch to a different PHY technique if channel characteristics in their environment change over time. In other words, our goal is to enable spatio-temporal *transitions* among exchangeable PHY techniques to adapt to the real-world environment in WMNs.

1.2.4 *Practical Validation*

Practical effects such as Carrier Frequency Offset (CFO), Symbol Time Offset (STO), or gain misadjustments have a significant impact on PHY performance. These effects often become even more severe in WMNs since WMNs lack a central instance that can correct such variations homogeneously on all nodes. Hence, practically validating our mechanisms that enable state-of-the-art PHYs in WMNs is crucial. The problem stems from the practical implementation of techniques at the PHY, which may require a prohibitive effort. Even using a Software-defined Radio (SDR) platform might require costly implementation on a Field Programmable Gate Array (FPGA) to validate mechanisms at the PHY. As a result, even validation cycles of simple mechanisms may take up to several months.

To keep validation cycles short, we need rapid prototyping. The challenge is to develop a framework that allows us to implement advanced PHYs in a multi-hop environment easily. Our SDR platform provides such a rapid prototyping framework based on Matlab[6] for a single transmitter-receiver pair. Hence, we need to extend it to support (a) multiple

---

[6] Throughout this work we use the term *Matlab* to refer to MATLAB and Communications Toolbox Release 2013a, The MathWorks, Inc., Natick, Massachusetts, United States, `http://www.mathworks.com/`

hops, and (b) an arbitrary number of nodes. Rapid prototyping in Matlab comes at the price of non-real-time execution, that is, signals are processed offline. However, our system needs to be online and interactive in order to, e.g., react to CSI updates. Hence, our goal is to find a trade-off between rapid prototyping and online processing.

MAIN GOAL. All in all, the main goal of this thesis is to improve the practical performance of WMNs in terms of throughput, robustness, and scalability. To this end, we enable advanced PHYs in WMNs in practice to exploit cooperative spatial diversity.

1.3 CONTRIBUTIONS

To address the aforementioned challenges, we propose Corridor-based Routing. We (a) introduce the general operation of our scheme, and (b) adapt a number of advanced PHYs to WMNs using Corridor-based Routing. In particular, our contributions are as follows.

1.3.1 *Architecture and Operation of Corridor-based Routing*

Corridor-based Routing is a multi-hop forwarding paradigm that supports spatial diversity. Essentially, data is forwarded along corridors, which are the result of widening a hop-by-hop path in order to span multiple nodes at each hop. Figure 1.2 shows an example for a corridor of width three. We call each widened hop a *stage*. Stages realize the aforementioned fully connected subsets of nodes. Each stage can have a different width and use a different stage mechanism—that is, a different PHY technique—to forward data. This implements spatial PHY transitions. To realize temporal PHY transitions, stages periodically measure channel conditions and adapt the stage mechanism in use based on the number of nodes at each side, the CSI of its links, and the amount of data at each transmitter. We call this *stage maintenance*. Finally, *stage coordination* provides consensus among nodes both within a stage, e.g., for resource allocation, and among stages, e.g., to ensure that the stage mechanisms of two subsequent stages are compatible.

1.3.2 *Stage Mechanism based on OFDMA*

We design and implement a stage mechanism based on OFDMA for Corridor-based Routing. The result is modular, that is, we can integrate it seamlessly into the architecture described in Section 1.3.1. OFDMA builds on Orthogonal Frequency-Division Multiplexing (OFDM) at the PHY, which splits the channel bandwidth into narrowband subchannels.

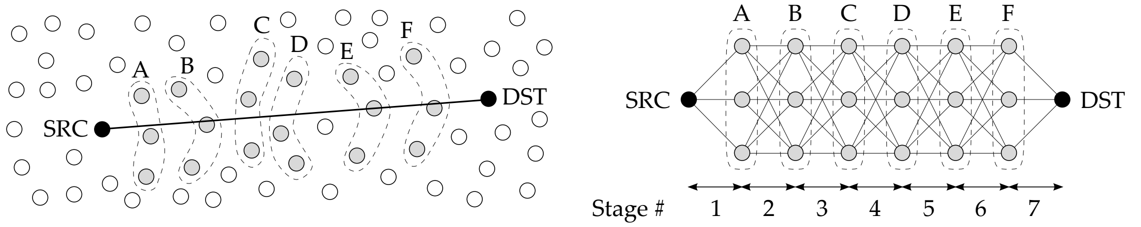


Figure 1.2: Corridor example. We select nodes during corridor construction, as shown on the left. For clarity, we often represent corridors in a structured manner, as shown on the right.

Additionally, OFDMA can assign each subchannel to a different link in the network. If the channel is frequency selective, each subchannel on each link experiences a different quality in terms of Signal-to-Noise Ratio (SNR). Hence, the PHY can allocate each subchannel to the link on which it has the largest SNR to improve transmission quality. We adapt this technique to a stage by allowing transmitters to share the available subchannels. We investigate both fair and unfair algorithms—the former allocate the same number of subchannels to each stage link while the latter share subchannels unevenly. As a result, unfair algorithms may cause bottlenecks if a node receives more data in one stage than it can forward in the next stage. Further, we analyze two metrics for resource allocation, namely, the SNR, and the stage transmission time, which takes into account the aforementioned bottlenecks. Data splits and joins at nodes as it traverses the corridor, thus becoming disordered. We include identifiers to allow the destination to reconstruct the data. We show that our OFDMA stage mechanism achieves 30% average and up to 2× throughput gains in practice compared to a traditional mechanism not using corridors.

1.3.3 *Stage Mechanism based on IA*

We also adapt IA to operate as a stage mechanism module. IA can operate in the space, time, and frequency domains. We choose IA in the frequency domain because it does not require nodes to have multiple antennas. In frequency IA, three transmitters send unicast data to three receivers using three subchannels. By aligning interference, one of the transmitters can send two packets on one resource, that is, the overall system can deliver four packets using only three subchannels, achieving a 33% throughput gain. We identify a crucial limitation of IA in practice—unfavorable channel conditions *augment* noise at the receiver prohibitively, rendering IA inoperable. However, we realize that this noise augmentation strongly depends on which transmitter sends data to which receiver on which subchannels, since CSI has a different impact in each case. If a stage has more than three transmitters or receivers, we can also choose which nodes participate in IA. Additionally, we can use IA only on beneficial subchannels and resort to OFDMA on others. In other words, IA can choose among a large set of possible resource allocations. We define a metric that allows to predict the aforementioned noise augmentation based on CSI measurements, and use it to find the best resource allocation for a given set of transmitters and receivers. We take into account the amount of data each transmitter in a stage needs to forward, and choose the allocation which minimizes the transmission time. We implement our design in practice, and show that our resource selection mechanism enables throughput gains in the range of 20% to 30% compared to a scheme not using IA.

1.4 OUTLINE

Figure 1.3 shows a block diagram of the structure of this thesis, which is divided into three parts—Introduction, Corridor-based Routing, and Discussion. The first part highlights the research question that we deal with. In particular, Chapter 1 presents the limitations of existing WMNs, and explains the challenge of including state-of-the-art PHYs in such networks. Next, in Chapter 2, we discuss previous work that aims at improving the performance of WMNs and relate it to our goals.

The second part investigates Corridor-based Routing, which is our approach to enable state-of-the-art PHYs in WMNs. Initially, in Chapter 3 we present the architecture and

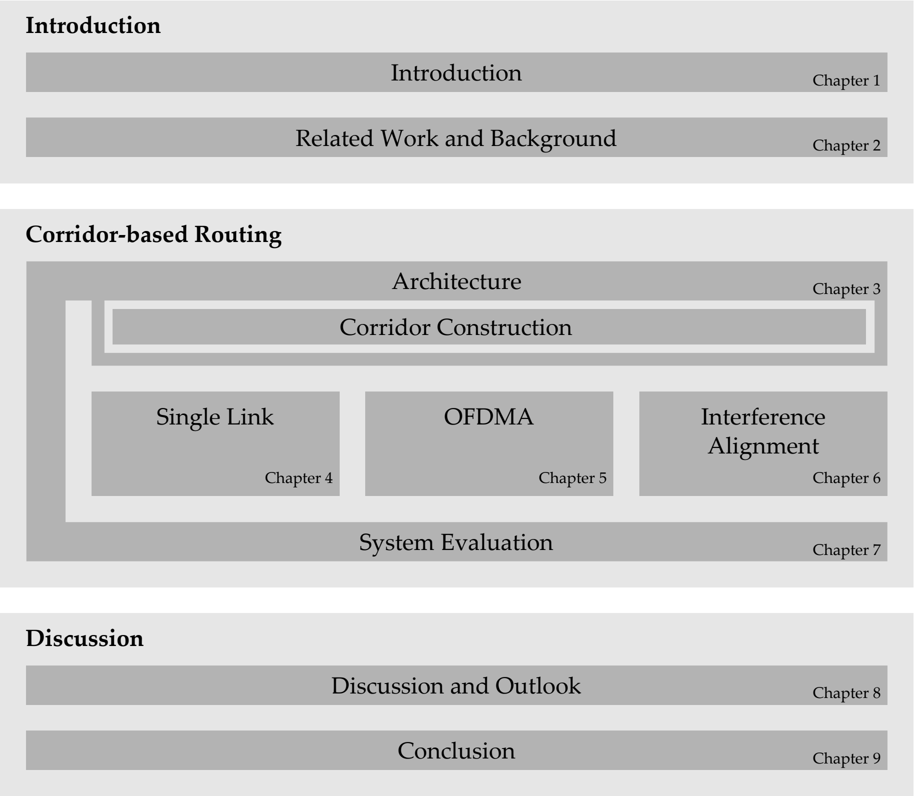


Figure 1.3: Structure of the chapters of this thesis.

operation of our system. The former deals with the general concept underlying our approach, while the latter defines a specific protocol that implements it. This includes corridor construction. In the subsequent chapters, we investigate three different stage mechanisms that corridors can use as PHY modules to forward data. First, we describe a single link stage mechanism in Chapter 4. This mechanism only transmits on one link of a stage at a time using OFDM. Second, in Chapter 5 we present in detail the OFDMA stage mechanism we sketch in Section 1.3.2. We also discuss practical results which show the improved throughput and robustness of a corridor using OFDMA compared to a traditional WMN. Third, in Chapter 6 we dive into the specific operation of the IA stage mechanism that we summarize in Section 1.3.3. We present results focused on the operation of the stage mechanism itself, that is, on the resource selection for IA. We then conclude this part of the thesis with a system evaluation in Chapter 7 which includes the full operation of the corridor, as well as all the aforementioned stage mechanisms.

In the third part of this thesis, we first discuss our results and give an outlook on future work in Chapter 8. This includes, for instance, a sketch on the possible benefits of operating multiple simultaneous corridors in a network. Finally, we summarize and conclude the thesis in Chapter 9.

2

RELATED WORK AND BACKGROUND

In Chapter 1 we present the goal of this work—enable advanced PHYs in WMNs to improve their practical performance. Our approach is based on the cooperation of multiple nodes at the PHY to exploit spatial diversity. To enable this cooperation, we need to coordinate nodes in a multi-hop environment, which inevitably involves the NET. As a result, being between the PHY and the NET, the Link Layer (LNK) is also a fundamental part of our approach. In other words, we span all three lower layers of the protocol stack.

A significant amount of related work deals with improving the performance of WMNs on each of the aforementioned layers, as well as combinations of these layers. Initially, many approaches focused exclusively on the NET [83] since the NET forms the WMN in the first place. However, it soon became clear that the limitations of WMNs were rooted at the PHY [40]. Hence, research efforts on WMNs have moved towards dealing with the characteristics of the wireless medium [68, 67, 22] in order to tackle the source of the problem directly. That is, state-of-the-art WMNs are by no means limited to the NET but are strongly related to the PHY.

Related work exhibits different levels of interaction with the PHY: PHY-aware, cross-layer, and PHY-interactive. First, PHY-aware approaches operate exclusively at the NET but modify its behavior to account for the characteristics of the wireless medium without interacting with it. For instance, these approaches often do not use the hop count as a metric at the NET but rather more sophisticated metrics such as the Expected Transmission Count (ETX) [30]. That is, they take into account that a low hop count typically implies long, unreliable wireless links, and thus favor short, high quality links which cause less retransmissions resulting in a smaller end-to-end transmission time. However, PHY-aware approaches do not obtain any additional information from the lower layers, nor do they change their parameters. Second, cross-layer schemes violate the layered protocol stack model to obtain information from the lower layers, thus achieving a higher level of interaction with them than PHY-aware approaches. For instance, at the LNK, such cross-layer information can help identifying interference patterns in a multi-hop environment [155], thus allowing for simultaneous transmissions which a traditional Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol would have prevented unnecessarily. At the NET, cross-layer mechanisms allow nodes to, e.g., split packets and forward each fragment opportunistically via different routes based on the PHY decoding reliability at each intermediate hop [67]. Finally, state-of-the-art PHY-interactive approaches advance even further towards the PHY. However, related work at this level of interaction quickly becomes limited. Only a few existing approaches directly change or parameterize the PHY—most of them assume a PHY similar to the 802.11 standard [57]. Moreover, schemes that do modify the lower layers typically consider only one specific PHY mechanism, such as using OFDMA instead of OFDM [148]. In contrast, in this work we study how to enable virtually any PHY in WMNs, thus increasing the interaction with the PHY one step further than the current state-of-the-art.

In this chapter, we survey existing efforts to improve the performance of WMNs as well as background on the specific PHY techniques that we enable for WMNs. We categorize

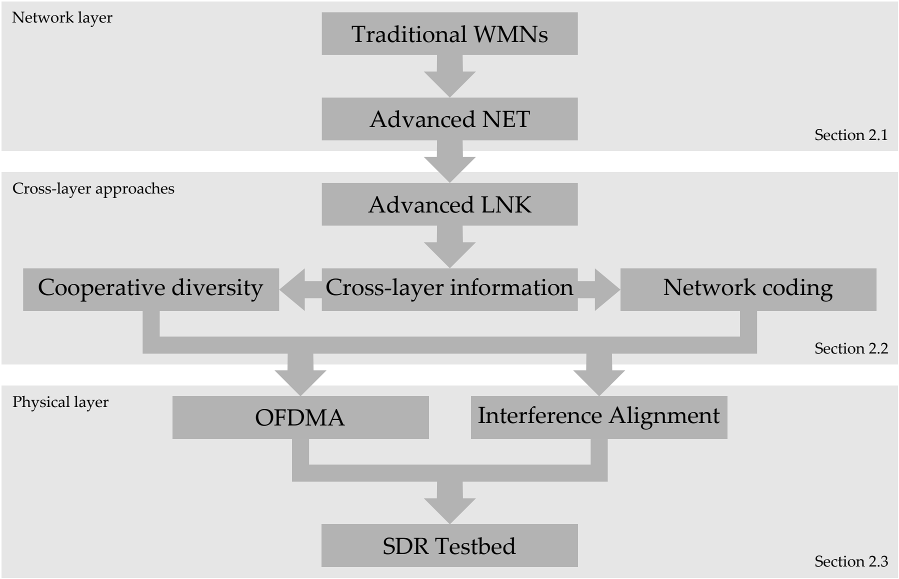


Figure 2.1: Overview on the structure of our survey on related work.

related work according to the three aforementioned levels of interaction with the PHY, that is, PHY-aware (Section 2.1), cross-layer (Section 2.2), and PHY-interactive (Section 2.3). As a result, we follow a top-down approach, starting from the NET and advancing step-by-step towards state-of-the-art WMN schemes operating close to the PHY. At that level of interaction, we then seamlessly move on to the second part of this thesis, which deals with Corridor-based Routing. Figure 2.1 shows the structure of this chapter and highlights our top-down approach. In Section 2.3, we also present related work on validating wireless techniques on SDRs, which is a crucial step to ensure their practicability and thus plays a fundamental role in our work.

2.1 PHY-AWARE NETWORK LAYER

In this section, we first sketch the limitations of traditional WMN schemes to highlight the importance of taking into account the characteristics of the wireless medium. Then, we present an overview of the aforementioned PHY-aware approaches.

2.1.1 *Traditional WMN Schemes*

Traditional routing mechanisms for WMNs address the problem of finding a path from a source node to a destination node on a network graph which may change frequently due to the fluctuations of the wireless medium. Existing solutions [17] can be either topological or geographical, that is, either using a link-based or a distance-based metric. Topological approaches can be further classified into reactive, such as Ad hoc On-Demand Distance Vector Routing (AODV) [118] or Dynamic Source Routing (DSR) [62], or proactive, such as Optimized Link State Routing (OLSR) [26]. While the former only initiates path discovery when a source node generates a packet, the latter periodically exchanges link information. Still, both essentially need to flood the network, either with route request messages or link updates, to determine the network topology and compute the end to end path. This potentially significant overhead is unavoidable since the topology is initially unknown to the protocol and no further information about the network is available. In contrast, geographic approaches such as greedy or face routing [104] avoid this overhead entirely—the key difference is that they assume that nodes are aware of their locations. Nodes locally share their location with their neighbors, and source nodes include the location of the destination in the headers of each packet they generate. Hence, intermediate nodes can compute which of their neighbors provides most geographical progress towards the destination without knowing the topology. Further, nodes can also forward packets along any arbitrary geometric curve from source to destination [112].

A further WMN approach that avoids flooding the network with control messages is backpressure routing [145]. Intuitively, this scheme forwards data similarly to how water flows through pipes, that is, along pressure gradients[1]. Each node keeps an independent packet queue for each destination. If a node has more packets in a certain queue than a neighbor—that is, a higher *pressure*—it forwards data to that neighbor. Since the destination consumes data and thus always has an empty queue, packets ultimately flow towards the destination. However, the original scheme [145] requires a central controller which is aware of all queue backlogs and optimizes which nodes transmit in each time slot. As a result, the scalability of backpressure routing is limited. While this scheme is mainly theoretical, recent work implements it in practice, and shows how to overcome its limitations [80, 113]. In particular, some approaches propose self-organized variants which do not require a central controller but in turn are not throughput optimal [114]. Also, backpressure routing may result in unnecessarily long routes as it uses all links of the network to forward packets. However, hybrid schemes combining backpressure with topological routing can alleviate this problem [165].

All of the aforementioned protocols abstract from the PHY and LNK. That is, they build on a simplified model of the lower layers, which in turn leads to an inefficient design. In the following, we present selected examples of this inefficiency described in the literature.

- LINK MODEL. Traditional schemes assume that a link in a WMN either exists or not. However, this strongly depends on the Modulation and Coding Scheme (MCS) and the subcarriers in use. A link may be unusable at a high MCS but still deliver packets when using a lower modulation and a stronger error correction code. However, a deep fade may impede communication even at the most robust MCS available. In such cases, the PHY can still operate the link by switching off the subcarriers close to the deep fade [122]. While this slightly reduces the available bandwidth, it may be very beneficial for the NET if, e.g., the link enables a shortcut on an otherwise prohibitively long multi-hop path. Finally, the PHY can set the MCS of each subcarrier individually, instead of using the same MCS on all of the subcarriers. This further improves the performance of links which otherwise would be inexistent.

- PACKET LOSS. The NET typically assumes that packets forwarded on a WMN link either arrive correctly or cannot be decoded at all. A decoding error can happen, for instance, as a result of channel fluctuations or interference. However, such errors often affect only a portion of a packet, either in the time or frequency domain. Hence, nodes can achieve a significant gain if they only retransmit the affected bits instead of the entire packet [61]. On a multi-hop path, the impact of such

---

[1] The water metaphor is widely used to describe backpressure routing, such as in “Notes on Backpressure Routing” by Michael J. Neely. Available at `http://ee.usc.edu/stochastic-nets/docs/backpressure.pdf`

retransmissions increases dramatically, as such costly packet retransmissions can occur randomly at any intermediate hop of a potentially long path.

- CONCURRENT TRANSMISSIONS. Most traditional routing protocols assume that two nodes in range that transmit simultaneously inevitably cause a collision resulting in a packet loss. The 802.11 LNK [57] provides multiple mechanisms to avoid collisions, such as Request-to-Send (RTS) and Clear-to-Send (CTS) messages, which inform nodes that they are within range of two communicating nodes, and thus should not transmit. However, these mechanisms may avoid simultaneous transmissions unnecessarily. For instance, depending on the timing of messages and the transmission power of nodes, a receiver may be able to decode two collided packets [42, 136]. In other words, traditional collision avoidance builds on an interference model which naturally causes a wastage of resources, and design inefficiencies.
- TOPOLOGY MODELS. Theoretical analysis of traditional WMNs is often based on simplified models, such as the unit disk graph [25], which assumes that two nodes are in range whenever they are closer than a certain distance value. However, in practice, real-world effects such as uneven antenna radiation patterns and multipath propagation result in a significantly different behavior [106]. As a result, designs that build on such models may not be practical at all, that is, they may address problems that have a low impact in practice while ignoring crucial issues.

The above examples illustrate that NET protocols agnostic to the PHY are highly inefficient. However, even if protocols adapt to the PHY, the intrinsic characteristics of standard PHYs, such as the 802.11 PHY, limit the scalability of the WMN [51]. In particular, the lack of concurrency at the PHY causes the maximum concurrent flow and the minimum cut-capacity to scale as $\Theta\left(1/r\left(n\right)k\right)$ [64], i.e., it tends to zero as the number of communicating node pairs k increases for a certain node range $r\left(n\right)$. In other words, according to this result WMNs do not scale. Nevertheless, recent work [38, 64] shows that allowing nodes to transmit or receive multiple packets concurrently overcomes this limitation. Specifically, the capacity scales as $\Theta\left(nr(n)/k\right)$ for concurrent receptions, and as $\Theta\left(n^2r^3(n)/k\right)$ for both concurrent transmissions as well as receptions [64], with n the number of nodes in the network. That is, concurrency at the PHY enables WMNs to scale better. To achieve this, not only the NET needs to adapt to the PHY but also the PHY to the NET by using state-of-the-art PHY techniques that enable nodes to send and receive multiple packets concurrently according to the needs of the NET.

Signal processing is key to enable this potential in terms of scalability [40]. For instance, many state-of-the-art PHY techniques build on combining baseband signals from multi-antenna systems to achieve concurrency, and thus a better performance [126, 60]. However, the capacity limit when using such techniques in a WMN is unclear [6]. In other words, WMNs have a large potential but their maximum theoretical performance in terms of throughput is unclear. The key problem is modeling the inherent complexity of WMNs. Existing work [6] defines three major hurdles, namely, modeling the feedback overhead for network coordination, taking into account the spatio-temporal fluctuations in WMNs, and including additional constraints such as delay or reliability. All of these issues significantly impact the performance of WMNs in practice, and are thus crucial for obtaining the aforementioned performance limit. While current approaches such as [92] determine the capacity of WMNs, they are often limited to certain PHY capabilities and thus cover only a fraction of the potential of WMNs.

2.1.2 *Advanced Routing Schemes*

In the following, we survey PHY-aware mechanisms which partially exploit the aforementioned potential of WMNs. Specifically, we focus on wireless backbones, multi-path routing, and opportunistic routing.

2.1.2.1 *Wireless Backbones*

Topological routing protocols (c.f. Section 2.1.1) often limit the performance of WMNs when flooding control messages to find paths. Due to the broadcast nature of the wireless medium, such flooding may block all transmissions in the network for a significant amount of time. To alleviate this issue, wireless backbones [162, 24] introduce a hierarchy in the network, that is, only a certain subset of so-called cluster-heads participate in the routing process. All other nodes are assigned to a nearby cluster-head, which locally relays packets addressed to them. In other words, similarly to our approach introduced in Section 1.3.1, wireless backbones divide the network into local groups of nodes. However, the key difference is that we envision such groups to *cooperate* at the PHY, while wireless backbones aim at reducing overhead without introducing any cooperation at all. Moreover, wireless backbones use clustering techniques [167] to form groups of nodes, generally without taking into account how packets flow in the network. In contrast, our design sketched in Section 1.3.1 forms groups to transport data *in a certain direction*. Still, work on wireless backbones identifies crucial issues when forming groups of nodes in a WMN, such as choosing cluster-heads [162], guaranteeing connectivity among clusters [24], and routing packets efficiently both within as well as among clusters [21]. For the latter, a combination of topological and geographic routing is most beneficial [21]. In particular, topological routing is well suited for local clusters, while geographic approaches keep overhead low on potentially long routes among cluster-heads.

2.1.2.2 *Multi-Path Routing*

Besides reducing control overhead using wireless backbones, PHY-aware schemes also can exploit spatial diversity using multi-path routing. The underlying PHY principle is that two links whose start points or end points are separated more than $\lambda/4$ are uncorrelated with high probability [106], where λ is the wavelength. For wireless systems operating at 2.4 GHz, this translates into about 3 to 4 centimeters. In other words, any two links in a WMN are likely to have different channel quality. A straightforward approach to exploit spatial diversity at the NET is to use multiple paths—if one path provides poor quality, using a different path may help [108].

A first approach to spatial diversity is local repair. Essentially, whenever a path breaks, nearby nodes locally find an alternative route that connects again the two path fractions [100]. Traditional WMNs can easily incorporate such an approach. For instance, AODV can generate a route request with a small Time To Live (TTL) value at a node detecting the path break [83]. As a result, the request is only flooded locally but is likely to reach a node which is part of the original path. The Castor protocol [37] implements a similar concept but provides a more sophisticated solution including routing security. Essentially, nodes forward packets based on per-destination reliability indicators that they keep for each neighbor. Initially, a node floods data packets because all estimators are zero. A node sets the reliability estimator for a neighbor n_c according to the number of acknowledgments that it receives via n_c for packets that it has previously forwarded to n_c. If a path breaks, the reliability of the path at each node drops. As a result, nodes forward packets via their neighbor with the next higher reliability for the current destination. While such local repair schemes can be very effective, they can only exploit spatial diversity at the level of links. In contrast, mechanisms at the PHY provide much higher granularity as they can select individual OFDM subcarriers of individual links.

A second approach to spatial diversity is using multiple paths *simultaneously*—either to achieve a higher throughput, or to provide redundancy [152, 85]. Additionally, related work proposes only including paths of equal length in the multi-path to avoid long detours, and thus undesired delays [152]. However, such an approach causes significant interference at the PHY since paths of similar lengths connecting a certain node pair are often geographically very close. In other words, instead of cooperating to forward data more efficiently, nodes on the paths *compete* for the channel, resulting in poor performance. A similar approach for multicast scenarios [85] leads to similar problems—messages are flooded within multicast groups to provide redundancy but cause interference. To avoid such issues, spatial diversity schemes must involve the PHY.

2.1.2.3 *Opportunistic Routing*

In contrast to multi-path routing, opportunistic approaches exploit spatial diversity *a posteriori*—that is, they decide after a transmission which node shall forward data. As a result, opportunistic routing forwards packets on-the-fly, and is thus able to react immediately to the fluctuations of the wireless channel. Essentially, nodes relay packets to a set of potential forwarders [16]. Depending on the channel conditions on each link during transmission, typically only a subset of these forwarders is able to decode the packet. Out of this subset, the protocol chooses the node which provides most progress towards the destination as the next forwarder. Hence, such an approach exploits spatial diversity without needing to discover multiple paths. The key challenge is coordination. That is, the protocol must ensure that at least one of the forwarders receives the packet, but also that at most one of the forwarders relays it, to avoid unnecessary transmissions.

Related work has extended the concept of opportunistic routing to include, e.g., partial packet recovery [55], and support for adaptive MCS [79]. The former allows nodes to forward packet fragments to exploit weak but long links, and to alleviate the impact of packet losses (c.f. Section 2.1.1). The latter is key to the performance of opportunistic routing since the MCS determines whether a link is operable at a certain SNR. Both analytical and practical results show that choosing a suitable MCS improves performance by up to an order of magnitude [79]. Practical work on opportunistic routing solves further issues such as supporting the Transmission Control Protocol (TCP) and using off-the-shelf hardware [55]. Additionally, opportunistic routing can optimize on-the-fly to which final destination it delivers packets—this can provide significant gain if the source indicates multiple possible destinations, such as multiple gateways in a WMN [81].

In contrast to opportunistic routing, our corridor-based approach (c.f. Section 1.3) takes routing decisions *a priori* based on CSI feedback. Opportunistic routing builds on the assumption that such CSI is prohibitively expensive in a WMN. However, we counteract this limitation by splitting corridors into stages. Moreover, work in this area shows that some feedback is also needed in opportunistic routing for rate adaptation [79]. In some cases, one-bit PHY feedback is enough [129]—that is, feedback for opportunistic routing can be of the same order of magnitude than feedback for corridors.

2.2 CROSS-LAYER APPROACHES

While the previous section deals with protocols that operate exclusively at the NET, in the following we survey mechanisms that span two or more layers out of NET, LNK, and PHY. Our goal is to identify cross-layer techniques that improve the performance of single-hop or multi-hop wireless networks based on the characteristics of the PHY. We later design our Corridor-based Routing system such that it supports and integrates cross-layer techniques. We start with mechanisms that improve the efficiency of channel use in Section 2.2.1. Next, we present schemes that exploit cross-layer information in Section 2.2.2. Finally, we introduce two areas which extend such cross-layer approaches to an entire WMN, namely, network coding in Section 2.2.3 and cooperative diversity in Section 2.2.4.

2.2.1 *Advanced Link Layer*

The LNK coordinates both channel access and channel coding. Recent work at the PHY on full duplex challenges some of the fundamental design principles of channel access. Hence, we also sketch the state-of-the-art in that area.

2.2.1.1 *Channel Access*

At the LNK, two problems limit the performance of channel access, namely, (a) deciding which node shall transmit on which resource, and (b) allowing the highest possible number of transmissions per time unit. Regarding (a), the 802.11 LNK [57] uses an often inefficient CSMA/CA protocol with binary exponential backoff to coordinate channel access. Recent cross-layer work suggests translating this protocol to the frequency domain [132]. Essentially, instead of wasting transmission time during backoff phases, nodes send a pilot on a random OFDM subcarrier. The node which picks the smallest subcarrier wins the contention. Hence, nodes need a second antenna in order to transmit their pilot and simultaneously receive the pilots from neighboring nodes. However, full duplex techniques (c.f. Section 2.2.1.2) address this requirement. Also, similar work in this area exploits non-contending neighbor nodes to circumvent the need for two antennas [128]. Essentially, neighbors provide feedback to the contending nodes on which node won. Further, since two nodes may choose the same subcarrier, such schemes introduce multiple rounds—the probability of two nodes choosing the same subcarrier twice is negligible [132]. Translating contention to the frequency domain results in an OFDMA transmission as multiple nodes transmit simultaneously on different subcarriers. However, it does not need to deal with CSI feedback and tight node synchronization, since nodes only need to transmit a simple on/off pilot. In contrast, schemes using OFDMA as an advanced PHY scheme to achieve higher throughput, must address these issues.

Beyond lightweight contention mechanisms, OFDMA can also avoid contention entirely [87]. Basically, nodes transmit multiple copies of their data simultaneously on a set of randomly chosen subcarriers. Hence, collisions may happen if two nodes choose intersecting sets of subcarriers. However, since subcarriers are typically linearly independent, collisions on multiple subcarriers result in a system of linear equations of the form $A\mathbf{x} = \mathbf{b}$. The vector of unknowns $\mathbf{x}$ represents the transmitted packets, the matrix of coefficients A stands for the CSI of each subcarrier, and vector $\mathbf{b}$ contains the received packet collisions. As long as transmitters send packets via at least as many subcarriers as nodes collide, the receivers can decode them. In other words, collisions become uncritical and contention thus unnecessary. However, in contrast to the aforementioned techniques which translate contention to the frequency domain, in this case nodes must be tightly synchronized since they transmit actual data on the individual subcarriers. Still, CSI feedback is not needed in this case either, since only the receivers need to know matrix A. Overall, we conclude that supporting advanced PHYs such as OFDMA enables wireless systems to improve—or even avoid—contention at the LNK.

Regarding (b), traditional LNK schemes such as CSMA/CA are typically conservative regarding the number of simultaneous transmissions. In particular, when using RTS/CTS, any neighbor of both the transmitter and the receiver must not transmit. However, such a restriction is often excessive in practice, and can result, for instance, in the exposed node problem [14]. That is, two neighboring transmitters are prevented from transmitting even if their receivers are at locations such that they would only receive the signal from their respective transmitter. Identifying such situations is hard—in practice, the range of nodes is highly irregular and strongly depends on the physical environment a node is located in. In other words, simple models such as unit disk graphs [25] do not capture the complexity of practical wireless networks. Hence, state-of-the-art mechanisms that aim at increasing the number of simultaneous transmissions build on empirical approaches. For instance, nodes can construct a collision map that reflects at which node each transmitter interferes [155]. Further, a transmitter can also try to exploit the capture effect opportunistically [160]. That is, instead of waiting for the channel to be available, the node directly probes each possible receiver. If one or more receivers answer with an acknowledgment, channel conditions allow for a simultaneous transmission. The node then transmits to the receiver which received its probe message best. Such empirical approaches at the NET and LNK assume that the PHY is random and unpredictable. However, involving the detailed PHY CSI directly into channel access decisions may allow for mechanisms that not only learn from past experiences but also predict them. This motivates our work on supporting advanced PHY schemes in WMNs. Higher layers can also help improving the efficiency at the LNK by, for instance, merging TCP and LNK acknowledgments [130]. However, their efficiency is bounded to the performance of the PHY they build on.

2.2.1.2 *Full Duplex*

The aforementioned mechanisms for channel access typically assume that a node cannot transmit and send simultaneously on the same frequency. The reason is a technical limitation—even if a node has two antennas and two transceiver chains, the outgoing signal drowns the incoming one since it is orders of magnitude larger. As a result, the incoming signal is likely to fall below the quantization noise floor of the receive chain's Analog-to-Digital Converter (ADC), which adjusts its range to the stronger outgoing signal. Thus, canceling the outgoing signal from the received signal to obtain the incoming signal after the ADC is infeasible.

To solve this problem, existing approaches cancel the outgoing signal *prior* to the ADC using analog hardware [32, 60]. This enables full duplex and hence opens the door to highly efficient channel access [60], as well as, for instance, low latency feedback channels [93]. The latter is crucial to support advanced PHYs, particularly in multi-hop scenarios (c.f. Chapter 1). State-of-the-art full duplex Single-Input Single-Output (SISO) systems are practical, and support current wireless standards such as 802.11ac [13]. They build on a large body of work which studies different techniques to cancel the aforementioned outgoing signal [32], and how to integrate full duplex in existing standards [60]. Further, recent work shows that full duplex MIMO is feasible in practice, too [12, 8]. In this case, complexity grows significantly since nodes not only have to cancel the outgoing signal of one antenna but of M antennas to receive M incoming signals simultaneously. Initial work in this area explores the combination of full duplex with MIMO but requires *four times* the amount of antennas than MIMO data streams to operate. However, the latest results show that full-duplex MIMO with M data streams and M antennas is (a) feasible and (b) scalable [12]. To this end, authors exploit that the antennas of a node are located close to each other and thus experience similar channel conditions. Further, they leverage that transmitting pilot symbols over all M antennas improves signal cancellation.

Supporting full duplex in a WMN may significantly increase its efficiency. While not all nodes might support full duplex, a full duplex node can simultaneously transmit to and receive from two half duplex nodes, respectively [143]. However, if the half duplex nodes are in range of each other, the half duplex transmitter would interfere at the half duplex receiver. To solve this issue, the nodes can additionally use spatial IA to ensure that the interference aligns at the half duplex receiver and that it can thus decode the signal [143]. Related work shows the feasibility of such an approach for single-hop, but translating it into multi-hop is challenging (c.f. Section 1.2). Full duplex can be beneficially combined with further techniques at the physical layer, such as backscatter communication [93]. Such systems are particularly interesting for IoT scenarios, which also motivate our work.

2.2.1.3 *Channel Coding*

Apart from channel access, the LNK is also responsible for channel coding. That is, it adapts data encoding to the current channel conditions to ensure that the receiver can decode it. This is particularly challenging when enabling advanced PHYs in a distributed environment such as a WMN. The reason is that such PHYs may deliver data not just from one transmitter to one receiver but may split streams among multiple pairs of nodes. Hence, the LNK must encode data such that each receiver can decode it. Traditional mechanisms [57] use acknowledgments and simply retransmit packets using a more robust coding in case of an error. This is very costly, specially if the PHY must retransmit all streams when decoding fails in one of them. Rateless codes [48, 56] address this issue—instead of retransmitting all data, they allow the transmitter to send only additional information to help the receiver decode. Most importantly, the transmitter does not need to know which part of the original packet produced the decoding error.

In this work, we use Strider [48] as a rateless code to deal with the aforementioned coding challenges. Instead of sending packets with a specific discrete MCS, in Strider nodes code packets into batches. If channel quality is high, the receiver only needs a few batches to decode, whereas if channel quality is low, more batches are transmitted. The key advantage of such an approach is that no batch is in vain—all batches contribute to a successful decoding. Moreover, Strider does not need to estimate the channel quality prior to transmitting data in order to choose the best rate, because it just sends batches until the receiver can decode. This simplifies the operation of the LNK significantly. However, Strider still needs some minimal feedback from the receiver to know whether it can decode or if it needs more batches. To this end, transmitters can learn the decoding Cumulative Distribution Function (CDF) to estimate how many batches a receiver needs [56]. Thus, receivers only need to send feedback when the estimated value is too small.

2.2.2 *Cross-Layer Information*

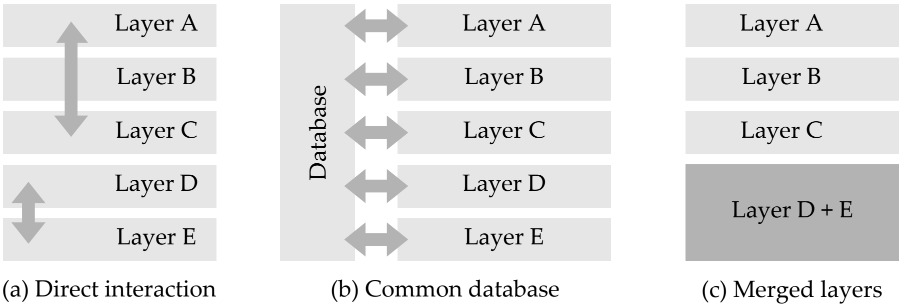
In the following, we focus on cross-layer interactions that involve the PHY, and thus help us in the design of the system we sketch in Section 1.3. In particular, from the work we discuss in this section, we learn the crucial components of cross-layer approaches which we then include in our design.

2.2.2.1 *Cross-Layer Fundamentals*

Allowing layers of the network stack to share information about data flow helps solving some of the challenges in wireless networks [41]. Early work in this area proposes different designs for such cross-layer interactions [141, 28]. Essentially, layers can (a) share information via dedicated interfaces, (b) use a common database to store such information, or (c) merge completely. Figure 2.2 gives an overview. While (c) allows for full interaction among layers, it results in a highly complex system which is difficult to maintain. In contrast, (b) provides an efficient manner for information sharing—any information that any layer publishes in the database is immediately available for any other layer that can benefit from it. Moreover, it retains the layered structure from traditional protocol stacks, including all its benefits. However, such an approach requires standardized interfaces to the database, and a common understanding on how it operates. In other words, it would imply a significant change compared to existing systems, which makes its adoption unlikely. The most widespread approach is (a), which is a compromise. It does not require any common elements but just allows layers to communicate directly via dedicated interfaces. Hence, only the layers exchanging information need to adapt to enable the cross-layer interaction. We use such an approach for our design in Chapter 3, and discuss existing work that follows a similar approach throughout this section.

2.2.2.2 *Channel State*

Cross-layer protocols that interact with the PHY often query for the state of the channel in use. For instance, a routing protocol may choose the next hop based on a Link Quality Indicator (LQI) computed at the PHY. Such approaches may require this state either at the transmitter or at the receiver. The PHY can quantify the channel state according to very diverse metrics, which range from an abstract LQI to detailed CSI. The former is often based on simplified models (e.g., the two-ray ground model), while the latter refers to the individual complex OFDM channel coefficients of each subcarrier, that is, the amplitude and phase of each coefficient. In order to obtain the channel coefficients, transmitters



(a) Direct interaction (b) Common database (c) Merged layers

Figure 2.2: Overview on cross-layer approaches.

typically send pilots symbols in each data frame. Since the CSI is often both time and frequency dependent, transmitters include pilots at regular intervals in the respective dimensions of the frame [139]. Pilot symbols are known at the receiver—hence, it can use them to deduce how the channel modifies symbols both in amplitude and phase at the time and frequency of the pilot. This information is the detailed CSI but the receiver can also process it to obtain LQIs such as the average SNR or the signal strength. While traditional WMN routing protocols often use LQIs as routing metrics, practical experiments show that the underlying PHY models do not capture the complexity of, for instance, urban environments [36]. Moreover, advanced PHYs typically require detailed CSI to operate. Most importantly, they require it *at the transmitter*, that is, the receiver must feed the CSI back to the transmitter after computing it.

Timely CSI is challenging not only in the multi-hop case but also in single-hop scenarios [39]. In Section 1.3 we sketch a technique to avoid multi-hop CSI feedback in a WMN but we still must provide single-hop CSI feedback, for which we build on existing practical approaches from, e.g., cellular networks. In the following, we summarize the insights from the extensive work in this area [96] which we use as a basis to realize CSI feedback in our system. Most practical systems, such as LTE [1], quantize CSI since full CSI feedback is often infeasible. Essentially, sending full amplitude and phase information would result in a large overhead. In contrast, codebooks known to both the transmitter and the receiver allow for efficient feedback—for each CSI value, the receiver finds the most similar codebook entry and only sends back its index. The larger the codebook, the more similar values the receiver can find, but also the larger the codebook entry indices and thus the feedback overhead. Hence, finding a suitable codebook size for a certain PHY technique is crucial. For OFDM and OFDMA, the PHY typically uses CSI feedback to identify poor subcarriers. Performance improves significantly if such subcarriers use a more robust coding [129], do not transport critical data [15], or are not used at all [122]. In this case, the most basic feedback is a single bit per subcarrier indicating whether its performance is poor. In other words, the codebook only has two entries. Even with such minimal feedback, related work [129, 23] achieves large enhancements in system performance. Theoretical studies in this area analytically show the impact of such a limited rate feedback channel on OFDM [4]. However, PHYs that precode data at the transmitter, such as IA, typically need more detailed CSI feedback (c.f. Chapter 6), and thus incur more overhead than a one-bit indicator.

2.2.2.3 *PHY Hints*

While channel state describes the medium at a certain point in time, PHY hints [153] provide information on how the medium affected a particular received data symbol. Basically, PHY hints capture the decoding confidence of the receiver for each symbol, and help the receiver on deciding how to process it. That is, PHY hints always refer to specific received data, in contrast to channel state. Hence, they are used at the receiver and do not need to be fed back. In the simplest case, the decoding confidence of a symbol in the IQ plane is directly related to the distance of the symbol to the nearest constellation point. If the distance is small, the decoding confidence is high, and vice versa.

PHY hints allow a receiver to infer much more precise information about a transmission than traditional indicators, such as the signal strength of a received frame. For instance, related work uses PHY hints to estimate the best suited MCS more accurately [154]. Specifically, receivers can compute the likelihood of packet decoding failures for each single frame even if no bit errors occur. Hence, this allows for fine-granular rate adaptation algorithms, and results in a much higher throughput than schemes that use frame errors or frame SNRs to select bit rates. In a WMN which supports multiple PHYs, this opens the door to further gains. To be specific, the PHY choice depends on the amount of time each PHY needs to successfully deliver a frame to the destination, which is directly related to the bit rate. Hence, the more accurate the bit rate selection, the more suitable PHY the WMN chooses. This motivates our design in the following chapters, which not only considers the CSI to choose the PHY in a WMN, but also the bit rate the PHY supports.

2.2.2.4 *Partial Packets*

The aforementioned PHY hints enable Partial Packet Recovery (PPR) [15, 61]. In case of a decoding error, receivers can use them to deduce which parts of a frame are likely to contain symbol errors. Hence, instead of retransmitting the entire frame, the transmitter can (a) retransmit only the frame parts with errors [61], or (b) send additional redundancy only for those parts [89]. In both cases (a) and (b), throughput improves significantly. Again, we can beneficially combine such a scheme with a WMN supporting multiple PHYs. In particular, the amount of data nodes need to transmit influences the PHY the WMN chooses. Since the amount of data is different in the retransmission than in the original transmission when using PPR, the WMN can potentially find a more efficient PHY to transport the retransmitted data. Hence, the interplay of PHY and PPR results in a number of mechanism combinations. That is, we can find the PHY and the PPR technique which are most suitable for a given scenario. This results in a highly adaptive system.

Related work on PPR shows that such adaptability enables gains even when not involving the PHY. Specifically, a receiver can decide after receiving a packet whether to request again the frame parts with errors, or only request additional redundancy [159]. While the redundancy variant is more efficient, it may cause a prohibitive CPU load at the receiver if a frame contains many errors. Thus, partial frame retransmission is more suitable in such cases. In other words, error patterns in the received frames play a fundamental role for PPR. The wireless medium often results in burst errors but the hardware both at transmitter and receiver may cause additional error patterns [53]. For instance, error probability often increases at the end of the frame due to clock drifting. That is, the CFO estimation that the receiver computes using the frame preamble becomes outdated over the course of the frame. Further, errors may occur due to the interplay of transmitter and receiver gain adaptation. Each effect causes a different error pattern which calls for a different PPR technique.

The throughput gains of PPR strongly depend on channel conditions [89]. More precisely, PPR provides gains in indoor environments with high SNRs when channel conditions result in the link operating at the border between two different rates. A system without PPR would typically choose the lower rate to avoid errors—and thus costly retransmissions—whereas a system with PPR can afford such errors [61]. In other words, PPR allows for a higher Bit Error Rate (BER) at the PHY. Our goal is to enable WMNs to switch to advanced PHYs which result in better channel conditions. Hence, combining both approaches yields a promising synergy since both advanced PHYs and PPR contribute to support a higher rate. In contrast, in outdoor environments with mobile nodes, links are significantly more unpredictable and thus rate selection cannot find a stable operating point [89]. In such a scenario, PPR provides significant gains not only at the border between rates but for any channel conditions since errors are frequent.

In the uplink of a wireless access network, PPR can operate *without* using retransmissions. Essentially, if a packet of a Mobile Station (MS) arrives with errors at two or more Access Points (APs), the APs can use PHY-hints to obtain the correct parts of the packet and then exchange them via a wired Ethernet backbone [158]. The key idea is that errors in the received packet are often *different* at each AP due to spatial diversity. The more APs overhear a packet, the higher the probability that they can reconstruct the full packet. However, such an approach requires a wired backbone, and is thus unsuitable for infrastructure-less wireless networks such as WMNs.

2.2.2.5 *Collision Decoding*

While PPR addresses bit errors regardless of the underlying cause, collision decoding provides mechanisms for the specific case of errors arising from simultaneous transmissions. Essentially, receivers can decode data even after a collision as long as the Signal-to-Interference-plus-Noise Ratio (SINR) is high enough [161]. That is, if the signal of interest is strong enough compared to the interference, the receiver can decode it. Related work suggests a number of techniques that enable collision decoding, such as Successive Interference Cancellation (SIC) [49], or Message in Message (MIM) [101]. Theoretical results show that reception of multiple packets allows WMNs to increase throughput by a factor of $\Theta(\log n)$, where n is the number of nodes [38]. In particular, a combination of multiple packet reception and network coding is promising to allow WMNs to scale.

Recent work shows that collision decoding is practical. For instance, ZigZag decoding [42] exploits that two interfering 802.11 nodes typically collide multiple times on the same packet if they experience the hidden node problem. Basically, after a collision, both nodes wait for a random time and try to retransmit their packet. Since they are mutually *hidden*, they collide again. However, packets overlap differently in each collision as each transmitter waits for a different amount of time. This collision offset enables the receiver to bootstrap decoding, since it can use the non-overlapped symbols from one collision to obtain overlapped symbols from the second collision using SIC and vice versa. In case of a single collision, the receiver can still resort to decoding only the strongest signal. However, typical receiver hardware locks on the signal which arrives first. If the strong signal arrives later, nodes can switch to it using techniques such as MIM [101]. Selecting a suitable code rate for collision decoding is particularly challenging, since transmitters often cannot determine the SINR *during* the collision. To address this, recent work suggests using rateless codes, which allow the receiver to choose the rate *a posteriori* [49].

Hence, we conclude that collision decoding (a) offers a large potential for throughput improvement in WMNs [38, 136], and (b) is feasible in practice. Still, the aforementioned techniques typically assume standard 802.11 PHY and LNK layers. In contrast, we aim at enabling advanced PHYs for WMNs. Hence, collision decoding is orthogonal to our approach—if a WMN chooses a PHY and LNK that may result in collisions, the above techniques are suitable to mitigate their effect.

2.2.3 *Network Coding*

The previous section introduces how the LNK can benefit from cross-layer information from the PHY. Next, we present related work which extends such an approach to the NET of WMNs to improve performance, and which thus deals with the main goal of our work. In particular, we survey cross-layer approaches which use network coding [5] in the form of random network coding and PHY network coding to achieve a higher throughput in wireless networks. The basic principle of network coding improves the throughput of flows intersecting at a router. Briefly, if two nodes A and B exchange packets via router R, a traditional scheme would require four transmissions—two for A and B sending their respective packets to R, and another two for R forwarding them. However, if R forwards the XOR of both packets, A and B can still decode as they still know the packets they sent, resulting in only three transmissions. In the following, we explain how such an approach interacts with the PHY, and how it relates to our goal.

2.2.3.1 *Random Network Coding*

The aforementioned network coding concept is directly applicable to WMNs. Basically, if the network needs to forward two flows in opposite directions, the intermediate routers can use network coding to relay packets using less transmissions. Further, similar schemes are beneficial in other topologies with more than two flows, too [68, 65]. In order to create more coding opportunities, nodes overhear and store packets not intended for them. As a result, they can decode coded packets that involve the previously overheard packets. This allows neighbors to code more packets together, and thus increase the amount of data conveyed with each single transmission. However, this is also the limitation of such a scheme—transmitters must be aware of the packets available at their neighbors in order to know which packets they can code together to deliver them in one transmission. This is particularly challenging, since nodes might not overhear a transmission as expected due to the random nature of the wireless medium. Random network coding [99] tackles this issue. Intuitively, source nodes code native packets of a flow into batches, that is, linearly independent combinations of the native packets. The destination can decode as soon as it receives as many coded packets as native packets were coded together at the source. The key insight is that the destination can use *any* combination of coded packets to decode, that is, it does not matter which packets intermediate nodes forward since they all contribute to the decoding process at the destination. Hence, intermediate nodes do not need to be aware of which packets are available at their neighbors.

Random network coding yields large gains in practice [22]. Still, exploiting PHY-hints (c.f. Section 2.2.2.3) results in even better performance. More precisely, intermediate nodes can code together and forward not only any packets but also *partial* packets [67]. In other words, such an approach operates at symbol level. Packets feature a header which indicates which symbols the intermediate nodes that forward it estimate to be correct. The destination decodes each symbol individually as soon as it receives enough coded versions of it. Hence, intermediate nodes can forward any data they receive—they neither need to coordinate with neighbors nor do they need to forward full packets. Further work in this area extends symbol-level random network coding to improve end-to-end flow control. For instance, intermediate nodes can send an acknowledgment to the source as soon as they can decode a batch to allow the source to start sending coded packets of the next batch [90]. Moreover, by introducing a transmission window such as in TCP, the WMN can forward multiple data segments simultaneously [91].

In summary, symbol-level random network coding is tailored to the characteristics of the wireless medium as observed from the point of view of the NET. That is, links in the WMN are random and thus unpredictable. However, this only holds at the *timescale* *of the NET* (c.f. Section 2.2.2.3). Actions at the NET such as route establishment require in the order of seconds to complete. Indeed, the PHY changes randomly over such a timespan even in a typical indoor environment. However, the PHY operates at a timescale orders of magnitude smaller which provides enough temporal resolution to perform meaningful CSI measurement and feedback. While symbol-level random network coding assumes that the NET must cope with this timescale mismatch, we aim at *decoupling* both timescales using our design sketched in Section 1.3. Specifically, we create a multi-hop structure at the NET which allows us to use advanced single-hop techniques at the PHY. In other words, we address a similar problem than symbol-level random network coding but solve it from a different perspective.

2.2.3.2 *Signal-Level Network Coding*

Although traditional network coding can exploit cross-layer information, such as PHY-hints in symbol-level random network coding, it operates primarily at the NET. In other words, packets—or symbols—are coded in the digital domain. Nodes receive and transmit packets individually. That is, traditional network coding assumes that the LNK avoids collisions. In contrast, signal-level network coding exploits collisions instead of avoiding them, since collisions *naturally* code packets together [170]. As a result, signal-level network coding further reduces the number of transmissions when two nodes A and B exchange data via an intermediate router R. Concretely, both nodes A and B can transmit their packets *simultaneously*. Router R directly receives the coded packet and relays it. Hence, such an approach requires two transmissions only instead of four (c.f. Section 2.2.3). Most importantly, signal-level network coding is beneficial also for multi-hop scenarios. Figure 2.3 shows an example. In a traditional WMN, successive packets on a path must be spaced at least two hops to avoid interference at the node in between. For instance, in Figure 2.3, nodes B and D do not transmit their packets simultaneously to avoid a collision at intermediate node C, which receives P_1. However, with signal-level network coding, C can still decode P_1 even if it collides with P_2—it obtains P_1 XOR P_2 and can thus decode P_1 if it still stores P_2. Hence, nodes can forward packets faster.

Signal-level network coding comes in two variants, namely, Physical-layer Network Coding (PNC) [88] and Analog Network Coding (ANC) [66]. The difference lies in the behavior of the router. In PNC, the router deduces the XOR of the collided packets from the received signal. That is, the decoding process automatically yields the XOR in the digital domain. Since the signals do not add modulo two in the analog domain, the receiver *maps* the received constellation points to the corresponding XOR bit in the digital domain [170]. For instance, assuming transmitters use Binary Phase-Shift Keying (BPSK) and ignoring channel distortions for simplicity, if both transmitters send symbol $1 + j \cdot 0$,

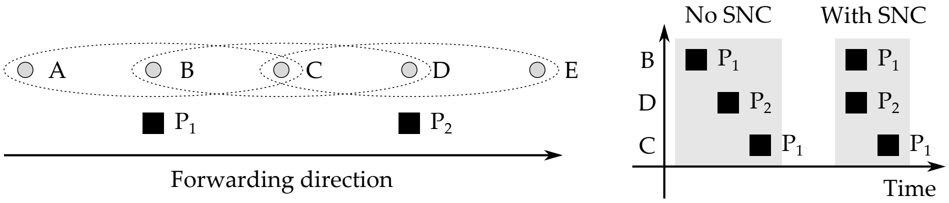


Figure 2.3: Multi-hop signal-level network coding (SNC). The gray areas represent the transmission time with and without SNC. Figure adapted from [66].

the receiver gets the sum, that is, $2 + j \cdot 0$. To obtain the XOR, the receiver maps such a received signal to a binary zero. Other symbol combinations have equivalent mappings. While straightforward for BPSK and ideal channels, PNC becomes challenging for higher order modulations and realistic channel distortions. Still, practical work in this area shows its feasibility [98]. Further, PNC also needs to deal with synchronization issues. Surprisingly, depending on how data packets are encoded and modulated, asynchrony is actually beneficial in some scenarios [97].

In contrast, ANC follows a more simple design. Basically, intermediate nodes do not decode the XOR of the collided packets but just amplify-and-forward the overlapped signals [66]. That is, in the aforementioned scenario with nodes A and B exchanging data via R, R forwards the analog sum of the signals of A and B. Node A can then obtain the signal of node B by subtracting its own signal from the sum. Node B decodes the signal of node A similarly. Hence, unlike PNC, ANC does not need to deal with mapping received signals to binary XOR values at R. However, the drawback is that such an amplify-and-forward approach also amplifies noise [102], whereas PNC decodes data at each hop. Both PNC and ANC can operate on networks similar to the structure we suggest in Figure 1.2. Hence, we can integrate them in our design as one of the advanced PHYs we support. Further, Natural Network Coding (NNC) [121] also fits on such a structure. NNC is in-between PNC and ANC since it decodes data at each node but incorporates the randomness of the channel. More precisely, NNC realizes random network coding at the PHY, that is, the coding coefficients are directly related to the CSI. When nodes A and B simultaneously transmit a packet, router R decodes the overlapped symbols by mapping them to bits according to their distance in the received constellation diagram, which includes the distortion of the channel. In other words, NNC decodes data similarly to PNC but instead of obtaining the XOR of the overlapped packets, it computes a function resulting from the CSI. Such an advanced PHY is also suitable for the system design we present in Chapter 3.

2.2.4 *Cooperative Diversity*

Wireless techniques that enable nodes to cooperate in order to deliver data to a destination, such as the above discussed mechanisms, fall into the broad category of cooperative diversity techniques [133, 134]. Specifically, cooperative diversity refers to distributed nodes which combine their antennas to achieve a higher spatial diversity. For instance, an MS in a cellular network may share its data with a second MS via an ad-hoc link in order to send its data via the link of the second MS to the base station in case its own link suffers strong fading [142].

2.2.4.1 *Relays*

A simple form of cooperative diversity is relaying, either following an amplify-and-forward or a decode-and-forward strategy. For both cases, transmitters can operate in a number of different modes. For instance, the relay may transmit on the same frequency or time resource than the source in order to form a virtual MIMO array. Alternatively, the source and the relay may transmit on orthogonal channels, allowing the destination to combine both packet receptions to improve decoding [164]. A large body of work studies such cooperative approaches in theory. Further, recent results also show their practicability [109]. In practice, cooperative diversity schemes must deal with issues such as time and frequency synchronization. CFO is particularly challenging when two nodes transmit on the same resource. Basically, the CFO is different for each node but the receiver receives a combination of both signals. Hence, cooperative diversity schemes must resort to advanced CFO correction techniques, such as transmitter-side CFO compensation. Similar practical issues arise for the PHY mechanisms we consider in Chapters 5 and 6.

Further cooperative schemes include mechanisms that exploit relays to forward partial information about a signal. For example, a relay may forward soft information such as PHY-hints of a packet it cannot decode. The destination can then combine this information with its own received signal to decode the packet correctly. Related work shows that such a scheme can yield a diversity order of two in block Rayleigh fading channels [169]. However, results in this area are often theoretical, and practical implementations that exploit them are missing. This motivates our focus on implementing advanced PHY schemes in practice.

2.2.4.2 *Multi-Hop Scenarios*

A corridor structure such as in Figure 1.2 exploits cooperative diversity over multiple hops. More precisely, nodes cooperate at the PHY in order to exploit spatial diversity and deliver data to the destination more efficiently. Related work studies simplified variants of such structures. For instance, nodes can use CSI to choose the best sequence of links to deliver data from a source to a destination [29, 50]. In particular, nodes can find links (a) locally, (b) over a certain number of hops, or (c) globally. In (a), nodes only have CSI information about their own links. Hence, at each stage (c.f. Section 1.3.1) the node which receives the packet chooses the next hop locally. In (b), nodes have CSI of a certain number of next stages. Hence, they can avoid forwarding packets to a next hop with very poor outgoing links, which would result in a bottleneck. Finally, in (c) nodes have global CSI information and thus can find the optimal end-to-end path to the destination.

Further cooperative diversity schemes suggest using all links of a stage with OFDMA [29], similarly to our approach in Chapter 5. However, existing work in this area is limited and focuses on a strictly theoretical study on outage performance. In other words, it ignores the *system* needed to operate a multi-hop cooperative scheme using OFDMA, as well as practical issues. Such a system spans the NET, LNK, and PHY, including amongst other components tailored resource allocation mechanisms, scheduling schemes, and coding techniques. Moreover, our goal is to support not only OFDMA in WMNs but virtually any advanced PHY that exploits spatial diversity. Hence, our work clearly stands apart from existing approaches in this area.

Most multi-hop cooperative diversity techniques are theoretical, since practical implementations would require costly CSI feedback over multiple hops, which is often infeasible (c.f. Section 1.2). We aim at designing a system that enables such schemes in practice. However, certain cooperative diversity techniques do not require timely CSI at the transmitters, and thus pose less challenges for WMNs. For example, multiple nodes overhearing a packet can coordinate in order to relay it simultaneously and thus achieve a higher SNR at the intended receiver [125]. To this end, transmitters must achieve tight time and frequency synchronization. Any node overhearing a certain packet can participate in the joint transmission without detailed per-subcarrier channel knowledge. In contrast, techniques such as IA, OFDMA, or MIMO require CSI at the transmitter to determine the parameters of the transmission.

2.3 PHY-INTERACTIVE SCHEMES

The previous sections discuss related work that aims at improving the performance of WMNs predominantly at the NET, the LNK, or both. In the following, we continue our top-down survey, and focus on advanced PHY schemes that we expect to be highly beneficial for WMNs. Concretely, we discuss OFDMA and IA because we realize them in Chapters 5 and 6 as example PHYs for our system that we sketch in Section 1.3. Finally, we conclude this section with an overview on SDR platforms to highlight the need for a rapid prototyping SDR framework for WMNs.

2.3.1 *Orthogonal Frequency-Division Multiple Access*

OFDMA is a state-of-the-art multiple access scheme that builds on top of OFDM. Basically, OFDM splits the bandwidth of the channel in use into narrowband subcarriers. As a result, each of the subcarriers experiences flat fading, that is, the channel transfer function is approximately constant throughout the subcarrier's bandwidth. OFDM transmits each data symbol via an individual subcarrier instead of spreading it over the entire channel bandwidth. Hence, each of the subcarriers acts as a one-top channel, which is to say that the channel distortion on the symbols that OFDM transmits via a certain subcarrier is a multiplication with a complex channel coefficient. This allows for a simple and efficient decoder at the receiver, who can easily compensate for the channel distortion.

Due to multi-path propagation effects, each subcarrier typically experiences a different channel distortion. Such frequency selectivity often translates into some subcarriers providing better channel quality than others. Essentially, multi-path reflections add up differently at a receiver for different frequencies, that is, for different subcarriers. Reflections adding up destructively result in a channel coefficient with an amplitude smaller than one. Hence, noise has a stronger impact on that subcarrier, and the transmitter is limited to low order MCSs to avoid decoding errors. The CSI of an OFDM link captures this frequency selectivity since it is the set of complex coefficients that describes the link.

Spatial diversity causes links to be uncorrelated. That is, two links typically experience different distortions on a certain subcarrier, even if they are located close to each other. OFDMA exploits this behavior to achieve better performance. Essentially, OFDMA allocates disjoint sets of subcarriers to different links, and uses them simultaneously. In order to improve performance, it allocates each subcarrier to the link whose channel conditions are best for that subcarrier. More precisely, it chooses the link whose channel coefficient for that subcarrier has the largest amplitude, which results in the least noise impact. Hence, links use on average subcarriers with better channel conditions than without OFDMA. Thus, links can use higher MCSs, which translates into a higher overall throughput. In the following, we first discuss practical issues that affect both OFDM and OFDMA. After that, we present existing single-hop OFDMA schemes which we use as a reference for our multi-hop OFDMA corridor in Chapter 5. Finally, we survey work on multi-hop OFDMA, which is typically limited to simulation and theoretical results.

2.3.1.1 *Practical Issues*

The aforementioned frequency selective behavior of OFDM subcarriers suggests that a transmitter should use individual MCSs on each subcarrier to achieve a good adaptation to the channel conditions. However, this poses practical problems since transmitters need to (a) obtain per-subcarrier CSI via feedback, and (b) encode each subcarrier individually. While (b) only requires a more complex design at transmitter and receiver, (a) results in costly overhead. Hence, schemes such as 802.11g use the same MCS on all subcarriers [47]. This translates into a strong limitation since the worst subcarrier determines the MCS. For instance, a deep fade on a single subcarrier may force the system to use a very low MCS even if all other subcarriers support a much higher MCS. In other words, such an approach wastes channel capacity. Related work extends 802.11g to support per-subcarrier MCS [47]. Alternatively, a transmitter can use power loading, that is, instead of adapting the MCS, it can adapt the energy it transmits on each subcarrier [122]. Per-subcarrier modulation and/or power loading becomes essential for schemes using wide channels, such as 802.11ac, which allows for 80 MHz and 160 MHz channels. Essentially, the wider the channel, the higher the probability that at least one subcarrier experiences poor channel conditions. Similar issues occur in multi-hop scenarios with the additional constraint that subsequent hops should support similar data rates to avoid bottlenecks [172]. This typically requires disseminating CSI over multiple hops, which is particularly challenging in practice. Moreover, since OFDMA systems building on OFDM allow multiple node pairs to share the channel subcarriers, all of these nodes require channel feedback.

A further practical issue of both OFDM and OFDMA systems is synchronization. This is specially challenging for OFDMA since multiple nodes transmit simultaneously [107]. To avoid interference, nodes need to synchronize both in time and frequency. That is, all transmitters must (a) start transmitting at exactly the same time, and (b) share the same clock. Regarding (a), time misalignments result in subcarriers not being orthogonal, which is to say that the subcarrier sets of different nodes interfere with each other. For (b), small clock drifts cause a characteristic symbol rotation in the IQ-plane. However, larger drifts ultimately also cause interference among subcarriers. A large body of work addresses these issues for infrastructure-based networks such as LTE. For instance, theoretical studies derive error rates for different timing [117] and CFO estimation errors [173]. Traditional systems address the former by choosing a Cyclic Prefix (CP) [106] large enough to cancel out any timing errors, and the latter by including pilot symbols that allow the receiver to track the clock drift. Both approaches reduce channel utilization—the CP introduces redundant data, and pilots take up OFDM symbols otherwise available for data. More advanced schemes aim at reducing the impact of such approaches by, e.g., using the preamble [106] to estimate the CFO, or exploiting unused subcarriers [11]. In WMNs, time alignment becomes more challenging since transmissions must synchronize at multiple receivers *simultaneously*. Hence, single-hop schemes which, for example, introduce individual delays at each transmitter to ensure that all signals arrive at the same time at the base station, are not always suitable for multi-hop [52, 116]. Still, this is only a concern for large WMNs whose links are long enough to experience significant delay differences. In contrast, single-hop frequency synchronization techniques can often be translated to WMNs using OFDMA since CFO correction occurs on a per-subcarrier basis.

2.3.1.2 *Single-Hop OFDMA*

OFDMA is mainly used in infrastructure-based scenarios such as cellular networks. Initial work in this area [157] suggests (a) allowing the base station to use different subcarriers to transmit unicast data to individual users according to their channel conditions on each subcarrier, and (b) enabling users to transmit simultaneously on disjoint sets of subcarriers to the base station. Similar schemes are suitable not only for cellular but also for 802.11 networks [146]. Moreover, systems can combine OFDMA with other PHY techniques such as MIMO to obtain further performance improvements [34].

While OFDMA opens the door to a better channel utilization compared to OFDM, it also raises a number of issues. In particular, systems using OFDMA must decide on how to allocate subcarriers, which MCS to use on each subcarrier, and how to distribute transmission power among subcarriers. That is, the system must decide how to allocate its resources. A large body of work deals with these issues, predominantly from a theoretical perspective [3]. The uplink and the downlink result in different allocation constraints since the underlying assumptions differ—on the downlink, the base station transmits on all subcarriers, while on the uplink, multiple nodes share the available subcarriers. In both cases, optimal resource allocation is highly complex. Hence, most approaches split the problem into multiple sub-problems, and often resort to greedy schemes [71]. Still, in some cases, sub-optimal approaches yield virtually identical results as the theoretical optimal solutions [135]. Further, data scheduling also plays a fundamental role along with subcarrier and power allocation. That is, the amount of data the base station needs to deliver to the users in the downlink directly influences the number of subcarriers and OFDMA symbols the base station allocates to each user [27]. Similarly, resource allocation on the uplink is directly related to the amount of data each user wants to transmit to the base station. The base station typically coordinates such uplink scheduling. Hence, users must signal to the base station both their CSI and the amount of data that they need to transmit, which may result in a significant overhead. Alternatively, probabilistic approaches allow for distributed allocation and scheduling. Essentially, users claim resources in a reservation phase, and transmit data in a transmission phase. Collisions only occur in the reservation phase, which is much shorter than the transmission phase. As a result, the impact of collisions is limited [163].

The operation of OFDMA becomes particularly challenging in dense and unplanned scenarios, such as femtocell deployments. In contrast to traditional cells, femtocells may interfere with each other significantly. To address this issue, overlapping femtocell base stations can share the available subcarriers. In other words, this results in a scenario with multiple transmitters and receivers for both the uplink and the downlink. Such a setting is somewhat similar to OFDMA in WMNs—however, the key differences are that (a) femtocell base stations can coordinate via a wired backbone, and (b) communication is still single-hop. Related work on OFDMA for femtocells studies coordination both in a centralized [7] and distributed [166] manner. Similarly to traditional cells, allocation and scheduling schemes can optimize all of the aforementioned parameters, such as subcarrier allocation and power distribution among subcarriers [140].

2.3.1.3 *Multi-Hop OFDMA*

The concept underlying multi-hop OFDMA is related to multi-radio WMNs since both approaches aim at exploiting spatial and frequency diversity. For instance, work on multi-radio WMNs jointly optimizes the end-to-end path selection and the channel at which each link operates. However, existing approaches are limited. First, results are often simulated, and do not consider quick channel fluctuations at the PHY [156]. This is crucial, since such fluctuations strongly limit performance. Specifically, by the time the scheme establishes an end-to-end path, channel conditions typically have changed (c.f. Section 1.1). Second, and most importantly, multi-radio approaches must deal with a number of inherent limitations. Radios can typically operate only on one channel at any point in time. Hence, they can only exchange control information with radios currently operating on the same channel, which results in an inefficient design. In contrast, OFDMA operates only on one channel, that is, nodes do not get cut off from certain portions of the network because of using a specific resource. In other words, while multi-radio WMNs and multi-hop OFDMA are conceptually related, design constraints are very different.

Similarly to the single-hop case, theoretical work on OFDMA for WMNs focuses on resource allocation. We categorize existing approaches according to their assumptions on channel state and traffic knowledge. First, some schemes assume global network knowledge. The design of the resulting algorithms builds on a central network controller which coordinates allocation [70], as well as traffic scheduling [174]. While this allows the network controller to optimize the performance of the network as a whole, it poses significant practical challenges. Basically, providing timely feedback to the controller is often infeasible. A second class of OFDMA WMN schemes proposes *partially* centralized solutions. In this case, the network controller only decides on a subset of the allocation problems, while the nodes locally handle the rest. For instance, the controller decides on subcarrier allocation, which might require only limited feedback (c.f. Section 2.2.2.2), and the nodes handle power allocation, which requires detailed CSI [35]. Further schemes combine such an approach with game theory [82]. Specifically, these schemes model resource allocation as a game because nodes have to find a compromise regarding how many subcarriers they keep, and how many they give up for other nodes to use. Finally, such game theoretic mechanisms are also suitable for our third class of resource allocation schemes, which follows a purely distributed approach [54]. While distributed schemes avoid feedback to a central controller, they often can find suboptimal solutions only.

The above OFDMA WMN schemes provide solutions for resource allocation. However, a WMN can only use them when translated into a *protocol*. Related work on protocols for OFDMA WMNs is limited, and mostly presents simulation results. Practical implementations are inexistent. The functionality of OFDMA WMNs is often realized at the LNK to allow for simultaneous access to the medium [151]. Simulations of resource allocation mechanisms based on, e.g., the SINR of individual subcarriers [149], include all necessary control messages to coordinate nodes. However, in contrast to the aforementioned theoretical approaches, implementations often focus on reducing the resulting overhead. For instance, protocols often aggregate data packets, that is, instead of allocating resources for each individual packet, multiple packets use the same allocation as long as channels remain stable [147]. Further schemes aim at reducing the control overhead itself by assigning resources in a partially probabilistic manner [148, 150]. While the aforementioned protocols enable the benefits of OFDMA for WMNs, their practicability is limited. First, they focus on the LNK only, that is, the multi-hop paths at NET are not involved at all in the operation of OFDMA even though they determine on which links resources are needed. In other words, the aforementioned OFDMA WMN approaches do not incorporate at all the multi-hop component of WMNs. Second, existing work only provides simulation results and often builds on strong assumptions regarding the PHY, such as simplified interference models [148]. Finally, the above protocols force all nodes in a network to use OFDMA even though other schemes such as OFDM may be more beneficial in some scenarios. For instance, CSI feedback may become prohibitive in a highly mobile network area, and thus OFDM may result in better throughput performance than OFDMA. In our work, we build on the concept sketched in Section 1.3 to address all of these issues.

2.3.2 *Interference Alignment*

In information theory, a very recent and promising approach is IA, which allows to achieve an increased Degree-of-Freedom (DOF). This means that, in a system with K interfering transmitter-receiver pairs, each pair gets more than a $1/K$ resource share. Thus, the sum of all shares is *larger than one*. This surprising result was first established by Cadambe and Jafar [20]. The key idea behind IA is to align interfering signals into the same subspace, while desired signals lie in orthogonal subspaces. In other words, two or more aligned interferers affect a certain receiver as if there was only one interferer. Hence, receivers can decode the desired signal by combining "aligned" unknowns in an underdetermined system of equations.

A toy example of IA would work as follows. Assume an interference domain with K transmitter-receiver pairs which exchange data at the same time. Further, all *desired* links—that is, all links connecting pairs—have real channel coefficients while all *interference* links have purely imaginary coefficients. As a result, the intended receiver of a signal gets the signal multiplied by a real value, while all other receivers get it multiplied by an imaginary value. Thus, if all transmitters send real signals, receivers can discard the interference by ignoring the imaginary part of the received signal. Since pairs can only exchange real signals, they only use half the resources, but they use them all the time, that is, each pair gets "half the cake". While this toy example is only valid for carefully selected channels, transmitters can precode signals based on CSI to achieve alignment in random channels. Further, transmitters can align interference in multiple domains—time, frequency, space, and signal level (c.f. Section 2.3.2.2). IA in space enables gains equivalent to our above example but requires multiple antennas. In contrast, the achievable DOF in the time and frequency domains is upper-bounded by $1/2$ per user. More precisely, IA in both time and frequency requires so-called *symbol extensions*, which means that IA is done over multiple time slots or subcarriers. The additional "symbols" provide overflow space for interference that does not align perfectly. The achievable DOF increases with larger extensions but the larger the extension, the higher the required SNR is [127]. Still, for three transmitter-receiver pairs and the smallest possible extension, which comprises three symbols, transmitters can send a total of four packets using only three symbols, which translates into a 33% gain compared to no IA. Transmitters using IA need the channel coefficients of all symbols in the extension to ensure alignment. For time extension, this means knowing *future* CSI, which limits its practicability. In contrast, in the frequency domain, nodes can obtain CSI estimates for multiple subcarriers in parallel. Hence, frequency IA is directly applicable to the vast majority of nowadays wireless networks, which are based on OFDM and thus allow for IA over multiple subcarriers. Related work investigates IA mainly in single-hop scenarios and focuses on spatial IA. In Chapter 6, we present the first practical frequency IA system. Initially we study the single-hop case but then we adapt it to a multi-hop environment. Hence, our work clearly stands apart from previous results in this area.

2.3.2.1 *Practical Issues*

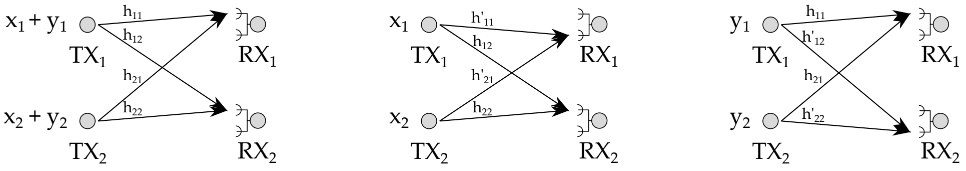
While IA is a promising approach to improve throughput in wireless networks, its practicability is limited [33] due to its high SNR requirements [127] and the need for detailed CSI at the transmitters. In contrast to OFDMA, one-bit feedback (c.f. Section 2.2.2.2) is not enough to compute pre-coding vectors—instead, IA needs the channel coefficients both in amplitude and phase. Still, recent work shows that IA is operable in certain practical scenarios [44, 43, 105, 9]. In particular, IA is feasible in indoor and/or dense scenarios featuring high SNRs. Further, related work shows that signals do not need to align perfectly in order to achieve throughput gains [137]. This "best effort" approach eases the implementation of IA in practice. Finally, a number of techniques allow IA to operate at lower SNRs. For instance, the original IA scheme [20] chooses the all-one vector as a basis to derive the pre-coding vectors at the transmitters. However, transmitters can choose *any* initial vector. Thus, if they choose this vector according to CSI, they can maximize the sum-rate at the receivers [69].

2.3.2.2 *IA Dimensions*

Interference can align in time, space, frequency, and signal level. However, practical time IA is challenging since transmitters need to know future CSI (c.f. Section 2.3.2). Signal level IA [19] also stands apart from other IA variants since it does not achieve alignment in signal space but exploiting symbol lattices. More precisely, two or more interfering signals using a regular Quadrature Amplitude Modulation (QAM) symbol lattice have the same effect than a single interferer if their signal level at the receiver is the same. That is, if the interfering signals are *aligned*. Essentially, the alignment condition is that the minimum distance among the lattice points of the sum of the interference signals is the same than the minimum distance in the lattice of a single interferer. However, achieving a specific signal level at a receiver is often challenging in practice.

In contrast to time and signal level IA, spatial IA is widely studied in related work. Similarly to the above toy scenario (c.f. Section 2.3.2), its gain becomes larger the more users participate in IA. However, close-form solutions to compute the precoding vectors only exist for the three-user interference channel. Implementations on MIMO SDR testbeds for this particular case show that spatial IA is feasible [44]. However, practical issues such as outdated CSI or correlated channels limit its performance. Beyond three users, state-of-the-art IA schemes resort to iterative algorithms, such as alternating minimization [120], to compute the precoding vectors. Again, practical SDR implementations show the feasibility of IA also for this case [103]. Most work on spatial IA focuses on the single-hop case. However, some authors also investigate the multi-hop case. In particular, nodes in a WMN can cooperate to align or cancel interference even if they are not one-hop neighbors [86]. Work in this area identifies network motifs in multi-hop topologies which are beneficial for spatial IA. This directly fits the corridor concept we introduce in Section 1.3 since such motifs could be stages of a corridor. Moreover, the corridor structure would provide both CSI and information about data flows, which is essential for operating spatial IA in practice. Still, in contrast to work on spatial multi-hop IA, in Chapter 6 we investigate multi-hop IA in the frequency domain.

A highly promising variant of spatial IA is blind IA since it does not require CSI feedback at all [45, 59]. The key to enable such operation is twofold. Namely, (a) the receiver must have two antennas among which to *switch*, and (b) the coherence time of the channel must be long enough to allow the transmitters to send packets multiple times without CSI changing significantly. Figure 2.4 shows an example scenario. Each transmitter must have one packet for each receiver. In the first time-slot, transmitters send the sum of their packets, and receivers receive via their first antenna. Next, in the second time-slot, transmitters only send their first packet, and the first receiver switches to its second antenna while the second receiver still uses its first antenna. Finally, transmitters send



First time-slot Second time-slot Third time-slot

Figure 2.4: Blind IA scenario. Packets at the transmitters indicate which signals each transmitter sends in each time-slot. Equation 2.1 shows the received signals.

their second packet in the third time-slot, and both receivers switch to the first and second antenna, respectively. As a result, the first receiver r_1 has the data shown in Equation 2.1. It can cancel out the packets y_1 and y_2 in I using III to obtain IV. Then, it can obtain x_1 and x_2 solving the linear system of II and IV. Similarly, the second receiver can obtain y_1 and y_2. In other words, blind IA delivers four packets using only three time-slots, thus achieving a 33% gain. Practical implementations realize blind IA in real-time [105], and compare it to other state-of-the-art multi-antenna PHYs such as MIMO [9].

$$\begin{aligned}
&\text{I:} && r_1(1) && = h_{11} \cdot (x_1 + y_1) + h_{21} \cdot (x_2 + y_2) \\
&\text{II:} && r_1(2) && = h'_{11} \cdot x_1 + h'_{21} \cdot x_2 \\
&\text{III:} && r_1(3) && = h_{11} \cdot y_1 + h_{21} \cdot y_2 \\
&\text{IV:} && r_1(1) - r_1(3) && = h_{11} \cdot x_1 + h_{21} \cdot x_2
\end{aligned} \tag{2.1}$$

In contrast to spatial IA, work on frequency IA is limited. We discuss its operation in detail in Chapter 6. Improvements building on the original frequency IA design [20] show how to achieve a larger SNR at receivers by orthonormalizing the precoding matrices [138]. This allows frequency IA to become operable in a wider range of situations. Still, practical implementations of frequency IA are inexistent. The only attempt to obtain insights into its performance in practice is based on offline traces of real-world channels [18]. Essentially, authors jointly measure radio channels to three base stations in an urban macrocell scenario. Then, they use those traces as an input to a frequency IA simulation. In Chapter 6, we build a practical frequency IA *system* that allows us to (a) investigate its limitations, and (b) show the effectiveness of our solutions to those limitations.

2.3.2.3 *IA System Improvements*

While the above discussion deals with IA itself, the performance of IA can improve significantly when considering it as part of a whole *system*. For instance, an IA system can divide its users into groups which use orthogonal channels to mitigate the overhead impact resulting from providing timely CSI feedback [119]. Concretely, although IA gains theoretically increase with the number of users, the aforementioned overhead impact grows faster. If the number of users participating in IA becomes too large, the throughput ultimately tends to zero. Hence, by dividing users into smaller groups that take turns for transmission, overhead becomes manageable. Further, in combination with interference cancellation, IA can help a MIMO system to transmit more simultaneous streams than the number of antennas at each node [43]. In particular, APs connected via a wired backbone can exchange packets decoded with IA to bootstrap decoding based on interference cancellation. Finally, related work on IA systems shows that spatial IA can operate using only one antenna at the transmitters, and without needing CSI feedback [2]. In general, if two one-antenna transmitters interfere at a receiver with two antennas, their signals are not aligned. Hence the receiver cannot separate interference from a third desired signal. However, if the receiver can move one of its antennas, in most cases it can find a small displacement which results in the two interfering signals being aligned [2]. As a result, the two-antenna receiver can decode *without* precoding at the transmitters.

2.3.3 *Software-defined Radios*

One of our goals is to validate our mechanisms in practice (c.f. Section 1.2). To this end, we use an SDR platform which gives us full control over the PHY while allowing us to span also the LNK and the NET. As part of our contribution, we design a framework which enables us to carry out *multi-hop* experiments directly from the Matlab workspace. In the following, we survey existing SDR platforms and highlight the need for such a multi-hop framework. The flexibility of SDRs comes at the price of processing delays, since functionality is realized in software instead of application specific hardware. These delays hinder interactive protocols at the LNK and NET layers. However, certain SDR architectures [111, 180] execute code closer to the hardware than others [144, 177], resulting in different delays. Current approaches can coarsely be categorized into two classes, namely FPGA-based and host-based schemes.

First, FPGA-based solutions implement most functionality on FPGAs directly on the SDR. This results in very low processing delays at the price of a much higher implementation effort compared to host-based approaches. Examples include the Wireless Open-Access Research Platform (WARP) and the Airblue SDR. The former achieves real-time performance [178, 179], while the latter focuses on a modular design [111]. Second, host-based solutions implement core functionality on General-Purpose Processors (GPPs). This approach is highly attractive since it allows for rapid prototyping of mechanisms—at the cost of a higher processing delay. The SORA [144] SDR aims at minimizing this delay by storing pre-computed waveforms of time-critical data. The widely used GNU Radio [176] with the USRP [177] SDR incurs higher delays than SORA but in turn features easier extensibility. While GNU Radio requires implementation in C++, WARPLab [181] takes rapid prototyping a step further allowing for implementation in Matlab. This is highly convenient for prototyping complex signal processing schemes but also places WARPLab at the highest end of the scale for processing delays.

Related work studies SDR processing delays extensively for the case of GNU Radio with USRP [123, 131]. It analyzes the feasibility of 802.11 and concludes that the minimal receive latency is already larger than the inter-frame spacing requirements of current LNK protocols. This issue becomes particularly critical for sophisticated LNK layers [171] which are based on, for instance, machine-learning techniques for cognitive radio. Hence, recent work categorizes LNK functionality based on its timing constraints [115]. Essentially, such a categorization splits LNK protocols, implementing critical components close to the hardware while leaving the rest to GPPs. Other approaches such as CalRadio [63] also deal with the LNK but assume a fixed 802.11 PHY, which limits their flexibility compared to SDRs. While the aforementioned research efforts focus on the challenges resulting from SDR processing delays at the LNK, these issues also propagate to the network layer. Existing SDR *network* testbeds [76] often use FPGA-based solutions to achieve real-time

		NET	LNK	PHY
PHY-aware NET	2.1.1 Traditional WMNs	■	□	□
	2.1.2 Advanced routing schemes	■	□	░
Cross-layer approaches	2.2.1 Advanced LNK	□	■	■
	2.2.2 Cross-layer information	□	■	▨
	2.2.3 Network coding	■	□	▨
	2.2.4 Cooperative diversity	▨	□	■
PHY-interactive schemes	2.3.1 OFDMA	□	■	■
	2.3.2 IA	░	□	■
	2.3.3 SDRs	□	□	■
Corridor-based Routing		■	■	■

Table 2.1: Overview on related work. Black cells indicate that the corresponding topic focuses on the marked layer. Gray cells indicate partial relation to a layer. Light gray cells refer to a topic dealing only superficially with a layer. White cells indicate no relation at all.

performance, which raises the bar regarding implementation effort. Alternatively, such approaches resort to standard 802.11 hardware, thus limiting cross-layer implementations to the LNK and NET. Further efforts to incorporate the NET into SDR testbeds use, e.g., a framework [31] which brings together the USRP SDR with the Click Modular Router [75], hence providing a tool that spans up to the NET. However, this framework focuses on the LNK and is still limited by the processing delay of the USRP. While the connection to Click allows for NET functionality, the design of the framework is tailored to one-hop wireless communication. A solution for rapid prototyping of cross-layer multi-hop protocols on SDRs that addresses the impact of long processing times is missing. As one of our contributions in this thesis, we propose an approach to close this gap.

2.4 SUMMARY

In this chapter, we survey related work that aims at improving the throughput performance of WMNs at the PHY, LNK, and NET. Table 2.1 shows an overview of our findings. For each topic, the shading of the cells indicates how much it deals with the three aforementioned layers. We observe that nearly all topics involve to some extent the PHY. In other words, improving how WMNs deal with the PHY is key to achieve better performance. Further, with a few exceptions, topics focus on a single layer. Most of them also deal partially with a second layer but interaction among layers is rather limited. For instance, network coding involves the PHY because it uses PHY hints, but the underlying PHY is still the 802.11 standard. Also, advanced routing schemes use metrics such as ETX to take into account the PHY but operate exclusively at the NET. Hence, we conclude that none of the existing approaches covers all three layers despite adapting the PHY to WMNs has a direct influence on all of them. In this thesis, we aim at filling this gap by designing a system that supports advanced PHYs on all the aforementioned layers. To this end, we propose Corridor-based Routing.

Part II

CORRIDOR-BASED ROUTING

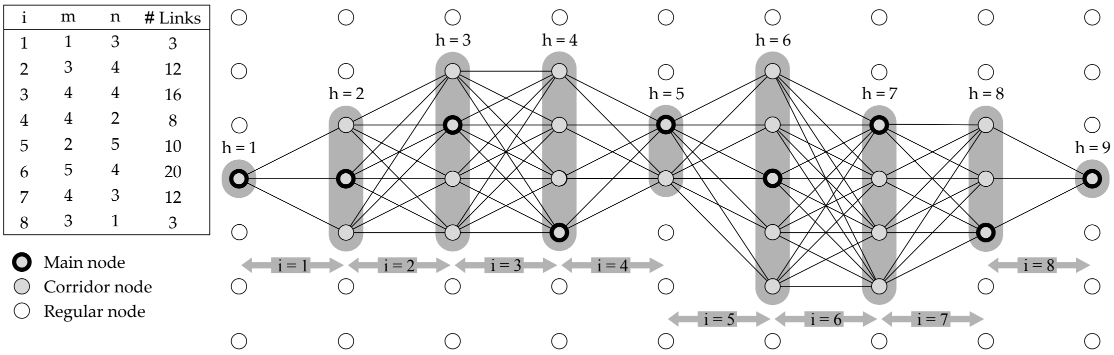
3

ARCHITECTURE

Our survey in Chapter 2 highlights that the key to providing better throughput performance in WMNs is the PHY. State-of-the-art WMNs use cross-layer approaches to adapt to the characteristics of the wireless medium. However, such approaches typically adapt a single layer out of PHY, LNK, or NET and, at most, partially influence the operation of a second layer. A system architecture that enables multiple advanced PHY techniques, and additionally supports the resulting changes at LNK and NET is missing. In this section, we propose such an architecture. In particular, we get into the details of the concept we sketch in Section 1.3, namely, Corridor-based Routing. In Section 3.1 we explain how the characteristics of corridors address the needs of advanced PHYs in WMNs. Next, we present a protocol that enables the operation of a corridor, which includes its construction, maintenance, and coordination. Finally, we conclude this chapter in Section 3.3.

3.1 CORRIDOR CONCEPT

Corridor-based Routing widens the performance bottleneck of WMNs by exploiting spatial diversity. In the following, we introduce a general view on Corridor-based Routing and explore its characteristics. A corridor is a set of successive groups of nodes that forward data along multiple stages—we index the former with h and the latter with i, as shown in Figure 3.1. Each group has a main node h_m that coordinates nodes in its group. The remaining nodes of a group are called "group neighbors". The corridor architecture does not specify which node is selected as a coordinator in each group but a number of designs are possible. For instance, the NET may define it according to a certain criteria, or at random. A stage involves two groups of nodes, namely, a group of m transmitters and a group of n receivers. Hence, the number of links in a stage is $m \cdot n$. Both m and n may have any value as long as the resulting stages are fully connected. That is, a stage may be widening ($m < n$), narrowing ($m > n$), or constant ($m = n$). Figure 3.1 shows an example of a corridor consisting of nine groups of nodes h and thus eight stages i.



i	m	n	# Links
1	1	3	3
2	3	4	12
3	4	4	16
4	4	2	8
5	2	5	10
6	5	4	20
7	4	3	12
8	3	1	3

Figure 3.1: Corridor notation example. We denote groups of forwarding nodes as h, stages as i, number of transmitters in a stage as m, number of receivers in a stage as n, and the coordinating node of a stage as h_m. We represent nodes at regular intervals for clarity.

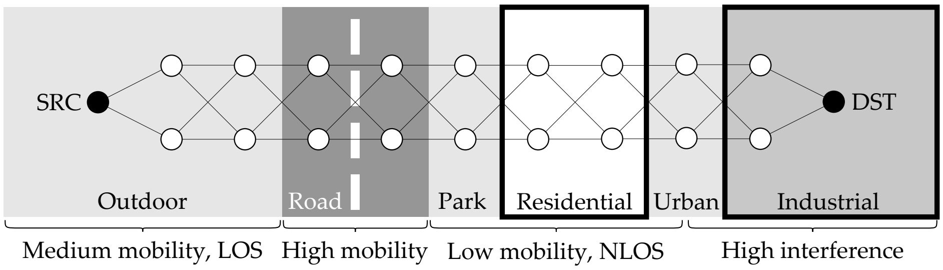


Figure 3.2: Example of the impact of a diverse urban environment on channel conditions along a corridor. LOS and NLOS stand for Line-Of-Sight and Non-Line-Of-Sight, respectively.

The construction of a corridor is equivalent to finding a traditional hop-by-hop path from a source to a destination. In Section 3.2 we present a NET protocol that builds corridors. The source node initiates the construction process and specifies a certain intended corridor width w. The larger w, the more spatial diversity the corridor features, but also the larger the overhead required for operating it. Moreover, the corridor construction protocol may not be able to achieve the intended corridor width w at all corridor groups. For example, the protocol may be forced to narrow the corridor in sparse network areas. Hence, w specifies the maximum corridor width and thus limits the corridor operation overhead, but does not guarantee a certain minimum corridor width. In general, corridors are not constant on all stages but can have any shape, as shown in Figure 3.1.

In contrast to state-of-the-art WMN techniques (c.f. Chapter 2), Corridor-based Routing enables WMNs to use a different PHY at each hop, that is, at each stage. Hence, each stage can locally choose the mechanism which best suits the current channel conditions, thus providing high adaptability and flexibility. For instance, a network deployed in a urban environment may span multiple heterogeneous areas, as in Figure 3.2. Each area has a different impact on channel conditions, that is, *different* PHY mechanisms are best suited in each area. Using corridors, multi-hop transmissions traversing multiple such areas can switch among PHYs at each stage accordingly. A stage only needs to coordinate with its neighboring stages in order to guarantee that the used PHY is compatible to the mechanisms in use at its neighbors. For instance, if the PHY in stage i splits data among the receiving nodes and the PHY in stage $i+1$ assumes that all nodes have the same data, the PHYs are incompatible. Further, a certain PHY may require a minimum number of transmitters, and is thus incompatible to a previous stage with a smaller number of receivers. In other words, the transmitting nodes at each stage choose the most suitable PHY according to (a) the channel state of the stage, (b) the shape of the stage, and (c) the amount of data that each transmitter needs to transmit.

3.1.1 *Corridor Characteristics*

In the following, we explain how a corridor addresses the needs of advanced PHYs, namely, spatial diversity, coordination, network awareness, timeliness, and locality.

3.1.1.1 *Spatial Diversity*

Most state-of-the-art PHYs exploit spatial diversity to achieve a higher performance. In other words, they require using multiple antennas at different locations. While nodes in a WMN provide naturally for such diversity, this has a direct impact on the LNK and the NET. In particular, the LNK must allow for simultaneous transmissions in the same interference domain, and the NET must deliver data to each of the transmitting nodes instead of forwarding it to just one node. Without a corridor, both the NET and the LNK would incur a significant overhead in determining which nodes require which data, and which nodes are allowed to transmit at the same time, respectively. A corridor facilitates both since it "modularizes" the path into stages. Hence, only the nodes within a stage need to transmit simultaneously. Further, each stage has well-defined requirements on which of the nodes in its transmitter group requires which data according to the PHY in use. For instance, if the PHY requires all transmitters to have the same data, the NET can take this into account when forwarding data in the previous stage.

3.1.1.2 *Coordination*

Advanced PHYs require multiple entities to cooperate, and thus need coordination. For instance, in a single-hop scenario using multi-user MIMO, multiple APs need to synchronize accurately [126]. This applies directly to WMNs using such PHYs, too. In particular, WMNs may require coordination at different levels of granularity, ranging from knowing which nodes have received a certain packet to precise synchronization. This results in control overhead at the upper layers. Concretely, the LNK must provide channel access to disseminate such control messages among nodes in range of each other, and the NET must provide means to forward control data over multiple hops, if needed. The latter becomes critical when the PHY requires tight coordination since multi-hop routing may incur significant delays. Corridor-based Routing addresses this issue by using advanced PHYs on fully connected groups of nodes only, that is, on corridor stages.

3.1.1.3 *Network Awareness*

In WMNs, the PHY typically assumes that the upper layers determine which nodes participate in a transmission. More precisely, when forwarding data along a traditional path, each hop translates into a specific transmitter and receiver. In contrast, a PHY involving multiple nodes raises the question as to which nodes shall participate. Potentially, any node in range along the path to the destination could participate. Finding an ad hoc consensus on which nodes shall participate for each individual transmission would result in a large control overhead both at the LNK and at the NET. Specifically, the NET would need to ensure that the nodes actually contribute to transport data closer to the destination, and the LNK would need to keep track of the link quality of any possible transmitter-receiver pair to select an appropriate MCS. The latter is crucial since MCS adaptation often builds on link statistics to choose an appropriate encoding. Corridor-based Routing significantly simplifies the operation of advanced PHYs since its stage structure predefines which nodes act as transmitters and receivers at each hop.

3.1.1.4 *Timeliness*

A large number of advanced PHYs requires timely CSI feedback to adapt to fast channel fluctuations. While this is feasible in single-hop scenarios, it is highly challenging in the multi-hop case. The underlying reason is that channel fluctuations occur at a much smaller timescale than multi-hop forwarding. Hence, by the time CSI feedback has traversed multiple hops, it becomes outdated. To enable multi-hop CSI dissemination, both the LNK and the NET would need to operate following a tight scheduling. As a result, distributed schemes such as CSMA/CA become unsuitable. However, a centralized controller for scheduling is not scalable, either. In other words, such an approach is prohibitive for WMNs. In contrast, Corridor-based Routing *decouples* both aforementioned timescales. Since it only uses a certain PHY within a fully connected stage, CSI feedback is always single-hop. Similarly to infrastructure-based systems, the LNK within such a stage may follow a time-scheduled approach. Further, it does not require a central controller for the entire network since the main node of each transmitter group can schedule transmissions locally. Such single-hop CSI feedback allows for timely adaptation to *short-term* link quality variations at the PHY timescale. Further, the corridor can adapt to *long-term* variations at the NET timescale if, for instance, an external interference permanently causes poor links in a certain network area. To this end, the corridor could switch to a more robust stage mechanism, or even change its shape to avoid the affected area.

3.1.1.5 *Locality*

Finally, PHY techniques are inherently local since transmitters have a limited range. Hence, and as a result of supporting multiple PHYs, a WMN may use simultaneously different PHYs in different network areas (c.f. Figure 3.2). This poses a number of challenges at the LNK and NET. At the LNK, the MCS adaptation mechanism must keep track not only of all possible transmitter-receiver pairs but also for each possible PHY. In particular, the link quality for a certain pair may vary significantly according to the PHY in use. Further, each PHY splits and joins data differently as it forwards data through the network. For instance, while OFDM delivers the same data to all receivers, blind IA as depicted in Figure 2.4 delivers one packet of each transmitter at each receiver. Hence, the NET must allow data to spread among the receivers of each transmission, that is, it needs to ensure that each individual data segment reaches the destination instead of only keeping track of one single packet as in traditional WMNs. The stage structure in Corridor-based Routing allows the LNK and the NET to easily handle the resulting complexity. Essentially, the LNK only needs to keep track of the link quality of each PHY within one stage. Further, data segments are confined to the corridor, and the succession of stages ensures that, at the NET, nodes route each segment towards the destination.

3.1.2 *Corridor Trade-offs*

The above corridor characteristics enable advanced PHYs to operate in WMNs. Basically, Corridor-based Routing follows a divide-and-conquer approach that yields the aforementioned characteristics but also results in a number of trade-offs. Instead of finding the optimal combination of transmitters, receivers, and PHY technique in the whole network for each single transmission, Corridor-based Routing (a) limits the nodes that it considers to corridor nodes which provide progress towards the destination, and (b) divides the corridor into fully connected stages. While this reduces the overhead dramatically, it also may result in sub-optimal solutions. For instance, including an additional node in a stage may yield better performance than when considering only the stage nodes that the corridor construction protocol selects. Further, limiting CSI feedback to a single stage may result in the previous stage delivering data in a disadvantageous manner to the transmitting nodes. That is, a node in a group may receive more data than it can forward in the next stage, resulting in a bottleneck, because the previous stage does not know the quality of the outgoing links of that node. The impact and frequency of such situations depends on each PHY. However, Corridor-based Routing can shift the trade-off balance to counteract such effects. In the following, we present two mechanism blueprints that fit the corridor architecture and enable such a shift.

3.1.2.1 *Cross-Layer Corridor Construction*

Corridor construction protocols aim at building a corridor from a source to a destination with a certain intended corridor width w. In dense network areas, the corridor may find more than w suitable nodes for a certain node group h. While the construction protocol may simply choose w nodes at random for the current group, it may also use cross-layer information to choose the w nodes which are most suitable for a certain PHY that the resulting stage may use later on. However, such a selection is only beneficial when the corresponding PHY can benefit from long-term channel characteristics since the corridor structure itself only adapts at large intervals (c.f. Section 3.1.1.4). In other words, selecting the nodes for group h based on short-lived channel characteristics is unprofitable. Further, exploiting cross-layer information to select nodes during corridor construction comes at the price of a larger construction overhead since the protocol needs to obtain this information for all potential nodes. Hence, this technique shifts the trade-off balance towards a higher overhead but may also result in better performance.

3.1.2.2 *Coarse Corridor-wide Feedback*

A second approach to shift the divide-and-conquer trade-off in Corridor-based Routing is to allow a limited amount of corridor-wide feedback traffic. Essentially, stages within a corridor may exchange limited information on their long-term channel characteristics to avoid the aforementioned bottlenecks. This results in multi-hop feedback which, similarly to the mechanism in Section 3.1.2.1, is unprofitable for short-lived CSI—only long-lived and thus coarse channel state feedback may be beneficial. Again, the price for such feedback is a larger overhead. However, the corridor can control the overhead impact by limiting the number of stages that forward the feedback, similarly to Reference [50]. The more stages forward the feedback, the less likely bottlenecks are, but also the larger the overhead is. The most suitable dissemination range of such feedback, as well as the disseminated information itself, depend on the particular PHY in use. The reason is that different traffic patterns lead to bottlenecks for each PHY.

3.2 CORRIDOR OPERATION

In this section, we introduce the detailed operation of Corridor-based Routing. More precisely, we design a protocol consisting of multiple phases, namely, corridor construction, stage maintenance, profile matching, stage coordination, and data transmission. The design of most phases is different for each individual PHY. Thus, in Section 3.2.1 we give a general description, and leave the specific details for the following chapters, which deal with particular PHYs. However, the corridor construction phase is common for all PHYs since the corridor is the base structure on top of which all of them operate. Hence, in Section 3.2.2 we present the design, implementation, and evaluation of a specific corridor construction protocol. In the following chapters, we build on this protocol.

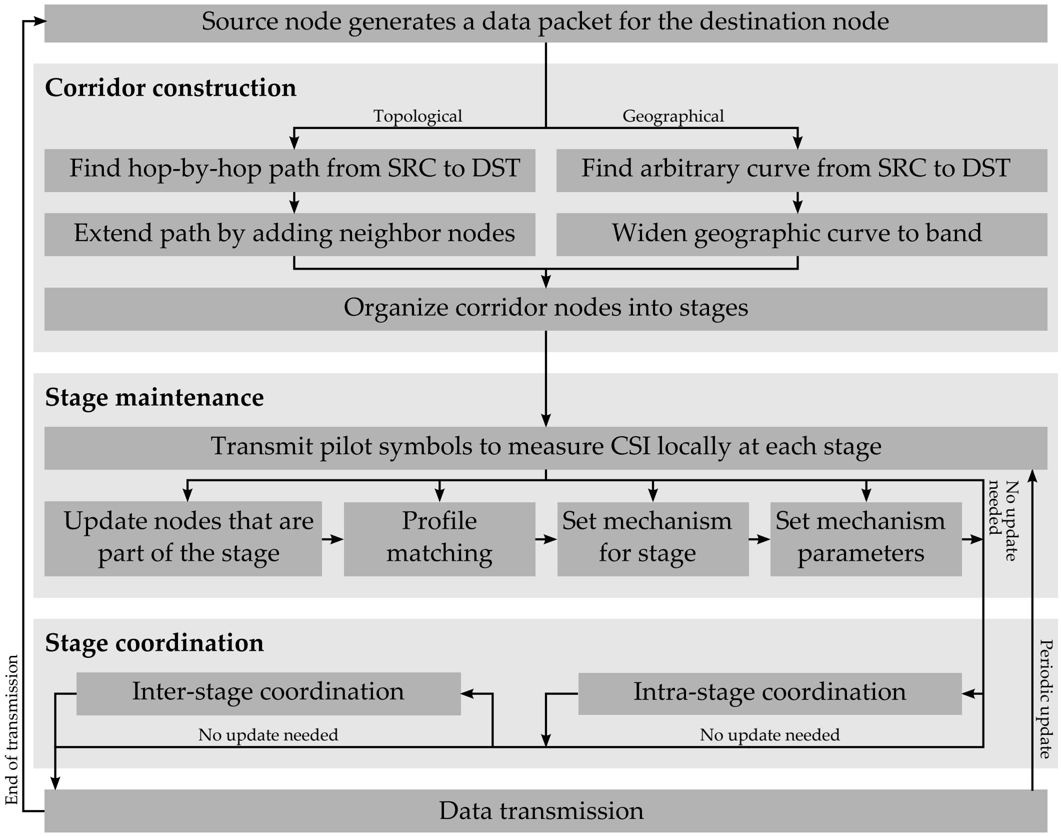

Source node generates a data packet for the destination node
Corridor construction
Topological
Geographical
Find hop-by-hop path from SRC to DST
Find arbitrary curve from SRC to DST
Extend path by adding neighbor nodes
Widen geographic curve to band
Organize corridor nodes into stages
Stage maintenance
Transmit pilot symbols to measure CSI locally at each stage
Update nodes that are part of the stage
Profile matching
Set mechanism for stage
Set mechanism parameters
No update needed
Stage coordination
Inter-stage coordination
Intra-stage coordination
No update needed
No update needed
Periodic update
End of transmission
Data transmission

Figure 3.3: Protocol phases for Corridor-based Routing.

3.2.1 *Protocol Phases*

Figure 3.3 shows an overview of our Corridor-based Routing protocol. In the following, we explain the operation of each individual protocol phase.

3.2.1.1 *Corridor Construction*

In a first step, our protocol builds the corridor itself, similarly to how traditional WMN approaches establish a routing path. The goals are to (a) find a suitable corridor placement in the network, (b) decide which nodes are part of the corridor, and (c) organize nodes in stages. To this end, the corridor construction mechanism can build on well-known topological or geographical routing mechanisms. In the topological case, routing protocols such as AODV [118] or DSR [62] offer mechanisms to find a hop-by-hop path to the destination. After the initial setup, the protocol widens each hop to a group by adding neighboring nodes. The intended corridor width w determines how many nodes the protocol adds to each group. In the geographical case, trajectory-based forwarding [112] enables corridor construction to find a path along an arbitrary curve from source to destination. The protocol then extends the curve to a band whose width is given in geographic units. All nodes falling into that band become part of the corridor. As opposed to the topological case, the width in terms of nodes may vary, since it depends on node density. Further, corridor construction can also use a combination of topological and geographical routing mechanisms, such as the protocol we propose in Section 3.2.2.

For all cases, the routing metric determines the corridor shape. For instance, in the topological case, using the hop count compared to using a metric such as ETX [30] often results in two different hop-by-hop paths. Hence, the resulting corridors have different shapes. Similarly, in geographical routing the corridor shape is related to the underlying geographical curve. This enables the construction protocol to select a metric that avoids areas that provide only substandard service due to long-term inferior channel conditions.

3.2.1.2 *Stage Maintenance*

As a result of corridor construction, nodes are aware of (a) the corridor they are part of, and (b) the stage they are involved into. The nodes of a stage periodically probe each other to locally maintain the control information required for data forwarding. Maintenance ranges from periodical "Hello" messages required for basic PHYs, to full CSI measurements required for advanced PHYs. Figure 3.3 reflects the latter case, which involves transmitting a sequence of known pilot symbols on each link to determine its CSI. After each maintenance measurement, the stage adapts its PHY forwarding mechanism if needed. To this end, the main node of the transmitter group of the stage collects the channel measurements from all other stage nodes, and decides to which extent adaption is needed. Such adaptation may include adjusting the parameters of the current PHY mechanism, switching to a different PHY mechanism, or changing the nodes involved in the stage. This allows for a quick reaction to short-term channel conditions within one stage. However, switching to a different PHY mechanism requires profile matching, which we discuss in Section 3.2.1.3, to ensure that the new PHY is compatible to the PHYs in use in the next and previous stages.

3.2.1.3 *Profile Matching*

To forward data, a stage may use any PHY which fits its $m \times n$ shape. We call a PHY which is adapted to operate on such a corridor stage a *stage mechanism*. Each stage mechanism has particular requirements both regarding the current stage as well as the previous and next stages. Regarding the current stage, a stage mechanism may require, for instance, a certain degree of time and frequency synchronization among transmitters, or a minimum coherence time to be beneficial. The former is crucial for PHYs such as distributed MIMO which require all transmitter nodes to start transmitting at exactly the same time and with a virtually identical clock. The latter is important for PHYs which generate a large amount of overhead—if channels change too often due to short coherence times, the overhead does not pay off. Further, regarding the previous and next stages, a PHY may require a certain amount or type of data at each transmitter node. For example, a PHY such as SourceSync [125], which aims at achieving a higher SNR at the receiver by transmitting a packet simultaneously from multiple nodes, requires all transmitters to have that packet. In other words, all transmitters must have the same data.

The *profile* of a stage mechanism captures its requirements. Essentially, the profile lists how the PHY assumes input data to be available at the transmitters, how it delivers output data to the receivers, and on what type of stages it fits. The latter includes the aforementioned synchronization and coherence time, but also the stage shape and any other characteristic that the PHY may require to be operable. Stages are aware of the profiles of their neighboring stages to ensure that they use a PHY which is compatible to them regarding input and output data. That is, stage maintenance uses the profiles of neighboring stages to ensure that, after adapting the stage mechanism, the profiles still match. Further, stage profiles are also a form of corridor-wide feedback (c.f. Section 3.1.2.2) that may help optimizing the PHY at each stage from an end-to-end perspective.

3.2.1.4 *Stage Coordination*

Stage coordination is the control overhead required to operate a certain PHY. We divide it into intra-stage and inter-stage coordination—the former refers to control messages within a stage, and the latter to control overhead among different stages. For instance, if nodes in a stage share a number of PHY resources, such as in OFDMA, the control messages that nodes exchange to agree on which transmitter-receiver pair uses which resource, are part of intra-stage coordination. In contrast, the overhead required to let the following stage know which information has traveled via which of the aforementioned resources, is part of inter-stage coordination. Further, exchanging PHY profiles (c.f. Section 3.2.1.3) among stages is also part of such coordination. As shown in Figure 3.3, stage coordination occurs after stage maintenance since the result of stage maintenance may require additional control messages, such as announcing that the stage has switched to a different PHY. However, slight adaptations such as adjusting mechanism parameters only, typically do not require additional coordination. Hence, the frequency of stage coordination is significantly lower than of stage adaptation, and thus also the resulting overhead. Moreover, the localized operation of stage mechanisms minimizes costly inter-stage coordination, and thus keeps multi-hop control data low.

3.2.1.5 *Data Transmission*

After a stage completes the above protocol phases, it starts transmitting data. Stages can exploit packet aggregation to minimize the impact of stage maintenance and coordination. That is, the stage does not execute those phases for each individual packet but for groups of packets. However, the coherence time limits the number of aggregated packets in each group. In other words, the duration of the transmission of a group of packets must not exceed the time during which the CSI is approximately constant. The reason is that, whenever the CSI changes, the stage must execute stage maintenance and coordination to ensure that it is still using the most appropriate PHY for the current CSI. The coherence time depends on the mobility of the WMN. More precisely, it is related to the maximum Doppler spread $f_{D_{\max}}$ as shown in Equation 3.1, which in turn depends on the maximum speed $v_{\max}$ at which nodes move, the center frequency f_C, and the speed of light c_l as reflected in Equation 3.2. If nodes are static, $v_{\max} = 0$ and thus $f_{D_{\max}} = 0$, which translates into a theoretically infinite coherence time. For typical indoor environments where nodes move at most at human speed, that is, at about 5 km/h, the coherence time for communication on 2.4 GHz is about 45 ms. Since packet durations are typically much smaller, stages can aggregate a significant number of packets. Nevertheless, this number drops for scenarios with higher node mobility. This results in a trade-off regarding the overhead that a stage mechanism generates and the mobility within the stage.

$$t_{\text{coherence}} \approx \frac{1}{2 \cdot f_{D_{\max}}} \tag{3.1}$$

$$f_{D_{\max}} = \frac{f_C \cdot v_{\max}}{c_l} \tag{3.2}$$

3.2.2 *Corridor Construction Protocol*

Using an *efficient* corridor construction protocol is crucial for Corridor-based Routing. The problem is that building a corridor results in a significantly larger control overhead than finding a traditional hop-by-hop path from a source to a destination. However, in contrast to such traditional paths, we expect it to be much more robust against channel fluctuations. Essentially, while a path may break as soon as one link becomes inoperable due to a change in CSI, a corridor can adapt (a) the parameters of the PHY in use, and (b) the PHY itself to counteract such fluctuations. In other words, while corridors may require a larger initial investment than traditional paths in terms of overhead, they pay off after using them for a number of packet transmissions. More precisely, the throughput gain of the corridor must be large enough to compensate for the effort required to set up and maintain the corridor. This raises a number of questions.

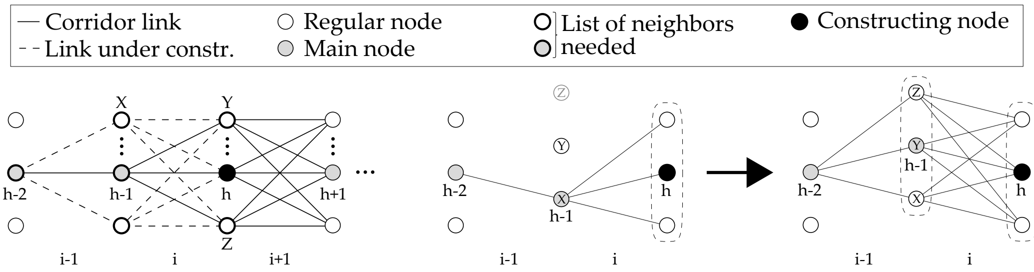
- How many packets are needed to achieve this turning point? Is the pay-off reached before the corridor has to be rebuilt due to, e.g., node mobility?
- How do network characteristics such as density and connectivity impact the construction of corridors, and the resulting performance?
- How large is the overhead of corridor control messages? How many packets are needed, and what information must they carry?

In this section, we design and evaluate a corridor construction protocol to answer the above questions. Basically, we combine the benefits of geographic and topological routing. First, we use location-based approaches to find a traditional hop-by-hop path from source S to destination D *without* flooding the network with route discovery messages. Then, we widen the resulting path to a corridor of a certain intended width w. To guarantee that all nodes in a stage are fully connected, nodes locally exchange neighbor lists and identify suitable neighbors. If the protocol does not find enough suitable neighbors for a certain node group h, it narrows the corridor and widens it again, if possible, at subsequent stages. In the following, we elaborate this design to a practical protocol.

3.2.2.1 *Construction Algorithm*

Our corridor construction algorithm operates on WMNs with randomly distributed nodes. If nodes move, we consider the worst-case scenario, that is, the algorithm needs to rebuild the entire corridor. Alternatively, the algorithm could also locally rebuild the corridor, resulting in less overhead. The input parameters for our algorithm are (a) the intended corridor width w, and (b) SNR_{min}, which is the minimum average SNR that all the links in the corridor must have. We build the corridor in the aforementioned two steps, namely, finding a traditional *single-path*, and widening it to a corridor.

STEP 1: SINGLE-PATH. Our single-path mechanism is inspired by well-known geographic routing approaches [104]. However, we do not use geographic routing to transport data packets but to find a single-path, that is, we use this approach with a different goal than typical geographic routing mechanisms. Source node S—and all subsequent nodes C selected as part of the path in this step—run Algorithm 1 to find the next node. Essentially, each node C forms a set T of neighbors N which are suitable to be the next



(a) Generic stage construction (b) Stage width limited to two (c) Stage width up to three

Figure 3.4: Stage construction (a) and reorganization (b/c). In (b/c), X and Z are not neighbors.

Algorithm 1 Single-path construction algorithm.

```
Require: Location of neighbors N of current node C, location of destination D, nodes P
    already part of the single-path, and $SNR_{min}$
Ensure: Find next node of a hop-by-hop path to D
set $\alpha$ to angle of $\overrightarrow{CD}$ with horizontal axis
for all neighbors N of C do
    set $\beta_N$ to angle of $\overrightarrow{CN}$ with horizontal axis
    if $|\beta_N - \alpha| < \pi$ and $SNR_N > SNR_{min}$
    and $N \notin P$ then
        add N to set T of suitable neighbors
    end if
end for
if $T \neq \emptyset$ then
    return neighbor $N \in T$ with minimum $|\beta_N - \alpha|$
else
    use existing mechanisms to escape local minima
end if
```

node of the path. In particular, T contains all neighbors who (a) are not part of the path already, (b) have a link to the current node C with an SNR larger than SNR_{min}, and (c) are located in the direction of the destination D. A neighbor N lies in the direction of D if the angle between $\overrightarrow{CN}$ and $\overrightarrow{CD}$ is less than 180°. Finally, node C chooses the neighbor in T which lies closest to the direction of D. If node C does not find any suitable node, that is, $T = \emptyset$, there is a local minima. To escape it, the algorithm can use planar graph routing [104]. However, we do not include this in our prototype implementation in Section 3.2.2.3.

STEP 2: WIDENING PATHS. Next, we define an algorithm to widen the single-path to a corridor. This process starts at the destination once Algorithm 1 finds the last hop of the single-path. We build one stage at a time towards the source. For each stage, the algorithm must ensure that all nodes of the two subsequent groups forming a stage are one-hop neighbors, and that the number of nodes in each group is equal to w, if possible. Algorithm 2 uses a topological approach to meet these requirements. The construction of a generic stage is shown in Figure 3.4a. Our algorithm involves three steps: (a) neighbor list collection, (b) neighbor matching, and (c) node selection. Node h_m coordinates these three steps for the construction of stage i, which connects groups h and $h-1$. In step (a), h_m collects the list of neighbors of the two next nodes along the single-path, $(h-1)_m$

Algorithm 2 Stage construction according to Figure 3.4a.

```
Require: List of neighbors of all nodes potentially part of the current stage $i$ being built.
Ensure: Find suitable nodes for group $h-1$ in order to build stage $i$.
 1: node $h_m$ requests list of neighbors of
      1. Group neighbors of $h_m$
      2. Group neighbors of $(h-1)_m$ (required to verify link symmetry)
      3. Main node of $h-1$
      4. Main node of $h-2$ via $(h-1)_m$
 2: filter out links in lists with SNR $<$ $SNR_{min}$
 3: find coincidences $V$ in all filtered lists
 4: if $|V| = w-1$ then
 5:   select all nodes $\in V$ as neighbors for group $h-1$
 6: else if $|V| < w-1$ then
 7:   if $\exists u \in V \,/\, u$ is neighbor $\forall V \wedge u$ has $w-1-|V|$ list coincidences $V'$ with $h$, $h-2$
      then
 8:     set $u$ as main node of group $h-1$
 9:     set nodes $\in V \cup V'$ as neighbors for group $h-1$
10:   else
11:     narrow corridor at group $h-1$
12:   end if
13: else
14:   select $w-1$ nodes $\in V$ with best average SNR
15: end if
```

and $(h-2)_m$. Additionally, h_m requires the two-hop neighbors of $(h-1)_m$ and itself to verify that links are symmetric. For instance, in Figure 3.4a node h_m needs the list of neighbors of X to verify that X can both send to and receive from Y, Z, h_m, $(h-1)_m$, and $(h-2)_m$. In step (b), h_m removes all links with SNR $<$ SNR_{min} from the lists, and then finds common nodes in all lists. The latter ensures that the stage is fully connected. The number of common nodes $|V|$ leads to three cases in step (c), which we discuss next.

First, if $|V| = w-1$, we have exactly the number of nodes needed to form a stage of width w. Hence, Algorithm 2 selects all nodes in V as neighbors for group $h-1$. Second, if $|V| < w-1$, we lack one or more nodes to build a stage of width w. However, changing the main node of group $h-1$ may still allow the algorithm to find enough group neighbors. For instance, in Figure 3.4b, node X is the main node of $h-1$ and only has node Y as a suitable group neighbor. Node Z, which is a neighbor of Y and of all nodes in group h, is not a neighbor of X. Since the main node must be a one-hop neighbor of all stage nodes to enable coordination, Figure 3.4b only allows for a corridor of width two. Hence, in this case the algorithm examines whether any node in V allows for the desired stage width if set as the main node of $h-1$. In our example, this is the case for node Y, which is thus set as the main node in Figure 3.4c. Such stage reorganization is also possible for wider corridors—the general condition making this possible is given in line 7 of Algorithm 2. If changing the main node does not allow for the desired width, we narrow the corridor at group h. Third, if $|V| > w-1$, we have found more nodes than needed to achieve w. In this case, the algorithm chooses the $w-1$ nodes with best average SNR. However, this also allows for cross-layer corridor construction (c.f. Section 3.1.2.1).

Packet type	**Contents and function**
Beacon	Node ID and location. Allows nodes to discover their neighborhood.
Single-path	Message for single-path construction. Includes destination ID, list of nodes which are already part of the single-path, and SNR_{min}.
List Request	Used by h_m to request the neighbor list of a node.
List Reply	Answer to "List Request". Includes the ID and the average SNR of each neighbor of the node sending it.
Stage ACK	Notifies group neighbors that the main node building the current stage has chosen them as part of the stage.
Corridor ACK	Final confirmation sent by the main node of each stage after the corridor is complete to signal that data transmission can begin.
Optimization	Allows for cross-layer corridor construction. Includes PHY state to enable h_m to select the most suitable nodes for a group if $\lvert V \rvert > w - 1$

Table 3.1: Control packets for corridor construction

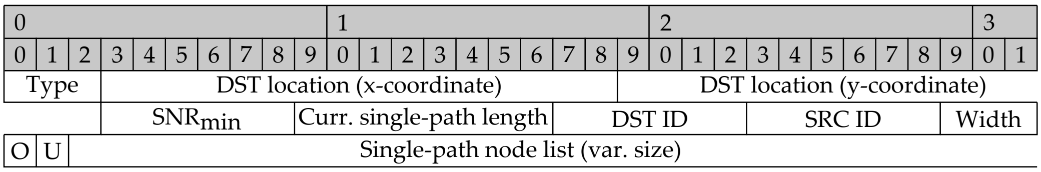
3.2.2.2 *Protocol Design*

To put our approach into practice, we define a protocol that implements the above algorithms, and specify control messages to coordinate nodes. Table 3.1 gives an overview. In the following, we explain the details of some selected control messages as an example.

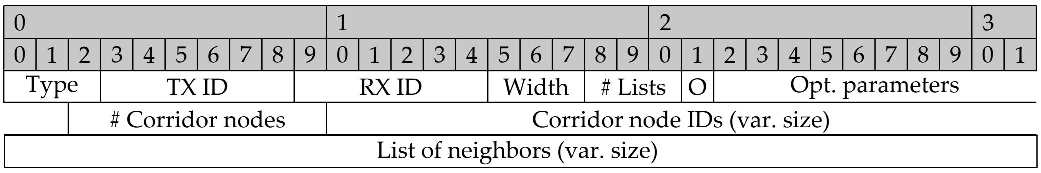
SINGLE-PATH MESSAGE. For single-path construction, each node on the path forwards the single-path control message to the next node which it chooses based on Algorithm 1. Figure 3.5a depicts the details of this message. It contains the location of the destination and the corridor parameters required at the destination to start the corridor construction once the single-path is finished. For instance, this includes the "O" field, which indicates whether the corridor construction protocol shall use cross-layer information to select nodes if $\lvert V \rvert > w - 1$. Each node forwarding the packet adds its identifier to the list of nodes at the end of the message. This prevents nodes from selecting a neighbor which is already part of the path as a next hop. For our prototype in Section 3.2.2.3, we encode the node identifier with six bits. Also, each node increases by one the path length field.

LIST REPLY MESSAGE. Figure 3.5b shows the details of the "List Reply" message, which carries the lists of neighbors needed at the main nodes to construct each individual stage. Similarly to the single-path message, it includes a list of all nodes which are already part of the corridor to prevent the algorithm from adding a node twice to the corridor. Since both the list of neighbors and the list of nodes are of variable size, the message includes appropriate length fields, such as "# Lists". A "List Reply" message can aggregate answers from multiple nodes in case they are relayed—for instance, in Figure 3.4a, $(h-1)_m$ relays the list of neighbors of $(h-2)_m$ and of its own neighbors. Further, the message includes the identifiers of the nodes involved in the list reply, as well as certain corridor parameters needed for correctly processing the reply.

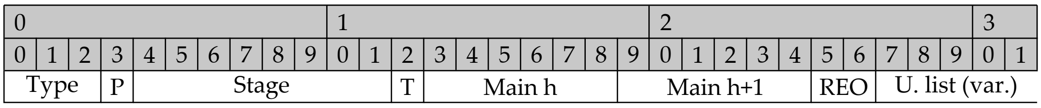
STAGE ACK MESSAGE. After running Algorithm 2, the main node building the current stage sends "Stage ACK" messages (Figure 3.5c) to inform all potential stage neighbors



(a) Control message structure for single-path construction.



(b) Control message structure for “List Reply”.



(c) Control message structure for stage acknowledgments.

Figure 3.5: Message structure for selected control messages.

whether they are part of the stage. The binary field “P” conveys this information. Besides, the “T” field is set to one if the node receiving the acknowledgment shall be the main node. The message also includes the group the receiving node is part of, as well as the identifier of the main node of its group, and of the next group. As a result, each node is aware of its position and function within the corridor under construction.

3.2.2.3 *Experimental Design*

We evaluate the overhead of our construction protocol using both simulations on random topologies, and practical experiments on an SDR testbed. The former allow us to evaluate scenarios our testbed cannot recreate, while the latter give us practical insights.

GOAL AND METRICS. Our evaluation aims at (a) verifying that our construction mechanism is practicable regarding overhead, and (b) measuring the quality of our corridors. The corridor quality is directly related to the performance that advanced PHY techniques achieve when using it. Hence, we use this as a metric. More precisely, we use a stage mechanism based on OFDMA (c.f. Chapter 5) on our corridors to quantify their throughput performance. Related theoretical work [77, 78] shows that, in similar scenarios, OFDMA yields throughput gains between 20% and 50% compared to traditional hop-by-hop forwarding based on OFDM. We use those values as a reference, that is, we investigate whether our corridors allow OFDMA to approach this theoretical limit. Hence, our first metrics is throughput including overhead. To measure it, we consider the data that the network layer delivers at the destination, and the overall time needed for the end-to-end transmission. Our second metric is the number of corridor nodes that our construction protocol finds. That is, we measure how often our protocol needs to narrow the corridor for a certain intended corridor width w. Finally, our third metric is the time required to build the corridor. We obtain this time by counting the PHY symbols needed for signaling.

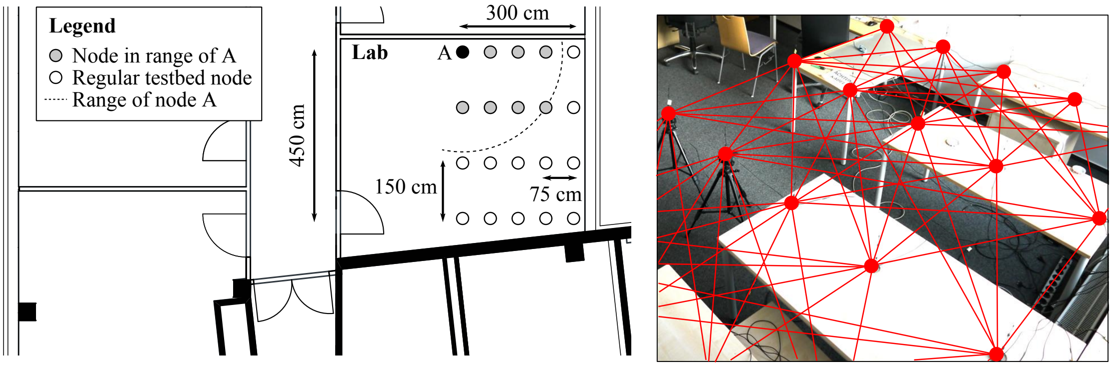
Name	Type	Area and Nodes	Range	Characteristic
RAND	Simulation	$50 \times 50\ m^2$, 50 nodes	12 m	Random location
UD-L	Simulation	$50 \times 50\ m^2$, 50 nodes	15 m	High connectivity
UD-S	Simulation	$21 \times 21\ m^2$, 49 nodes	6 m	Strong links
TBED	Practical experiments	$4.5 \times 3\ m^2$, 20 nodes	2.7 m	SDR testbed

Table 3.2: Simulation and practical scenarios.

MEASURING PROCESS. All our experiments follow the same pattern. We choose a random source-destination pair, which transmit multiple packets both on a traditional path using OFDM, and on an OFDMA corridor. At the beginning of the transmission, we build either a single-path or a corridor, respectively. This construction process is only needed for the *first* packet, since subsequent packets can use the same routing information as long as nodes do not move. That is, only the first packet is affected by the construction overhead. For that packet, we expect OFDMA to perform worse than OFDM, since the effort required to build a corridor is significantly larger than the one required for a single-path. However, once the corridor is available, OFDMA should perform better in terms of throughput (c.f. Chapter 5). Hence, after a certain number of packets, the initial penalty of OFDMA due to the corridor construction should pay off. In our experiments, we determine the number of packets needed to reach this turning point. At the LNK, we use a basic adaptive MCS. For each OFDM link or OFDMA stage, the MCS algorithm improves the modulation and coding after each successful transmission, until an error occurs. After an error, the MCS is not increased any further. Thus, the throughput initially increases with each packet, and later stabilizes when the link or stage reaches the maximum MCS it supports. Since OFDMA can combine beneficial subcarriers (c.f. Chapter 2), we expect it to reach a higher MCS than OFDM, which directly translates into throughput gain. For both OFDM and OFDMA, we use the same MCS on all subcarriers.

ASSUMPTIONS. We consider that nodes move randomly at human speeds. Moreover, we assume a number of worst-case conditions. First, we rebuild the entire corridor when a node moves, that is, we do not allow the corridor to rebuild stages locally. Second, we use the aforementioned basic MCS algorithm. A more advanced approach could use feedback to estimate the maximum MCS and thus further improve corridor performance. Finally, we do not use cross-layer information (c.f. Section 3.1.2.1), which would again improve the performance of OFDMA.

SCENARIOS. We test our corridor construction mechanism in four different settings, namely, three simulated scenarios and one SDR testbed scenario. Table 3.2 gives an overview. The simulated scenarios allow us to investigate the performance of our system for large networks and diverse situations. The first one is a WMN with 50 nodes deployed randomly on an area of 250 m^2, named as RAND. We assume nodes are in each other's range if they are less that 12 meters far apart. At the PHY, we assume Rayleigh channels, a path loss exponent of $\alpha = 2$, and Additive White Gaussian Noise (AWGN) with 40 dB average SNR at a distance of one meter to a transmitter. Due to the random location of nodes, the network topology may have "holes". However, indoor networks with a large number of connected devices may be very dense. Thus, our second scenario considers



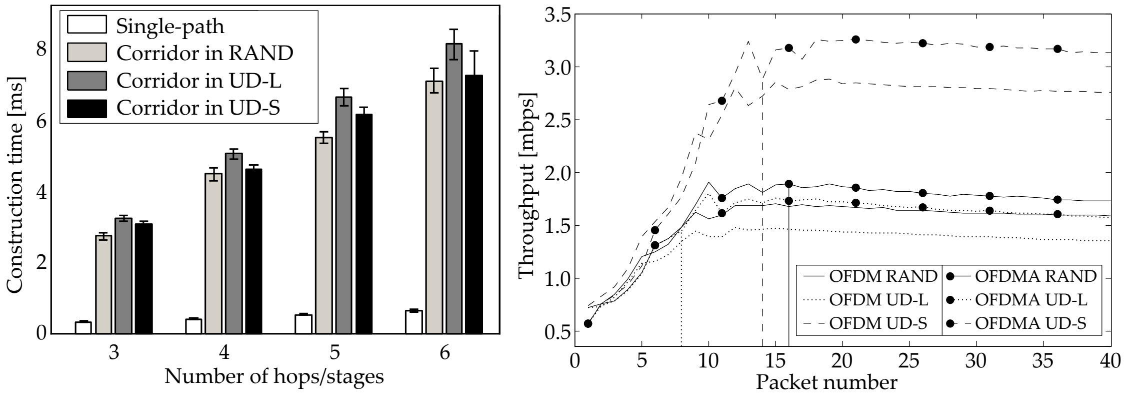
(a) Lab setup diagram.

(b) Picture of lab setup. Lines indicate links.

Figure 3.6: Practical setup for scenario TBED using SDRs.

a similarly large area with a uniform density, named as UD-L. While nodes are still placed randomly, we ensure that there is at least one node every ten meters. As a result, this scenario features high connectivity. Since our corridor construction scheme takes into account the average SNR of links, our third scenario considers stronger links than RAND and UD-L. To this end, we place about 50 nodes with uniform density in a smaller area. We name this scenario UD-S. Moreover, we lower the range, achieving a connectivity slightly lower than UD-L. Finally, our last scenario TBED is an SDR testbed which allows us to test our system in practice. We place 20 nodes at fixed locations in our lab as depicted in Figure 3.6a. Since we cannot change the topology randomly for each experiment run, we choose a regular pattern to better understand our results. We adjust the transmit power of nodes to achieve typical indoor SNRs, and discard packets on links which exceed 2.7 meters. As an example, Figure 3.6a shows which nodes are one-hop neighbors of node A for that range. This setup allows corridors of width four despite the small testbed size. Our testbed consists of six WARPv3 boards with four radio interfaces each. To allow for a network with 20 nodes, we use each interface as if it were an individual node. Still, we treat the data of each radio independently (c.f. Chapter 7).

IMPLEMENTATION. We implement our corridor construction mechanism on WARP Drive, which is our framework for prototyping WMN mechanisms on the WARP [180] SDR. We refer the reader to Chapter 7 for more details on this framework but sketch its most important characteristics in the following. WARP Drive builds on the WARPLab Reference Design that allows us to carry out over-the-air experiments directly from the Matlab workspace. We process signals in Matlab, send them via Ethernet to the WARP boards, transmit them over the wireless medium, and transfer them back to Matlab. While this results in non-real-time operation, we do not process data offline, but online and interactively. That is, over-the-air transmissions are not scheduled in advance but occur dynamically as a result of protocol actions. Moreover, WARP Drive allows us to switch seamlessly from simulations to practical experiments, that is, we use the same code for both cases. Our implementation includes (a) the corridor construction protocol, and (b) a stage mechanism based on OFDMA. For details on the latter, we refer the reader to Chapter 5. At the PHY, we use parameters equivalent to 802.11g, that is, 20 MHz channels, 48 usable subcarriers, 312.5 kHz subcarrier spacing, and 12.5% CP. Moreover, we use



(a) Time required for constructing corridors in each of the simulation scenarios.

(b) Turning point. The vertical lines indicate from when on the corridor pays off the construction overhead.

Figure 3.7: Corridor construction overhead analysis.

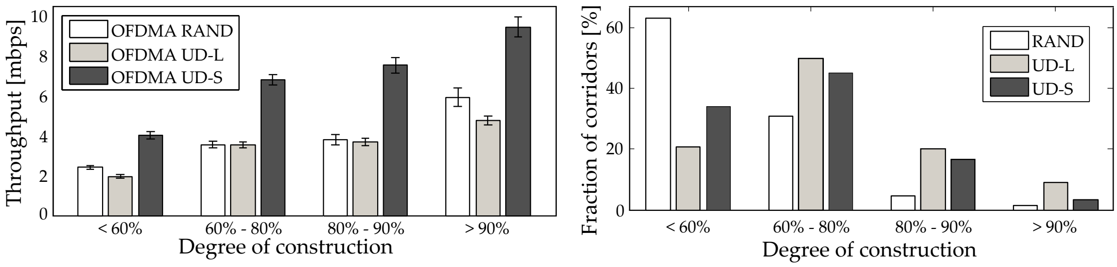
802.11 short and long preambles—the former trains the Automatic Gain Control (AGC) and the latter allows for frame detection using correlation at the receiver.

3.2.2.4 *Simulation Results*

In the following, we present our simulation results in the three scenarios shown in Table 3.2. Concretely, we investigate (a) the construction time, (b) the turning point from which building the corridor pays off, (c) the number of nodes in the corridors we build, and (d) the impact of the intended corridor width w as well as of SNR_{min}.

CONSTRUCTION TIME. Figure 3.7a shows that corridor construction takes significantly longer than finding a single-path. The construction time is roughly proportional to the length of the corridor/single-path, since building each additional stage/hop requires control messages. For the single-path, we observe that the scenario barely influences the construction time, since the algorithm only needs to choose one suitable next node. Corridor construction in UD-L is slightly slower than in RAND and UD-S. For RAND, the reason is most probably the sparsity of the network in some areas, which results in less control messages, as nodes need to exchange less lists of neighbors. For UD-S the reason is twofold. First, the connectivity of UD-S is slightly lower than for UD-L, which results in the same effect than for RAND. Second, its strong links allow to transmit control data at high rates, and thus the time to build the corridor is less than for UD-L. Finally, the intended corridor width w and the SNR_{min} value barely impact the construction time, since these parameters only determine which nodes are chosen *after* requesting the lists of neighbors. Thus, the control overhead is mostly independent of w and SNR_{min}.

TURNING POINT. Next, we analyze after how many packets the construction overhead compensates. Figure 3.7b depicts the average end-to-end throughput of 40 subsequent packets sent on a single-path (OFDM) and a corridor (OFDMA). For all cases, the curve initially raises due to the adaptive MCS. That is, the MCS increases with each additional packet and, in turn, the throughput increases, too. In UD-S, the curves stabilize at a higher value than in RAND and UD-L since links are stronger and support a higher MCS.



(a) Throughput for different degrees of construction.

(b) DoC achieved for corridors of width three.

Figure 3.8: DoC analysis.

As expected, OFDMA performs initially worse than OFDM in all scenarios due to the cost of corridor building for the first packet. Still, because OFDMA combines the best links at each stage, the adaptive MCS scheme can reach a higher rate, and thus OFDMA outperforms OFDM after a few packets. The vertical lines indicate the turning point, that is, the time at which OFDMA exceeds OFDM regarding *cumulative* throughput. This occurs later than the intersection of the OFDM and OFDMA curves since OFDMA needs to compensate for its initial overhead before paying off. RAND and UD-S reach the turning point at packets 14 and 16, respectively, while UD-L compensates for the overhead already at packet 8. Although the turning points of RAND and UD-S are similar, the reasons behind are different. For RAND, the uneven density of the network results in narrow corridors more often than in UD-S and UD-L. Hence, the corridors provide less spatial diversity. From our results in Section 5.1.2, we deduce that this limits the gain of OFDMA, which translates into OFDMA needing more packets to pay off. In contrast, UD-S takes longer than UD-L to reach the turning point due to the adaptive MCS—since UD-S supports a higher MCS, the LNK needs more packets to reach it, hence delaying the turning point. Finally, UD-L reaches the turning point first because (a) the LNK finds the maximum MCS quickly, and (b) UD-L features higher connectivity than RAND, thus avoiding narrow corridors.

DEGREE OF CONSTRUCTION. Our previous experiment indicates that the performance of a corridor is closely related to the construction algorithm being able to find enough group neighbors for each group. We refer to this as Degree of Construction (DoC), which is the fraction of nodes that a corridor has compared to the number of nodes it should have given its width. For instance, the ideal corridor in Figure 1.2 has 20 nodes. If the algorithm needs to narrow some groups resulting in a corridor of 15 nodes, the DoC is 75%. Figure 3.8a shows the throughput of OFDMA for different DoC ranges. As expected, it improves for larger DoCs, since diversity increases. UD-S achieves the highest throughput due its strong links. Figure 3.8b depicts the distribution of the DoC for the case $w = 3$. In RAND, most corridors only achieve a DoC of 60% or less due to the low connectivity of the network. In contrast, in both UD-L and UD-S, our algorithm achieves in most cases DoCs up to 80%. UD-S has a lower DoC than UD-L due to its slightly lower connectivity.

WIDTH AND SNR$_{\text{MIN}}$. Some effects regarding corridor width and SNR$_{\text{min}}$ are contrary to each other, which complicates drawing general conclusions. For instance, a larger width results in more diversity as more nodes participate in each stage, and thus OFDMA achieves better throughput performance. However, we share resources in a "fair" manner among the nodes of each stage, that is, we assign the same number of subcarriers to each node. The wider a stage, the higher is the probability of a node having poor channel conditions, thus limiting the performance of the stage. Regarding SNR_{min}, we encounter a similar trade-off. A large SNR_{min} prevents our algorithm from including bad links in a stage, thus improving the performance of OFDMA. However, this also affects single-path construction, resulting in better OFDM performance and thus lowering gains. Overall, this limits our gains to about 15%. However, this is unrelated to the corridor construction scheme but is an effect of the particular OFDMA stage mechanism we use in our simulations. We address this issue both in Chapter 5 and in the following practical experiments, since the aforementioned effects become critical on the SDR testbed.

3.2.2.5 *Practical Results*

For our practical experiments, we adapt the design of our OFDMA stage mechanism to address the issues discussed in Section 3.2.2.4. Basically, we allow (a) each node to get a different amount of resources according to its CSI, and (b) per-subcarrier MCS adaptation. This added flexibility enables our OFDMA stage mechanism to perform better on links with moderate frequency selectivity. For further details on this alternative design and the underlying real-world effects, we refer the reader to Chapter 5. Next, we present our practical results using such a design in conjunction with our construction protocol.

Figure 3.9 depicts the throughput in our testbed of both OFDM and OFDMA for different widths and values of SNR_{min}. We achieve throughput gains mostly between 20% and 50%, which matches the results of related work. According to Section 3.2.2.3, we conclude that our construction algorithm builds good corridors. The throughput of OFDM varies in each of the three measurements although it does not depend on the corridor width. This is likely because channels in our lab fluctuate slowly during the course of the day, leading to different results for the same SNR_{min}. Still, channels are comparable for each corridor width. We observe that the throughput of OFDM increases with SNR_{min} since this parameter also affects the nodes chosen for single-paths. In contrast, OFDMA achieves a similar throughput for all SNR_{min} values. This shows the high degree of flexibility of OFDMA—weak links in a stage barely impact its performance. Regarding corridor width, we observe that throughput decreases slightly with larger stages. This is expected, since OFDMA exchanges CSI among all nodes of a stage as part of its stage maintenance overhead. Hence, the larger a stage, the larger the overhead. For corridors of width four, Figure 3.9 shows that this overhead exceeds the gains of OFDMA, resulting in a negative throughput gain. That is, this negative gain is not a result of the overhead for corridor construction, which is independent of the corridor width (c.f. Section 3.2.2.4). Moreover, Figure 3.9 illustrates that the diversity available in a corridor of width two already allows for large gains. That is, the corridor does not need to be wide to provide large benefits.

3.2.2.6 *Discussion*

At the beginning of this section, we identified a number of open questions regarding the overhead impact of corridor construction, and our results provide some answers. First, the overhead itself depends on the length of the corridor, and increases faster per stage than a single-path does per hop. For example, for a corridor of length five, the overhead is about ten times larger than that of a single-path. It consists mostly of control packets conveying neighbor lists. This leads directly to the second issue of compensating

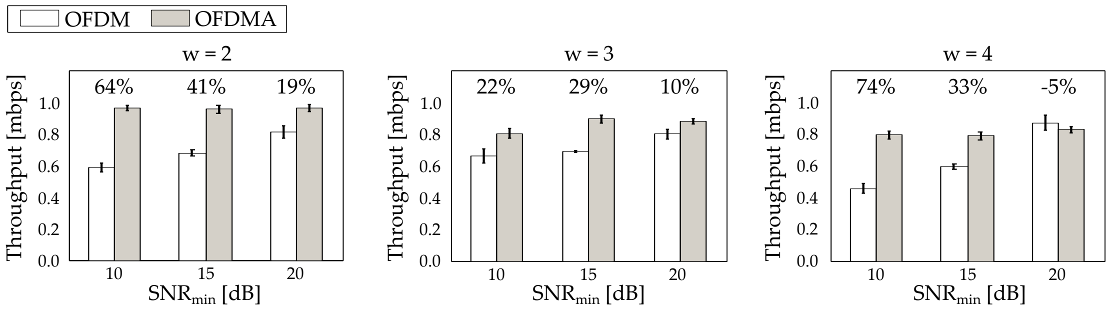


Figure 3.9: Practical end-to-end throughput achieved on corridors of intended widths two, three and four built using our construction algorithm. Percentages refer to gains.

for this overhead—that is, reaching the turning point—before the corridor fails. In our experiments, the slowest scenario needed an average of 16 packets to reach this point. Assuming a packet size of 1500 bytes, a worst-case link throughput of 1 mbps, and a corridor length of five stages, 16 packets require 0.96 seconds to reach the destination. Since we consider a range of six meters for our smallest scenario, on average a node needs to move three meters to leave a corridor. Hence, this requires nodes to move at about $3\,\mathrm{m}/0.96\,\mathrm{s} = 11.25\,\mathrm{km/h}$ for the corridor overhead not to compensate, which is significantly larger than the average human speed. This intuitive estimation becomes more beneficial assuming more realistic throughputs, and thus opens doors to using our construction scheme also in outdoor scenarios featuring higher mobility. However, a detailed measurement of the impact of node mobility is out of scope of our evaluation. Third, we also study the influence of topology characteristics. As expected, dense and connected networks result in better corridors. The less nodes are available, the more often our algorithm has to narrow a corridor. However, we observe that corridors do not need to be wide to provide large gains, that is, also sparse networks can benefit from corridors. Finally, for OFDMA stage mechanisms, we conclude that corridor widths up to three nodes and small SNR_{min} values are best. However, SNR_{min} should be above a minimum to avoid links with bad channel conditions on all subcarriers (c.f. Chapter 5).

3.3 SUMMARY

In this chapter, we first explain how the characteristics of Corridor-based Routing address the requirements of advanced PHYs in WMNs, including their impact on the LNK and the NET. More precisely, we conclude that corridors provide spatial diversity, simplify the coordination among nodes, take charge of choosing which nodes shall cooperate at the PHY, enable timely CSI feedback, and allow the WMN to use multiple PHYs simultaneously in different network areas. To this end, Corridor-based Routing forwards data along *groups* of nodes instead of individual nodes—intuitively, it widens a traditional hop-by-hop path in order to span multiple nodes at each hop. Further, we highlight that the key to provide the above characteristics is incurring a number of trade-offs. Essentially, instead of finding the optimal network-wide solution regarding which nodes shall cooperate using which PHY, corridors find the best solution within the aforementioned groups of nodes. While this results in a sub-optimal solution, we still expect to achieve throughput gains due to the vast diversity in the network. In other words, even if we consider a subset of all possible solutions only, the probability that the subset contains a beneficial solution is large. In Chapters 5 and 6 we show this for the case of OFDMA and IA, respectively.

Additionally, we introduce the operation of Corridor-based Routing. In particular, our design considers five protocol phases. First, corridor construction builds the end-to-end corridor structure. Second, stage maintenance determines the PHY which is most suitable for each corridor stage. Third, profile matching ensures that the PHY in use at each stage is compatible to the PHYs in use at its neighbors. For instance, if a certain stage expects all of its transmitters to have the same data, the previous stage must use a broadcast PHY. Fourth, stage coordination exchanges PHY control information both among nodes of a certain stage, and among multiple stages, if needed. Finally, the data transmission phase exploits long coherence times to forward data along the corridor with a low overhead impact. That is, it aggregates multiple packets and only exchanges control information when channel conditions change. The design of most protocol phases depends on the PHY in use. However, corridor construction is common to all PHYs. Hence, we also present a specific corridor construction protocol to evaluate its overhead in practice. Essentially, our mechanism widens a geographical path to a corridor using topological information. We implement it both in simulation and on an SDR testbed. We observe that the overhead strongly depends on the characteristics of the network topology, such as its density. Moreover, we conclude that the overhead of building a corridor typically pays off as long as (a) the PHY in use provides better throughput than a traditional OFDM scheme, and (b) the WMN can use the corridor for multiple packets.

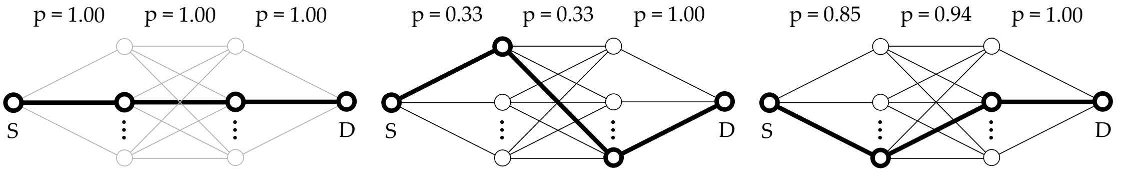
4

SINGLE LINK STAGE MECHANISM

In this and the following chapters, we present three stage mechanisms for Corridor-based Routing. In particular, this chapter deals with a Single Link (SL) stage mechanism, which is virtually equivalent to traditional hop-by-hop forwarding in WMNs. We later often use it as a baseline mechanism to compute the performance gains that our OFDMA and IA stage mechanisms achieve. Essentially, SL just transmits data on one link of a stage at a time using OFDM, that is, it involves only one transmitter and one receiver. An SL stage can operate in three different modes, namely, fixed link, random link, and best link. The mode determines which link the stage uses to forward data.

- In **fixed link**, a transmitter node of a stage always uses the same outgoing link to relay its data. Hence, the result is identical to a traditional path—the source always sends data to the same neighbor, which in turn always forwards it to the same next node, and so on. Since this mode does not exploit at all the diversity of the corridor, the corridor construction mechanism does not need to widen the single-path to a corridor but can just provide a single-path.
- In contrast to fixed link, the **random link** mode chooses at each hop a next node randomly out of the receiving stage nodes. Thus, each packet may follow a different sequence of nodes to the destination. In this case, the corridor construction algorithm must provide a complete corridor to allow each stage to choose a link.
- Finally, the **best link** mode operates similarly to random link but chooses the next node based on an LQI instead of randomly. Thus, this mode requires at least coarse channel state feedback to decide which link to use. Such feedback can be, for instance, the average SNR of each link over all subcarriers.

If all stages use SL, the result is a single hop-by-hop path to the destination for all three modes. That is, nodes do not split packets among multiple receivers but forward them as units to the next node. On average, fixed link and random link behave similarly—due to the randomness of the wireless medium, the fixed link mode may encounter a link with good or poor channel conditions at random, too. On the contrary, the best link mode actively avoids links with poor channel conditions and thus results in a better average



(a) Fixed link (b) Random link (c) Best link

Figure 4.1: SL modes. The thick lines show which links nodes choose to forward data. The probability values p indicate the probability of the transmitter choosing the highlighted links at each stage. The values of p in (c) are exemplary and unrelated to the topology.

throughput. Still, note that SL cannot choose among all links of the stage but only among the outgoing links of the node which is transmitting data. Figure 4.1 summarizes all three modes. The values of p indicate the probability with which the nodes forward data via the links highlighted in Figure 4.1. For fixed link, the probability is always one since nodes always choose the same next node. When using random link, nodes forward packets with the same probability on all their outgoing links. Finally, in best link, p is related to the channel state. Note that the probability for the last stage is always one since there is only one link to the destination.

4.1 SINGLE LINK DESIGN

For our SL stage mechanism design, we follow a best link approach. If not stated otherwise, we also choose this mode when we use SL in the following chapters. The reason is twofold. First, the best link mode yields a stronger baseline than the random or fixed link modes for comparison with our OFDMA and IA stage mechanisms. Since it considers the SNR of each link, it is also stronger than traditional routing protocols that use the hop-count as a metric. Second, the feedback overhead for per-link SNR values is typically negligible, in contrast to per-subcarrier values. Further, if the corridor allows for multiple PHYs, CSI feedback is anyhow necessary to enable the main node to decide which PHY is best suited according to the current channel state. Hence, in that case, the best link mode does not cause any additional overhead. Next, we describe the design of our SL stage mechanism.

4.1.1 *Profile*

We design SL as a highly flexible stage mechanism to allow Corridor-based Routing to combine it with any other PHY. In particular, we conceive SL as a basic mechanism which Corridor-based Routing can resort to if no other PHY can operate due to disadvantageous channel conditions or profile incompatibility. Table 4.1 shows an overview of the requirements of the profile of our SL stage mechanism.

Characteristic	**Description**
Input data	Allows for any amount of data at any of the nodes of the transmitter group.
Output data	May result in any amount of data at any of the nodes of the receiver group.
Stage shape	Allows for any stage shape.
Feedback	Coarse per-link SNR.
Time synchronization	Coarse time synchronization to allow for successive transmissions within stage.
Frequency synchronization	Receiver-side, pilot-aided CFO estimation similar to existing 802.11 systems.
Coherence time	No particular requirements on coherence time.
Coherence bandwidth	No particular requirements on coherence bandwidth.

Table 4.1: Profile of our SL stage mechanism.

We allow any amount of data at any of the input and output nodes of a stage using SL. Actually, this is unnecessary for corridors using SL only since nodes do not split packets among multiple receivers, as in Figure 4.1. However, it is crucial if the corridor uses other stage mechanisms such as OFDMA or IA. Further, we allow corridors to use SL on stages of any shape. That is, we do not impose a minimum or maximum number of transmitters and receivers. Essentially, transmitters transmit their data sequentially. The more transmitters, the more individual transmissions and, the more receivers, the more diversity, since each transmitter can choose to which receiver it sends its data. Due to the simplicity of SL, its profile only poses loose requirements regarding the PHY characteristics of a stage. First, SL only needs coarse per-link feedback, such as the average SNR, which typically incurs low overhead. Second, loose time synchronization is enough to ensure that the transmissions of the individual transmitters in the stage do not overlap in time. For frequency synchronization, nodes can use techniques from 802.11 systems, such as pilot-aided CFO estimation. This does not require any feedback since receivers can correct the CFO after a transmission has taken place. Finally, SL does not assume that channel characteristics remain constant for a certain interval, nor does it require that adjacent OFDM subcarriers behave similarly. Hence, it does not pose any requirements on coherence time and bandwidth, respectively.

4.1.2 *Operation*

In the following, we describe the operation of SL, which realizes the above profile. Figure 4.2 depicts our SL frame format. All transmissions use the entire available bandwidth, and multiple access occurs in the time domain only. The frame consists of three phases, namely, (a) CSI measurement, (b) control data transmission, and (c) data transmission. The corridor construction protocol establishes the transmission order during construction.

4.1.2.1 *CSI Measurement*

In the CSI measurement phase, each transmitter transmits a set of pilot symbols. All receivers process each set of pilots to determine the channel characteristics of all their incoming links. In particular, they compute the average SNR over all subcarriers. To this end, receivers first obtain the channel coefficient H^s of each subcarrier s using the transmitted pilots p_t^s and the received pilots p_r^s as in Equation 4.1. If the transmitters send more than one OFDM symbol with pilots, receivers average the estimated channel coefficients over all M OFDM symbols. Second, receivers equalize the received pilots

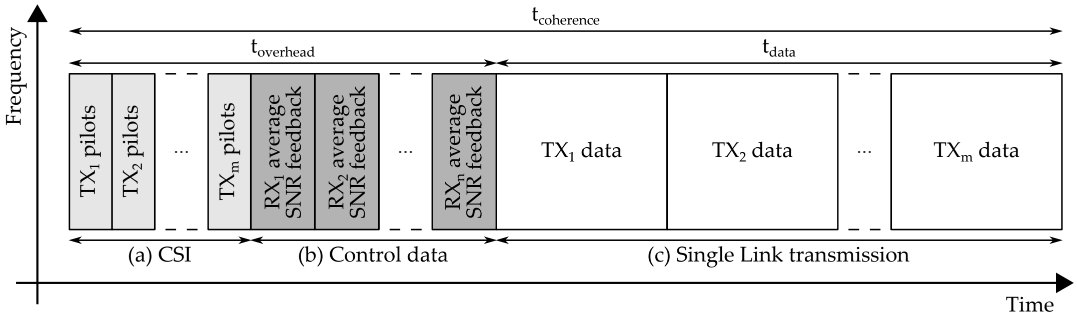


Figure 4.2: SL frame format. Shaded areas indicate control message overhead.

p_r^s using zero-forcing as in Equation 4.2 for each OFDM symbol. Third, they determine the per-subcarrier Error Vector Magnitude (EVM) based on the equalized pilots p_e^s and the transmitted pilots p_t^s as in Equation 4.3. The EVM is the average distance of the received symbols to the originally transmitted constellation points, and is thus directly related to the noise. Intuitively, it is a measure of the size of the point clouds around the constellation points. Finally, receivers compute the per-link SNR $\mathrm{SNR}_{\mathrm{link}}$ by deducing the per-subcarrier SNRs from the EVM and averaging the result, as in Equation 4.4.

$$H^s = \frac{\sum_{k=1}^{M} \frac{p_r^s(k)}{p_t^s(k)}}{M} \qquad (4.1) \qquad\qquad p_e^s(k) = \left.\frac{p_r^s(k)}{H^s}\right|_{\forall k \in M} \qquad (4.2)$$

$$evm^s = \frac{\sum_{k=1}^{M} |p_e^s(k) - p_t^s(k)|}{M} \qquad (4.3) \qquad\qquad \mathrm{SNR}_{\mathrm{link}} = \frac{\sum_{s=1}^{N} \frac{1}{evm^s}}{N} \qquad (4.4)$$

4.1.2.2 *Control Data Transmission*

In part (b) of the frame in Figure 4.2 the receivers send the computed SNR values back to the transmitters. To keep feedback overhead low, receivers use an efficient encoding. That is, instead of transmitting the SNR as, for instance, a plain double precision floating point value in linear scale, the receivers transmit a simplified variant, such as the integer part of the logarithmic SNR value. Alternatively, receivers can use a codebook approach—we discuss this technique in Chapter 5 for detailed per-subcarrier CSI feedback. For our prototype implementation (c.f. Section 4.2), we use such a codebook approach but our design allows for any technique that quantizes the per-link SNR computed at the receivers. Since nodes in a stage are fully connected, all nodes receive the feedback messages in frame part (b) without any additional overhead cost. That is, after that part, all nodes are aware of the average per-link SNR on all stage links. Still, note that the transmitters only need to know the SNR of their outgoing links since they can only forward data along their own links.

4.1.2.3 *Payload Data Transmission*

Finally, in part (c) of Figure 4.2, SL transmits data. Transmitters select their best outgoing link, and forward their data in sequence. The transmission time may differ for each node since each transmitter may have a different amount of data to forward, and each link may support a different MCS. If the overall transmission time exceeds the coherence time of the channel, SL must split the transmission into two or more frames (c.f. Section 3.2.1.5). The additional frames include again CSI measurement and feedback to capture potential channel state changes. Further, two or more transmitters may choose the same receiver node. As a result, that node receives both payloads, and merges them to a single data block, which it forwards in the next stage. Since transmitter nodes in SL always send all their data to a single receiver, data blocks may merge as nodes forward them through the corridor, but they never split. Splitting only occurs when a corridor combines SL with other PHYs such as OFDMA. Still, SL requires data blocks to include a header which indicates which part of the original packet they include. The original packet is the packet which the source transmits in the first stage of a corridor. The destination uses those headers to reassemble the packet once it receives all data blocks (c.f. Section 4.2.1).

4.1.3 *Corridor Protocol Phases*

The above design corresponds as follows to the corridor protocol phases we present in Section 3.2.1. Frame parts (a) and (b) in Figure 4.2 are the stage maintenance of SL. If the corridor supports other PHYs, nodes may substitute the coarse SNR feedback introduced in Section 4.1.2 with detailed CSI to enable the main node to evaluate whether other stage mechanisms are more beneficial than SL. Profile matching is virtually unnecessary for SL since it is highly flexible, and thus can interact with all stage mechanisms we consider in this and the following chapters. Moreover, its stage coordination overhead is minimal. In particular, intra-stage coordination is limited to ensuring that nodes do not transmit simultaneously. SL achieves this by ensuring that nodes transmit in a certain order which is implicitly given by the corridor. Regarding inter-stage coordination, SL must let the next stage know which receiver has received which data blocks. To this end, SL uses the headers sketched in Section 4.1.2.3. Finally, the data transmission phase corresponds to the back-to-back transmission of data blocks in part (c) of Figure 4.2.

4.2 IMPLEMENTATION

In this section, we describe our implementation of our above SL design. Concretely, we explain how we realize this design at the PHY, LNK, and NET.

4.2.1 *Network Layer*

Whenever the NET gets a packet from the upper layers, it determines the destination from the transport layer headers. If no corridor to the destination exists, it initiates the corridor construction protocol we introduce in Chapter 3. Once the corridor is available, it re-encodes the packet into so-called *segments*. Corridor-based Routing does this to allow nodes to split data blocks if needed. As discussed in Section 4.1.2.3, SL itself does not support data splitting but we design it to be compatible to other PHYs that do allow nodes to do so. Hence, SL must be able to handle such data segments. The key challenge is that (a) intermediate nodes must be able to route individual data segments, and (b) the destination must be able to reassemble them to the original packet. Hence, we do not allow nodes to split packets arbitrarily but only up to a certain granularity. A segment is the smallest data unit, and all data is coded into such segments. In other words, data blocks are groups of segments. Splitting a data block translates into dividing it into two groups of segments. To merge two data blocks, a node simply strings the corresponding groups of segments together. Each segment includes an identifier which allows the destination to deduce its position within the original packet. Thus, intermediate nodes do not need to adjust this header whenever they split or merge data blocks.

The segment size has a direct impact on the inter-stage coordination overhead since each segment includes the aforementioned identifier. The smaller the segment size, the more segments the source node needs for each packet, and thus the more—and larger—identifiers it uses. However, a small segment size allows for fine-granular data splitting, which helps intermediate nodes to fully exploit the capacity of stage links. We delve into this issue in Chapter 5 for the case of OFDMA. Further, the segment size is also related to channel coding since nodes must add error correction and detection codes to each individual segment. While nodes could aggregate all segments of a data block for

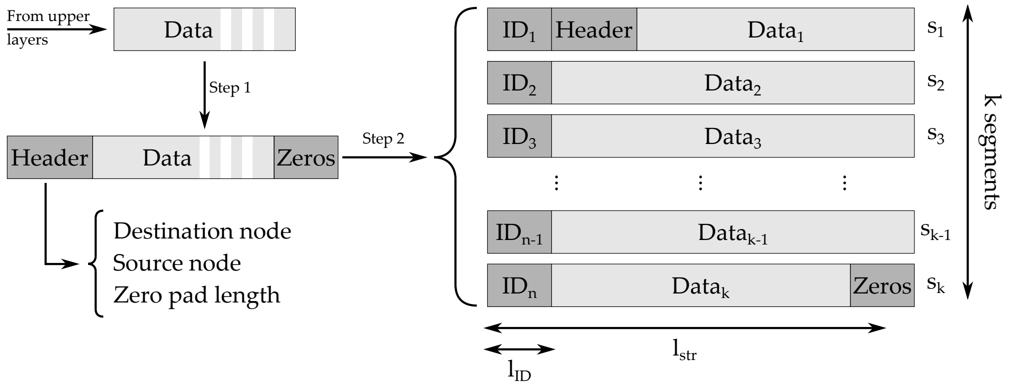


Figure 4.3: Packet segmentation at the NET for SL as well as other PHYs.

coding, this means that transmission errors would cause costly retransmissions of the entire data block. Hence, we encode each segment individually. To this end, the LNK (c.f. Section 4.2.2) can use either discrete-rate or rateless codes. In our implementation, we use a rateless code, namely, Strider[1] [48]. Since Strider only allows for certain input data sizes, we set the segment size accordingly. In particular, we process packets at the NET as depicted in Figure 4.3. First, we add a packet header and, if needed, a zero padding trailer. The latter is needed to ensure that the length of the resulting packet is a multiple of the data input length of Strider l_{str}. Second, we divide the packet into segments of length $l_{\text{str}} - l_{\text{ID}}$, and prepend an identifier of length l_{ID} to each segment. As a result, we obtain a set of segments that Strider can encode individually. As depicted in Figure 4.3, only the first packet segment includes the source and destination addresses. That is, only the node of each stage that receives that segment can decode this information—still, this information is needed at all stage nodes for routing. Hence, the node that receives the header, broadcasts it within the stage. It also becomes the main node for the next stage.

4.2.2 *Link Layer*

The LNK of our SL implementation deals with channel access and flow control. Regarding channel access, the LNK needs to deal with both internal and external collisions—the former refer to simultaneous transmissions of nodes that are part of the corridor, while the latter involve nodes outside the corridor. We avoid internal collisions using the frame structure shown in Figure 4.2, which we discuss in detail in Section 4.1.2. For our evaluation, we consider a single data transmission. Hence, we do not experience the aforementioned external collisions, and thus we do not implement specific mechanisms to deal with them. Still, Corridor-based Routing can easily build on existing solutions. For instance, we can extend the frame in Figure 4.2 to include a Network Allocation Vector (NAV)—each node of the stage sends such a message to prevent any simultaneous 802.11 transmissions. While the performance of the 802.11 LNK is limited, state-of-the-art improvements such as migrating backoff to the frequency domain [132], or combining LNK acknowledgments with higher layer signaling [130] mitigate this effect.

Flow control at the LNK coordinates detection and correction of transmission errors, including packet retransmissions. As introduced in Section 4.2.1, we use Strider for rateless coding. That is, we do not need to estimate the best suitable MCS for the current

---

[1] We use the Matlab implementation of Strider available at `http://snsg.stanford.edu/projects/strider/`

Strider parameter	Value
Batch size	33
Bits per Strider packet	362 bits
Bits per batch of packets	$33 \cdot 362 = 11946$ bits ≈ 1500 bytes
Number of batches	27
CRC bits	16 bits

Table 4.2: Strider parameters for SL as well as other stage mechanisms. We refer the reader to the original work on Strider [48] and Section 2.2.1.3 for details on each parameter.

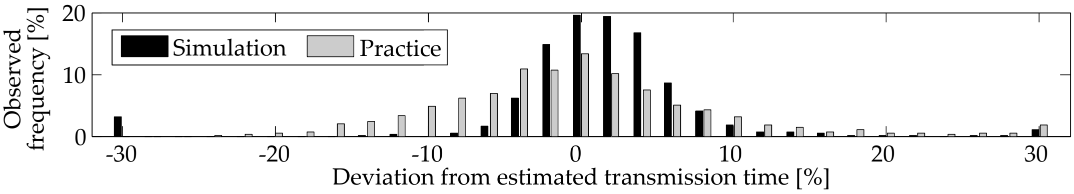
channel conditions but just send batches until the receiver can decode. However, based on the characteristics of Strider [48], we set the maximum number of batches to 27—further Strider parameters are shown in Table 4.2. In other words, if a receiver is not able to decode a segment after receiving 27 batches, flow control triggers a retransmission. Similarly, if Strider reports a Cyclic Redundancy Check (CRC) error for a segment, the transmitter node retransmits the segment. Flow control allows for a certain maximum number of retransmissions. If more retransmissions are needed, the LNK assumes that the link is not usable at all and reports a transmission failure.

4.2.3 *Physical Layer*

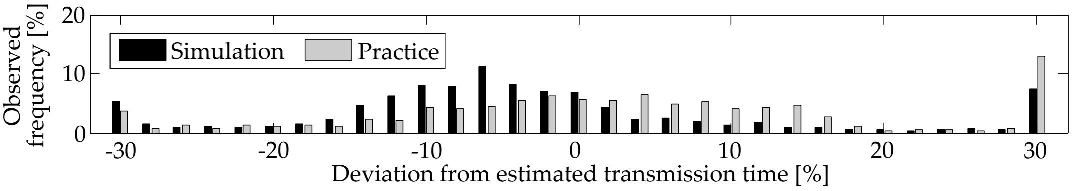
At the PHY, SL transmits batches for each Strider segment until the receiver can decode. A critical issue for rateless codes is knowing how many batches the receiver needs. Basic approaches assume that the transmitter keeps transmitting batches until it receives an acknowledgment from the receiver. However, this requires a feedback channel and may incur significant signaling overhead. State-of-the-art solutions to this problem build on statistical models [49], that is, they estimate the number of batches the receiver needs based on channel characteristics. In our practical SL implementation, we combine both approaches. More precisely, we first estimate the number of batches based on the coarse SNR feedback which is anyhow available for link selection. However, for evaluation purposes, we then transmit all 27 batches that Strider generates at the transmitter (c.f. Section 4.2.2), and determine how many batches the receiver actually needs. This allows us to investigate the accuracy of our estimation. In other words, we estimate in advance the number of batches but allow the destination to request more if needed.

4.2.3.1 *Batch Number Estimation*

In the following, we present practical results from our SDR testbed on the accuracy of the aforementioned batch number estimation, which is directly proportional to the transmission time. In Section 4.2.3.3, we then show its impact on our testbed measurements. Figure 4.4 depicts how much the actual transmission time of Strider segments deviates from the estimated transmission time for both simulated and practical experiments. In particular, Figure 4.4a shows this deviation for packet transmissions on groups of adjacent subcarriers. We observe that the result roughly follows a normal distribution. However, the variance is significantly larger in the practical case than in simulation. That is, in practice we infer from the SNR either too long or too short transmission times more



(a) Estimated vs. actual transmission time per subchannel.



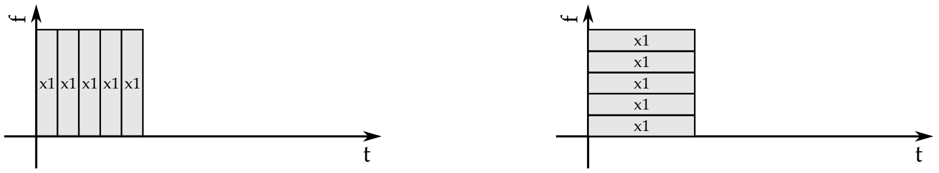
(b) Estimated vs. actual transmission time per stage.

Figure 4.4: Deviation from the estimated transmission time in simulation and practice. While (a) shows the accuracy for an individual subchannel, (b) depicts the effect on the entire stage. The values at ±30% include the deviations in $[-\infty \cdots -30\%]$ and $[30\% \ldots \infty]$.

frequently than in simulation. According to Figure 4.4a, deviations are often negative in practice. However, as we discuss in Section 4.2.3.2, the impact of the positive deviations is much larger. As a result, the data transmission phase of a stage takes longer than expected, and the signaling overhead to request further batches increases.

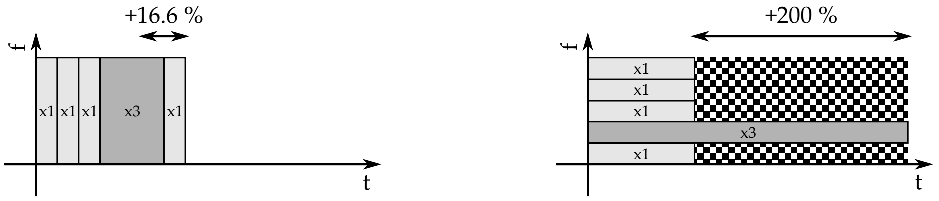
4.2.3.2 *Scheduling Approaches*

The impact of the batch number estimation accuracy on the overall transmission time strongly depends on the scheduling technique at the PHY. Essentially, SL can use either wideband or per-subchannel scheduling. In wideband scheduling, a transmitter uses all subcarriers for each batch, as depicted in Figure 4.5a. On the contrary, in per-subchannel scheduling, a transmitter transmits each batch on an individual subchannel. Figure 4.5b shows an example. A subchannel is a group of adjacent subcarriers with similar channel characteristics. The PHY often groups subcarriers to subchannels to reduce overhead. For instance, instead of providing feedback for each individual subcarrier, the PHY can send feedback only per subchannel, since adjacent subcarriers behave similarly. The advantage of per-subchannel operation is that transmissions can adapt better to frequency-selective channels. While poor subchannels use more batches to convey a segment, subchannels with good channel conditions require only a few batches. In contrast, in wideband scheduling the worst subchannel typically limits the throughput. Hence, the PHY can save transmission time by using a per-subchannel approach, and scheduling more segments on the good subchannels than on the poor ones. If all subchannels have similar channel conditions, wideband and per-subchannel scheduling should perform identically, as in Figures 4.5a and 4.5b, respectively. Concretely, in those examples, all segments require the same number of batches, which we indicate with “x1”. The drawback of per-subchannel scheduling is that the transmitter requires feedback for each subchannel. The reason is that it needs to estimate the transmission time on each subchannel in order to schedule segments beneficially, that is, to transmit more segments on the better subchannels. While this results in a larger feedback overhead, it potentially reduces the transmission time.



(a) Estimated transmission time for wideband scheduling.

(b) Estimated transmission time for per-subchannel scheduling.



(c) Actual transmission time for wideband scheduling.

(d) Actual transmission time for per-subchannel scheduling.

Figure 4.5: Example of impact of inaccurate batch number estimation on wideband and per-subchannel scheduling. The percentage indicates the increase in frame length in the actual transmission compared to the estimated transmission.

Moreover, if stage maintenance anyhow provides detailed CSI feedback to enable the main node to decide which PHY is most beneficial for the stage, per-subchannel scheduling does not incur any additional overhead. In our evaluation in Section 4.3, we compare the performance of both per-subchannel and wideband scheduling for SL.

4.2.3.3 *Scheduling Impact*

From the description in Section 4.2.3.2, per-subchannel scheduling provides higher throughput than wideband scheduling. However, this only holds if transmitters can estimate the transmission time of segments on each subchannel accurately. Our results in Figure 4.4a reveal that, in practice, this estimation is challenging. The impact of inaccurate transmission time estimation differs for each of the scheduling approaches we present earlier. For wideband scheduling, the impact is limited. In the example shown in Figure 4.5c, the segment marked as "x3" unexpectedly requires three times as many batches as the other segments. Our above analysis on the batch number estimation shows that such deviations are not due to inaccurate feedback but occur randomly. As a result, the transmission in Figure 4.5c takes 16.6% more time compared to the estimation in Figure 4.5a. In contrast, in per-subchannel scheduling, the same effect results in a 200% increase in transmission time, as shown in Figure 4.5d. The key difference is that wideband scheduling uses all subchannels to absorb the increased transmission time, whereas per-subchannel scheduling is limited to a single subchannel. Hence, all other subchannels remain idle and thus waste resources, as the checkerboard area in Figure 4.5d indicates. In other words, inaccurate transmission time estimations may result in a significant resource wastage for per-subchannel scheduling since transmitters cannot allocate segments to subchannels properly. This may undermine the advantage of using per-subchannel scheduling entirely. For SL using per-subchannel scheduling, Figure 4.4b shows the practical deviation of the overall transmission time for an entire *stage*—instead of a single subchannel—compared to simulations. As expected, we observe that the inaccuracy of practical estimations results in significantly longer stage transmission times than in simulation. Note that the value at 30% in Figure 4.4b includes all time deviations in the range $[30\% \dots \infty]$. Our data shows that the practical values in that range are, on average, considerably larger than the simulation ones. In other words, inaccurate transmission time estimations have a large impact on the length of stage frames and thus on throughput, as suggested in Figure 4.5. Hence, in practice, wideband scheduling may be more beneficial for SL than per-subchannel scheduling.

4.2.3.4 *Physical Layer Parameters*

Our prototype uses OFDM at the PHY with parameters similar to the 802.11 standard. In particular, we use 20 MHz channels which we divide into 48 usable subcarriers. The resulting subcarrier spacing is 312.5 kHz. We form subchannels comprising three adjacent subcarriers, that is, subchannels span approximately 937.5 kHz. To ensure that subcarriers in a subchannel behave similarly, subchannels must not exceed the coherence bandwidth B_C. The coherence bandwidth is related to the propagation delay spread τ as $B_C \approx \frac{1}{\tau}$. For typical indoor environments, τ is at most 700 ns [58]. Thus, the coherence bandwidth is about 1.42 MHz, which is significantly larger than our subchannel width. In other words, we can safely assume that all subcarriers within our subchannels behave similarly. Further, our OFDM system uses a 12.5% CP.

4.3 EVALUATION

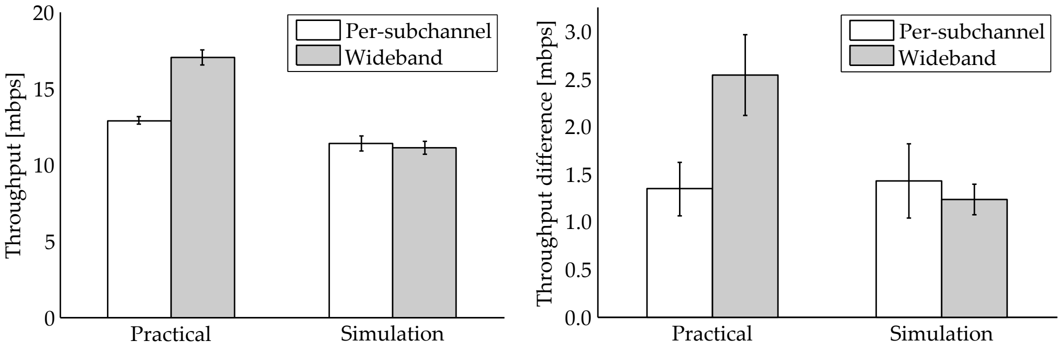
In the following, we evaluate our SL stage mechanism. To this end, we perform both simulated and practical experiments. In Section 4.3.1 we introduce the scenarios we use for our experiments, and in Section 4.3.2 we present our results.

4.3.1 *Scenario*

We perform our experiments in scenarios similar to the ones we use in Section 3.2.2.3. In particular, we carry out simulations in the UD-S and UD-L scenarios from Table 3.2. This allows us to study random network topologies with different average SNRs. For each case, we use the corridor construction protocol that we introduce in Chapter 3 to build corridors of intended widths two, three, and four. However, in contrast to our experiments in Chapter 3, we use a rateless code. Thus, we do not observe the step-by-step MCS adaptation to the available channel capacity, such as in Figure 3.7. Instead, Strider achieves that capacity immediately by sending additional batches until the receiver can decode. For our practical experiments, we use again the testbed depicted in Figure 3.6. In this case, we build corridors of intended width $w = 4$.

4.3.2 *Results*

For both our practical and simulated scenarios, in Section 4.3.2.1 we use the end-to-end throughput as a metric to investigate the performance of wideband and per-subchannel scheduling. Moreover, we study the impact of the corridor length and width in Section 4.3.2.2, as well as the number of failed transmissions in Section 4.3.2.3.



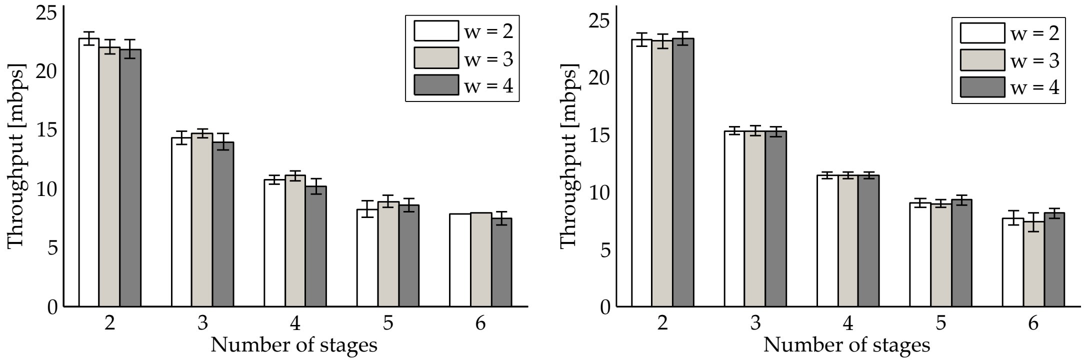
(a) End-to-end throughput. (b) Throughput penalty for corridor construction.

Figure 4.6: SL throughput analysis for corridors of length three and intended width $w = 4$.

4.3.2.1 *End-to-End Throughput*

Figure 4.6 shows our throughput analysis of SL. Concretely, we investigate the end-to-end throughput for corridors with three stages and intended width $w = 4$. Moreover, we compare our practical measurements with the results that we obtain from simulations in our UD-L scenario. For UD-S, results are equivalent but the absolute throughput values are larger due to the stronger links. Figure 4.6a shows that we achieve roughly 10 to 15 mbps. As our previous discussion on scheduling approaches in Section 4.2.3.3 suggests, we observe that wideband and per-subchannel scheduling behave differently in practice than in simulation. More precisely, in simulation both approaches result in virtually the same throughput. Still, per-subchannel scheduling achieves a slightly larger average throughput than wideband scheduling since it can adapt better to the frequency selective behavior of links. If our simulated links were more frequency selective, we would expect the difference to become larger in favor of per-subchannel scheduling. In contrast, our practical results exhibit the opposite behavior—per-subchannel scheduling performs significantly worse than wideband scheduling. Based on our measurements in Figure 4.4, we deduce that the reason is the difficulty of accurate batch number estimation, as we discuss in Section 4.2.3.1. This results in resource wastage for per-subchannel scheduling, as shown in Figure 4.5d. Essentially, occasional deviations of the transmission time of a segment cause per-subchannel scheduling to allocate too many segments on certain subchannels. Hence, those subchannels limit the throughput. On the contrary, wideband scheduling uses all subchannels for each segment, and thus does not waste resources in case of the aforementioned deviations (c.f. Figure 4.5c). This counter-intuitive result highlights the practical challenges of SL. While this issue is related rather to link adaptation than to Corridor-based Routing, understanding its implications is crucial to correctly assess the benefits of corridors. Basically, it affects the performance of SL, which we use as baseline in the following chapters. Hence, its behavior influences the gains that we obtain. Finally, the difference regarding throughput in Figure 4.6a between simulation and practice is most probably due to the different link strengths for both cases—in the practical case, our lab environment determines the link quality. Thus, our results do not imply that SL performs better in practice than in simulation.

In all of our experiments, we first build the corridor using our protocol from Chapter 3, and then start transmitting packets. We compute the throughput of each packet as the size



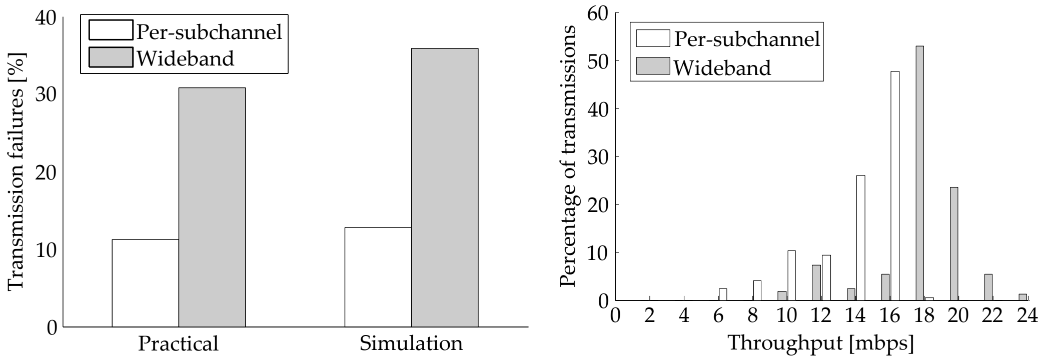
(a) Throughput for per-subchannel scheduling. (b) Throughput for wideband scheduling.

Figure 4.7: Impact of corridor length on SL for $i \in [2..6]$ and $w \in [2..4]$ in our UD-S scenario.

of the packet divided by the end-to-end transmission time. This includes the time required for corridor construction. If a packet can reuse an existing corridor, the construction time is zero. In Figure 4.6b, we depict the throughput penalty for the packets that cannot reuse such a corridor, and thus incur overhead. In our case, this only affects the first packet of each of our experiments. However, if the network topology changes frequently, more packets would incur this penalty. In simulation, the penalty is roughly 1 mbps for both per-subchannel and wideband scheduling. As in Figure 4.6a, both scheduling approaches perform similarly due to the characteristics of our simulated channels. In contrast, the difference becomes larger for the practical case. Essentially, since the corridor construction duration is the same for both scheduling approaches, its impact is larger on the faster wideband scheduling. In other words, the corridor construction time takes up a much larger fraction of the overall transmission time for the wideband case than for per-subchannel scheduling. Still, in comparison to the absolute throughput values in Figure 4.6a, we observe that the throughput penalty is, at most, about 15%. Hence, we conclude that the corridor construction overhead is reasonably low.

4.3.2.2 *Impact of Corridor Length and Width*

Figure 4.7 shows the end-to-end throughput for corridor lengths up to six stages for both per-subchannel and wideband scheduling. In this case, all our results are simulated since our testbed does not allow us to investigate arbitrary corridor lengths. Concretely, we carry out simulations in our UD-S scenario. As expected, the throughput drops for longer corridors. This is intrinsic to WMNs not using full duplex (c.f. Section 2.2.1.2) since a node, or group of nodes, cannot forward a packet until fully received. Thus, throughput drops as $1/i$, which matches our results in Figure 4.7. Moreover, we observe that both scheduling approaches achieve virtually the same performance. Similarly to our earlier results, this is most probably due to the characteristics of our simulated channels (c.f. Section 4.3.2.1). Interestingly, the achieved throughput is similar for all intended corridor widths w. This is surprising since, the wider the corridor, the larger the diversity, and thus the higher the probability of SL forwarding data on links with good channel conditions is. The reason is threefold. First, SL selects links based on their average SNR over *all* subchannels. A single deep fade often has a limited impact on this metric but



(a) Fraction of failed transmissions.

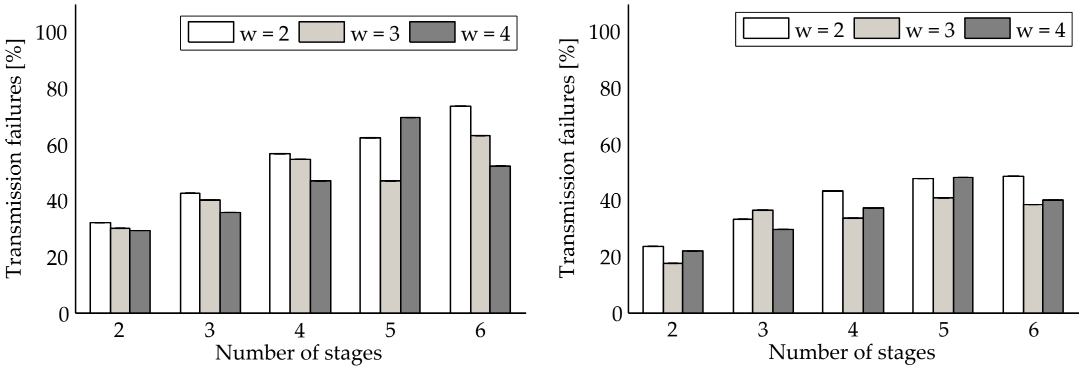
(b) Throughput histogram of the practical case.

Figure 4.8: Failed transmissions in SL for corridors of length three and intended width $w = 4$.

affects performance significantly. In other words, SL cannot identify links with deep fades reliably, and thus wide stages provide limited benefit only. Our data confirms this explanation, since additional experiments show that the best and random link modes perform similarly in a scenario with typical indoor SNRs. Second, our measurements show that the average DoC becomes lower for larger values of w, similarly to our results in Chapter 3. Hence, large values of w do not necessarily imply that nodes can choose out of more links. Third, the average link SNR in UD-S is large due to nodes being deployed on a small area. That is, most links offer good channel conditions, and thus being able to choose among more links does not provide a large benefit. In contrast, the average link SNR in UD-L is significantly lower. Hence, we would expect a better throughput for wider stages. Indeed, our data shows such a behavior. However, the average throughput improvement for larger values of w is limited—overall, our results for UD-L are roughly similar to Figure 4.7 with respect to the corridor width. Most probably, this is due to the low deep fade detection capability of SL, and the low DoCs for large values of w. From the above discussion, we conclude that corridors only provide a limited benefit for SL. In terms of throughput, the advantage of spatial diversity as available in corridors only becomes apparent for advanced PHYs which can fully exploit it, such as OFDMA and IA.

4.3.2.3 *Robustness*

Our results in previous sections reflect the average performance of all *successful* SL end-to-end transmissions. However, we also observe a significant number of transmission failures, which show that the robustness of SL is limited. A failure occurs when a stage exceeds five retransmissions. Figure 4.8a shows the percentage of failed transmissions for a corridor length of three stages and $w = 4$. For both simulation and practice, roughly 30% of the transmissions fail when using wideband scheduling. This surprisingly large number is most probably because channels in our testbed are static. If channels would change between transmission attempts, the probability of retransmissions being successful would increase. In contrast, the fraction of transmission failures drops to about 10% for per-subchannel scheduling. This is expected, since per-subchannel scheduling considers the channel state of each subchannel, while wideband scheduling aggregates all subchannels. For instance, if a channel features a deep fade, per-subchannel scheduling does not transmit any segment on it, while wideband scheduling must use it, resulting in



(a) Transmission failures in UD-L.

(b) Transmission failures in UD-S.

Figure 4.9: Failed transmissions in SL for corridors using wideband scheduling with $i \in [2..6]$.

frequent retransmissions. Similarly to results from related work [47, 122], this improves the performance of per-subchannel scheduling significantly. Moreover, Figure 4.8a shows that this effect is critical both in simulation and practice.

Retransmissions prior to a transmission failure also have a significant impact on the end-to-end throughput. This becomes apparent in Figure 4.8b, which shows a histogram of the throughput of our successful practical experiments. As in Section 4.3.2.1, wideband scheduling achieves higher throughput values than per-subchannel scheduling. For both cases, we observe that most successful transmissions achieve a comparatively high throughput value, and only a few result in low throughputs. We deduce that this throughput difference is due to retransmissions. However, from Figure 4.8a, we know that a significant number of the transmission attempts using wideband scheduling result in such retransmissions, and ultimately transmission failures. This means that only a small fraction of the attempts that incur retransmissions actually benefit from them to deliver data to the destination. Hence, most of the times transmissions either succeed without any retransmissions, or fail entirely. Only a limited number of transmissions lies between both extremes. This is most probably because of the aforementioned steadiness of the channels in our lab. Further, in Figure 4.8b we also observe that barely any transmissions using wideband scheduling achieve throughputs below the peak at 18 mbps. In contrast, per-subchannels scheduling exhibits a larger number of transmissions below its peak at 16 mbps. From the operation of per-subchannel scheduling, we infer that this difference reflects the fine-granular adaptation of the latter approach. More precisely, in some cases where wideband scheduling causes a retransmission, per-subchannel scheduling can avoid it by allowing poor subchannels to transmit more batches. Hence, we conclude that per-subchannel scheduling is more robust than wideband scheduling.

In Figure 4.9 we analyze the fraction of failed transmissions for corridors with up to six stages using SL with wideband scheduling in simulation. The per-subchannel case is equivalent but results in less transmission failures. As expected, the number of failed transmissions in Figure 4.9 increases with the corridor length. Essentially, the more stages, the larger is the probability that at least one of them exceeds the maximum number of retransmissions. Still, we observe that transmission failures occur more often in UD-L than in UD-S. In the former scenario SL exceeds 60% transmission failures, while in the latter it reaches only about 40% for the same corridor length. This is expected, since links in UD-S have a larger average SNR than in UD-L. Regarding the impact of corridor width, larger values of w do not always result in a smaller fraction of transmission failures due to the reasons discussed in Section 4.3.2.2. Nevertheless, Figure 4.9 shows larger differences among values of w than Figure 4.7b. That is, even if larger values of w do not necessarily increase the average throughput of the successful transmissions, they do lower the rate of failed transmissions in the cases where the best link mode successfully identifies deep fades. Basically, the probability of all links suffering a deep fade is lower the wider the stage is. For instance, in most cases in Figure 4.9a, corridors with $w = 4$ achieve lower transmission failure rates than corridors with $w = 2$. Hence, we conclude that, while Corridor-based Routing may not improve the throughput of SL, it often improves its robustness by enabling SL to avoid deep fades.

4.4 SUMMARY

In this chapter, we introduce our Single Link (SL) stage mechanism, which is equivalent to traditional hop-by-hop routing in WMNs. Essentially, each transmitter in a stage forwards all its data to a single receiver node. Transmitters can either forward their data always to the same receiver, to a receiver selected at random, or to the receiver to which they experience best channel conditions. For our SL stage mechanism, we use the latter approach. Moreover, we design SL to be highly flexible and thus compatible to any other stage mechanism. Hence, Corridor-based Routing can resort to it when no other PHY can operate due to disadvantageous channel conditions or profile incompatibilities.

Further, we explain the operation of SL in detail. We use a frame format consisting of three parts, namely, CSI measurement, control data transmission, and payload data transmission. Each part corresponds to one of the corridor protocol phases we introduce in Chapter 3. At the NET, we allow Corridor-based Routing to split and join packets as they traverse the corridor. To this end, we add an identifier to each packet segment, which enables the destination to reassemble the packet. At the LNK, we use Strider [48] for rateless link adaptation and, at the PHY, we allow for two scheduling approaches. Namely, wideband scheduling, which distributes packet segments over all available subchannels, and per-subchannel scheduling, which transmits each segment on an individual subchannel. As a result, the latter adapts to the frequency selective behavior of channels and should thus achieve higher throughputs. However, our practical experiments show that per-subchannel scheduling often performs worse than wideband scheduling. The reason is that the former needs to estimate how long it takes to transmit a packet segment on each subchannel, which becomes challenging in practice. While this issue is related to the domain of link adaptation, it is crucial, to correctly understand our results in the following chapters, since we often use SL as a baseline for our experiments.

We evaluate our SL stage mechanism both in simulation and practice. Our results show the aforementioned behavior of our two PHY scheduling mechanisms. Moreover, we observe that per-subchannel scheduling can deal efficiently with deep fades. In contrast, wideband scheduling cannot avoid them, and thus results in high transmission failure rates. Finally, we conclude from our results that SL only benefits marginally from corridors. Basically, it cannot fully exploit the spatial diversity of corridor stages since it only uses one link at a time. For SL, the only advantage of corridors is that it can choose which link to use. In contrast, advanced PHYs that combine multiple links can leverage much more of the potential of a stage. In the following chapters, we discuss such PHYs.

5

OFDMA STAGE MECHANISM

In this chapter, we present our second stage mechanism, which is based on OFDMA. In contrast to SL, OFDMA uses more than one link of a stage at the same time. Essentially, OFDMA enables transmitters to use only the subchannels of a link on which they experience good channel conditions, instead of forcing nodes to use either all or none of the available subchannels. The subchannels that a certain transmitter does not use are available for other transmitters in the stage. Since the links of a stage are uncorrelated with high probability, subchannels with poor performance on a certain link often exhibit good channel conditions on a different link, and vice versa. Hence, if the stage allocates each subchannel to a link on which it performs well, on average the overall channel quality is better compared to allocating all subchannels to the same link as in SL. In other words, the m transmitters of a stage *share* the available subcarriers to transmit data to the n receivers in the stage. However, to avoid interference, the OFDMA stage mechanism must ensure that transmitters use disjoint sets of subchannels to transmit data.

Figure 5.1 depicts an intuition of the above OFDMA scheme for a stage of width two. Since the stage has two transmitters and two receivers, OFDMA can use four links. Figure 5.1 shows the channel transfer function and the subchannels of each individual link. Due to spatial diversity, each link performs better on different subchannels. Basically, the larger the amplitude of a subchannel, the higher its SNR. For instance, the first link in Figure 5.1 performs best on the first four subchannels. With OFDMA, the stage can pick the best link for each subchannel. In our example, the stage uses the first link for the subchannels one to four, the second link for subchannels nine to ten, the third link for subchannels five to six, and the fourth link for subchannels seven to eight. The result is a combination of all four links which exhibits good channel conditions on all subchannels. While each transmitter can only use the fraction of the bandwidth allocated to its outgoing links, the overall performance is better compared to a single transmitter using the entire bandwidth. However, such an approach raises a number of questions.

- How much throughput gain can an OFDMA stage mechanism achieve?
- How wide must a stage be to fully exploit the advantages of OFDMA?
- How much CSI detail does a stage need to find a suitable resource allocation?

In the following, we address these issues and validate our solutions on a practical OFDMA stage mechanism prototype. Regarding the first question, Figure 5.2 provides an

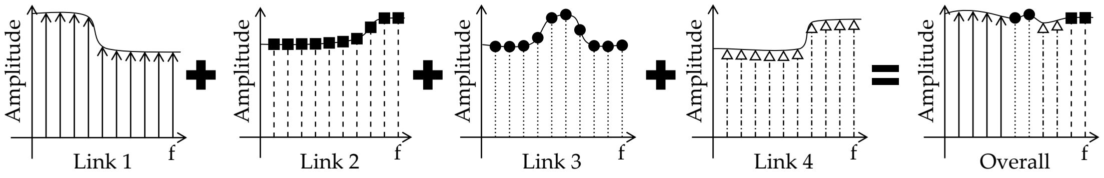


Figure 5.1: Intuition of our OFDMA stage mechanism for a stage of width two. Instead of using only a single link, we use the best subchannels of each link to form a "combined" link.

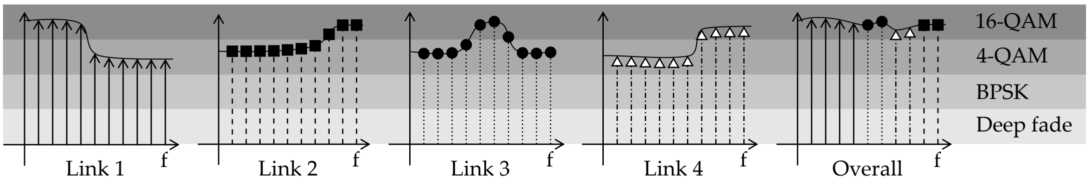


Figure 5.2: OFDMA throughput gain intuition. Each individual link can use 16-QAM on a few subchannels. In contrast, with OFDMA, the stage can use 16-QAM on all subchannels.

intuition of the potential gains OFDMA can provide, particularly in discrete-rate systems. That is, systems which use a discrete set of MCSs instead of a rateless code. Essentially, transmitters select the MCS based on certain SNR thresholds, which we represent as gray bands in Figure 5.2. In particular, on all of the four links in our example, most subchannels support 4-QAM, while only a few can operate at 16-QAM. Hence, each OFDM symbol can carry at most four bits on the subchannels which are above the 16-QAM threshold, and two bits on the remaining subchannels. In the best case, this translates into $4 \cdot 4 + 2 \cdot 6 = 28$ bits per OFDM symbol when using links one or four with, e.g., SL. On the contrary, when using OFDMA, all subchannels support 16-QAM since we allocate each subchannel to a link on which it exceeds the 16-QAM threshold. Thus, each OFDM symbol can carry $4 \cdot 10 = 40$ bits. That is, OFDMA achieves a throughput gain of 42.9% in our example. Moreover, if SL uses wideband scheduling similarly to many traditional systems, the worst subchannel determines the maximum MCS. In that case, the subchannels below the 16-QAM threshold limit SL to $2 \cdot 10 = 20$ bits per OFDM symbol, which translates into a $2\times$ throughput gain for OFDMA. While such large gains are a result of the combination of OFDMA and the threshold-based nature of discrete-rate MCSs, in Section 5.4 we show that our OFDMA stage mechanism also provides significant gains for rateless codes.

5.1 FUNDAMENTAL GAINS

The example in Figure 5.2 shows that the gain of OFDMA strongly depends on the underlying system assumptions, such as whether the system allows for per-subchannel scheduling, or whether it uses a rateless code. However, the fundamental gain of OFDMA stems from the combination of subchannels with good channel conditions. In this section, we investigate this gain to quantify the primary benefit of using OFDMA on a single corridor stage. We analyze the gain without considering particular system parameters which may amplify or diminish its impact on the actual throughput. To this end, we use the channel capacity as a metric since it captures the highest possible throughput for a certain SNR. Moreover, we compare the capacity in theory, simulation, and practice.

5.1.1 *System Model*

For our analysis in this section, we consider a stage of constant width, that is, a stage with $m = n$. Hence, the stage has n^2 links. Further, we assume that the n transmitters always have data to send to the n receivers, and that all nodes have full CSI. Since we do not deal with the CSI feedback overhead for this study, we consider individual subcarriers instead of subchannels (c.f. Section 4.2.3.2). We use a simple "best-out-of-n^2" allocation mechanism which assigns each of the N available subcarriers to the best available link.

Algorithm 3 Best-out-of-n^2 allocation mechanism.

```
Require: average noise power $P_N^{awgn}$ and CSI $H(f)$ of all $n^2$ links of the stage
Ensure: allocation of disjoint sets of subcarriers to the $n^2$ available links
set allocation[N] to zeros
for all subcarriers sc do
  set linkPerformance[$n^2$] to zeros
  for all senders $s$ do
    for all receivers $r$ do
      set currentLink = $n \cdot (s - 1) + r$
      set linkPerformance[currentLink] to $\left|\frac{P_N^{awgn}}{H(sc)}\right|$
    end for
  end for
  set allocation[sc] to the index of the smallest element in linkPerformance
end for
```

While this may lead to some nodes not getting any resources, it allows us to get insights on the best possible performance of OFDMA on a corridor stage. Algorithm 3 describes our allocation mechanism. Essentially, we assign each subcarrier to the link with the smallest $\left|\frac{P_N^{awgn}}{H(f)}\right|$ coefficient, which is the noise at the receiver after equalizing the channel. In particular, P_N^{awgn} is AWGN noise and $H(f)$ is the channel transfer function. Note that our allocation algorithm does not just distribute the subcarriers among the n senders, but among the n^2 available links, since each link of a transmitter may have a different quality. For example, while the link from transmitter s_1 to receiver r_1 might be good, the link from the same transmitter to receiver r_2 might be very poor.

5.1.2 *Analysis*

In the following, we analyze the capacity of the above system in theory, simulation, and practice. We study whether the results in all three domains match. Moreover, we use our analysis to show that we can realize theoretical OFDMA gains in practice.

5.1.2.1 *Theory*

We derive theoretical capacity formulas for a baseline equivalent to SL, and for our "best-out-of-n^2" OFDMA mechanism. For SL, which is based on OFDM, the derivation of the capacity is well-known. The Probability Density Function (PDF) p_γ of the SNR γ assuming Rayleigh fading channels is given by Equation 5.1. That is, γ is exponentially distributed with $\bar{\gamma}$ being the average SNR. We derive the capacity C_{OFDM} of our baseline using the Shannon-Hartley theorem, as shown in Equation 5.2, where B is the bandwidth of the channel in use, and $E_1(x)$ is the exponential integral.

$$p_\gamma = \frac{1}{\bar{\gamma}} e^{-\frac{\gamma}{\bar{\gamma}}} \tag{5.1}$$

$$C_{\mathrm{OFDM}} = B \int_0^\infty \log_2(1+\gamma) p_\gamma \, d\gamma = \frac{B}{\ln 2} e^{\frac{1}{\bar{\gamma}}} E_1\left(\frac{1}{\bar{\gamma}}\right) \tag{5.2}$$

Next, we obtain the capacity of OFDMA in combination with the "best-out-of-n^2" allocation scheme. The key difference to the OFDM case is that we consider n^2 channel realizations instead of only one. The CDF of the best SNR γ_{max} out of all these realizations is given by Equation 5.3. Essentially, it is the multiplication of the CDF F_γ of the SNR of each individual channel. Hence, we integrate the PDF of the SNR from Equation 5.1 to obtain the CDF and then substitute it in Equation 5.3.

$$F_{\gamma_{max}} = F_{\gamma_1} \cdot F_{\gamma_2} \cdot \ldots \cdot F_{\gamma_{n^2}} = (F_\gamma)^{n^2} = \left(1 - e^{-\frac{\gamma}{\overline{\gamma}}}\right)^{n^2} \tag{5.3}$$

Next, in Equation 5.4 we differentiate our result from Equation 5.3 to obtain the PDF $p_{\gamma_{max}}$ of the SNR of the "combined" OFDMA channel when using the "best-out-of-n^2" allocation scheme. In other words, we obtain the PDF of the OFDMA channel via the CDF of the individual channel realizations by integrating, combining, and differentiating again.

$$p_{\gamma_{max}} = \frac{dF_{\gamma_{max}}}{d\gamma_{max}} = n^2 \left(1 - e^{-\frac{\gamma_{max}}{\overline{\gamma}}}\right)^{n^2-1} \frac{1}{\overline{\gamma}} e^{-\frac{\gamma_{max}}{\overline{\gamma}}} \tag{5.4}$$

With the PDF of the OFDMA channel we can now obtain its capacity by again integrating the Shannon-Harley capacity formula multiplied by the PDF, similarly to Equation 5.2.

$$C_{OFDMA} = B \int_0^\infty \log_2(1 + \gamma_{max}) \frac{n^2}{\overline{\gamma}} \left(1 - e^{-\frac{\gamma_{max}}{\overline{\gamma}}}\right)^{n^2-1} e^{-\frac{\gamma_{max}}{\overline{\gamma}}} \, d\gamma_{max} \tag{5.5}$$

To solve this integral, we use Equations 5.6 and 5.7 from Reference [46]. The former allows us to rewrite Equation 5.5 in the form of the latter integral, for which the solution is well known. $E_1(x)$ in Equation 5.7 is again the exponential integral function.

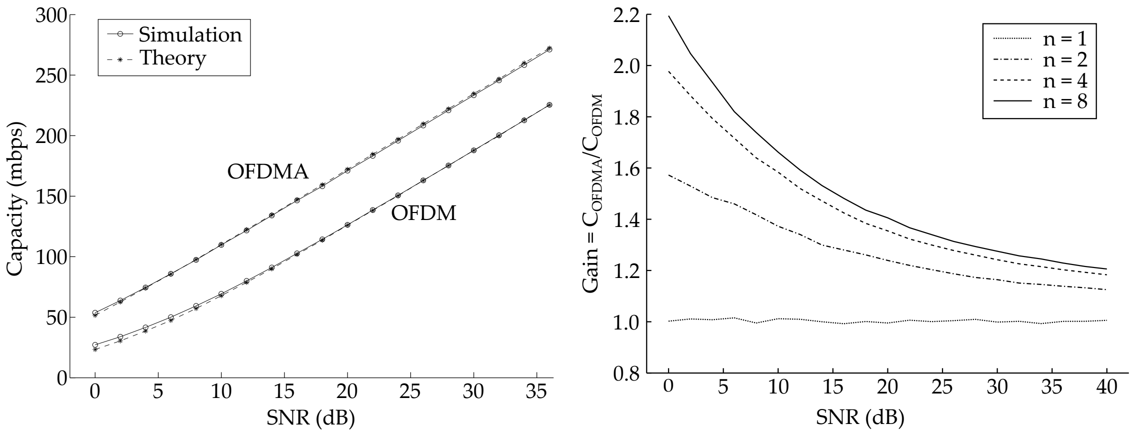
$$(1 - x)^N = \sum_{k=0}^{N} \binom{N}{k} (-1)^k x^k \tag{5.6}$$

$$\int_0^\infty \log_2(1 + x) e^{-\frac{a \cdot x}{b}} \, dx = -\frac{b}{a \cdot \ln 2} e^{\frac{a}{b}} E_1\left(\frac{a(x+1)}{b}\right) - b e^{-\frac{a \cdot x}{b}} \ln(1 + x) \tag{5.7}$$

The aforementioned simplification of Equation 5.5 using Equation 5.6 results in Equation 5.8. Finally, using Equation 5.7 with $E_1(\infty) = 0$, we obtain the OFDMA capacity C_{OFDMA} as shown in Equation 5.9. It depends exclusively on the channel bandwidth B, the average SNR $\overline{\gamma}$, and the stage width n. For $n = 1$, Equation 5.9 simplifies to Equation 5.2 since an OFDMA stage with a single link is equivalent to SL.

$$C_{OFDMA} = \frac{B \cdot n^2}{\overline{\gamma}} \sum_{k=0}^{n^2-1} \binom{n^2-1}{k} (-1)^k \int_0^\infty \log_2(1 + \gamma) e^{-\frac{\gamma(k+1)}{\overline{\gamma}}} \, d\gamma \tag{5.8}$$

$$C_{OFDMA} = \frac{B \cdot n^2}{\ln 2} \sum_{k=0}^{n^2-1} \binom{n^2-1}{k} (-1)^k \left(\frac{1}{k+1}\right) e^{\frac{k+1}{\overline{\gamma}}} E_1\left(\frac{k+1}{\overline{\gamma}}\right) \tag{5.9}$$



(a) Simulation vs. theoretical results for $n = 4$.
(b) Capacity gain of OFDM vs. OFDMA in simulation.

Figure 5.3: Capacity of OFDMA in simulation.

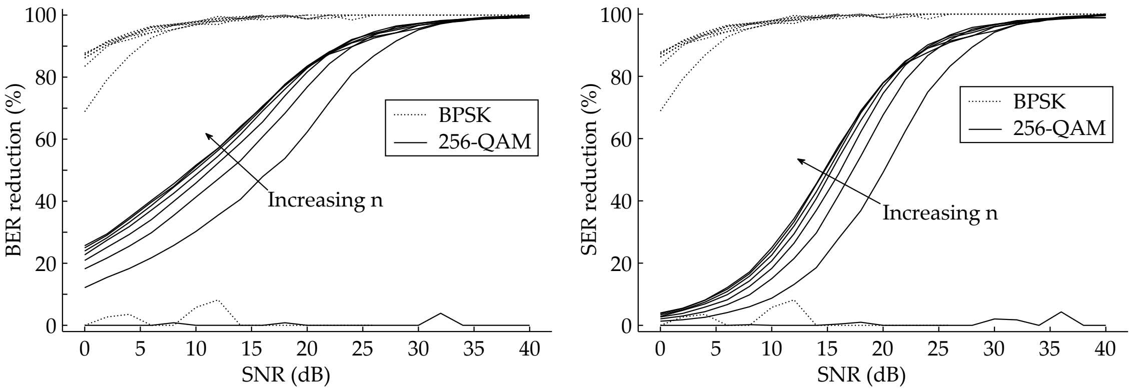
Equations 5.2 and 5.9 are our main results, that is, the capacities of OFDM and OFDMA, respectively. We use them to obtain the capacity gain of OFDMA compared to OFDM. We choose values for the bandwidth B and the average SNR $\overline{\gamma}$ similar to our testbed experiments in Section 5.1.2.3, namely, $B = 18.824$ MHz, $\overline{\gamma} \approx 30.5$ dB and $n = 4$. While we could analyze a range of values for n, we leave this for the simulation study in Section 5.1.2.2, which allows us to get also insights on the bit and symbol error rates. With these values, we obtain $C_{\text{OFDM}} = 175.29$ mbps and $C_{\text{OFDMA}} = 222.12$ mbps. Hence, OFDMA provides a theoretical gain of 26.71% compared to OFDM in a stage of width four.

5.1.2.2 *Simulation*

In the following, we use simulations to study the capacity gains as well as the Bit Error Rate (BER) and Symbol Error Rate (SER) improvement of a stage using OFDMA compared to OFDM. In contrast to our previous theoretical study, this approach allows us to investigate multiple MCSs and measure the resulting error rates. Concretely, we obtain the BER/SER values from the actual decoding process, and the capacity from the measured SNR using the Shannon-Hartley theorem. In our experiments, we vary three parameters, namely the stage width $n \in [1 \dots 8]$, the SNR $\gamma \in [0 \dots 40]$ dB, and the Bits Per Symbol (BPS) using BPSK as well as M-QAM modulation schemes with $M \in [4, 16, 64, 256]$. We implement our OFDMA corridor stage in Matlab using a Rayleigh channel model. All transmissions are uncoded. The results we present in this section are the average of 1000 experiment repetitions. The 95% confidence intervals virtually match the average values.

VALIDATION. We first validate our simulation model by comparison with the analytical formulas derived in Section 5.1.2.1. Figure 5.3a shows a nearly perfect match of the capacity results for both simulation and theory. Hence, we conclude that our simulation is correct, and analyze in the following further factors for the chosen parameters.

CAPACITY. In Figure 5.3b we analyze the OFDMA capacity gain. As expected, for $n = 1$ OFDMA does not provide any gain since the stage does not offer spatial diversity. For larger values of n, gains increase significantly. However, this increase is not linear with each additional node, but becomes smaller as n increases. This is also expected, since



(a) BER improvement of OFDMA compared to OFDM.

(b) SER improvement of OFDMA compared to OFDM.

Figure 5.4: BER and SER improvement in simulation when using OFDMA.

from a certain value of n onwards, additional links only increase diversity marginally. That is, the stage already offers good links for all subcarriers and thus adding more links does not improve performance. Figure 5.3b also shows that gains become larger for smaller SNR values, even exceeding a $2\times$ capacity improvement. While this result is correct, the achievable data rates at such SNR values are small. Nevertheless, this result is interesting for communication in harsh environments. Moreover, for typical indoor SNR values from 25 dB onwards, gains are still above 30% for $n = 4$.

BER. Figure 5.4a depicts the BER improvement for two selected modulation schemes, namely, BPSK and 256-QAM. For both schemes, we plot the performance for different stage widths n. The curves of the remaining modulation schemes lie in between but we leave them out for clarity. Similarly to Figure 5.3b, the flat curve at 0% improvement represents the case $n = 1$. The curve irregularities are due to floating point imprecisions. For larger values of n, we obtain promising results. In particular, for a typical indoor SNR value of 30 dB, OFDMA can reduce the BER by over 90% when using 256-QAM, even for a stage width of only $n = 2$. Increasing the stage width beyond this value at high SNRs only improves the BER marginally. The reason is that channel quality at such SNRs is, on average, very high. Thus, whenever a link performs poorly on a certain subcarrier due to, for instance, a deep fade, the probability that a different link is good at that frequency is high. The range of SNRs for which this happens depends on the modulation scheme. Basically, the results of each scheme are similar but shifted in terms of SNR. As expected, for medium SNRs the benefits for $n > 2$ are larger. Similarly to Figure 5.3b, we observe that, for all SNRs, values of n above a certain threshold do not increase diversity significantly. Overall, we conclude that OFDMA provides large BER improvements.

SER. Figure 5.4b shows the diagram equivalent to Figure 5.4a for the SER. As expected, we observe similar effects. However, the curves are significantly steeper than those of the BER. The only exception is BPSK, since its BER and SER are identical by design. We deduce that this steep shape occurs because we use gray coding. Essentially, symbol errors have the smallest possible impact on the BER. That is, even if symbols are decoded with errors, part of the decoded bits are often correct when using high order modulation schemes. The resulting effect is that the BER starts improving at much lower SNRs than the SER.

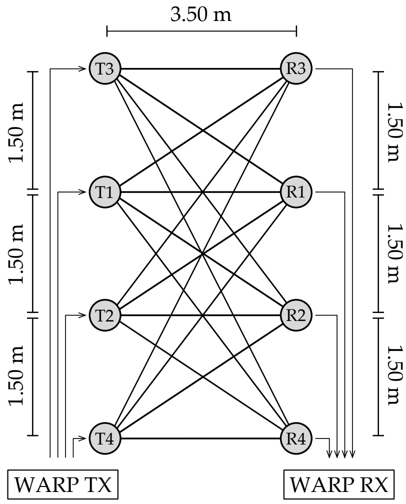


Figure 5.5: OFDMA single stage setup.

Parameter	**Value**
Number of subcarriers	128
Pilot symbols	20
Data symbols	20
Bits per symbol	$[1, 2, 4, 6, 8]$
OFDM symbol duration	6.8 µs
OFDM CP duration	400 ns
Subcarrier spacing	147.059 kHz
Lowest baseband frequency	73.529 kHz
Highest baseband frequency	9.485 MHz

Table 5.1: OFDMA single stage experiment parameters.

5.1.2.3 *Practice*

Lastly, we implement on SDRs the OFDMA stage mechanism which we use for our fundamental gain analysis. This allows us to (a) validate our theoretical as well as simulation results, and (b) show that an OFDMA stage mechanism is feasible in practice. To this end, we use the WARP SDR in combination with the WARPLab Reference Design. As described in Section 3.2.2.3, this allows us to carry out experiments directly from the Matlab workspace. We do not use WARP Drive (c.f. Chapter 7) because we only consider a single stage, that is, we do not evaluate yet a multi-hop scenario. For the practical multi-hop OFDMA case, we refer the reader to Section 5.4. Figure 5.5 shows our testbed, which consists of a stage of width $n = 4$. We connect all transmitters and all receivers to a single WARP board, respectively, to guarantee synchronization among the nodes at each side of the stage. This is possible since WARP boards feature four radio interfaces. However, we treat each radio independently, that is, as if each antenna were an independent node with a single radio interface. For our analysis in this section only, we abstract from practical issues such as the aforementioned synchronization. However, we address these issues later in Section 5.2.2, and include them in our multi-hop evaluation in Section 5.4. Table 5.1 gives an overview of the PHY parameters we use in our experiment. We perform five series of experiments, each with a different modulation scheme. We conduct the experiments in a time frame of 96 minutes. Since the average channel conditions may change during this timespan, we interleave the experiments of the five series in order to ensure that they are equally affected. The results shown in the following are the average measurements for each series. The average SNR on each subcarrier is about 30.5 dB.

VALIDATION. To validate our measurements, we compare the theoretical, simulative, and practical capacity results in Table 5.2 for $n = 4$. Similarly to the previous sections, we obtain the practical capacity using the Shannon-Hartley theorem based on the measured SNR. The matching in all three domains is nearly perfect. The only slight disagreement occurs for the practical case, where the capacity gain is about 3% higher than for the other approaches. However, the reason for this is that the measured OFDM capacity is 3 mbps smaller due to practical issues that arise when some of the subcarriers exhibit poor quality. In these cases, the CFO estimation becomes more challenging, which results

Metric	Theory	Simulation	Practice
Capacity of OFDM	175.29 mbps	175.35 mbps	172.00 mbps
Capacity of OFDMA	222.12 mbps	220.96 mbps	222.94 mbps
Capacity gain	26.71%	26.01%	29.61%

Table 5.2: Comparison of our capacity results in theory, simulation, and practice.

Metric	BPSK	4-QAM	16-QAM	64-QAM	256-QAM
OFDM BER	0.017%	0.034%	0.209%	0.772%	2.897%
OFDMA BER	0.000%	0.000%	0.000%	0.000%	0.147%
OFDM SER	0.017%	0.061%	0.692%	3.780%	17.435%
OFDMA SER	0.000%	0.000%	0.000%	0.000%	1.136%
BER improvement	~100%	~100%	~100%	~100%	94.93%
SER improvement	~100%	~100%	~100%	~100%	93.48%

Table 5.3: Practical BER and SER results for OFDM as well as OFDMA. Zero error rates indicate that the actual values are below the minimum that we can measure in our experiment.

in a slight rotation of the received constellation. This in turn leads to a lower SNR estimate, and hence a lower capacity. In contrast to OFDMA, this issue affects OFDM more often since it cannot avoid poor subchannels. Thus, we conclude that, in some cases, practical issues may increase the OFDMA gain compared to the results we expect based on our theoretical and simulation studies in Sections 5.1.2.1 and 5.1.2.2, respectively.

BER/SER. Table 5.3 presents the BER and SER that we measure in our practical experiments [1]. The first four rows compare the BER/SER for the cases of OFDM and OFDMA, while the last two rows show the reduction as a percentage. The values match our simulations, since OFDMA reduces error rates significantly. In particular, for all modulations up to 64-QAM, we measure zero bit and symbol errors for OFDMA. This does not mean that both BER and SER are zero, but that they drop below the minimum we can measure. To obtain the actual error rates in those cases, we would need to transmit longer frames.

5.1.3 *Discussion*

Our analysis in theory, simulation, and practice shows that an OFDMA stage provides a fundamental capacity gain of about 25% to 30% compared to OFDM for typical indoor SNRs. For systems using discrete-rate MCSs, this translates into a significant improvement of the BER and SER of each modulation scheme. Thus, we conclude that (a) OFDMA is a promising PHY technique for Corridor-based Routing, and (b) OFDMA is feasible in practice for many-to-many communication scenarios. In the following, we design the *system* needed to realize an OFDMA stage mechanism for Corridor-based Routing at the NET, LNK, and PHY. While such a system exploits the fundamental OFDMA gain that we analyze in this section, the actual gain strongly depends on its particular design.

---

[1] Some results in Table 5.3 extend Reference [94], which is a joint publication with Robin Klose based on his Master Thesis [72] that he elaborated under my supervision. The additional results are also from his thesis.

5.2 OFDMA DESIGN

In this section, we extend the single stage OFDMA scheme that we introduce in Section 5.1 to a full-blown stage mechanism for Corridor-based Routing. We depart from many of the assumptions in our earlier basic design, such as global CSI knowledge and constant width stages, and address practical issues. Concretely, in Section 5.2.1 we present the profile of our OFDMA stage mechanism. After that, in Section 5.2.2 we explain how we deal with practical issues. Then, we introduce our OFDMA resource allocation mechanisms in Section 5.2.3, which include both an approach based exclusively on CSI and an approach which also considers traffic information. Finally, Section 5.2.4 deals with the operation of our OFDMA stage mechanism as a system, that is, the interplay of PHY, LNK, and NET.

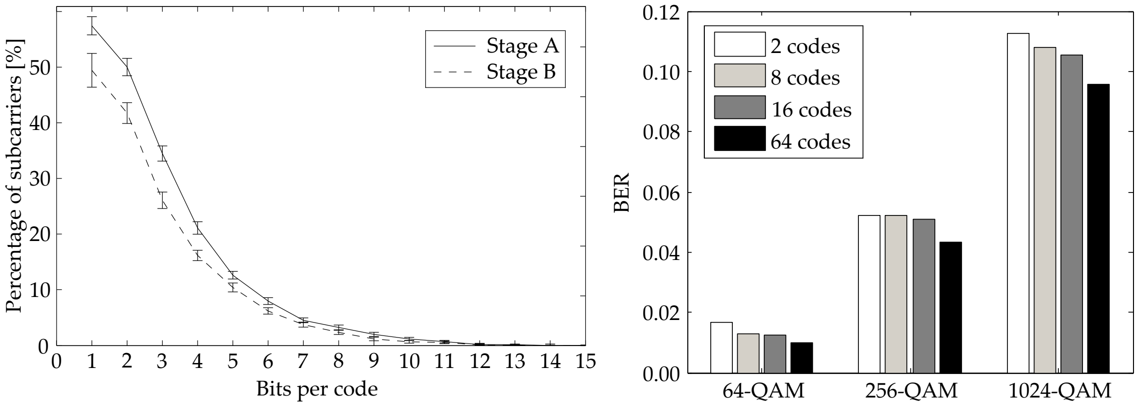
5.2.1 *Profile*

In contrast to the SL profile in Chapter 4, the profile of OFDMA has higher requirements regarding both the stage shape and its physical characteristics. Still, we design our OFDMA stage mechanism to result in a profile which is as loose as possible, and thus compatible to a large number of other stage mechanisms. Table 5.4 summarizes the profile of OFDMA.

While certain OFDMA allocation schemes require all transmitter nodes to transmit the same amount of data (c.f. Section 5.2.3), we design a scheme which allows for any amount of data at each transmitter. Moreover, it does not require each link to get the same amount of subchannels. As a result, our OFDMA profile allows for any input data but may also result in any amount of data at any of the receiver nodes. Regarding the stage shape, we allow for constant, widening, and narrowing stages—the amount of links only influences how much subchannels each link gets. However, the profile

Characteristic	**Description**
Input data	Allows for any amount of data at any of the nodes of the transmitter group.
Output data	May result in any amount of data at any of the nodes of the receiver group.
Stage shape	The number of transmitters m must not exceed the number of subchannels. Otherwise, the stage may have any shape, that is, constant, increasing, or decreasing.
Feedback	Coarse per-subchannel SNR.
Time synchronization	Transmitters must start transmitting simultaneously within the duration of a CP.
Frequency synchronization	Receiver-side, pilot-aided, per-subchannel CFO estimation based on traditional OFDM systems.
Coherence time	The coherence time must be long enough to allow for packet aggregation and thus overhead compensation.
Coherence bandwidth	No particular requirements on coherence bandwidth.

Table 5.4: Profile of our OFDMA stage mechanism.



(a) Percentage of subcarriers allocated differently with full CSI compared to quantized CSI for two stages.

(b) BER of our OFDMA stage mechanism for different codebook sizes and modulation orders.

Figure 5.6: Impact of CSI quantization on OFDMA allocation and performance.

requires that the number of transmitters does not exceed the number of subchannels. This ensures that each transmitter gets at least one subchannel and can thus forward data. In contrast to SL with wideband scheduling, OFDMA requires per-subchannel CSI feedback. While this adds overhead, SL with per-subchannel scheduling requires this, too (c.f. Section 4.2.3.2). Moreover, in Section 5.2.2.1 we show that coarse feedback is sufficient for OFDMA subchannel allocation. Still, due to this overhead, the coherence time of the stage must be long enough to allow for packet aggregation, and thus a lower feedback overhead impact. Since each subchannel operates independently, OFDMA does not require any minimum or maximum coherence bandwidth. Finally, synchronization is critical for OFDMA. In the time domain, transmitters must start transmitting at the same time within the duration of the CP, which is often 12.5% of the OFDM symbol duration. For systems similar to 802.11, this translates into 400 ns. Measurements in our lab show that nodes can achieve such synchronization over the wireless channel (c.f. Section 5.2.2.4). However, this may become challenging in harsh scenarios, such as outdoors. In the frequency domain, OFDMA can build on existing CFO correction techniques to address clock synchronization (c.f. Section 5.2.2.3), and thus does not have any particular requirements.

5.2.2 *Practical Issues*

In this section, we present solutions to the practical challenges that OFDMA faces in a many-to-many stage scenario. In particular, we explain how OFDMA can (a) provide timely CSI feedback for channel allocation without incurring excessive overhead, (b) estimate per-subchannel SNRs, (c) correct per-subchannel CFO, and (d) achieve tight time synchronization wirelessly.

5.2.2.1 *CSI Feedback*

OFDMA requires per-subchannel CSI feedback. To obtain the CSI itself, the transmitters transmit pilot symbols to the receivers, which in turn send the CSI back. To this end, related work typically uses codebooks (c.f. Section 2.2.2.2). The challenge lies in deter-

Codebook size	16 codes	64 codes	512 codes	8192 codes
OFDMA BER	0.0179	0.0170	0.0146	0.0139

Table 5.5: OFDMA BER for 64-QAM with increasing codebook size.

mining a suitable codebook size for a particular PHY. The larger the codebook, the more detailed the CSI but also the more overhead it incurs. For our OFDMA stage mechanism, we find a suitable codebook size based on practical experiments. Concretely, we measure how many bits per CSI value we need to obtain the same subchannel allocation than with full CSI. Figure 5.6a depicts the result for two different stages of a corridor of width two. We observe that both curves are similar, which means that the required codebook size does not depend on the specific channels of a stage. In other words, we do not need to determine a suitable codebook size for each individual stage but obtain roughly similar results if we use a fixed codebook size for all stages. To achieve virtually identical allocations compared to full CSI, Figure 5.6a shows that we need to quantize feedback using 13 bits per code, that is, a codebook with 8192 entries.

However, our survey on related work in Section 2.2.2.2 suggests that much smaller codebooks should be feasible, too. The key to such small codebooks is that we do not necessarily need to obtain the same subchannel allocation than with full CSI but an allocation which results in a similar OFDMA channel quality. That is, we may use different links for each subchannel as long as they have similar channel quality than the links we would use with full CSI. We use the BER as a metric to capture the channel quality. Figure 5.6b shows our results for different modulation schemes. We observe that even a codebook with only two entries achieves roughly similar performance than a codebook with 64 entries. In contrast, Figure 5.6a shows that the difference compared to full CSI in terms of allocated subcarriers drops from about 50% to less than 10% for those values. Moreover, Table 5.5 shows the numerical BER for codebooks ranging from 16 to 8192 entries. Although the codebook with 8192 entries incurs more than three times the overhead than the one with 16 entries, their BER is roughly the same. Thus, we conclude that large codebooks only provide marginal improvement for the case of OFDMA.

The extreme case in Figure 5.6b which only uses two codes is equivalent to the one-bit CSI feedback schemes in Section 2.2.2.2. Essentially, nodes just indicate which subcarriers they prefer with a binary one. This may have a significant impact on the design of the allocation mechanism, since it cannot distinguish which link performs better on a certain subchannel if multiple links would like to use it. For our one-bit feedback results in Figure 5.6b, we use an allocation mechanism which follows a rotating priority order. Initially, we order the links of the stage according to an arbitrary priority. Next, we allocate to the link with the highest priority all subchannels which that link marked with a binary one in its one-bit feedback. After that, we allocate to the link with the second highest priority the subchannels which it prefers out of the remaining ones. The mechanism finishes once it allocates all subchannels. To achieve a fair allocation, the priority order rotates for the next transmission, that is, we cyclically shift the priority order of each link by one.

While one-bit feedback may provide good results in our testbed, different wireless environments may require more precise CSI feedback to find beneficial OFDMA allocations. Hence, we design our OFDMA stage mechanism to support any arbitrary codebook size. If not stated otherwise, we use four bits per code to keep overhead low.

5.2.2.2 *SNR Estimation*

The CSI feedback in Section 5.2.2.1 conveys a quantized version H_s^q of the channel coefficient for each subcarrier and each link. Since stages are fully connected, all nodes receive the feedback and obtain H_s^q. OFDMA allocation schemes can operate directly on H_s^q—essentially, a large $|H_s^q|$ indicates that a subcarrier experiences good channel conditions. However, if the allocation scheme requires the per-subchannel SNR, we cannot obtain it using the approach in Section 4.1.2.1. The reason is that the nodes receiving the CSI feedback are only aware of H_s^q but not of the received pilot symbols, which we need to compute the EVM. However, since the average power of the AWGN noise is similar at all nodes of a stage, we can still use H_s^q to compute the per-subchannel SNR. Basically, nodes deduce the average AWGN noise power P_N^{awgn} either from the idle channel, or from other frame receptions. This allows nodes to obtain the per-subchannel noise power P_N^s of any stage link based on the received CSI H_s^q as in Equation 5.10. Finally, since transmitters normalize their per-subcarrier output power P_S^s, nodes can use P_N^s to estimate the per-subcarrier SNR as in Equation 5.11.

$$P_N^s = \frac{P_N^{awgn}}{|H_s^q|} \qquad (5.10)$$

$$snr^s = \frac{P_S^s}{P_N^s} = \frac{1}{P_N^s} = \frac{|H_s^q|}{P_N^{awgn}} \qquad (5.11)$$

5.2.2.3 *CFO Correction*

Synchronization in the frequency domain is crucial to avoid CFO. Essentially, the receiver must compensate for the clock drift between transmitter and receiver to avoid additional noise. For OFDMA, the challenge lies in correcting the CFO per-subcarrier, since each pair of transmitter and receiver nodes experiences a different CFO. Hence, receivers must correct the CFO individually for each set of subcarriers allocated to a different transmitter. In other words, since the transmitters share the available subcarriers, the CFO may be different for each subcarrier. To estimate the CFO, we use a pilot-aided technique, which is to say that we average the differences between phase shifts of successive pilot symbols in order to identify the systematic phase drift in a transmission. The key insight is that we can estimate the CFO individually for each subcarrier. Instead of averaging the CFO over all subcarriers, we only average it within each subcarrier, or set of subcarriers, that a certain transmitter uses to transmit data to a certain receiver. Additionally, the phase noise may cause the CFO to vary throughout a packet transmission. However, an OFDMA stage mechanism can use existing techniques from single-hop OFDMA systems to address this issue. Moreover, our prototype in Section 5.3 only transmits short packets and thus we do not need to address this in our implementation.

We consider three effects that determine the phase of a symbol, namely, (a) the transfer function of the channel, (b) AWGN noise, and (c) the CFO. The noise component averages out over multiple pilot symbols since it has zero-mean. Further, the channel adds a constant phase shift, which cancels out when comparing the phases of two successive symbols. Hence, both phase components (a) and (b) do not influence our CFO estimation. We can thus obtain the phase offset $\hat{\phi}_\Delta$ as shown in Equation 5.12.

$$\hat{\phi}_{\Delta,tr} = \arg\left(\sum_{s\in S_{tr}} \sum_{i=1}^{N_{pil}-1} e^{j\cdot(\phi_{r,s,i+1}-\phi_{r,s,i})}\right) \qquad (5.12)$$

In Equation 5.12, N_{pil} is the number of pilots, S_{tr} the subcarriers allocated to transmitter t for transmitting data to receiver r, and $\phi_{r,s,i}$ the phase shift of the i^{th} pilot symbol on subcarrier s at receiver r. In other words, we average the CFO over all pilot symbols and all subcarriers that a certain pair of nodes uses to communicate, that is, all symbols which exhibit the same clock drift. To compensate for the CFO, we multiply the received signal on subcarrier s with a complex sinusoidal whose frequency is the inverse of the estimated CFO $\hat{f}_\Delta = \frac{\hat{\phi}_\Delta}{T_{S,CP} \cdot 2\pi}$, where $T_{S,CP}$ is the duration of an OFDM symbol including the CP. While this approach performs well when the quality of all subcarriers in set S_{tr} is overall good, our practical experiments show that it performs poorly when some of the subcarriers have a low SNR due to, for instance, deep fades. The reason is that the strong impact of noise on such subcarriers causes the offset between successive pilot symbols to be many times larger than the offset due to the CFO. Since we average the CFO of all subcarriers in set S_{tr}, this influences the overall CFO estimate. Thus, we use an enhanced CFO estimation technique [73], which weights the contribution of each subcarrier to the overall estimate depending on its channel conditions, as shown in Equation 5.13.

$$\hat{\phi}_{\Delta,tr} = \arg\left(\sum_{s \in S_{tr}} \frac{\sum_{i=1}^{N_{\text{pil}}-1} e^{j\cdot(\phi_{r,s,i+1} - \phi_{r,s,i})}}{\sum_{i=1}^{N_{\text{pil}}-1} (\phi_{r,s,i+1} - \phi_{r,s,i})^2}\right) \tag{5.13}$$

The term in the denominator of Equation 5.13 is the inverse weight for subcarrier s. Essentially, it is the magnitude of phase shift differences between successive pilot symbols. The larger this phase difference, the higher the probability that subcarrier s exhibits poor performance. However, the larger the denominator in Equation 5.13, the less the contribution of that subcarrier to the overall CFO estimation. That is, we use the phase difference as a metric to achieve the desired weighting. We cannot directly use the subcarrier SNRs as weights for the CFO because we need to correct the CFO *prior* to estimating the SNR with the technique in Section 4.1.2.1. Measurements in our testbed show that our approach in Equation 5.13 is significantly more robust to noise than the basic scheme in Equation 5.12. Hence, we use the former in our practical experiments.

5.2.2.4 *Time Synchronization*

To avoid Inter-Symbol Interference (ISI) and thus keep subcarriers orthogonal in OFDMA, all transmitters must start transmitting simultaneously. In particular, the timing constraint is the difference of the duration of the CP and the multi-path delay spread τ. For typical indoor environments, this translates into tens or, at most, a few hundreds of nanoseconds. To this end, we provide mechanisms for (a) synchronizing the transmitters among themselves (intra-group) and (b) synchronizing the receivers to the senders (inter-group).

INTRA-GROUP. To enable simultaneous transmission within the above timing constraint, the main node of an OFDMA stage sends a preamble known to all nodes. The other transmitters correlate the received preamble with the transmitted preamble. The resulting signal exhibits a clear peak at a certain point in time, which all transmitters use as a common reference time to transmit simultaneously their OFDMA data. Practical measurements in our lab [175] using a real-time implementation of such a scheme show its feasibility. The average time offset among transmitters is about 8 ns, which is sufficient for our purposes. In the best case, the offset is close to 1 ns and, in the worst case, it does not exceed 30 ns. Since we use a CP of 400 ns, we can ensure subcarrier orthogonality even in the worst case with high probability. The performance of such a technique depends on the specific preamble the main node uses, and the 802.11 channel on which nodes transmit. The accuracy improves for preambles defined in standards, such as the 802.16 (WiMAX) preamble, compared to preambles chosen randomly. Further, interference on busy channels also increases the average time offset due to occasional collisions of the preamble with data packets. Still, in combination with LNK medium access control techniques, we expect such collisions to be infrequent. Finally, slight variations among the sampling clocks of the transmitters also have a significant impact on the achieved accuracy. Experiments using wired synchronization of the sampling clocks show that the average time offset drops even below 8 ns. Still, OFDMA does not require such tight synchronization and thus the inaccuracy of the sampling clocks is not critical.

INTER-GROUP. We use a similar preamble-based technique to synchronize the receivers to the transmitters. Essentially, the main node transmits a second preamble prior to the OFDMA data frame. Receivers correlate with that preamble to determine the beginning of the frame. In particular, we use both the short and long preambles from the 802.11 standard. The long preamble serves the aforementioned correlation, while the short one trains the AGC. Slight inaccuracies in the detection of the frame start are not critical since the CP compensates for them. However, such inaccuracies result in a phase shift, that is, the channel coefficient of a subcarrier may appear to have a different phase in two or more subsequent transmissions. While this affects the perceived CSI, it does not have an impact on the allocation of subcarriers since OFDMA allocation mechanisms only require the amplitude of channel coefficients (c.f. Section 5.2.3). In other words, even if the pilot symbols that we use for CSI estimation and the actual OFDMA data frame experience a different channel phase, the allocation based on the CSI feedback of the pilots is still valid. Hence, OFDMA does not pose strong constraints on inter-group time synchronization, which allows us to use preamble-based techniques based on traditional PHYs.

5.2.3 *Resource Allocation*

In the following, we deal with subchannel allocation techniques. This is a crucial component of our OFDMA stage mechanism since a beneficial allocation of resources is the key to achieve performance gains compared to traditional WMN schemes. In particular, we discuss two approaches to subcarrier allocation. While the first one decides on subchannel allocation using CSI feedback only, the second one also takes into account the amount of data each transmitter in a stage needs to forward. For the bulk of our experiments, we use the latter approach since it provides more flexibility.

5.2.3.1 *CSI-based*

DESIGN. Our first type of allocation mechanisms uses CSI feedback as an input for subchannel allocation. The most simple form of CSI-based subchannel allocation is the “best-out-of-n^2” allocation scheme in Algorithm 3. Basically, this scheme allocates each subchannel to the stage link on which it performs best. While this results in the best possible overall channel quality, it may allocate no subchannels at all to certain transmitters. This is unproblematic for transmitters which anyhow have no data to

Algorithm 4 Fair CSI-based allocation mechanism.

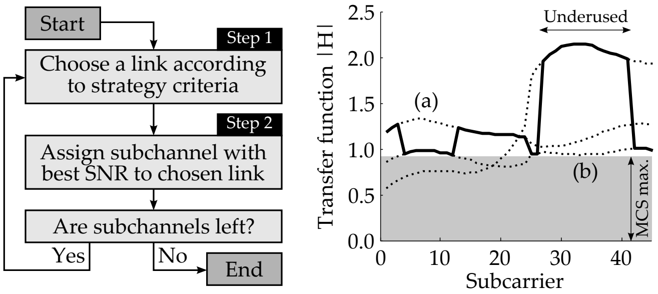
```
Require: average noise power $P_N^{awgn}$ and CSI $H(f)$ of all $m \cdot n$ links of the stage
Ensure: fair allocation of disjoint sets of subcarriers to the $m \cdot n$ available links
set available_subchannels to all subchannels
set subchannels_allocated_to_link[$m \cdot n$] to zeros
while available_subchannels $\neq \emptyset$ do
    find link l which has the subchannel with the largest $\frac{P_N^{awgn}}{H(sc)}$ and
    for which subchannels_allocated_to_link[l] < fair_share
    find subchannel sc on which l exhibits its smallest $\frac{P_N^{awgn}}{H(sc)}$
    allocate sc to l
    remove sc from available_subchannels
    increase subchannels_allocated_to_link[l] by one
end while
```

transmit but becomes critical otherwise. Since this type of allocation mechanisms does not take traffic into account, the “best-out-of-n^2” scheme cannot avoid such issues.

Alternatively, CSI-based schemes can also follow a fair approach, that is, allocate the same number of resources to each node. Still, in this case, determining which link shall get which subchannel is not straightforward. The goal of the allocation scheme is to assign a fair share of subchannels to each link such that it maximizes the overall OFDMA channel quality. However, comparing all possible subchannel allocations is typically infeasible in terms of computational effort. Thus, we follow an iterative approach which allows for different allocation strategies [74]. Essentially, each iteration of our fair CSI-based scheme is divided into two steps, as shown in Figure 5.7a. First, the scheme chooses a stage link which does not yet have its fair share of subchannels. To choose the link, it follows a certain strategy criteria. Second, the scheme allocates to the chosen link the subchannel on which it performs best, out of the subchannels which are not yet allocated. The scheme repeats these two steps until it has allocated all subchannels.

The key to finding good subchannel allocations is the aforementioned criteria which the scheme uses in the first step of each iteration to choose a link. Practical measurements in our lab show that the best strategy is to choose the link which has the subchannel with the *lowest* SNR. In the second step, the scheme allocates to that link the subchannel on which it achieves its *highest* SNR. The rationale behind this somewhat counter-intuitive approach is that the strategy tries to ensure that links with very poor SNR on some subchannels get at least the fair share of subchannels on which they perform best. Hence, the criteria that the scheme uses in the first step defines which links it prioritizes in the first iterations of the algorithm. These links can choose out of a larger number of still available subchannels. If the scheme would follow a strategy which prioritizes links with high SNR, it might happen that in the last iterations the scheme is forced to assign subchannels with very poor channel conditions to the links with low SNR. Besides choosing the best or the worst link in the first step, a third strategy could also choose a link randomly. If not stated otherwise, for fair CSI-based subchannel allocation we always use the strategy which chooses the worst link first, as it performs best. Algorithm 4 summarizes its operation.

LIMITATIONS. Since the above allocation mechanisms do not take into account the amount of data that nodes need to forward, OFDMA designs that build on them typically



Effect (a)
The allocation assigns the same amount of resources to each link. Thus, some subcarriers are not allocated to the best link available.

Effect (b)
Since all subcarriers use the same MCS, the subcarrier with poorest quality limits the throughput. Good subcarriers cannot be exploited.

(a) CSI-based allocation [74].

(b) Testbed channel. The thick line is the OFDMA allocation.

Figure 5.7: Operation and limitations of fair CSI-based allocation.

enforce that all nodes (a) receive exactly the same amount of data, and (b) use the same MCS. This simplifies the design significantly. In particular, since each node has the same amount of resources available for transmission than for reception, and all nodes need to forward the same amount of data using the same MCS, the transmission time of all nodes is the same. Thus, bottlenecks are virtually impossible. However, such a design has significant drawbacks. Figure 5.7b depicts an example based on an actual channel measurement from our testbed. We observe two key limitations, namely, (a) the aforementioned fair allocation of subchannels forces us to use poor links, and (b) the worst subchannel in a stage imposes the maximum MCS since all subcarriers use the same rate. As a result, the time to transport data across the stage increases prohibitively. This issue becomes particularly critical for highly frequency selective indoor channels since their conditions fluctuate strongly among different subchannels. To mitigate this effect, and to better exploit channel capacity, in Section 5.2.3.2 we design a second type of allocation mechanism that (a) can assign a different number of subchannels to each link of a stage, and (b) can use a different coding rate on each subchannel. This increases complexity but we expect in turn a better channel utilization.

5.2.3.2 *CSI-based and Traffic-based*

Our second type of allocation mechanism addresses the limitations shown in Figure 5.7b. To this end, it considers both the CSI of all stage links, and the amount of data at each transmitter. The goal of the allocation mechanism is to distribute the available subchannels among the stage links in a way that minimizes the transmission time locally at that stage. While minimizing the end-to-end transmission time would yield better performance, it would require nodes to be aware of global CSI, that is, the CSI on all stages. However, this is often infeasible and undermines the corridor design, which builds on decoupling the operation of the stages as much as possible.

To minimize the stage transmission time, we must estimate how long it takes to transmit a segment on a certain subchannel. While for rateless codes this translates into estimating the approximate number of batches needed for decoding, for discrete-rate codes it means estimating the MCS that each subchannel supports. Either way, we use the CSI as a basis. This approach cancels out the potentially open-loop operation of rateless codes. Still, it also solves one of the main limitations of such codes, namely, determining when the receiver has enough batches to decode a packet. In our case, the challenge is to

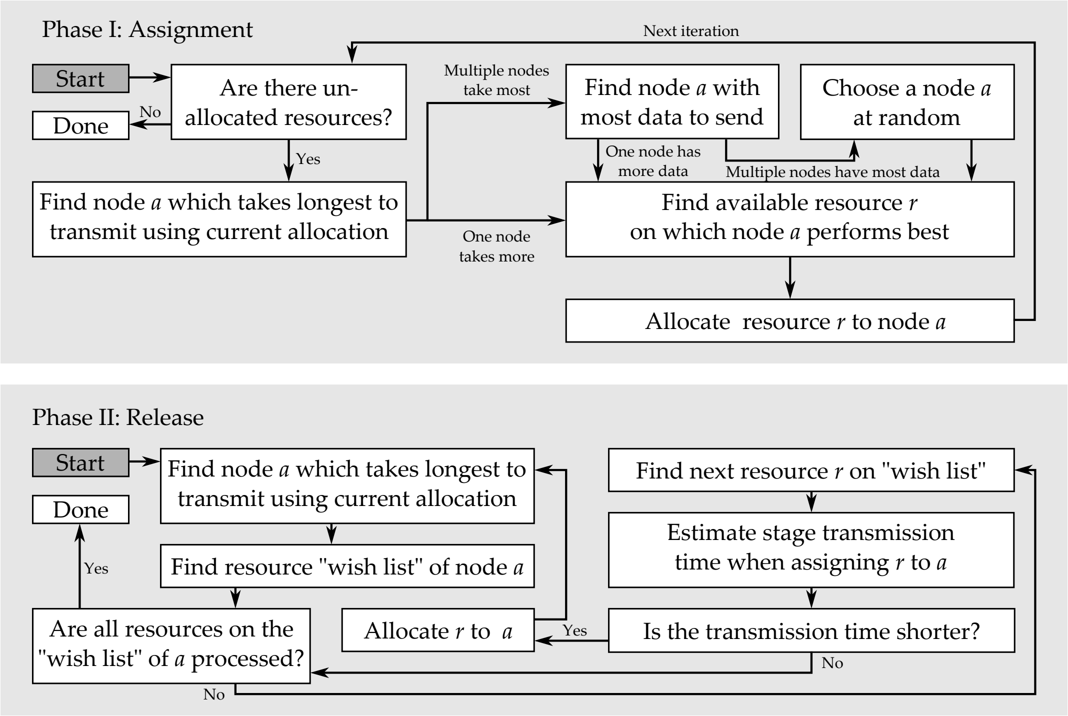

Phase I: Assignment
Start
Done
Are there un-allocated resources?
No
Yes
Find node *a* which takes longest to transmit using current allocation
Multiple nodes take most
One node takes more
Find node *a* with most data to send
Choose a node *a* at random
One node has more data
Multiple nodes have most data
Find available resource *r* on which node *a* performs best
Allocate resource *r* to node *a*
Next iteration
Phase II: Release
Start
Done
Find node *a* which takes longest to transmit using current allocation
Find resource "wish list" of node *a*
Are all resources on the "wish list" of *a* processed?
Yes
No
Allocate *r* to *a*
Yes
Find next resource *r* on "wish list"
Estimate stage transmission time when assigning *r* to *a*
Is the transmission time shorter?
No

Figure 5.8: CSI-based and traffic-based subchannel allocation mechanism.

estimate a priori how many batches receivers need, instead of notifying the transmitters a posteriori as soon as the receivers are ready to decode. However, the key benefit of using rateless codes in our setting is that, even if the estimation is wrong, all batches contribute to packet decoding, in contrast to discrete-rate approaches. In Section 4.2.3.1, we evaluate in practice the accuracy of the aforementioned estimation.

Figure 5.8 summarizes our allocation mechanism. It consists of two independent phases—assignment and release—and follows a greedy iterative approach. In each iteration, the assignment phase assigns one of the available subchannels to one of the transmitter nodes. Essentially, the mechanism selects the node which would take most time to transmit its data using the subchannels currently allocated to it. Next, the mechanism allocates to that node the subchannel on which it requires the least time to transmit a Strider segment, out of the still available subchannels. This process is repeated until no more subchannels are left. In other words, in each iteration, the assignment phase allocates resources to the transmitter which is currently limiting the overall stage throughput. If multiple transmitters limit the throughput, that is, if they all take the same amount of time to transmit their data, the mechanism chooses the transmitter which has most data to transmit. This occurs, for instance, always in the first iteration of the mechanism—no node has any subchannels yet, and thus they all take an infinite amount of time to transmit their data. Further, if multiple nodes limit the throughput with the current allocation and, additionally, all of them have the same amount of data, the mechanism selects a node at random. As a result, the assignment phase involves both the link CSI and the amount of data at each node, since both are needed to determine the time that a node needs to transmit its data on a certain set of subchannels.

The release phase tries to further reduce the stage transmission time by reallocating subchannels—since the assignment phase follows a greedy approach, its result is not necessarily optimal. Basically, we find the node a which limits the throughput and compute its so-called "wish list". The wish list of a node contains the subchannels that the node does not own but which it would like to use because its outgoing links experience good channel conditions on them. The wish list is ordered, that is, the first subchannel on the list is the one on which node a would perform best. Next, we estimate the stage transmission time assuming that node b, who owns the first subchannel on the wish list of a, releases that subchannel to a. If the new stage transmission time is shorter than the initial one, the subchannel is reallocated to a. Otherwise, the release phase repeats the process with the next subchannel on the wish list. The release phase ends when it reaches the end of the wish list of a without reallocating any subchannel. In other words, it ends when no subchannel release reduces the stage transmission time. However, if the release phase reallocates a subchannel, a different node a may limit the throughput of the stage. In that case, the release phase starts over again, which is to say that it finds the wish list of the new node a and steps through it from the beginning.

While the allocation scheme in Section 5.2.3.1 only ensures that each node gets the same amount of subchannels, the above mechanism assigns an amount of *resources* to each node which is proportional to the data that the node needs to transmit. A key advantage is that the scheme takes into account the MCS—or number of batches—that each subchannel supports. For instance, two transmitters may get the same amount of subchannels even if they have a different amount of data to transmit. Basically, if the MCS of the subchannels allocated to the node with less data is lower than the MCS of the subchannels allocated to the node with more data, the amount of allocated resources is still proportional. This allows for a flexible and efficient assignment of subchannels.

5.2.4 *Operation*

Our OFDMA stage mechanism transmits data according to the frame format in Figure 5.9. Similarly to our SL frame format in Figure 4.2, it consists of three phases. The main difference is that the segment transmissions in part (c) of the frame do not use the entire bandwidth available but only a subchannel. Each subchannel may be allocated to a different transmitter, that is, Figure 5.9 shows the overlapping of all transmissions.

5.2.4.1 *CSI Measurement*

In part (a) of our OFDMA frame format, nodes transmit pilot symbols to measure CSI. While Figure 5.9 shows that the transmitters send these pilot symbols to the receivers, the

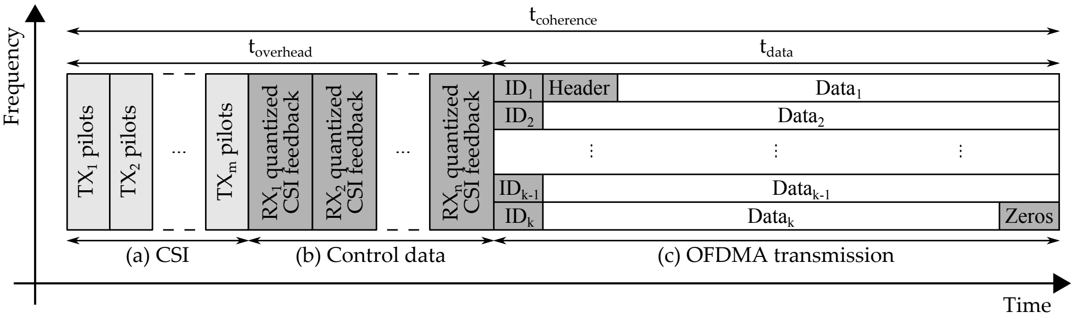


Figure 5.9: OFDMA frame format. Shaded areas indicate control message overhead.

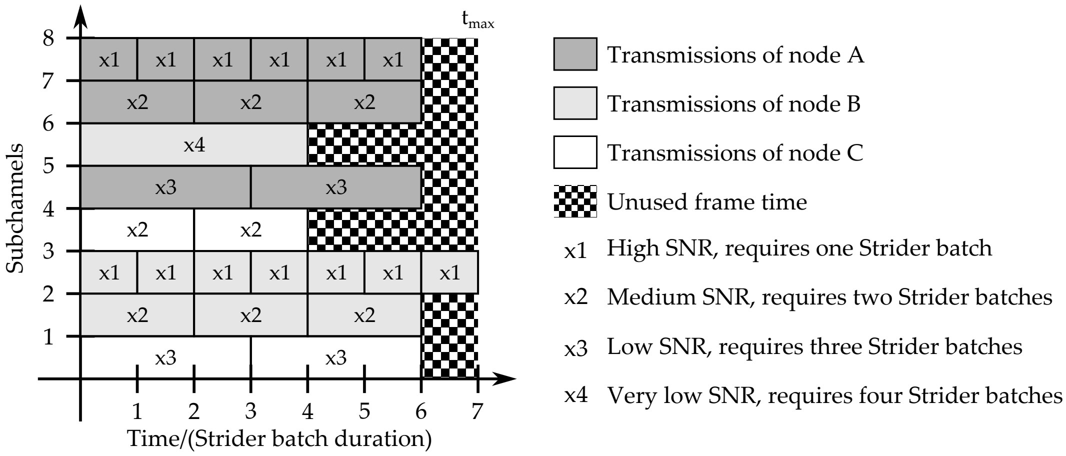

tmax
Subchannels
Time/(Strider batch duration)
1 2 3 4 5 6 7 8
x1 x2 x3 x4
Transmissions of node A
Transmissions of node B
Transmissions of node C
Unused frame time
x1 High SNR, requires one Strider batch
x2 Medium SNR, requires two Strider batches
x3 Low SNR, requires three Strider batches
x4 Very low SNR, requires four Strider batches

Figure 5.10: Scheduling example for an OFDMA frame. Subchannel three limits the throughput.

opposite case is feasible, too. That is, receivers may also transmit the pilot symbols to the transmitters. This is possible only if channels are reciprocal, that is, if their CSI is the same in both directions, which is the case if both directions operate on the same frequency, such as in our testbed. We implement both approaches in Section 5.3 but use mostly the variant in Figure 5.9. The advantage of the inverse measurement is that transmitters have *full* CSI of their *outgoing* links. Still, to compute the subchannel allocation according to Section 5.2.3, they need the CSI of *all* stage links, which they get via *quantized* CSI feedback from the other transmitters in the stage. Moreover, as discussed in Section 5.2.2.1, OFDMA only needs coarse per-subchannel CSI to find beneficial allocations. Hence, having full CSI at the transmitters for some stage links does not provide a significant advantage.

5.2.4.2 *Control Data Transmission*

After measuring CSI, our stage mechanism sends quantized feedback to the transmitters using a codebook-based approach (c.f. Section 5.2.2.1). In particular, nodes share the channel transfer function that they obtain from channel measurements, which includes the channel coefficients of each subchannel. Transmitters use this information to compute the per-subchannel SNR as in Section 5.2.2.2. Once transmitters know the CSI of all $m \cdot n$ links of a stage, each transmitter determines the subchannel allocation *independently*, instead of letting the main node compute it centrally. That is, we exploit that all transmitters receive the CSI feedback of all receivers to save the overhead that the main node would incur if it would have to share the allocation after computing it. Further, the previous stage needs to let the current stage know which data segment it forwards on which subchannel. However, similarly to SL, this control overhead is encoded as an identifier in each segment (c.f. Section 4.2.1). Hence, it does not add overhead to part (b) of the frame.

5.2.4.3 *Payload Data Transmission*

As a result of subchannel allocation based on the above control data, each node owns a set of subchannels for transmitting data. While the allocation mechanism considers the overall amount of data at each node, it does not specify the scheduling of each individual Strider segment. That is, each transmitter must decide how it uses the subchannels assigned to it. The key idea is to find a scheduling which results in roughly the same

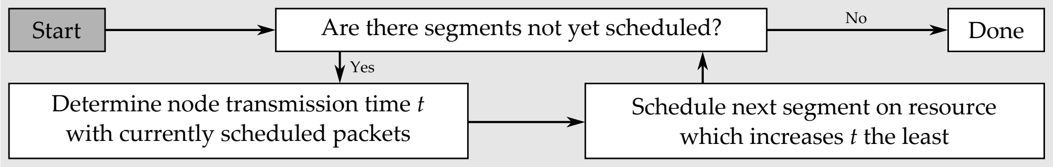

Start
Are there segments not yet scheduled?
No
Done
Yes
Determine node transmission time t with currently scheduled packets
Schedule next segment on resource which increases t the least

Figure 5.11: Scheduling algorithm to assign Strider segments to subchannels.

transmission time on all subchannels. In other words, subchannels with high SNR should transport more Strider segments than subchannels with low SNR. Figure 5.10 gives an example. The goal is to minimize the checkerboard area at the end of the frame, which stands for the time some subchannels are idle while one or more other subchannels are still in use. In Figure 5.10, node B is limiting the throughput since it takes longest to finish transmitting on all of its subchannels, compared to nodes A and C. However, note that no other scheduling at that node would result in a shorter stage transmission time. In particular, if it reschedules one of its packets on subchannel three to any of its other two subchannels, the overall duration increases since the transmission of a packet on those subchannels takes significantly longer than on subchannel three. In contrast, node A achieves perfect scheduling since all of its subchannels finish simultaneously. Still, the slowest node out of all transmitters determines the stage transmission time.

We use the iterative scheduling algorithm depicted in Figure 5.11 to minimize the checkerboard area in Figure 5.10. Each node runs this algorithm locally, that is, we minimize the idle time among the subchannels allocated to a certain node. Jointly scheduling Strider segments on all nodes might result in a shorter stage transmission time but would require nodes to exchange Strider segments, which is typically infeasible due to the incurred overhead. Essentially, in each iteration, our algorithm schedules a Strider segment on the subchannel of the current node on which transmitting the segment incurs the least increase of the node transmission time. The algorithm repeats this process until all segments are scheduled. Once all segments are scheduled, all nodes transmit simultaneously on their allocated subchannels. In case of a transmission error, which is to say if a receiver cannot decode a segment with the estimated number of Strider batches, we send more batches. If we reach the maximum number of batches defined in Table 4.2, we start over, and retransmit the entire frame.

Similarly to the SL case (c.f. Section 4.1.2.3), we split the frame into two or more smaller frames if the frame transmission time is longer than the coherence time. For each frame, we include again CSI measurement and feedback to adjust the allocation of subchannels in case the channel state changes. Moreover, the operation of our OFDMA stage mechanism results in both splitting and joining of data blocks. Essentially, data blocks join when a receiver node receives multiple groups of Strider segments via different subchannels allocated to different links. In turn, data splitting occurs when a transmitter node forwards its data segments via multiple of its outgoing links. Since nodes decide how to schedule segments based on how long the transmission on each subchannel takes, data splits and joins according to the current channel state as it traverses the corridor.

5.2.5 *Corridor Protocol Phases*

The operation of our OFDMA stage mechanism corresponds as follows to the corridor protocol phases. Stage maintenance includes the coarse per-subchannel CSI measurement in frame part (a), and the CSI feedback in frame part (b). Since OFDMA allows for any amount of data at the transmitters and supports virtually any stage shape (c.f. Section 5.2.1), profile matching is not critical—that is, it is compatible to nearly any other stage. Intra-stage coordination consists primarily of allocating disjoint sets of subchannels to each transmitter. However, this occurs in a decentralized manner since all nodes can compute the allocation independently based on the CSI. Hence, the resulting overhead is low. Moreover, intra-stage coordination also includes time synchronization among transmitters according to Section 5.2.2.4. To this end, the main node only transmits a preamble, which again causes a marginal overhead. Similarly to SL, inter-stage coordination is limited to the identifiers in each segment which allow the destination to reassemble the original packet. Finally, the data transmission phase corresponds to the OFDMA transmission in frame part (c), which allows all nodes to transmit simultaneously.

5.3 IMPLEMENTATION

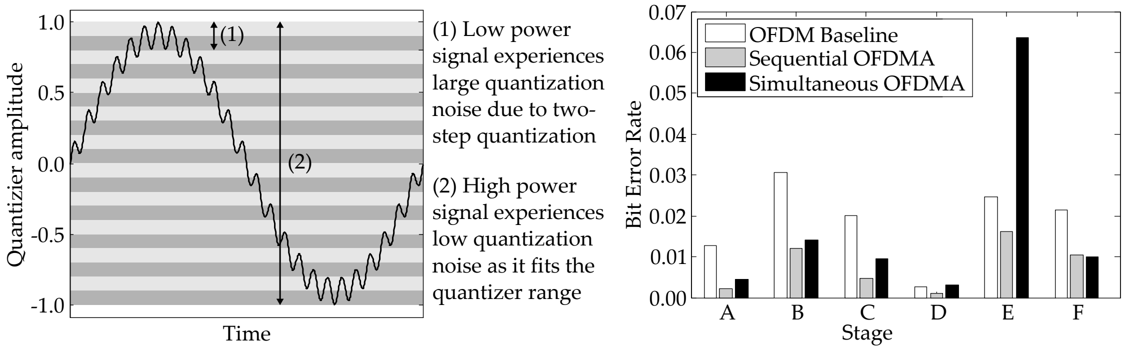
In this section, we explain how we realize our OFDMA stage mechanism at the NET, LNK, and PHY. We adopt some components from our SL implementation in Chapter 4 which are common to all stage mechanisms for Corridor-based Routing, and focus on the particularities of the OFDMA case.

5.3.1 *Network Layer*

At the NET, we divide packets into segments in the same way than for SL in Section 4.2.1. Moreover, we use the corridor construction protocol that we present in Section 3.2.2. The resulting corridor may have stages of any shape, that is, narrowing, constant, and widening. We do not expect the shape of stages to have a clear impact on our results since a narrowing stage does not necessarily imply a bottleneck, and a widening stage does not cause a waste of resources. In particular, the amount of subcarriers at each stage is the same for any m and n. While larger m and n values translate into sharing the same number of subcarriers among more links, it also means that, on average, each link transports less segments. However, the shape of a stage does affect its spatial diversity. For instance, at a stage that narrows down to a single receiver node ($n = 1$), transmitters must transmit to that node even if their link to the receiver exhibits poor channel conditions on all subcarriers. Still, if channel performance is good, a stage with $n = 1$ can transmit data as efficiently as a stage with large m and n. Hence, we allow for any stage shape in the NET implementation of our OFDMA stage mechanism to obtain the average performance of both beneficial and disadvantageous cases.

5.3.2 *Link Layer*

Our OFDMA LNK implementation is similar to the SL LNK case in Section 4.2.2. That is, we deal with internal collisions using our frame format in Figure 5.9, and do not consider external collisions. Moreover, we use Strider as a rateless code with the parameters in Table 4.2. However, in the OFDMA case, the LNK triggers a frame retransmission if any of the subchannels exceeds the maximum number of batches. That is, retransmissions are more costly than in the SL LNK case, since we retransmit the segments on all subchannels.



(1) Low power signal experiences large quantization noise due to two-step quantization

(2) High power signal experiences low quantization noise as it fits the quantizer range

(a) Example sketch of gain misadjustments and resulting quantization noise. Gray bars indicate quantizer steps.

(b) BER variations due to quantization noise on six independent stages using 64-QAM.

Figure 5.12: Impact of quantization noise on OFDMA signals.

Alternatively, the OFDMA LNK could also retransmit the failed segments only using all subchannels simultaneously. This would be equivalent to transmitting most of the data using OFDMA, and retransmitting failed segments using SL. While such an approach is feasible, we do not consider it in our implementation.

5.3.3 *Physical Layer*

At the PHY, our OFDMA stage mechanism deals with the same batch number estimation issue than SL. We refer the reader to Section 4.2.3 for further details. Since OFDMA transmits segments on individual subchannels, it can only use per-subchannel scheduling. In our OFDMA evaluation in Section 5.4 we compare the throughput performance of OFDMA to a baseline using SL with both wideband and per-subchannel scheduling. However, since estimating the number of required batches on a subchannel is challenging in practice, only a baseline using per-subchannel scheduling results in a fair comparison. While in theory such scheduling is beneficial for SL, our results in Section 4.3 show that SL performs better when using wideband scheduling. Still, this issue is not related to Corridor-based Routing but to link adaptation. In the following, we explain how we deal in our implementation with the practical issues that we discuss in Section 5.2.2.

5.3.3.1 *Gain Control*

Gain control in OFDMA is challenging due to the overlapping of multiple signals *in time*. The receiver can decompose the resulting signal using the Fast Fourier Transform (FFT) *after* quantization. Still, before quantization the receiver can only operate on the sum in the time domain. If one signal arrives with more power than others at the receivers, the smaller ones suffer from higher quantization noise, as shown in Figure 5.12a. The key problem is that the receiver can only adjust the largest signal to the input range of the quantizer, while all others are sampled with less accuracy.

We show the impact of this issue in Figure 5.12b, which depicts the BER when using a discrete-rate code on six different stages of width two. We use the BER as a metric of the aforementioned noise impact. In particular, we transmit data using 64-QAM on all subcarriers. The OFDM baseline refers to the BER when only using one link of the stage, while the simultaneous OFDMA case uses all links. Additionally, we show the performance of a *sequential* OFDMA variant, which allows nodes to send in sequence on the subcarriers allocated to them. Hence, signals do not overlap in time, and thus the gain control issue is circumvented. This is not how we envision the scheme to operate once deployed, but it allows us to illustrate the impact of gain control. As shown in Figure 5.12b, the impact of gain misadjustments varies for each stage, since it depends directly on the physical environment surrounding the stage. We observe that simultaneous OFDMA performs worse than sequential OFDMA, specially at stage E. We deduce that, to solve this problem, transmitter nodes would need to adjust their gains to ensure that receivers receive all signals with similar power. This is intrinsic to OFDMA, and orthogonal to the gains achievable by subchannel allocation. Hence, we do not tackle it in our implementation. To obtain the actual allocation gain, for the bulk of our experiments we show the sequential OFDMA results, and compute throughput as if transmissions were simultaneous.

5.3.3.2 *CSI Feedback*

Based on our results in Section 5.2.2.1, in our implementation we use a codebook with 16 entries to provide quantized CSI feedback. To prevent decoding errors, we transmit feedback with a robust encoding, namely, BPSK and a 1/2 error correction code. If the feedback transmission fails, the LNK triggers a retransmission which counts towards the maximum number of retransmissions allowed on a stage. In the bulk of our experiments, we transmit pilot symbols from the transmitters to the receivers, and send feedback from the receivers to the transmitters. While the inverse is also feasible, it does not provide significant advantages in our case (c.f. Section 5.2.2.1).

5.3.3.3 *CFO Correction*

For our practical experiments in Section 5.4.2.1, we perform per-subchannel CFO correction as discussed in Section 5.2.2.3. Our results show that we can correct CFO for our OFDMA stage mechanism in practice. For the remaining experiments in Section 5.4, we use wired synchronization among the WARP boards and focus on the performance gains. To this end, we set one WARP board of our setup in Section 5.4.1 as master node, and share its radio-frequency clock with the remaining nodes using the WARP Clock Module.

5.3.3.4 *Time Synchronization*

Our study in Section 5.2.2.4 shows that node synchronization for simultaneous transmissions within the time constraints of OFDMA is feasible in practice. However, since we use the sequential OFDMA approach described in Section 5.3.3.1 to address gain misadjustments in our experiments, nodes do not transmit simultaneously and thus we do not need to ensure tight inter-group time synchronization. However, we still need intra-group time synchronization, which we provide as described in Section 5.2.2.4, that is, transmitting 802.11 preambles and correlating the received signal with them.

5.3.3.5 *Physical Layer Parameters*

We base our OFDMA stage mechanism on an OFDM PHY with parameters as described in Section 4.2.3.4. The key difference is that transmitters transmit zeros on the subchannels not allocated to them. Additionally, each transmitter adjusts its output signal strength to ensure that the overall transmission power is the same for OFDMA than for the SL stage mechanism that we use as a baseline. In other words, in an OFDMA stage with m transmitters, each transmitter only uses a $1/m$ fraction of the overall available power. This guarantees a fair comparison of both schemes. Moreover, except for our experiments in Section 5.4.2.1, we build our OFDMA stage mechanism on WARP Drive, which is a framework that facilitates wireless multi-hop experiments using WARP directly from the Matlab workspace. For further details on WARP Drive, we refer the reader to Chapter 7.

5.3.3.6 *Throughput Computation*

The processing delays of WARPLab and thus WARP Drive (c.f. Chapter 7) prevent us from measuring throughput directly, since these delays would strongly affect the result. Moreover, the large coherence times in our testbed would lead to CSI measurements at much larger intervals than in a real-world scenario. To circumvent these limitations, we obtain throughput by extrapolating our measurements. We consider an indoor scenario and assume a realistic coherence time for such a setting, namely, 45 ms (c.f. Section 3.2.1.5). For overhead calculations, we consider that CSI measurements at a stage, and thus the resulting node coordination overhead, must take place at least each 45 ms. In other words, while we might only need to coordinate nodes once at the beginning of a transmission due to the stability of the channels in our lab, we do take into account that such coordination would be required more frequently in a real-world environment. We compute the end-to-end throughput at the interface of the network layer to the upper layers, that is, we include the overhead of the NET, LNK, and PHY.

At the source, we pass roughly 125 kBytes of data to the network layer, which divides it into segments of size 1500 bytes that Strider encodes into batches (c.f. Table 4.2). Similarly to related work on rateless codes (c.f. Chapter 2), we abstract from the minimal feedback that a receiver sends to a transmitter when it needs more batches to decode. Instead, we always transmit all 27 batches that Strider generates at the transmitter, and then determine how many the receiver actually needs, similarly to our implementation in Chapter 4. However, transmitting all 27 batches increases the processing time per Strider segment significantly. To keep the experiment runtime reasonable, we transmit one segment via each subchannel and extrapolate the resulting performance to the remaining Strider segments on each subchannel.

5.4 EVALUATION

In this section, we evaluate our OFDMA stage mechanism both in practice and simulation. First, we introduce our evaluation scenarios for both cases in Section 5.4.1. After that, we present and discuss our results in Section 5.4.2.

5.4.1 *Scenarios*

We carry out our experiments in four different scenarios, out of which we use two for simulations and two for practical experiments. Our two simulation scenarios are UD-S and UD-L from Table 3.2. Similarly to our evaluations in Chapters 3 and 4, we use them to investigate the performance of our stage mechanism on random topologies with two different average SNRs. Further, our practical scenarios are (a) the mesh setup

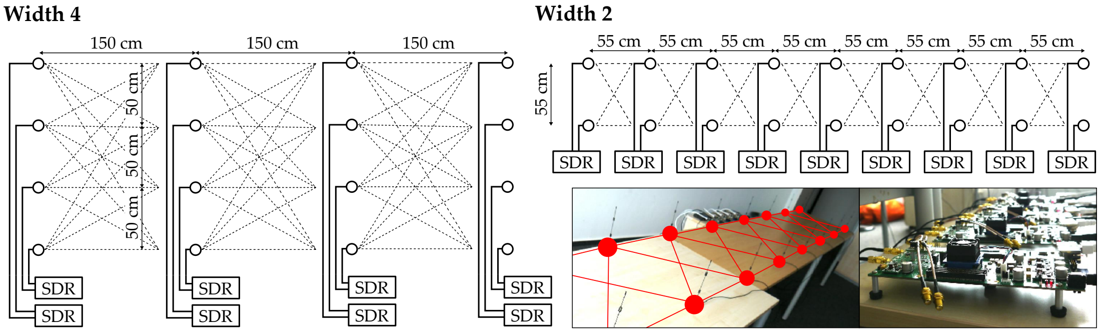


Figure 5.13: SDR testbed with predefined corridors of widths two and four. The pictures on the lower right show the setup for the corridor of width two. Red lines indicate links.

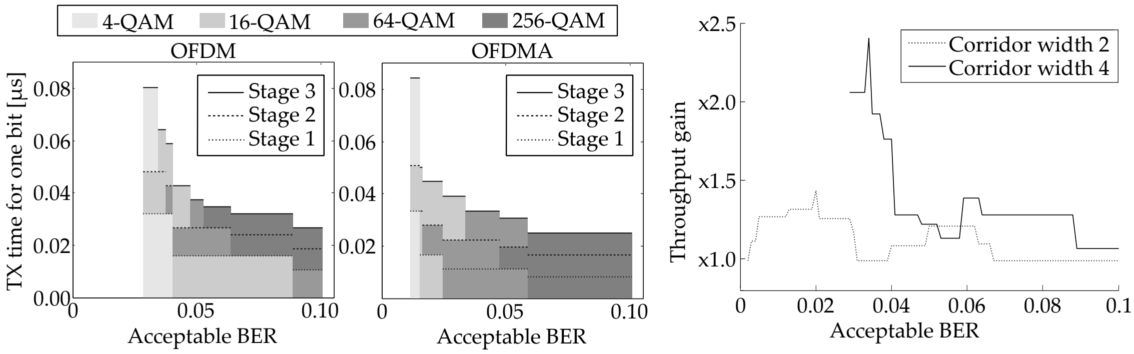
with 20 nodes in Figure 3.6, and (b) two predefined corridors of widths two and four. While testbed (a) builds on top of WARP Drive and thus includes effects at NET, LNK, and PHY, testbed (b) focuses on the PHY only. In particular, we use (b) to study the raw throughput of OFDMA without the impact of the upper layers. However, in contrast to our fundamental gain analysis in Section 5.1, we do not consider the capacity but the actual throughput. That is, we investigate how much data the destination actually can decode, instead of only computing how much data the channel can transport in theory. Figure 5.13 depicts testbed (b), and shows the distance among nodes. Similarly to WARP Drive, we use each radio of each WARP board as if it were an independent node. Moreover, we adjust the transmission power to achieve typical indoor SNRs, that is, 20 dB to 30 dB.

5.4.2 *Results*

In the following, we evaluate OFDMA for WMNs. First, in Section 5.4.2.1 we present the raw throughput gains at the PHY using discrete-rate coding. Our goal in this first part is to investigate the potential gains of OFDMA. After that, in Section 5.4.2.2 we study the throughput gain of the entire system when using our OFDMA stage mechanism. In this case, we use Strider as a rateless code. Our goal in this second part is to analyze how much of the raw gain our OFDMA system actually achieves.

5.4.2.1 *Physical Layer Gain*

In this experiment, we aim at determining in practice the raw throughput gain of OFDMA compared to a mechanism that does not exploit spatial diversity, such as SL. To this end, we use the testbed in Figure 5.13. In particular, we transmit data encoded with different discrete-rate MCSs, and compute the BER for each case. We do not yet consider traffic information to find subchannel allocations but use our algorithm in Section 5.2.3.1 which considers the CSI only. That is, we allocate a fair share of subchannels to each node. Moreover, we use the same modulation on all subchannels—while we use per-subchannel coding in our later system evaluation (c.f. Section 5.4.2.2), this more simple approach allows us to decouple the gains stemming from spatial diversity from the benefits of exploiting frequency-selective fading. As a baseline, we use SL in random link mode. That is, we use the same corridor than OFDMA but choose a random node of each stage as a forwarder. In the following descriptions and figures, we use the terms SL and OFDM interchangeably, since we often refer to the actual PHY that SL uses to highlight



(a) Normalized end-to-end delay in a corridor of width four.

(b) End-to-end throughput gain.

Figure 5.14: Practical OFDMA gain at the PHY.

the difference compared to OFDMA. Our experiment produces the BERs for a range of modulations. However, the actual throughput depends on the amount of bit errors that the channel code in use can correct. Since we later compare the results of this experiment to a rateless code, we analyze the throughput gain for an entire range of "acceptable BERs". In other words, we do not limit our results to the error correction capability of a particular code but consider a range of values. This allows us to obtain broader insights on the gains of OFDMA.

We expect the gain of OFDMA to stem from being able to choose higher modulations than our OFDM baseline for a certain acceptable BER. In Figure 5.14a we depict the end-to-end delay incurred when transporting data through our corridor of width four for any acceptable BER up to 10%. We "normalize" the result dividing the time by the amount of sent data to highlight that higher modulations transport each single bit faster. The gray tones indicate the modulation used in each stage. For example, for OFDM and 10% acceptable BER, the first stage uses 64-QAM, while the second and the third use 256-QAM. The transmission time decreases, as expected, with increasing acceptable BER, since the larger the number of errors the code can correct, the higher the modulations that become possible, and thus the faster the data is transported. For any acceptable BER, OFDMA can use higher modulations and thus requires less time than OFDM to transport data, which directly translates into throughput gain. For small acceptable BER values, both OFDM and OFDMA cannot operate, since the BER is too high even for the lowest modulation scheme in at least one of the stages. This is shown by the left white areas in Figure 5.14a. However, note that OFDMA can already operate at about 0.012 acceptable BER, while OFDM requires at least 0.029 error correction capability to become feasible.

In Figure 5.14b we show the end-to-end throughput gains resulting from our observations in Figure 5.14a. The corridor of width four *doubles* the throughput for certain acceptable BER values, and the corridor of width two achieves up to 1.4× gain. We conclude that gains increase with corridor width—the more links, the higher the probability that OFDMA can allocate links with good channel conditions to each subchannel is. The curve for the corridor of width four starts at 0.029 acceptable BER since OFDM cannot operate for lower values and thus no gain can be computed. That is, gain would be *infinite* in the range from 0.012 to 0.029 acceptable BER. This shows that OFDMA not only provides better performance than SL in terms of throughput but also in terms of robustness due to its fine-granular resource allocation.

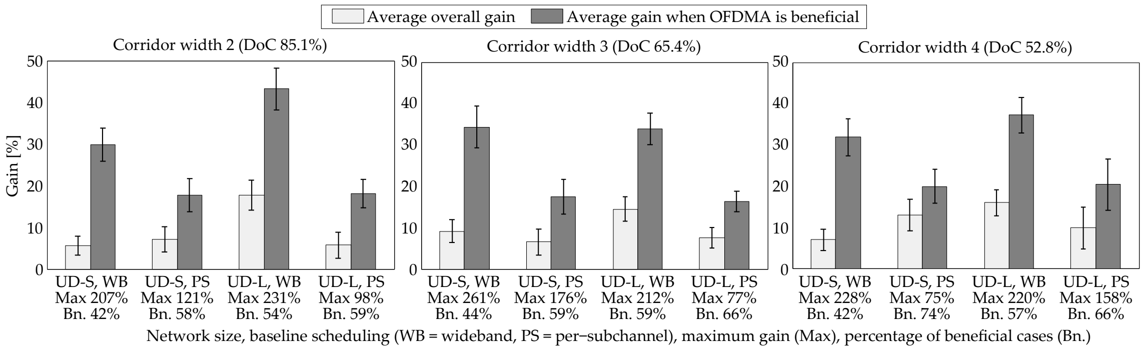


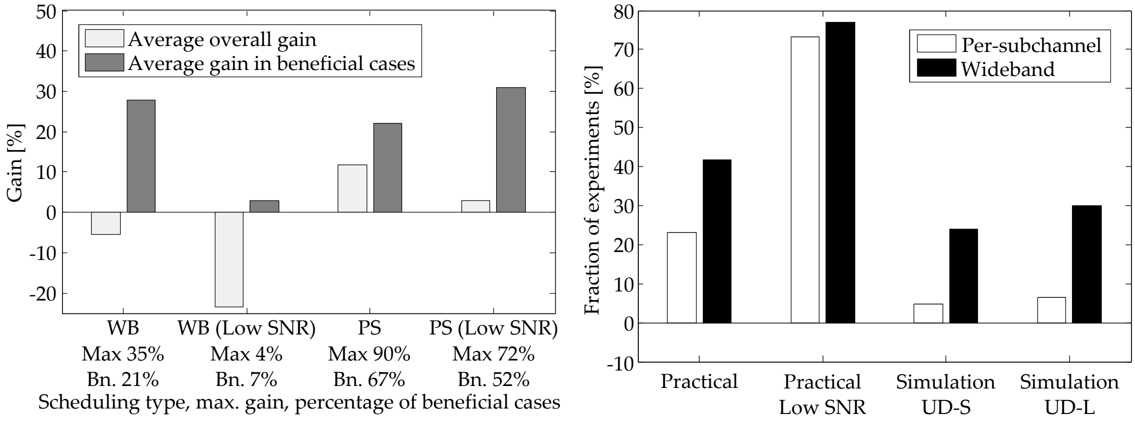
Figure 5.15: Simulation results in our UD-S and UD-L scenarios. We compare the performance of OFDMA to SL both using wideband scheduling and per-subchannel scheduling.

5.4.2.2 *System Gain*

In the following, we evaluate our OFDMA stage mechanism at the PHY, LNK, and NET both in simulation and practice. The former allows us to study random network topologies while the latter gives us insights into real-world effects.

SIMULATIONS. Figure 5.15 shows an overview of our simulation results. We depict the throughput gain of OFDMA compared to our SL baseline, which uses OFDM. Specifically, for SL we consider both wideband scheduling and per-subchannel scheduling (c.f. Section 4.2.3.2). Further, we measure the performance of our system in corridors of intended widths two, three, and four. For each case, we compute the DoC, which reflects the spatial diversity of the corridor. In particular, the DoC is the percentage of the actual number of nodes that form a corridor compared to the canonic number of nodes a corridor of the given length and width should have (c.f. Section 3.2.2.4). For each network size and scheduling type in Figure 5.15, we compute the average throughput gain for (a) all experiment repetitions, and (b) the experiment repetitions where OFDMA is *beneficial*. That is, in some cases it does not pay off to use OFDMA because it results in a *negative* gain compared to SL. This occurs, for instance, when a forwarder with poor outgoing links needs to forward a large amount of Strider segments as a result of local allocation in previous stages. For each scenario, we show the percentage of beneficial cases out of all experiments. Moreover, we also indicate the maximum gain in each case.

Figure 5.15 shows that our OFDMA system achieves up to roughly 2× throughput gain in the best cases, similarly to our results at the PHY in Section 5.4.2.1. Further, we observe that the average gains depend strongly on whether we consider all cases or only the beneficial ones—the former yield gains in the range from 10% to 20%, while the latter achieve up to 40% better throughput than our baseline. In other words, estimating whether OFDMA is beneficial for a certain end-to-end connection is crucial. The results show that the percentage of beneficial cases itself depends on (a) the network scenario, and (b) the type of baseline scheduling. Regarding (a), OFDMA tends to pay off more frequently in UD-L than in UD-S. Figure 5.3b suggests that the reason is related to the average SNR, which is lower for UD-L than for UD-S. While UD-L achieves a lower throughput, relative gains increase as the impact of subchannels with poor channel conditions is larger. In contrast, at high SNRs, most subchannels only require few batches to convey a Strider segment. Thus, using OFDMA to combine the best subchannels provides less gains. As to (b), we



(a) Practical system gains.

(b) Fraction of OFDMA-only transmissions.

Figure 5.16: Practical results. The diagram on the right refers to the fraction of experiments in which our baseline is not operable due to real-world channel conditions.

observe that the number of beneficial cases is larger for per-subchannel scheduling than for the wideband case. The reason is that, in the per-subchannel case, the baseline needs to schedule transmissions similarly to OFDMA (c.f. Section 5.2.4.3). That is, it inevitably wastes transmission time due to some subchannels finishing transmission earlier than others, such as in Figure 5.10. Hence, OFDMA is beneficial in a wider range of situations.

Still, the percentages of beneficial cases in Figure 5.15 show that the SL baseline using per-subchannel scheduling only wastes transmission time occasionally. In all other cases, such scheduling enables SL to exploit the frequency-selectiveness of channels. As a result, the baseline achieves a higher throughput than in the wideband case, which ultimately results in OFDMA gains mostly below 20%, as shown in Figure 5.15. Since OFDMA must use per-subchannel scheduling, this suggests that only a comparison with a baseline also using per-subchannel scheduling would be fair. However, the practical effect we describe in Section 4.2.3.1 results in the opposite behavior. That is, wideband scheduling is more beneficial in practice for our baseline than per-subchannel scheduling. This issue highlights the difference between theory and practice—we discuss this in detail along with our practical results below. Finally, in Figure 5.15 we also show the average DoC for each corridor width. We observe that the DoC becomes smaller for wider corridors. Basically, the wider the intended corridor width, the more difficult it is for the corridor construction protocol to find enough nodes for each stage (c.f. Section 3.2.2.4), forcing the corridor to narrow more often. This explains why wider corridors do not necessarily result in larger gains in Figure 5.15. In contrast, our results in Section 5.4.2.1 exhibit such a relation of the width to the gains since corridors always have 100% DoC in that case.

PRACTICAL RESULTS. Figure 5.16 depicts our practical OFDMA measurements in our mesh SDR testbed (c.f. Figure 3.6). Again, we consider both wideband and per-subchannel scheduling for the baseline scheme. Additionally, we investigate a low SNR case. To this end, we add a node to our testbed which continuously sends noise on all subchannels. As a result, the average SNR at the other nodes drops. In Figure 5.16a we show the OFDMA throughput gain. Strikingly, our OFDMA system yields *negative* gains when we compare it to our SL baseline that uses wideband scheduling. In other words, it performs worse than traditional hop-by-hop forwarding. The reason is the practical effect introduced in Section 4.2.3.1, that is, the challenge of estimating in practice the amount of Strider batches that the receiver needs to decode, and its impact on the per-subchannel frame structure (c.f. Figure 4.5). Figure 5.16a suggests that this effect is critical at low SNRs—most probably, the added noise hinders the estimation of the transmission duration. As a result, the throughput gain drops to −20%. In other words, the baseline performs better than OFDMA even though it is limited to wideband scheduling and thus cannot exploit frequency-selective link adaptation. From our above simulation results, we would expect the baseline to perform even better when switching to per-subchannel scheduling, resulting in a larger negative gain. However, in Figure 5.16 we observe the opposite behavior—a baseline using per-subchannel scheduling yields a *positive* OFDMA gain. Essentially, in this case the baseline also suffers occasional mismatches regarding the estimated number of Strider batches. As a result, SL and OFDMA are on a par with respect to scheduling, revealing the actual spatial diversity gain of OFDMA.

Other than the impact of estimating the transmission duration in practice, we come across similar effects than in simulation. For instance, per-subchannel OFDMA gains tend to become larger at lower SNRs, similarly to the simulation case. Further, we find a significant difference regarding the average gain of all experiments compared to the gain of the cases beneficial for OFDMA. That is, deciding whether OFDMA is helpful to transport data is also crucial in practice. Still, the increase in favorable cases when switching from wideband to per-subchannel scheduling is larger in practice than in simulation. Figure 4.4 explains this behavior. Basically, in practice, OFDMA frames are often larger than baseline frames using wideband scheduling due to the inaccurate transmission time estimation.

Further, we observe a substantial difference between simulation and practice regarding the fraction of cases in which our baseline is not operable but OFDMA successfully delivers data. Most probably, the reason for such cases are deep fades in the channel transfer function, that is, subchannels at which multipath effects result in destructive interference for certain links. Such deep fades are more frequent in practice because our simulated channels are less frequency selective than the channels in the lab. While our baseline must use all subchannels on a link, OFDMA can allocate subchannels suffering a deep fade on a stage link to a different stage link. As a result, OFDMA can evade deep fades effectively, and is thus more robust. Our results in Figure 5.16b show the aforementioned difference clearly. For instance, in the per-subchannel scheduling case, the fraction of experiments for which our baseline is not operable is only 6% in simulation but over 20% in practice. We observe a similar effect in the wideband case, which results in even larger fractions of OFDMA-only transmissions. Deep fades have a stronger impact in the wideband case because they affect all Strider segments instead of only the ones scheduled on the affected subchannels. In other words, a scheme not using OFDMA cannot operate *at all* in a significant number of cases. Hence, even though OFDMA may provide sometimes only moderate throughput gains, it is often essential to transport at least some data, theoretically yielding an *infinite* gain. This corresponds directly to the similar behavior we observe in our experiments at the PHY in Section 5.4.2.1. Finally, Figure 5.16b shows that the fraction of OFDMA-only transmissions increases for lower SNRs both in practice and simulation. Essentially, the lower the average SNR, the higher the probability that at least one subchannel of a given link cannot transport data. Hence, the baseline becomes inoperable more often. OFDMA still can evade such situations, which shows its robustness.

5.4.3 *Discussion*

Our results show that OFDMA for Corridor-based Routing (a) is feasible in practice, and (b) improves performance in terms of throughput and robustness significantly. The overhead is not critical since we achieve large gains while accounting for all control messages our system needs for stage maintenance and coordination. We achieve large throughput gains even when using coarse CSI feedback. That is, gains do not stem from a fine-granular classification of subchannels according to their channel conditions, but just from coarsely identifying subchannels with very poor channel conditions. Further, corridors using OFDMA enable communication at SNR values at which a traditional hop-by-hop forwarding scheme cannot operate at all, which highlights its robustness. While the throughput gain of OFDMA becomes larger for wider corridors, our experiments show that this effect is sometimes limited. The reason is twofold, namely, (a) additional stage links only improve spatial diversity slightly, and (b) finding enough nodes to achieve a large stage width is challenging. Further, we conclude that the benefit of OFDMA strongly depends on how data splits and joins as it traverses the corridor since this might result in bottlenecks. Hence, deciding a priori whether to use OFDMA is crucial.

When moving from OFDMA for WMNs in theory to a practical OFDMA system, we encounter a number of challenges. In particular, our results show that the combination of per-subchannel operation and the need to estimate the duration of transmissions is critical. Using individual subchannels is inevitable since OFDMA builds on top of frequency selective links. The problem is that imprecise estimation of transmission durations results in idle subchannels and thus wasted channel time. In other words, it hinders the scheduling algorithm to minimize the checkerbox area in Figure 5.10. This issue raises for both discrete-rate and rateless approaches. The former causes idle subchannels due to retransmissions, while the latter results in such behavior due to additional batches. Our results show that improving the accuracy of transmission duration estimations to sort this issue out is challenging. The key problem is that we need to decide *a priori* how to schedule Strider segments. A future direction to tackle this problem is to find a method to decide on the scheduling *a posteriori*—for instance, using an approach inspired by rateless codes. Alternatively, in a scenario with multiple corridors, a stage could signal idle subchannels to nearby corridors that could use them. However, finding lightweight signaling is essential for such an approach.

5.5 SUMMARY

In this chapter, we introduce a stage mechanism based on OFDMA. Essentially, OFDMA enables all links of a corridor stage to share the available OFDM subchannels at the PHY. As a result, the PHY can (a) use the best links for each subchannel to achieve throughput gains, and (b) avoid using subchannels on links that exhibit poor subchannel conditions to provide robustness. As a first step, we determine the analytical capacity gain that we can achieve on a corridor stage using OFDMA, and compare it to simulation and practical results. The capacity allows us to get insights into the fundamental improvement in terms of channel conditions when using OFDMA. Our results in theory, simulation, and practice match. In particular, we show that the capacity improves by roughly 30% at typical indoor SNRs for a stage of width four. Next, we build on this preliminary but motivating result to design our OFDMA stage mechanism as a component of our Corridor-based Routing system that we introduce in Chapter 3. The key difference to the above capacity study is that we aim at measuring the throughput gain of OFDMA. To this end, we need to take into account not just the PHY but also the LNK and the NET. Moreover, we address practical issues, and show the feasibility of our solutions based on testbed experiments. For instance, we show that we can achieve tight time and frequency synchronization for OFDMA using preamble sequences as well as pilot symbols. Further, we show that coarse CSI feedback is sufficient to find suitable subchannel allocations for a stage. We discuss two types of subchannel allocation mechanisms, namely, a traffic-agnostic and a traffic-aware scheme. The traffic-agnostic scheme decides on subchannel allocations using CSI only. As a result, it allocates a fair share of resources to each transmitter, which may cause bottlenecks if some nodes have more data to transmit than others. In contrast, our traffic-aware scheme distributes resources according to the amount of data at each node. After subchannel allocation, transmitters use a scheduling mechanism to decide how many data segments to send on each of their subchannels in order to minimize the transmission time. To realize our OFDMA stage mechanism, we use a frame format consisting of CSI measurement, CSI feedback, and payload data transmission. The resulting stage profile has loose requirements, and is thus compatible to most other PHYs.

We implement our system both in simulation and practice. Our LNK and NET are similar to the SL case since they implement functionalities which are common to any stage mechanism. We abstract from some PHY issues but show that we can deal with them in practice. Finally, we evaluate and discuss our OFDMA stage mechanism. In particular, we study the throughput performance (a) at the PHY only, and (b) for the entire system. For (a), we show that OFDMA corridors achieve up to $2\times$ throughput gain. Further, OFDMA becomes operable at lower SNRs than SL, that is, OFDMA is significantly more robust. In case (b), we achieve similar maximum gains. However, we observe that correctly estimating the transmission duration of a packet on an OFDMA subchannel is essential for beneficial scheduling at the LNK, and thus for fully exploiting the gain of OFDMA. Still, this does not affect the aforementioned robustness of OFDMA since it can still avoid subchannels affected by deep fades efficiently. Overall, we conclude that our OFDMA stage mechanism provides up to roughly $2\times$ throughput gain in the best case, and $1.3\times$ on average in the cases for which the gain is positive. Moreover, we find that OFDMA gains strongly depend on CSI and traffic conditions. Hence, identifying disadvantageous cases in advance is crucial.

6

INTERFERENCE ALIGNMENT STAGE MECHANISM

In this chapter, we introduce a stage mechanism based on IA. In particular, we build on IA in the frequency domain (c.f. Chapter 2). Essentially, we allow multiple transmitters to transmit simultaneously on the same resources in a way such that the resulting interference cancels out at the intended receiver node. That is, while the intended receiver can decode the data, all other receivers observe a collision. To this end, transmitters precode data according to the CSI. Figure 6.1 shows an intuition of this concept. The three observers A, B, and C stand for the receiving nodes, while the three objects represent the transmitters. Instead of transmitting data, they reflect visible light at the observers. Assuming that the observers wish to take a picture of two different pairs of objects, they would have to place at most two objects in the scene, and take turns at taking pictures. This is analogous to how nodes use Time Division Multiple Access (TDMA) to communicate without interference. However, if the observers position themselves at suitable locations, they can actually have all objects in the scene simultaneously, thus reducing the time that they need to take the pictures. For a communication scenario, this translates into nodes transmitting more data per time unit, resulting in a higher throughput. Concretely, observer A in Figure 6.1 only sees the square and the star since the interfering circle is in the shadow area of the square. In other words, the interference *is aligned*. Similarly, C only sees the square and the circle. Thus, their pictures only contain two objects. In contrast, at observer B the interference does not align. As a result, she obtains a picture with three objects, that is, a picture which does not meet the requirement of depicting only a single pair of objects. The key idea behind our IA example is that all observers are looking at the same scene but from different angles, hence taking a different picture—either a different pair of objects or a collision, depending on their location.

As a result of the above IA concept, the characteristics of a stage mechanism using IA at the PHY differ significantly from the stage mechanisms in earlier chapters. Basically, while SL and OFDMA avoid collisions using time- and frequency division, respectively, IA explicitly allows for collisions. Most importantly, IA achieves a DOF larger than one, while SL and OFDMA are limited to one. In turn for this higher throughput, only the intended receivers can decode data in IA. In particular, we build on a variant of frequency IA which considers three pairs of transmitters and receivers. While the first pair can

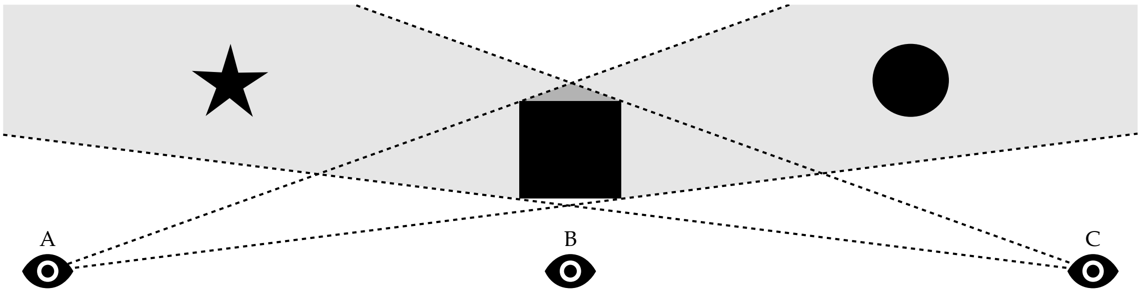


Figure 6.1: IA intuition. The eyes represent observers that look at the scene from different angles.

exchange two symbols in one time unit, the second and the third pair can only exchange one. That is, by design one receiver receives *double* the amount of data than the others, which significantly impacts how data flows through a corridor. In addition to the uneven link configuration, the channel conditions of *all* stage links affect the throughput that a certain pair of communicating nodes can achieve. Hence, the operation of an IA stage mechanism is significantly more complex than in the case of SL and OFDMA.

Further, while theoretical work studies IA in the frequency domain (c.f. Chapter 2), no practical implementations exist. In the following, we introduce the first system that (a) overcomes the practical limitations of frequency IA, (b) discusses its performance based on testbed measurements using SDRs, and (c) shows that frequency IA can perform closely to the theoretical boundaries when using our design. Concretely, we present our IA stage mechanism in two steps. First, in Section 6.1 we introduce a design that enables practical frequency IA in a single-hop scenario. After this first step, we extend IA to the multi-hop case in Section 6.2. Finally, we implement and evaluate our IA stage mechanism in Sections 6.3 and 6.4, respectively.

6.1 ENABLING FREQUENCY IA

In this section, we study practical IA in the frequency domain for a single-hop scenario, that is, within a single stage. The key limitation of frequency IA is that it requires a high SNR to operate reliably. Otherwise, IA is actually detrimental—it may cause *negative* gains since the intended receivers cannot decode data, which means that the overall stage throughput is *lower* than with traditional PHYs such as SL. To address this issue, we propose multiple high-granularity selection algorithms to *choose* which transmitters shall send data to which receivers on which subcarriers using IA. Moreover, if channel quality makes IA infeasible, we allow nodes to switch per subcarrier to more robust mechanisms such as OFDM. In other words, we provide mechanisms to adapt IA to the current channel state. This allows us to achieve low BERs, which makes IA feasible altogether and translates into a higher throughput. To this end, we first explain the operation of frequency IA in Section 6.1.1. After that, we describe the resource combinations among which frequency IA can choose in Section 6.1.2, and introduce resource selection algorithms in Section 6.1.3. Our insights from this section serve as a basis for our IA stage mechanism in Section 6.2.

6.1.1 *Operation of IA in the Frequency Domain*

In the following, we summarize how frequency IA works both from a theoretical and a practical perspective. The former explains how signals align in frequency, while the latter introduces a frame format to support such operation.

6.1.1.1 *Background*

We consider the case of $K = 3$ pairs of transmitters and receivers, which translates directly to a stage of width three. Moreover, we use a symbol extension over $N_P = 3$ subcarriers, that is, we align interference over three subcarriers (c.f. Section 2.3.2). For the general case, which uses a symbol extension of $N_P = 2c + 1$ subcarriers with $c \in \mathbb{N}$, we refer the reader to Reference [20]. For our IA stage mechanism, we build on the case $N_P = 3$ since larger symbol extensions require an even higher SNR, and only provide marginal gains. In

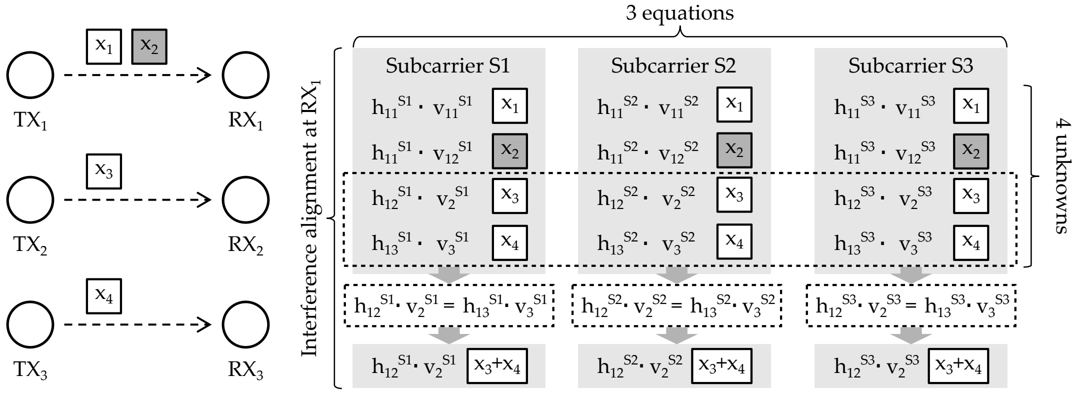

3 equations
TX1
RX1
TX2
RX2
TX3
RX3
x1
x2
x3
x4
Interference alignment at RX1
Subcarrier S1
Subcarrier S2
Subcarrier S3
4 unknowns
h12S1· v2S1 = h13S1· v3S1
h12S2· v2S2 = h13S2· v3S2
h12S3· v2S3 = h13S3· v3S3
x3+x4

Figure 6.2: Frequency IA example for the case of RX$_1$. The identities in the dashed boxes are the alignment conditions for RX$_1$. The lower gray boxes represent the aligned interference.

particular, transmitter 1 encodes two packets, x_1 and x_2, represented by a 2×1 column vector $\mathbf{x}_S$, with precoding matrix $\mathbf{V}_1 \in \mathbb{C}^{3\times2}$. That is $\mathbf{V}_1$ consists of two precoding vectors having the size of the symbol extension, one for each packet in $\mathbf{x}_S$. Transmitters 2 and 3 encode only one packet each, denoted as x_3 and x_4 with precoding matrices $\mathbf{V}_2$, $\mathbf{V}_3 \in \mathbb{C}^{3\times1}$, respectively. In other words, each transmitter sends the same data over all three subcarriers, but precoded with a different value. Transmitter 1 sends a linear combination of x_1 and x_2, that is, $x_1 V_{1,1} + x_2 V_{1,2}$, where $V_{1,k}$ is the k-th column of $\mathbf{V}_1$.

At the receiver side, each node receives the superposition of all transmitted signals on each of the three subcarriers. As a result, each receiver has three copies of the same overlapped data but affected by different channel coefficients, which is to say that each receiver has a system of three linearly independent equations. Still, there are *four* unknowns, one for each packet. Specifically, we can write the received signal $\mathbf{r}_n$ at the n-th receiver as in Equation 6.1.

$$\mathbf{r}_n = \mathbf{H}_{n1}\mathbf{V}_1\mathbf{x}_S + \mathbf{H}_{n2}\mathbf{V}_2 x_3 + \mathbf{H}_{n3}\mathbf{V}_3 x_4 + \mathbf{z}_n \tag{6.1}$$

In Equation 6.1, $\mathbf{H}_{nm}$ is the 3×3 matrix $\mathbf{H}_{nm} = \mathrm{diag}(h_{nm}[1], h_{nm}[2], h_{nm}[3])$ $m, n \in [1, 2, 3]$, where $h_{nm}[i]$ denotes the complex channel coefficient of the channel from transmitter m to receiver n on subcarrier i, and $\mathbf{z}_n \sim \mathcal{CN}(0, 1)$ denotes AWGN noise at receiver n. To solve the aforementioned linear system, two of the four packets must *align*. As a result, they occupy a common signal space, that is, we can combine them to one unknown representing interference, yielding a system of three equations and three unknowns. We solve this system, and discard the solution which represents the sum of the two aligned interfering signals. Still, the other two solutions are packets that we can decode. Figure 6.2 summarizes this process for the case of the first receiver. Basically, RX$_1$ in Figure 6.2 receives the superposition of packets x_1, x_2, x_3, and x_4 three times, namely, once on each of the subcarriers of the symbol extension. The upper grayer boxes represent the resulting linear system of equations—while the packets are the unknowns, the channel coefficients h and the precoding vectors v are the coefficients of the system. We choose the precoding vectors such that the received signals at RX$_1$ fulfill the alignment conditions in the dashed boxes in Figure 6.2. As a result, the lower two unknowns x_3 and x_4 *merge* to a single unknown $x_3 + x_4$. Hence, RX$_1$ can solve the system and obtain x_1, x_2, and $x_3 + x_4$. While RX$_1$ cannot decode x_3 and x_4 individually, it is anyhow not interested in them since they are interference for it. An analogous approach allows RX_2 and RX_3 to decode x_3 and x_4, respectively. In these cases, the alignment conditions force that a different pair of unknowns merges. For instance, at RX_2, x_1 and x_4 merge to $x_1 + x_4$, allowing RX_2 to decode x_2 and x_3. This matches our intuition in Figure 6.1. The scene contains all four packets but each receiver observes them from a different angle. The viewing angle is equivalent to the rotation due to the precoding vectors, which ensures that at least two packets fall in the same dimension and merge. Similarly to Figure 6.1, a *different* set of packets, or shapes, aligns at each receiver, or observer, respectively.

The key to frequency IA are the precoding matrices. Basically, the transmitters use CSI feedback to compute them such that the alignment conditions at each receiver hold. In particular, we use a design for the precoding matrices [20] which realizes the transmission pattern in Figure 6.2, that is, it allows us to decode x_1 and x_2 at receiver 1, x_3 at receiver 2 and x_4 at receiver 3. Essentially, we define the precoding matrices as $\mathbf{V}_1 = [\mathbf{w} \quad \mathbf{T}_1\mathbf{w}]$, $\mathbf{V}_2 = \mathbf{T}_2\mathbf{w}$ and $\mathbf{V}_3 = \mathbf{T}_3\mathbf{T}_1\mathbf{w}$, where vectors $\mathbf{T}_1$, $\mathbf{T}_2$, and $\mathbf{T}_3$ are as follows.

$$\mathbf{T}_1 = \mathbf{H}_{12}(\mathbf{H}_{21})^{-1}\mathbf{H}_{23}(\mathbf{H}_{32})^{-1}\mathbf{H}_{31}(\mathbf{H}_{13})^{-1}, \tag{6.2}$$

$$\mathbf{T}_2 = (\mathbf{H}_{32})^{-1}\mathbf{H}_{31}, \tag{6.3}$$

$$\mathbf{T}_3 = (\mathbf{H}_{23})^{-1}\mathbf{H}_{21}, \tag{6.4}$$

Vector $\mathbf{w}$ in the above definitions is the 3×1 initial precoding vector, which we can choose freely. That is, the constraints resulting from the alignment conditions do not fully define the precoding matrices, and thus we can define $\mathbf{w}$ arbitrarily. We follow the approach in Reference [20], which suggests defining $\mathbf{w}$ as an all-one vector. At receiver n, we zero-force the interference using the 3×3 zero-forcing filter matrix $\mathbf{A}_n = (\mathbf{B}_n)^{-1}$, which we design according to the precoding matrices. More precisely, the k-th column $B_{n,k}$ of matrix $\mathbf{B}_n$ is as defined in Equation 6.5.

$$B_{n,k} = \begin{cases} \mathbf{H}_{n1}V_{1,k} & k = 1,2 \\ \mathbf{H}_{n2}V_2 & k = 3 \ \cup \ n = 1,2 \\ \mathbf{H}_{33}V_3 & k = 3 \ \cup \ n = 3 \end{cases} \tag{6.5}$$

Essentially, B_n contains the coefficients of the aforementioned linear system of equations at receiver n, that is, with aligned interference. These coefficients are the multiplication of the channel coefficients with the precoding vectors, as shown in Figure 6.2. To solve the linear system, we invert B_n and multiply the result with the received data. The outcome is the zero-forced signal $\mathbf{s}_n$, which contains the original packets x_1, x_2, x_3 and x_4. Since each receiver can decode two out of the four packets, as a side-effect receivers 2 and 3 get one packet more than needed, that is, x_2 at receiver 2 and x_1 at receiver 3.

6.1.1.2 *Frame Format*

To realize IA at the PHY, we design a frame format which supports simultaneous transmissions on the same time and frequency resources. As shown in Figure 6.3, we follow a design similar to SL and OFDMA for parts (a) and (b) of the frame. That is, we first transmit pilots to measure the CSI at the receivers in part (a), and then send quantized feedback to the transmitters in part (b). Although the frequency IA variant we present in

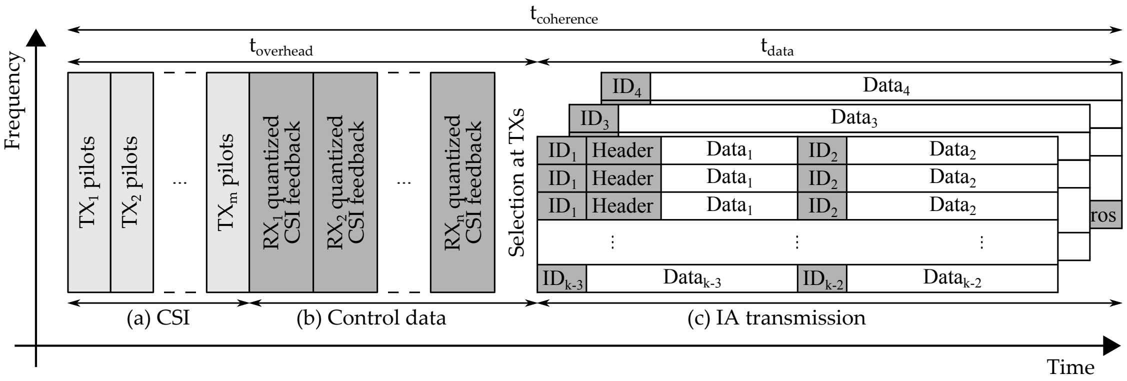


Figure 6.3: IA frame format. Shaded areas indicate control message overhead.

Section 6.1.1.1 considers three node pairs, we measure all links in a stage even if $m > 3$ or $n > 3$ since we *choose* which of the available resources to use. That is, we need the CSI of all links to decide which ones to use. In Figure 6.3, we show this selection of resources as the last step in part (b). The selection does not require any communication overhead but may incur a significant processing time due to the large amount of possible combinations, as we discuss in Section 6.1.2. Finally, we transmit the actual payload data in part (c). In contrast to SL and OFDMA, our IA frame format explicitly allows three signals to overlap at the same time and frequency. The foremost data frame in part (c) corresponds to the IA link which transports twice as much data per time unit than the remaining links. Hence, in Figure 6.3 we fit two packet segments in this frame, while the two data frames in the back only have capacity to transmit one packet segment.

Further, each data frame contains each segment three times, that is, once on each of the three subcarriers that we use as a symbol extension. In other words, the first three rows of each data frame correspond to the three gray columns in our example in Figure 6.2. Concretely, that example represents the reception of the foremost data frame in Figure 6.3 since it shows the decoding of the IA link with double capacity. Still, the receiver also receives the two frames in the back since they overlap in time and frequency. For the segments $k-3$ and $k-2$ in Figure 6.3, we only represent the last of the three subcarriers for simplicity. The hindmost data frame includes the last segment, which may contain zero padding to ensure that the data that the NET receives from the upper layers is a multiple of the segment size (c.f. Section 4.2.1).

6.1.2 *Resource Diversity*

In the following, we explain the resource combinations among which nodes can choose in our single-hop stage scenario when using IA. We assume m transmitters and n receivers which are fully connected, that is, which form a stage. Since we do not yet consider the flow of data through a multi-hop corridor, we further assume that all transmitters have data to send to all receivers. As a result, the impact of which of the transmitters can use the IA link with double capacity is low, since they all have enough data to transmit in order to fully exploit it. This differs for the multi-hop corridor because, in that case, nodes can only transmit the data which they received in the previous stage. Hence, the allocation of the double capacity link does have a significant impact. We discuss this issue further in detail in Section 6.2.2. For our analysis of the single-hop case in this section, we assume that the first transmitter always gets the double capacity link.

Moreover, we build IA on top of OFDM. That is, all nodes transmit data on up to N subcarriers and use the same channel. While we precode data over a symbol extension comprising multiple subcarriers, the underlying transmission technique is OFDM. In other words, compared to other schemes based on OFDM such as SL, the main differences when using IA are (a) multiple transmitters transmit simultaneously causing collisions, and (b) transmitters precode data to align interference. However, the latter only affects the phase and amplitude of the symbols we transmit. The IA PHY transmits these data symbols similarly to any other OFDM system. To distinguish our SL baseline from IA, we often refer to it as "plain OFDM" since it transmits data without any precoding nor collisions.

6.1.2.1 *Combinations*

We consider IA in the frequency domain as described in Section 6.1.1.1, that is, an IA scheme which allows three transmitters to send *four* streams to three receivers using only *three* OFDM subcarriers. To deploy such a scheme in a network, we must arrange nodes in stages with at least six nodes, namely, three transmitters and three receivers. The throughput performance of the scheme is directly related to the channel quality of the links in each group. Hence, if a stage has more than three nodes on each side, arranging these nodes in groups of six which are beneficial for IA is key to achieve the theoretical 33% gain. Moreover, since we implement IA in the frequency domain, we can also group nodes per subcarrier. That is, different groups of six nodes within a stage may share the available bandwidth similarly to an OFDMA system by allocating different groups of three subcarriers. Hence, the number of possible resource combinations is significant. For ease of exposition, throughout this chapter we use the following terms to refer to the different types of grouping that we perform to improve the performance of IA.

- NODE GROUP. We define as *groups* the aforementioned 3×3 sets of three transmitters and three receivers, which is the minimum number of nodes for frequency IA.
- SUBCARRIER TRIPLET. We call a subcarrier *triplet* the sets of three subcarriers that we use for IA. If $N > 3$ subcarriers, we can use IA in parallel on multiple triplets.
- NODE PAIR. We call a node *pair* two nodes that exchange a data stream using IA within a node group. That is, each node group features three node pairs.
- NODE SUBSET. We define *subsets* as portions of the network which form an $m \times n$ stage. Since subsets can have any size, they may contain multiple 3×3 node groups.

The selection of resources for Corridor-based Routing occurs in two steps. First, the corridor construction protocol defines the node subsets as a result of forming fully-connected stages. Essentially, stages directly become node subsets for IA. Second, our IA stage mechanism can select within each node subset a certain node group which uses a specific subcarrier triplet to exchange data among three node pairs. Basically, while node subsets remain constant for longer periods of time since they are part of the NET, the PHY may adapt the groups, triplets, and pairs in use more frequently based on the current CSI. We call the selection of a subset, group, pair, and triplet a resource *combination*. Hence, IA can optimize throughput performance by choosing resource combinations according to the following four selection criteria.

- Node Selection. IA can allocate different subcarrier triplets to different 3×3 groups. Thus, multiple groups can use different parts of the spectrum simultaneously.
- Subcarrier Selection. Nodes can choose which triplets to use. In particular, IA works best when the channel coefficients of a triplet are linearly independent.
- Stream Selection. Within a node group using a triplet, we can choose which transmitter sends to which receiver. There are six possible pairings (Figure 6.4).
- Mechanism Selection. Transmitters can decide to resort on certain subcarriers to plain OFDM if IA cannot perform well with the given CSI.

Essentially, our IA stage mechanism estimates performance for each possible resource combination and chooses the best one. To this end, we use CSI feedback, that is, we do not need to test each combination using actual data transmissions. Our experiments in Section 6.1.5.4 show that this selection is crucial to operate IA since a disadvantageous resource combination results in worse performance than plain OFDM, and may even prevent communication entirely. The number of possible resource combinations is very large. To make this exhaustive search tractable and hence allow for real-time communication, we use heuristics that reduce the complexity of the resulting combinatorial problem.

6.1.2.2 *Heuristics for Scalable Selection*

In this section, we determine how much each of the selection criteria in Section 6.1.2.1 contributes to the overall number of possible combinations. For the critical cases, we introduce heuristics based on characteristics of the wireless medium that reduce this number, and thus the search space for beneficial combinations, significantly.

SUBCARRIER SELECTION. The number of possible triplets of three subcarriers is given by the binomial coefficient $\binom{N}{3}$. While 802.11g only has 48 subcarriers, which leads to 17296 possible triplets, 802.11n or 802.11ac feature 112 and 484 subcarriers respectively, what causes the number of triplets to exceed 18 million. Estimating IA performance for each triplet to find the best one is infeasible. We propose exploiting the nature of wireless channels to overcome this problem. While subcarriers behave similar if they are close to each other in the frequency domain, they are likely uncorrelated when they are apart far enough. This effect is known as the coherence bandwidth B_C. To ensure that subcarriers are uncorrelated with high probability, we only need to allocate them sufficiently far apart, that is, further than B_C. Assuming an indoor propagation delay spread of at most 700 ns, $B_C \approx \frac{1}{700\text{ns}} = 1.42$ MHz (c.f. Section 4.2.3.4). In the 802.11 standards, subcarrier spacing is 312.5 kHz. Hence, there must be at least $\lceil \frac{1.42\text{MHz}}{312.5\text{kHz}} \rceil = 5$ subcarriers in between subcarriers of the same triplet.

Our heuristic to keep the number of possible triplet combinations low is to define a *fixed* allocation of triplets which maximizes the spacing between subcarriers of the same

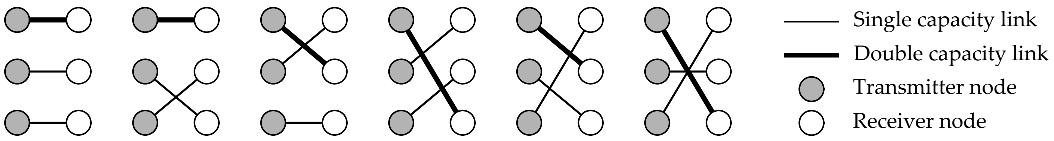


Figure 6.4: All possible node pairs within a 3×3 node group.

triplet. More precisely, subcarrier i is always in the same triplet as subcarriers $\frac{N}{3}+i$ and $\frac{2\cdot N}{3}+i$, for $i \in \left[1 \dots \frac{N}{3}-1\right]$. For 802.11g this means that subcarriers of a combination are separated by 5 MHz $> B_C$. For 802.11n/ac, the separation is even larger, as there are more subcarriers. This does *not* ignore frequency selective fading, since we still do subcarrier-wise node, stream, and mechanism selection. Subcarrier selection just adds a dimension but incurs the aforementioned large number of combinations, which hinders the transmitters from finding a suitable combination before the CSI is outdated.

NODE SELECTION. In general, IA must select 3×3 node groups out of all the nodes in the WMN. This would lead to an unmanageable number of possible node groups for increasing network sizes. However, Corridor-based Routing inherently solves this issue as it limits the selection of node groups to the nodes of a stage, which becomes the node subset of a combination. Hence, our heuristic in this case is given by a fundamental characteristic of Corridor-based Routing, namely, splitting the WMN into smaller subsets and serve each subset one at a time. This reduces the combinatorial problem, as we only need to search within each subset the 3×3 group that optimizes performance. In Chapter 3, we present a corridor construction protocol which builds fully-connected stages based on neighborhood information, and hence implicitly provides node subsets for IA. In some cases, the protocol can choose among multiple candidate nodes to build a stage. By default, it selects the nodes which feature the highest average SNR on their links. However, the protocol allows also for alternative criteria. For instance, if it knows in advance the stage mechanism that the corridor shall use, it can select the nodes which best fit the characteristics of the PHY. In the following, we present such a selection mechanism to form node subsets which are beneficial for IA.

IA works best if each receiver in a node group has (a) homogeneous, and (b) high SNRs to each transmitter in the group [95]. To select nodes for a stage of size $m \times n$, we propose a greedy heuristic which provides suitable subsets. Broadly speaking, goal (a) requires the transmitters to be located at similar distances to all n receivers, and goal (b) requires transmitters to be close to the receivers. During the construction of a stage as in Section 3.2.2.1, the previous stage already defines the receiver nodes (c.f. Figure 3.4). That is, we can only choose the transmitter nodes because we build the corridor stage by stage from the destination towards the source. Hence, we divide the nodes which are available as potential transmitters into sets of m nodes each. We cluster transmitters which are close to each other, since then the equidistant point to all transmitters is also close to them. Next, we calculate at each receiver the average SNR to each transmitter in each set of transmitters. Finally, we select the set of transmitters to which all receivers experience the most similar SNR values. As a result, the corridor construction protocol builds stages where all nodes are close to each other, and at roughly similar distances. While the corridor construction protocol only allows us to select the transmitters but not the receivers, the transmitters we choose for one stage become the receivers of the next stage that the protocol builds. Hence, the receivers implicitly fulfill goals (a) and (b). Moreover, we do not require CSI per subcarrier in terms of phase and amplitude to choose the transmitters, but only per node SNR values, which causes much less overhead during corridor construction. While also combinatorial, our heuristic does not require per subcarrier calculations, which cuts down complexity significantly.

The larger the subset size $m \times n$, the higher the probability of finding a good resource combination within each subset. Still, we expect combinations to improve only marginally from a certain subset size on, since more nodes only improve diversity slightly. Also, the number of possible node groups increases quickly for larger subsets. In particular, we investigate subsets of sizes 3×3 and 4×4. The former allow for no node selection since the subset is of the same size than a node group, whereas the latter allow for 16 groups. A 5×5 subset leads to $\binom{5}{3} \cdot \binom{5}{3} = 100$ groups, which is already critical for scalability.

STREAM SELECTION. Within a 3×3 node group, and for each subcarrier triplet, we optimize which transmitter sends data to which receiver. As shown in Figure 6.4, we can only form node pairs in six different ways. Hence, finding the best one does not require further heuristics to reduce complexity.

MECHANISM SELECTION. Selecting which mechanism is best for a certain subcarrier does not raise concerns on scalability, since we only need to decide among IA or plain OFDM. Still, our IA stage mechanism needs an efficient metric to estimate the performance of both IA and plain OFDM. We discuss such a metric in Section 6.1.3.

6.1.3 *Selection Algorithms*

The goal of our selection algorithms is to find the best possible node group, stream pairings, and mechanism selection out of all possible variants in a given node subset. We do not consider subcarrier triplets, since we use the fixed allocation heuristic we introduce in Section 6.1.2.2. However, the resource combinations we consider are still per subcarrier, that is, we may choose a different node group, stream pairing, and mechanism for each of the fixed triplets.

6.1.3.1 *Noise Impact*

We use the EVM as a metric to determine the performance of each possible combination. Concretely, we estimate the EVM for plain OFDM and IA at each transmitter, and choose the combination which *minimizes* it. This requires CSI feedback, which is available, since it is anyhow needed to calculate the IA precoding vectors. In the following, we explain how we compute the EVM for each case based on CSI only.

PLAIN OFDM. In the frequency domain, the received signal Y of a plain OFDM transmission on a certain subcarrier is related to the sent signal X, the channel H, and the noise N as $Y = H \cdot X + N$. After zero-forcing at the receiver, the decoded signal D is $D = Y/H = X + N/H$. That is, D is the sum of the sent signal X and a noise term N/H. The latter is the EVM. Hence, we choose the nodes and streams for which the involved links feature a large $|H|$, and which thus minimize the EVM.

INTERFERENCE ALIGNMENT. In the case of IA, zero-forcing the received signal involves multiplication by an inverse matrix determined by the channels and the precoding vectors, as discussed in Section 6.1.1.1. In particular, the decoded signal $\mathbf{d}_n$ at each receiver n is as in Equation 6.6. We specify the aligned signal $\mathbf{s}_n$ individually for each n in Equations 6.7, 6.8, and 6.9, since data aligns differently at each receiver.

$$\mathbf{d}_n = \mathbf{A}_n \cdot \mathbf{r}_n = \mathbf{A}_n \cdot (\mathbf{B}_n \cdot \mathbf{s}_n + \mathbf{z}_n) = \mathbf{s}_n + \mathbf{A}_n \cdot \mathbf{z}_n \tag{6.6}$$

$$\mathbf{s}_1 = \begin{pmatrix} x_1 \\ x_2 \\ x_3 + x_4 \end{pmatrix} \quad (6.7) \qquad \mathbf{s}_2 = \begin{pmatrix} x_1 + x_4 \\ x_2 \\ x_3 \end{pmatrix} \quad (6.8) \qquad \mathbf{s}_3 = \begin{pmatrix} x_1 \\ x_2 + x_3 \\ x_4 \end{pmatrix} \quad (6.9)$$

In Equation 6.6, $\mathbf{d}_n$ is the zero-forced signal, $\mathbf{r}_n$ the received signal, $\mathbf{z}_n$ the noise at the receiver, and $\mathbf{A}_n = \mathbf{B}_n^{-1}$ the zero-forcing matrix for receiver n. The EVM results from the term $\mathbf{A}_n \cdot \mathbf{z}_n$, which is the noise after zero-forcing. Hence, we choose the node groups and stream pairs whose channels minimize the terms affecting the noise in $\mathbf{A}_n$. Not all terms in $\mathbf{A}_n$ contribute to the EVM, since receivers only decode at most two out of the three dimensions in $\mathbf{s}_n$—the third one is used to align interference. Equation 6.10 shows which rows of $\mathbf{A}_n$, as indicated by the arrows (→) next to them, affect which stream.

$$\mathbf{A}_n = \begin{pmatrix} {a_n}^{11} & {a_n}^{12} & {a_n}^{13} \\ {a_n}^{21} & {a_n}^{22} & {a_n}^{23} \\ {a_n}^{31} & {a_n}^{32} & {a_n}^{33} \end{pmatrix} \begin{array}{l} \rightarrow \text{Stream 1 at } n = 1 \\ \rightarrow \text{Stream 2 at } n = 1 \\ \rightarrow \text{Stream 3 at } n = 2\text{, and stream 4 at } n = 3 \end{array} \tag{6.10}$$

For our metric, we only consider the terms which contribute to the sum of the EVM at all receivers, which we call evm_{all}. Essentially, we find the resource combination which leads to the smallest evm_{all} for all of the receivers. Hence, we compute the matrices $\mathbf{A}_n$ with $n \in [1, 2, 3]$ for each resource combination—that is, we obtain three matrices corresponding to the three receivers in the node group of the current combination. The smaller the terms in the matrices $\mathbf{A}_n$, the smaller is the EVM and thus the noise impact. Thus, we compute evm_{all} as the sum of the coefficients of the matrices $\mathbf{A}_n$. However, we only consider the coefficients of each matrix $\mathbf{A}_n$ which affect the stream that receiver n decodes. In other words, for each value of n, we only sum the rows indicated in Equation 6.10. As a result, we obtain evm_{all} as in Equation 6.11.

$$\begin{aligned} \text{evm}_{\text{all}} = \; & |a_1|^{11} + |a_1|^{12} + |a_1|^{13} + |a_1|^{21} + |a_1|^{22} + |a_1|^{23} + \\ & |a_2|^{31} + |a_2|^{32} + |a_2|^{33} + |a_3|^{31} + |a_3|^{32} + |a_3|^{33} \end{aligned} \tag{6.11}$$

We calculate evm_{all} for all node groups/pairs in the subset and choose for each subcarrier triplet the smallest one. For instance, in an 802.11g-like system with 48 subcarriers and for a 4×4 node subset, we consider $48/3 = 16$ subcarrier triplets, 16 node groups, and 6 node pairings. Hence, for each of the 16 triplets, we compute $16 \cdot 6 = 96$ evm_{all} values, and choose the smallest one. This results in a total of $16 \cdot 16 \cdot 6 = 1536$ evm_{all} values. Our prototype in Section 6.1.4 shows that this is computationally feasible, and that the computation takes a reasonable amount of time. The latter is crucial, since IA must choose a resource combination before the CSI feedback becomes outdated.

6.1.3.2 *Optimizations*

Based on evm_{all}, we design six selection algorithms—two for our plain OFDM baseline mechanism and four for IA. For the plain OFDM algorithms, we consider node subsets of the same size than for the IA case to allow for a fair comparison.

- OFDM 3 × 3 **Fixed**. For our non-optimized baseline, we set the subset size to 3×3. We pair each receiver RX_e to a fixed transmitter TX_f, for $e = f$ with $e, f \in [1, 2, 3]$. Nodes share the medium using TDMA. This is equivalent to SL in fixed link mode.

- OFDM 4×4 **Optimized.** Our optimized OFDM variant works similarly, but chooses pairs out of a 4×4 subset, hence increasing diversity. This is our default baseline. We pair each receiver to the transmitter to which it has the smallest N/H.

Each of our four IA selection algorithms considers a different combination out of node, stream, and mechanism selection. That is, while the most basic algorithm does not allow for any selection at all, the most complex one includes all of the three aforementioned selection techniques. This allows us to assess the benefit of using each of the selections. In particular, our IA selection algorithms work as follows.

- IA 3×3 **Fixed.** This is a non-optimized variant of IA. There is no node selection as subsets and groups are of the same size. We also do not select streams nor mechanisms. In other words, this is frequency IA as described in theoretical work.
- IA 3×3 **Optimized.** In this case, we use stream selection to form pairs according to the evm_{all} metric, but we do not use node selection. Further, we use all subcarrier triplets for IA only, that is, we do not exploit mechanism selection.
- IA 4×4 **Optimized.** We increase the node subset size to allow for node selection. Concretely, we select 3×3 node groups out of 4×4 node subsets for each subcarrier triplet. Within each group, we use stream selection similarly to IA 3×3 Optimized.
- IA+OFDM 4×4. We build on the previous scheme but decide for each subcarrier triplet what mechanism to use, that is, whether to use IA or OFDM. Similarly to OFDMA, we allow this algorithm to allocate OFDM subcarriers to different links.

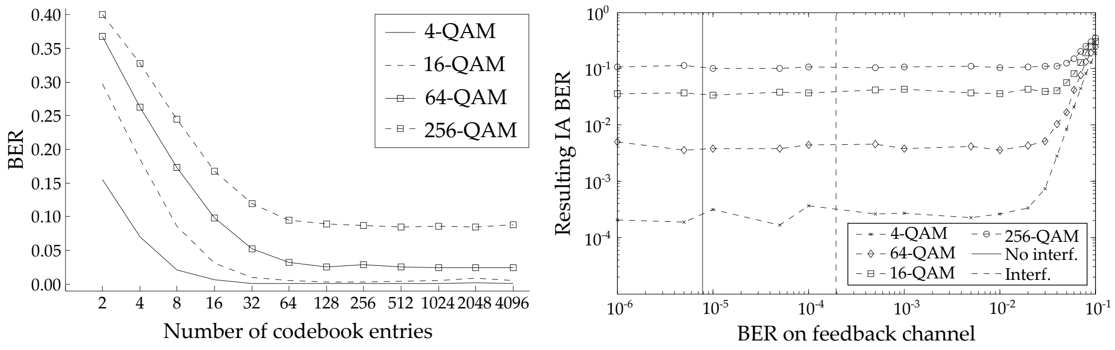
Table 6.1 gives an overview on which algorithm includes which of the selection techniques that we introduce in Section 6.1.2.1. In our evaluation in Section 6.1.5, we use the above algorithm names to specify which IA or OFDM variant we refer to.

6.1.4 *Implementation*

We implement frequency IA on the WARP SDR platform using WARPLab (c.f Section 2.3.3). For our single-hop IA study in this section, we focus on the PHY only. In Section 6.2, we build on this study, and implement IA as a full-fledged stage mechanism on WARP Drive,

Algorithm	**Stream selection**	**Node selection**	**Mechanism selection**	**Subcarrier selection**
OFDM 3×3 Fixed	No	No	No	No
OFDM 4×4 Optimized	Yes	Yes	No	No
IA 3×3 Fixed	No	No	No	Fixed
IA 3×3 Optimized	Yes	No	No	Fixed
IA 4×4 Optimized	Yes	Yes	No	Fixed
IA+OFDM 4×4	Yes	Yes	Yes	Fixed

Table 6.1: Overview on the selection techniques that each of our algorithms uses.



(a) BER with increasing feedback precision for IA.

(b) Impact of bit errors on feedback.

Figure 6.5: Impact of CSI quantization and feedback errors on IA performance.

that is, including the LNK and the NET. At the PHY, we base our design on the 802.11 standards (c.f. Section 4.2.3.4). Still, implementing practical frequency IA poses significant challenges, such as timely and accurate feedback. Next, we address these challenges and propose suitable solutions.

6.1.4.1 *CSI Feedback*

After receiving the pilots in part (a) of our frame design (c.f. Figure 6.3), the receivers quantize the estimated CSI using a codebook to send it back. This is a critical step, since too coarse CSI turns IA infeasible while too precise CSI has a large impact on overhead. In order to find an appropriate codebook size, we use an experimental approach. We estimate channels in our SDR testbed, quantize them with different codebook sizes, use them for IA, and measure the resulting BER. We test codebooks sizes ranging from 2 to 4096 entries, and M-QAM modulation orders from $M = 4$ to $M = 256$. Figure 6.5a shows our results. As expected, the BER becomes smaller for larger codebook sizes, since more precise CSI is available at the transmitter. Also, the higher the modulation order, the larger the BER, since errors are more likely in dense constellations. We observe that, for all modulation schemes, the BERs stabilize at a codebook size of about 64 entries. While a smaller codebook size is not precise enough for IA and causes errors, a larger codebook is unprofitable as it does not improve performance but increases feedback overhead. Thus, we conclude that 64 is a suitable codebook size for our testbed. Hence, we use this value as a default size for our experiments. Interestingly, our OFDMA stage mechanism provides good results using only one-bit feedback, that is, a codebook of size two. This significant difference is due to the underlying operation of IA and OFDMA. While OFDMA only needs feedback to decide whether a subcarrier experiences good channel conditions, IA uses feedback to compute precoding vectors in amplitude and phase. Thus, IA naturally needs more detailed CSI to operate.

We also study the effect of bit errors in the feedback frame in order to assess its robustness. Basically, these transmission errors modify the codebook entries, which means that the transmitters receive wrong CSI. As a result, they compute unsuitable precoding vectors and the interference does not align at the receivers. While we do not send feedback wirelessly in this single-hop study, we do account for its overhead. Further, we show experimentally that the number of bit errors in a typical wireless channel is small and thus does not affect IA performance at all. To this end, we first use simulations to assess the impact of wrong CSI on the IA frame. In particular, we study how the IA BER increases the more random bit errors we intentionally add to the CSI feedback, which we protect with a rate $1/2$ convolutional code. Next, we measure in practice the average BER of an OFDM feedback frame in our SDR testbed, and compare it to our simulative results to find out its impact on IA. Figure 6.5b shows our results. We observe that bit errors on the feedback frame for $\text{BER} \leqslant \text{BER}_{\text{ths}} = 10^{-2}$ do not affect the IA transmission at all, since curves are virtually stable up to that value. However, beyond that threshold, IA performance degrades as there are too many errors in the feedback frame. The continuous vertical line in Figure 6.5b represents the average OFDM BER in our testbed, which is about $8 \cdot 10^{-6}$. Thus, it is not critical for the CSI feedback.

However, external interference, which is not present in our lab, may also affect the CSI feedback transmission. Hence, we include an interferer in our practical setup and measure the OFDM BER when adding artificial noise. We set the transmit gains of the interferer to suitable values such that we do not fully jam the channel but still generate moderate interference. The dashed vertical line at $2 \cdot 10^{-4} < \text{BER}_{\text{ths}}$ in Figure 6.5b represents our result. Since the BER we measure is below the threshold BER_{ths}, above which the feedback errors affect the IA transmission, we conclude that feedback is also robust in the case of interference.

6.1.4.2 *Frequency Synchronization*

In IA, multiple transmitters transmit data simultaneously on the same subcarriers. However, the CFO varies for each pair of transmitter and receiver nodes. That is, a certain receiver receiving data from multiple transmitters experiences a different CFO for each transmitter. Hence, at an IA receiver, each of the components of an overlapped IA signal experiences a different CFO. In contrast to OFDMA, the receiver cannot separate the components to correct them individually but can only correct them as a whole. In other words, it can only correct the CFO for one transmitter but not for the others. As a result, it cannot decode the signal. To address this issue, we must ensure that all transmitters cause the *same* CFO at the receiver. In that case, the receiver only needs to correct one CFO value, which is feasible using pilot-aided techniques. To this end, transmitters can precode their signals to achieve the same CFO than a certain master transmitter node, which in Corridor-based Routing would be the main node of the current stage. That is, transmitters correct their signals prior to transmitting them. However, they first need to estimate their relative CFO to the aforementioned master transmitter. For this purpose, they can exchange pilot symbols among them. Precoding techniques for CFO correction can also eliminate CFO entirely at one single receiver. Still, this is not feasible in our IA scenario, since we have multiple receivers.

In our IA implementation, we abstract from the aforementioned CFO precoding among transmitters. Instead, we assume that all transmitters cause the same CFO at each receiver. Nevertheless, we still correct this CFO at the receivers. To this end, we use a mechanism similar to our approach in Section 5.2.2.3. To ensure that all transmitters share the same clock, we use wired synchronization. While this simplifies our design, it allows us to focus on the impact of our selection techniques, which are our core contribution in this single-hop study. Related work such as AirSynch [10] shows that practical CFO precoding is feasible for systems with multiple transmitters that transmit on the same time and frequency resources. Our IA design can build on such approaches.

6.1.4.3 *Time Synchronization*

Similarly to OFDMA, transmitters in IA must transmit simultaneously. For such tight intra-group time synchronization, we can again use an approach based on preambles as in Section 5.2.2.4. However, we also need inter-group time synchronization, that is, the receivers must be able to determine the beginning of a frame accurately. While we can also use a preamble in this case, IA is particularly challenging due to potential STOs. The key problem is that a receiver may detect the start of two subsequent frames with a marginally different time offset. For instance, while the receiver may detect the start of the first frame at exactly the correct time sample, for the second frame it may introduce an offset of a few samples. This does not hinder data decoding, since the CP compensates for such slight misadjustments, as long as the offset is shorter than the CP itself. In turn, this translates into a different rotation of the received signals in the time domain for each frame. The channel estimation mechanism at the receiver captures this rotation as part of the channel coefficients H. Thus, the receiver is typically oblivious to the STO. However, this means that the channel H that the receiver perceives is *different* in the first frame compared to the second frame. As a result, in case of STO, the precoding vectors based on the CSI that the receivers estimate from the first frame are invalid for the second frame.

To prevent this, we include pilot symbols not only in the first part of our frame in Figure 6.3, which we use for CSI estimation, but also in the last part of the frame that carries data. The channel itself does not change in between since the coherence time is larger than the entire frame. However, since frame parts (a) and (c) are two independent transmissions, the receivers might experience a different STO for each. Hence, they may obtain two different perceived channels for both, namely, H_1 and H_2. Before decoding the IA data, receivers filter frame part (c) as shown in Equation 6.12. Basically, they correct data a posteriori to emulate channel H_1 although the data experienced channel H_2. As a result, the precoding vectors obtained from the CSI feedback become valid again.

$$\text{data}_{\text{corrected}} = \text{data}_{\text{received}} \cdot \frac{H_1}{H_2} \tag{6.12}$$

Since the STO may be different for each pair of transmitter and receiver nodes, we must ensure that all transmitters start transmitting within the duration of a sample. The WARP SDR uses a 40 MHz clock, that is, each sample has a duration of 25 ns. Our practical time synchronization approach in Section 5.2.2.4 achieves on average a time offset of 8 ns. Hence, we do not expect this to become critical. For our IA prototype, we use wired intra-group synchronization since WARPLab does not allow for real-time operation. Still, we do ensure inter-group synchronization using the approach in Equation 6.12.

6.1.5 *Evaluation*

In the following, we evaluate our single-hop frequency IA system. In particular, our goal is to analyze whether frequency IA is operable in practice, and how well it performs when using the selection techniques that we present in Section 6.1.3.2.

6.1.5.1 *Metrics*

We use the overall PHY throughput as a metric to assess the performance of IA. If not stated otherwise, our baseline is the throughput that our scheme OFDM 4 × 4 Optimized achieves (c.f. Section 6.1.3.2). However, we cannot measure throughput directly since the delays that WAPRLab incurs would strongly affect the result. Still, we define a "raw" throughput metric to capture the impact of (a) the BER, and (b) the overhead. This metric abstracts from data framing and error correction, but we use it to *compare* the fundamental performance of our schemes, except in our last practical experiment in Section 6.1.5.4, where we do consider these issues. We take (a) into account by subtracting bit errors from the overall number of received bits, and (b) by including the overhead into the duration of the transmission. Also, the longer the IA frame is, the less weight the overhead carries. However, the coherence time gives the maximum length since we must update the CSI at the transmitters as soon as the channels change. We consider an indoor environment, that is, the coherence time is about 45 ms (c.f. Section 3.2.1.5). For calculations, we assume that our IA frame in Figure 6.3 is as long as the coherence time. However, since WARPLab buffers are not large enough to send data for such a comparatively long time, we obtain the raw throughput "thp" by extrapolating our measurement as follows. We measure the correct bits received for a duration of t_{measure}, calculate how many times t_{measure} fits into $t_{\text{coherence}}$, subtract the time for transmitting overhead and divide by $t_{\text{coherence}}$, as shown in Equation 6.13.

$$\text{thp} = \frac{(\text{bits}_{\text{TX}} - \text{bits}_{\text{ERR}}) \cdot \frac{t_{\text{coherence}} - t_{\text{overhead}}}{t_{\text{measure}}}}{t_{\text{coherence}}} \tag{6.13}$$

While our raw throughput metric takes into account the BER, a higher throughput does not implicitly mean a lower BER since IA has a larger DOF than plain OFDM. That is, if the IA BER impacts throughput less than the achievable 33% gain, IA has a higher throughput. Thus, in the following experiments we consider *both* metrics to evaluate whether our IA optimizations based on selection are successful. We aim at maximizing throughput using IA at comparable BERs. In most of our experiments we discuss the interaction of both metrics for fixed modulation schemes to understand how our selection techniques work. However, in our last practical experiment in Section 6.1.5.4 we bring both metrics together using adaptive modulation.

6.1.5.2 *Scenarios*

While we evaluate our single-hop IA system primarily in practice, we also perform simulations in order to study large and random topologies which we cannot recreate in our testbed. This ensures that our practical results are not specific to our testbed.

SIMULATION SETUP. The goal of our simulations is twofold, namely, (a) evaluate the performance of IA on random topologies, and (b) study the scheme to form node subsets during corridor construction that we sketch in Section 6.1.2.2. To this end, we assume a scenario with a large number of nodes deployed randomly in a space of size $s \cdot s\ m^2$. Additionally, we place four transmitters at fixed locations. These transmitters stand for the nodes that the corridor construction algorithm has already chosen during the construction of the previous stage. The larger s, the larger the distance among the transmitters. To achieve (b), in each simulation run we choose $n \in [3, 4]$ transmitters according to our node selection scheme in Section 6.1.2.2. To achieve (a), we then use the selection algorithms we define in Section 6.1.3.2 to transmit data from the transmitters to the receivers using IA. Depending on the selection algorithm, we either build 3×3

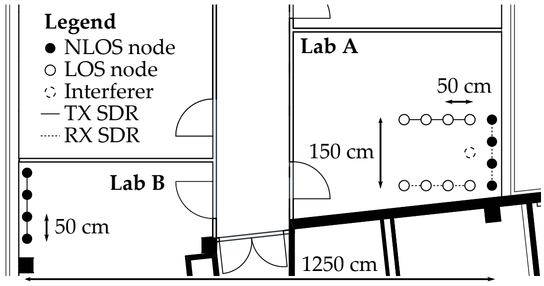


Figure 6.6: Practical single-hop IA setup

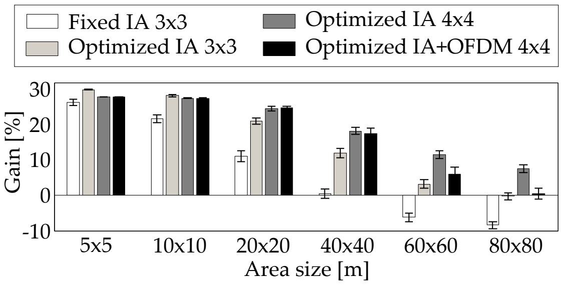


Figure 6.7: Single-hop IA simulation results

or 4 × 4 node subsets. For the former, we only use the first three transmitter out of the four that we place at fixed locations. Further, we assume a path loss exponent of $\alpha = 2$, Rayleigh channels, and 70 dB SNR at a distance of 1 meter to the transmitters.

PRACTICAL SETUP. Our practical testbed consists of two WARPv3 boards, each with four radio interfaces. We use each radio as if it were an individual node, that is, we treat the data of each interface independently in Matlab. All radios on the first board act as transmitters, while all radios on the second board are receivers. Such a setup allows us to model 3 × 3 and 4 × 4 node subsets. The transmitters are implicitly synchronized since we place them on the same WARP board. On the receiver side, having all nodes on one board is not required, but simplifies our testbed setup. If not stated otherwise, the two WARP boards are *not* synchronized among each other. In our experiments, we consider both a Line-Of-Sight (LOS) and a Non-Line-Of-Sight (NLOS) scenario, as depicted in Figure 6.6. For some experiments we add an interferer to study the robustness of our IA schemes. For both LOS and NLOS, we adjust the amplifier gains at the transmitters to keep SNRs in the range of typical indoor scenarios, that is, 20 to 30 dB.

If not stated otherwise, we use OFDM parameters equivalent to an 802.11g system, i.e., 18 MHz channels, 48 usable subcarriers, 312.5 kHz subcarrier spacing, and 12.5% CP. For experiments inspired by 802.11ac, we increase the number of subcarriers to 456, which in the standard corresponds to 160 MHz channels. Since WARP does not support such large bandwidths, we reduce the subcarrier spacing to 39 kHz, resulting in longer OFDM symbols. Still, this allows us to assess performance with a large number of subcarriers.

6.1.5.3 *Simulation Results*

In our simulations, we focus on the results that we cannot obtain in our testbed, namely, (a) random deployment of receivers in flexibly sized areas, and (b) a larger number of nodes. Figure 6.7 depicts the gains in terms of throughput of IA compared to OFDM 4 × 4 Optimized for increasing areas. The modulation scheme is 16-QAM, the number of nodes is fixed to 12, and we vary s from 5 to 80 meters. As expected, we observe an overall decline of gains with increasing area size, since receivers are on average further away from the transmitters, and thus experience lower SNRs. For the smallest area that we consider ($s = 5$), all algorithms achieve gains in between 25% and 30%. We cannot reach the theoretical 33% maximum because we include the feedback overhead with a codebook size of 64. The schemes using 4 × 4 subsets have slightly worse gains because they need to transmit CSI feedback of 16 links instead of only 9 links. However, this overhead pays off as soon as the room size increases and SNRs become lower—while the

Algorithm	12 nodes	24 nodes	36 nodes	48 nodes	60 nodes
OFDM 4×4 Optimized	59.4 mbps	59.4 mbps	59.4 mbps	59.4 mbps	59.4 mbps
IA 3×3 Fixed	72.2 mbps	71.6 mbps	71.8 mbps	71.8 mbps	72.0 mbps
IA 3×3 Optimized	76.2 mbps	76.2 mbps	76.2 mbps	76.0 mbps	76.1 mbps
IA 4×4 Optimized	75.7 mbps	75.8 mbps	75.8 mbps	75.8 mbps	75.8 mbps
IA+OFDM 4×4	75.8 mbps	75.8 mbps	75.8 mbps	75.8 mbps	75.8 mbps

Table 6.2: Average throughput each time a subset is served with increasing number of potential receivers. Throughput is essentially constant since more nodes barely increase diversity.

3×3 schemes quickly degrade, the 4×4 ones maintain gains close to 25% up to $s = 20$ due to our selection techniques. Concretely, 4×4 schemes can avoid combinations with particularly poor channel conditions using node selection, whereas 3×3 schemes must always use a given node group. For larger area sizes, we observe that the throughput gain decreases more for IA+OFDM than for 4×4 IA. The BER is responsible for this behavior. For low SNRs, IA performs worse than OFDM with respect to the BER, that is, while IA may achieve larger gross throughput, its BER cancels out most of it in terms of raw throughput (c.f. Equation 6.13). IA+OFDM gradually switches more subcarriers to OFDM in order to maintain a low BER, thus loosing throughput gain. While the throughput gain for 4×4 IA is larger, for $s = 80$ its BER is close to 20% (not shown in Figure 6.7). In contrast, the BER of IA+OFDM is still below 10%, similarly to the OFDM baseline. That is, IA+OFDM is able to dynamically adapt to channel conditions by transitioning individual subcarriers to OFDM when the SNR is too low for IA.

Table 6.2 shows our results for an increasing number of potential receiver nodes. The area is fixed to $s = 10$ and the modulation scheme is 16-QAM. In particular, we measure the throughput that each node subset achieves individually to assess the impact of higher node densities on forming subsets. That is, the more nodes, the more node subsets we form. We do not show the per subset throughput, which decreases with the number of subsets as $1/\text{subsets}$. This is expected, since we serve each subset in a TDMA fashion. Table 6.2 shows that the throughput is approximately constant with increasing node density. Thus, we conclude that our selection schemes do not require a large number of nodes to form suitable subsets, and achieve gains over both plain IA and plain OFDM.

6.1.5.4 *Practical Results*

In the following, we present the results of our practical experiments. Our first experiments focus on the practical performance of IA 3×3 Fixed, that is, IA without using our selection techniques. After that, we investigate the contribution of each of the individual selection techniques in scenarios that highlight their characteristics. Finally, in our last experiment we study the impact of adaptive modulation.

PLAIN IA. We first analyze the performance of IA *without* our selection mechanisms, and the impact of practical issues. Figure 6.8 depicts the gain in terms of throughput, and the BER in our LOS setup. First we focus on 802.11g, which is our default physical layer. We compare an idealized case, where we assume perfect CSI and synchronize both WARP boards to avoid CFO, with a realistic case where CSI is quantized, and the receivers

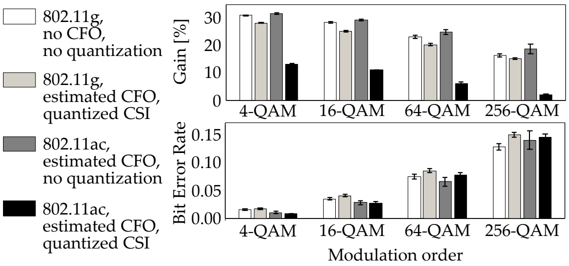


Figure 6.8: BER and throughput gain for IA 3×3 Fixed over OFDM 4×4 Opt.

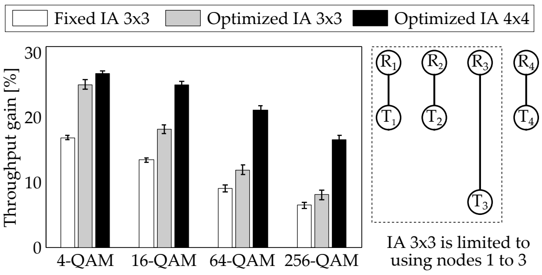
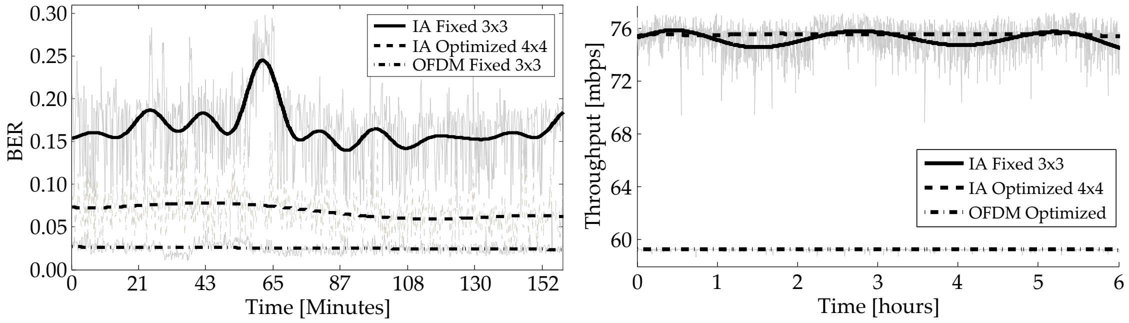


Figure 6.9: Throughput gain for one transmitter placed far away from the receivers.

must correct CFO. For perfect CSI, we assume that the feedback overhead is zero. The ideal case achieves close to 30% throughput gain compared to OFDM 4×4 Optimized for 4-QAM. Since we do not use our selection mechanisms, it degrades and incurs high BERs for higher modulations. The realistic case performs similarly, which means that we are able to cope successfully with CFO and CSI. However, gains are slightly lower due to these practical issues and the feedback overhead. Next, we study the performance of IA on PHY parameters inspired by 802.11ac. Since we observed that we can cope with CFO, for the case inspired by 802.11ac we keep estimating it realistically for both measurements in Figure 6.8. The first one assumes perfect CSI and achieves similar results than 802.11g, which shows that our IA system is well suited for encoding a large number of subcarriers. For the second measurement—with quantized feedback—the gain drops despite similar BERs. The reason is that the feedback overhead is much larger when sending CSI for 456 subcarriers than for 48 in 802.11g. However, our setup is limited to 20 MHz channels—in a 160 MHz system, the overhead impact for 802.11ac is similar to the 802.11g case.

TIME DEPENDENCE AND ROBUSTNESS. In our next experiment, we compare the behavior over time of IA 3×3 Fixed with IA 4×4 Optimized. Figures 6.10a and 6.10b show our results. In each measurement round, we measure all schemes, that is, they experience virtually the same CSI. We fit curves on our measurements to highlight their behavior. In both figures, IA 4×4 Optimized is more stable and robust than plain IA, since it adapts to changing channel conditions, while plain IA cannot avoid CSI fluctuations. In particular, Figure 6.10a depicts the BER during 2.5 hours for 256-QAM. At about minute 65, a change in channel quality increases the BER for plain IA for about 10 minutes. Meanwhile, IA 4×4 Optimized absorbs the CSI change by transitioning automatically to a suitable node and stream allocation. Thus, it is not affected at all. Moreover, Figure 6.10a shows a significant BER improvement of IA 4×4 Optimized compared to IA 3×3 Fixed. Figure 6.10b depicts the throughput of both schemes for six hours using 16-QAM, which is less prone to errors. As a result, both IA 4×4 Optimized and IA 3×3 Fixed perform similarly. However, note that the latter continuously fluctuates around the average value while the former is stable. Our measurements reflect the performance of our schemes during daytime. However, we also perform experiments over night. The results show that, as expected, channels are nearly constant, and thus all our schemes perform stably.

NODE SELECTION. In the following, we investigate scenarios where the selection algorithms benefit most, starting with node selection. To highlight its operation, we place the third transmitter of the LOS setup further away from the receivers, as depicted in



(a) BER over 2.5 hours for 256-QAM in LOS setup.

(b) Throughput over 6 hours (LOS setup, 16-QAM).

Figure 6.10: Robustness of frequency IA over time.

Figure 6.9. The idea behind this setup is that 3×3 algorithms are forced to use the far away node T_3, while 4×4 schemes can resort to T_4, which is much closer and thus experiences better channels. As expected, the results in Figure 6.9 show that both IA 3×3 Fixed and IA 3×3 Optimized perform significantly worse than IA 4×4. IA 3×3 Optimized still achieves good gains for 4-QAM since its stream selection—in combination with incurring less CSI feedback overhead than IA 4×4—improves performance just enough to achieve a similar throughput than the 4×4 case. However, for higher modulations this improvement is not enough, and thus its performance degrades considerably.

MECHANISM SELECTION. Next, we show how mechanism selection improves transmission in terms of BER for heterogeneous scenarios. To this end, we use our NLOS setup and place one transmitter closer to the receivers, as illustrated in Figure 6.11. When switching from NLOS to the "closer" setup, IA 4×4 Optimized experiences no improvement. This is expected, since it involves at least three nodes, and is limited by the SNRs of the far away ones. In contrast, IA+OFDM cuts down the BER to *half* its value in the "closer" setup. From our insights in Section 6.1.5.3, we deduce that IA+OFDM allocates individual OFDM subcarriers to the node with better SNR. In other words, it trades throughput for BER, which is essential if the channel code in use cannot cope with large BERs.

EXTERNAL INTERFERENCE. In what follows, we investigate the robustness of our selection techniques when adapting to external interference. We generate artificial noise on an increasing percentage of subcarriers using the interferer in Figure 6.6. We set low transmission gains to fit the setup. Thus, nodes on the right side of the LOS setup are more strongly affected than nodes on the left. Figure 6.12 shows our results for 64-QAM. While effects are similar for lower modulations schemes, they are less manifest. As expected, we observe that OFDM 4×4 Optimized degrades linearly with increasing noise bandwidth, since (a) it averages the performance of all nodes, and (b) all subcarriers affected by noise contribute equally to the BER. In contrast, noise barely affects IA+OFDM. Most probably, this scheme allocates resources predominantly to the left 3×3 group to adapt to the interferer on the right, and resorts to the more robust OFDM on most triplets.

For the IA algorithms without mechanism selection (dashed lines in Figure 6.12), we observe a common trend. For a noise bandwidth up to 30%, the BER rises, but beyond 30% it falls again. This behavior is consistent with the even distribution of subcarrier

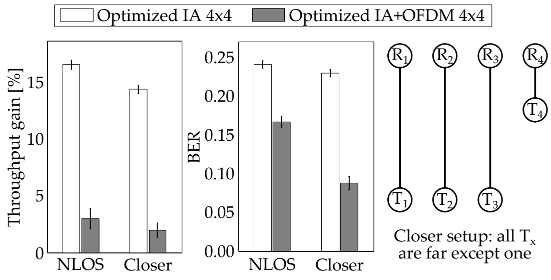


Figure 6.11: Throughput gain vs. plain OFDM and BER in the "closer" setup.

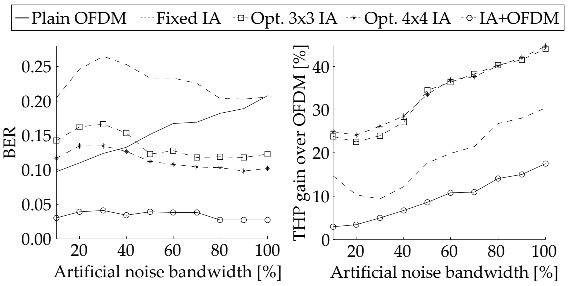


Figure 6.12: BER and throughput with the interferer in Figure 6.6 active (64-QAM).

triplets over the available bandwidth (c.f. Section 6.1.2.2). Basically, even narrow noise bandwidths quickly affect at least one subcarrier in each triplet, causing a steep initial rise of the BER. From the operation of our interferer, we infer that the subsequent decline beyond 30% is due to the increasing distribution of noise power. Our interference signal has the same power for all noise bandwidths, which means that for narrow bandwidths, a large noise affects only a few subcarriers, while a smaller noise affects many subcarriers for large bandwidths. Thus, we conclude that IA can cope better with small noise affecting all subcarriers in a triplet than with large noise affecting only one subcarrier. In other words, the noisiest subcarrier determines the performance of IA. Hence, in order to adapt efficiently to interference, an IA system using subcarrier selection should ensure that all subcarriers of a triplet have similar SNRs. For plain OFDM, noise power is also constant for all noise bandwidths, but Figure 6.12 shows that the BER improvement due to less noise per subcarrier is smaller than the degradation due to noise affecting more subcarriers. As expected, the overall performance of the optimized IA algorithms is better than for the fixed case, again due to selection. Regarding throughput gain, all mechanisms tend to improve with increasing noise because the baseline degrades linearly.

NLOS SETUP. We now investigate the performance of frequency IA in our NLOS setup (c.f. Figure 6.6). Figure 6.13 depicts our results. For low modulation schemes, all algorithms achieve significant gains. However, while plain IA 3 × 3 Fixed degrades quickly for increasing modulations, our selection algorithms enable optimized IA to provide gains above 15% even for 256-QAM. When switching from 64-QAM to 256-QAM, the gain of IA+OFDM decreases significantly. However, our measurements show that it achieves a BER much lower than the other IA schemes, and similar to OFDM 4 × 4 Optimized. This indicates that IA+OFDM transitions to OFDM on most triplets. All in all, we conclude that IA in NLOS scenarios is feasible, and benefits from node, stream, and mechanism selection.

ADAPTIVE MODULATION. In previous experiments, we study the IA throughput and BER gain for individual modulation schemes. Next, we analyze its behavior for adaptive modulation. Since IA has higher SNR requirements than plain OFDM, IA typically has larger BER for a given modulation. Hence, this raises the question whether for a given BER, plain OFDM could simply switch to a higher modulation than IA and thus achieve a higher throughput. To understand this issue, we analyze a system where transmitters adaptively choose for each frame the modulation and technique (IA or OFDM) that provides best throughput. Since we choose among both techniques, we never perform worse than plain OFDM despite the higher SNR requirements of IA. Similarly to a real-world system,

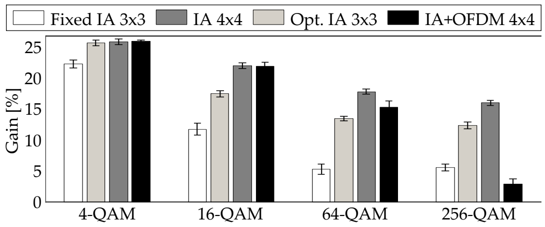


Figure 6.13: Throughput gain vs. OFDM 4 × 4 Optimized in our NLOS setup.

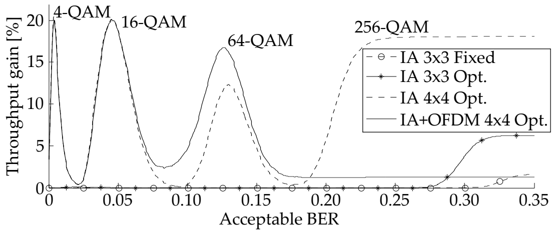


Figure 6.14: Throughput gain compared to OFDM with adaptive modulation.

transmitters aim at achieving a certain acceptable BER for each frame. The acceptable BER may be the error correction capability of the code in use, or the BER allowed by a loss-tolerant stream such as video. In Figure 6.14, we show the performance for a range of acceptable BER values, without limiting them to a specific coding scheme or stream type. For this experiment, we use an overnight measurement (9.5 hours) in our NLOS scenario. Hence, the average SNR is constant and provided by our setup. In particular, we continuously measure all our schemes for the M-QAM modulation orders $M \in [4, 16, 64, 256]$. Then, for each acceptable BER value, we find the best modulation and technique (IA or OFDM). We compare results to a baseline which always uses OFDM, but still adapts the modulation order. In other words, we do consider the case where our system and the baseline are using *different* modulation orders. In Figure 6.14, the gain peaks occur when the acceptable BER is larger than required for plain OFDM using a certain modulation order, but not enough for OFDM to switch to the next modulation. These are the margins where IA pays off, since this “excess” of the acceptable BER absorbs the higher SNR requirements of IA. For instance, such margins naturally occur for error correction codes. This happens because practical protocols usually provide a fixed set of codes, among which the transmitter can choose, and thus the code often corrects more errors than actually needed. Since each modulation order is feasible for a different range of acceptable BERs, we observe multiple peaks in Figure 6.14. Note that only 4 × 4 algorithms achieve gain, while the aforementioned margins are too small for all other IA schemes. This showcases how selection not only enables IA but also makes it profitable in a realistic setting. For the 64-QAM peak, IA+OFDM achieves even better results than IA 4 × 4. This is expected because it uses IA only on good subcarriers. By design, it stabilizes at a rather low gain value, since it transitions most of its subcarriers to OFDM for large BERs. This translates into no increased DOF. All other IA schemes stabilize at a high gain as soon as the acceptable BER covers the errors of IA at the highest modulation order.

6.1.5.5 *Discussion*

While we consider particular cases in our testbed measurements, we cover a broad range of heterogeneous scenarios to understand how our selection algorithms work. Moreover, our simulations with random node locations and variable SNRs in Section 6.1.5.3 provide equivalent results. Hence, we conclude that our testbed experiments are representative, and can be generalized to larger networks. The subset size is critical for feasibility, since our schemes require full CSI of each subset link, and the number of possible node selections explodes in large subsets. We expect the latter to be the constraining factor, since the overhead of CSI feedback has a limited impact in our experiments. Still, larger subsets increase the probability of finding good selections. In our testbed, a subset size of 4×4 is an operable trade-off. Further, we show that IA provides gains with realistic adaptive modulation. The gain is directly related to the acceptable BER, which in turn depends on the coding rate in use, the type of data sent, and ultimately on the application. Hence, further cross-layer optimizations to fully exploit the benefits of IA are possible. Also, while we select resources based on a PHY metric, our approach opens the door to metrics that include upper-layer requirements. Still, in this section we focus on PHY performance only. Finally, our experiments show how our selection algorithms can transition dynamically among different node, stream, and mechanism allocations on a subcarrier-wise basis in order to adapt to changing channel conditions. This is key to enable frequency IA, since adaptation is crucial in a wireless environment.

6.2 IA DESIGN

In this section, we extend the single-hop frequency IA system in Section 6.1 to a full-fledged multi-hop stage mechanism. In particular, we focus on the allocation of IA resources according to (a) the current CSI, and (b) the amount of data that each transmitter in a stage needs to forward. Ultimately, our goal is to minimize the transmission time within a stage following a similar approach to our allocation mechanism in Section 5.2.3.2.

6.2.1 *Profile*

Table 6.3 summarizes the profile of our IA stage mechanism. The constraints of frequency IA result in significantly higher requirements compared to the profiles of SL and OFDMA. Still, our resource allocation mechanism in Section 6.2.2 allows for any amount of data at

Characteristic	**Description**
Input data	Allows for any amount of data at any of the nodes of the transmitter group.
Output data	May result in any amount of data at any of the nodes of the receiver group.
Stage shape	The number of both transmitters and receivers must be at least three, that is, $m \geqslant 3$ and $n \geqslant 3$.
Feedback	Full per-subcarrier CSI.
Time synchronization	Transmitters must start transmitting simultaneously within the duration of a sample.
Frequency synchronization	Transmitters must synchronize to compensate their relative CFOs. Receivers must correct the resulting homogeneous CFO using pilot-aided techniques.
Coherence time	The coherence time must be long enough to allow for packet aggregation and thus overhead compensation.
Coherence bandwidth	The coherence bandwidth must be small enough to result in uncorrelated subcarrier triplets.

Table 6.3: Profile of our IA stage mechanism.

the transmitters and, similarly to SL and OFDMA, may result in any amount of data at the receivers. Regarding the stage shape, frequency IA requires at least three transmitters and receivers, that is, $m \geqslant 3$ and $n \geqslant 3$. For smaller stages, a corridor must resort to other stage mechanisms. Further, the profile requires full per-subcarrier CSI, since IA needs the amplitude and phase of the channel coefficients to compute the IA precoding vectors. As discussed in Sections 6.1.4.2 and 6.1.4.3, IA poses significant constraints for both time and frequency synchronization. In particular, transmitters must start transmitting within the duration of a sample such that their signals arrive with the same STO at a certain receiver. Similarly, transmitters must use CFO precoding to cause the same CFO at a receiver, thus allowing it to correct the CFO using pilot-aided techniques. In order to reduce control data and keep the overhead impact low, the coherence time of the stage must allow for packet aggregation. Finally, the channel must be frequency selective enough to ensure that subcarriers within a triplet (c.f. Section 6.1.2.2) are uncorrelated. That is, small coherence bandwidths are beneficial.

6.2.2 *Resource Allocation*

For our IA stage mechanism, we use a resource allocation mechanism that minimizes the transmission time. The key difference to the single-hop case in Section 6.1 is that transmitters do not always have data for all receivers. Instead, they only have a certain amount of data, which depends on how previous stages split and join packet segments. In other words, while in Section 6.1 transmitters can always fully exploit beneficial IA resource combinations, in a corridor stage the benefit of such combinations is limited if the transmitters that could use them have no data to transmit. Hence, for resource allocation, we must take into account both the CSI, and the amount of data at each node. We build on the same allocation scheme that we use for our OFDMA stage mechanism, since we essentially need to solve the same problem—assigning resources to nodes. However, in this case, resources are not OFDMA subchannels but IA subcarrier triplets. Moreover, instead of using each resource on an individual link, IA uses each resource for a certain combination of nodes, pairs, and mechanisms (c.f. Section 6.1.2.1). In the following, we explain how we translate the combinations we study in Section 6.1 into a resource allocation problem which our scheme in Section 5.2.3.2 can solve.

6.2.2.1 *Optimizations*

Similarly to the single-hop case, for our IA stage mechanism, we use the heuristics in Section 6.1.2.2 to reduce the number of possible resource combinations. However, instead of choosing the best combination for each triplet, we select the one which reduces the transmission time. In particular, we consider the following combinations.

- Node Subsets. The node subset is essentially the current stage. While in Section 6.1 we only consider 3×3 and 4×4 subsets, for our IA stage mechanism we allow for any stage size as long as $m \geqslant 3$ and $n \geqslant 3$, including cases for which $m \neq n$.
- Node Groups. The number of node groups in a stage is $\binom{m}{3} \cdot \binom{n}{3}$. For large m and n, this number becomes very large. Still, our corridor construction experiments show that achieving wide stages is often hard, and thus $m > 4$ or $n > 4$ is infrequent.

- NODE PAIRS. In contrast to Section 6.1, the performance of an IA stage depends on which transmitter uses the double capacity link. The reason is that transmitters do not necessarily always have enough data to fully exploit it. For instance, if the first transmitter in a node group barely has any data to send while the second transmitter has a large number of packet segments, the stage transmission time changes significantly depending on who can use the double capacity link. Hence, the allocation mechanism must consider 18 instead of only 6 different node pairings within each node group, as shown in Figure 6.15 in contrast to Figure 6.4.

As a result, for an 802.11g-like system and a corridor of intended width $w = 4$, we consider 16 subcarrier triplets, at most 16 node groups, and 18 node pairs. That is, the overall number of combinations is $16 \cdot 16 \cdot 18 = 4608$, which is in the same order of magnitude than the number of combinations we consider in the single-hop case. Hence, we conclude that we do not need further heuristics to reduce the number of combinations.

6.2.2.2 *Allocation Mechanism*

In order to use the allocation mechanism in Section 5.2.3.2, we must compute the estimated transmission time of a packet segment for each combination on each IA subcarrier triplet. This is equivalent to computing the estimated transmission time of a packet segment on each link and subchannel in an OFDMA stage, and implies similar challenges (c.f. Section 4.2.3.1). To this end, we use CSI feedback at the transmitters to obtain the noise impact for each IA resource combination, and deduce the resulting SNR. This allows us then to estimate the number of batches that Strider requires to transmit packet segments on that resource. However, in contrast to the assignment of a subchannel to a link in OFDMA, assigning a triplet to a combination allows *three* node pairs—instead of only one—to exchange a data segment. Hence, we compute three transmission times for each combination, namely, one for each IA stream. Further, one of the streams uses the double capacity link. Concretely, we exploit this additional capacity to transmit a single packet segment faster instead of scheduling two segments in parallel. Therefore, in Figure 6.3 we show two shorter segments in a row for the transmitter using the double capacity link. Since each of the two streams on that link may experience a different channel quality, we estimate the transmission time based on the worst of both streams.

In particular, we estimate the transmission time based on matrix $\mathbf{A}_n$ in Equation 6.10. In Section 6.1, we compute the overall noise impact of a combination as the sum of all matrix coefficients affecting each of the four IA streams. However, for our IA stage mechanism, we need the performance of each individual link. Thus, we compute the individual EVM of each stream, according to the rows highlighted in Equation 6.10. Equation 6.14 shows the EVM of each of the streams on the double capacity link, while Equations 6.15 and 6.16

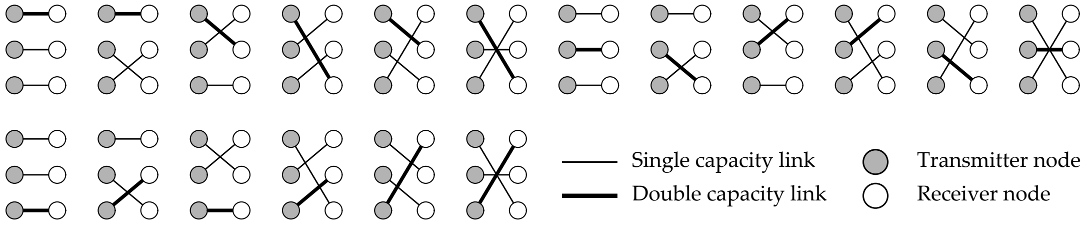


Figure 6.15: All possible node pairs within a 3 × 3 node group in a stage.

refer to the EVM of the single capacity links. We then use these EVM values to obtain the estimated SNR for each IA stream as in Section 4.1.2.1. Finally, we use the SNR to deduce the number of Strider batches that the transmitter nodes need to convey a data segment at the receivers.

$$\left.\begin{aligned} \mathrm{evm}_{\mathrm{A1}} &= |a_1|^{11} + |a_1|^{12} + |a_1|^{13} \\ \mathrm{evm}_{\mathrm{A2}} &= |a_1|^{21} + |a_1|^{22} + |a_1|^{23} \end{aligned}\right\} \quad \Rightarrow \mathrm{snr}_{\mathrm{A}} = \min\left(\frac{1}{\mathrm{evm}_{\mathrm{A1}}}, \frac{1}{\mathrm{evm}_{\mathrm{A2}}}\right) \tag{6.14}$$

$$\mathrm{evm}_{\mathrm{B}} = |a_2|^{31} + |a_2|^{32} + |a_2|^{33} \quad \Rightarrow \mathrm{snr}_{\mathrm{B}} = \frac{1}{\mathrm{evm}_{\mathrm{B}}} \tag{6.15}$$

$$\mathrm{evm}_{\mathrm{C}} = |a_3|^{31} + |a_3|^{32} + |a_3|^{33} \quad \Rightarrow \mathrm{snr}_{\mathrm{C}} = \frac{1}{\mathrm{evm}_{\mathrm{C}}} \tag{6.16}$$

Hence, we obtain for each resource combination on each subcarrier triplet two results, namely, (a) the three pairs of transmitters and receivers that can exchange data, and (b) the estimated transmission time for each of the three data segments that those nodes can exchange. The resource allocation mechanism takes this information as an input to decide how to use each of the available triplets in order to minimize the stage transmission time. Similarly to the OFDMA case in Figure 5.8, in each iteration the allocation mechanism assigns a subcarrier triplet to the transmitter node which takes most to transmit its data with the current resource allocation. In particular, it finds the triplet out of the still available triplets which features the combination that most benefits the aforementioned node, that is, the triplet which allows the node to transmit with the highest throughput. After allocating this triplet and combination to the chosen node, the mechanism recomputes the transmission time of that node, and of the other two nodes which, as a side effect of using the chosen combination, also can transmit a segment. The allocation mechanism repeats this process until no triplets are left. Finally, we perform a resource release phase following the same algorithm than in Figure 5.8.

6.2.3 *Operation*

Our IA stage mechanism uses the frame format in Figure 6.3, that is, it operates similarly to our single-hop prototype in Section 6.1. However, both the multi-hop corridor environment, and the scheduling of a limited number of packet segments impose additional constraints. We address the former in Section 6.2.3.1 and the latter in Section 6.2.3.2.

6.2.3.1 *Forwarding*

In contrast to our SL and OFDMA stage mechanisms, IA cannot operate on any stage shape but requires at least three transmitters and three receivers. Hence, the initial and last stages of a corridor, which typically have the shape $1 \times n$ and $m \times 1$, respectively, cannot use IA to forward data. As a result, a corridor using IA must support at least one additional stage mechanism that can operate on any stage shape. Prior to forwarding data on a stage, the transmitters must decide which stage mechanism to use. For our experiments in this chapter, we use IA whenever possible, that is, we consider the stage shape only to decide whether to use IA. We use OFDMA as an alternative stage mechanism. While we could also use SL, SL does not support splitting data among multiple nodes, and thus the corridor would never use IA. For results on corridors dynamically switching among PHYs according to CSI and traffic conditions, we refer the reader to Chapter 7.

6.2.3.2 *Scheduling*

Similarly to an OFDMA stage mechanism, after resource allocation, IA transmitters still must decide how to schedule the packet segments that they need to transmit on the resources allocated to them. To this end, we use again the iterative scheduler in Figure 5.11. In each iteration, we schedule a packet segment on the subcarrier triplet that incurs the smallest increase of the node transmission time, compared to the time required to transmit the segments scheduled so far (c.f. Figure 5.10). The scheduler repeats this process until no segments are left. As in the OFDMA case, each node schedules its segments individually. That is, we do not minimize the overall transmission time of all nodes but only the transmission time of each individual node (c.f. Section 5.2.4.3). Still, since transmitters are aware of both the CSI of all stage links, and the *amount* of data at each transmitter, they can compute in a distributed manner how many packet segments each node schedules for transmission on each triplet. This is crucial for the implementation of IA because using an IA triplet always involves three transmitters. In other words, even if only one transmitter decides to use a triplet, the other two nodes of the resource combination allocated to that triplet could transmit, too. We discuss this issue in detail in Section 6.3.

6.2.4 *Corridor Protocol Phases*

The corridor protocol phases that we define in Section 3.2.1 correspond to the operation of our IA stage as follows. Stage maintenance includes both CSI measurement using pilot symbols, and full CSI feedback. In comparison to our previous stage mechanisms, the latter has a significantly larger impact on the overhead since IA requires amplitude and phase of each channel coefficient. Profile matching is also more restrictive than for SL and OFDMA since it must ensure that (a) $m, n \geqslant 3$, (b) the frequency selectivity of the stage is large enough to result in uncorrelated subcarrier triplets, (c) the stage shape yields a tractable number of node groups, and (d) nodes have enough computational capabilities to find suitable resource combinations in a time significantly shorter than the coherence time. The latter is decisive for intra-stage coordination, since our resource allocation mechanism uses the aforementioned combinations as an input. As highlighted in Figure 6.3, the computational effort required to find these combinations may have an impact on the timing of the frame. In contrast, inter-stage coordination is limited to the identifiers included in each segment header that allow the destination to reassemble packets (c.f. Section 4.2.1). Finally, the data transmission phase comprises the deliberate collision of precoded signals in part (c) of our frame format in Figure 6.3.

6.3 IMPLEMENTATION

We realize our IA stage mechanism on WARP Drive (c.f. Section 7), that is, our implementation allows us to perform both simulations as well as over-the-air measurements directly from Matlab. WARP Drive also enables us to consider the PHY, LNK, and NET despite the non-real-time operation of WARPLab. In particular, we use the LNK and NET that we describe in Sections 4.2.1 and 4.2.2, respectively, since these components are common to all stage mechanisms for Corridor-based Routing. In the following we present the details of our PHY implementation, and explain how we deal with the practical issues that we also discuss for the single-hop case in Section 6.1.4.

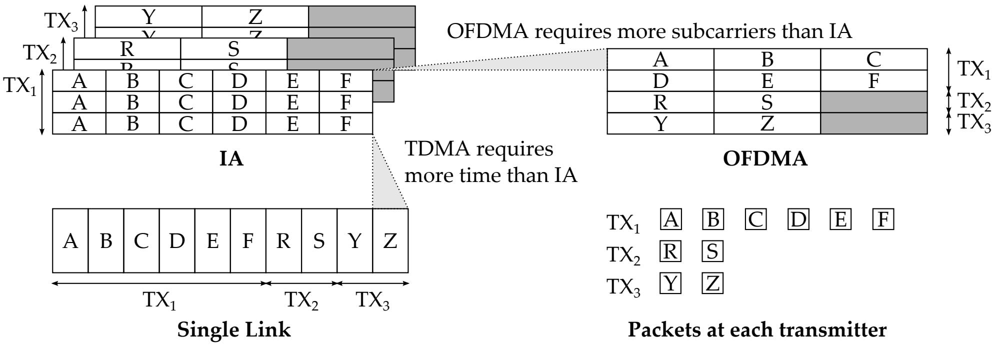


Figure 6.16: Empty transmission slots (shaded) in an IA data frame compared to OFDMA and SL.

6.3.1 *Physical Layer*

A key difference between IA and our previous stage mechanisms is that IA causes deliberate collisions. Hence, a node TX_1 using a resource r implies that two other nodes TX_2 and TX_3 can transmit simultaneously. Still, it may happen that these other nodes have not scheduled enough segments on resource r to fully exploit r. Figure 6.16 depicts an example. TX_1 needs to transmit six packets, whereas TX_2 and TX_3 have two packets each. Hence, our IA allocation and scheduling algorithms produce the frame shown in the upper left corner of Figure 6.16. Essentially, TX_1 uses the double capacity link, while TX_2 and TX_3 use the single capacity links. However, this results in one empty slot both for TX_2 and TX_3, as the shaded areas in Figure 6.16 indicate. Although we cannot fully exploit the capacity of IA in this example, IA still achieves a higher throughput than OFDMA and SL. Concretely, OFDMA would need an additional subcarrier to schedule all segments, and SL would require more time than IA to transmit the packets of all transmitters in a TDMA fashion. Figure 6.16 shows the resulting frames for both cases to scale. Nevertheless, the advantage of IA strongly depends on the particular CSI and traffic conditions. For instance, if TX_2 in Figure 6.16 would only have one packet to transmit, SL would achieve the same throughput as IA since IA would result in even more empty slots.

Implementations of our IA stage mechanism design may deal in different manners with the aforementioned empty transmission slots. For example, TX_2 and TX_3 may not transmit anything in these slots in order to save energy. This does not prevent the operation of IA—the receivers of the corresponding stream would simply receive zeros. Alternatively, TX_2 and TX_3 could exploit resource r to transmit local traffic such as control messages, and thus reduce the stage maintenance overhead. However, this requires additional signaling to let the receivers distinguish payload data from control data. In our implementation, TX_2 and TX_3 transmit random data if they have not scheduled any segments on resource r. The underlying reason is related to evaluation purposes. Similarly to our OFDMA stage mechanism implementation, instead of transmitting all scheduled segments, our prototype only transmits a limited number of segments on each resource. This keeps the experiment runtime reasonable. We then extrapolate our results to determine the transmission time of all segments. To this end, we average the transmission time of the segments that we actually transmit on each stream, and then multiply the result with the number of segments scheduled for that stream. The more segments we transmit, the more accurate our averaging is. Hence, we transmit random data on the aforementioned idle slots to improve our average for the corresponding stream. As in OFDMA (c.f. Figure 5.10), the stage transmission time is the transmission time of the stream that takes most to finish. Our scheduling algorithm in Section 6.2.3.2 aims at minimizing the resulting idle slots on the streams that finish earlier. However, this requires an accurate estimation of the number of Strider batches that a transmitter needs to convey a packet at a receiver. As we discuss in Section 4.2.3.1, this is challenging in practice, and may result in significant inefficiencies, as shown in Figure 4.5. Hence, using a baseline with per-subchannel scheduling is crucial to obtain comparable results in practical experiments.

6.3.2 *Practical Issues*

Our IA stage mechanism must deal with the same practical issues as our single-hop frequency IA system in Section 6.1.4. Hence, we adopt the same practical solutions as in the single-hop case. This includes the assumptions that (a) all transmitters share the same clock, and (b) all transmitters start transmitting within the duration of a sample in order to cause the same CFO and STO, respectively, at a certain receiver. In our single-hop implementation, we ensure both (a) and (b) by allowing all transmitters to share the same WARP board. However, for practical multi-hop experiments we need a larger setup, such as the mesh SDR testbed we describe in Section 3.6. Any combination of the 20 nodes in this testbed may need to act as transmitters of a stage if our corridor construction protocol chooses them as part of a corridor. However, we cannot place all 20 nodes on a single WARP board since each board can have at most four radio interfaces. Regarding frequency synchronization, we solve this issue using the WARP Clock Board, which allows us to set one board as a clock master and the remaining boards as clock slaves. In contrast, accurate time synchronization using WARPLab is more challenging. WARPLab transmits a "synch" packet to the boards via Ethernet to trigger transmissions. While this results in a simultaneous transmission of all boards which are set as transmitters, the "synch" packet may arrive with slightly different delays to each board due to the characteristics of Ethernet. To avoid this, we can set a certain output pin of each WARP board to a high level as soon as it receives the "synch" packet. Additionally, all boards can trigger a transmission not only when receiving a "synch" packet via Ethernet but also when sensing a high level at a certain input pin. Thus, we only need to wire the output pin of a certain board, which we define as the time master, to the input pins of all other boards. As a result, we obtain accurate time synchronization among transmitters. Alternatively, we could also achieve wireless synchronization using solutions such as AirSynch [10].

6.4 EVALUATION

We evaluate our IA stage mechanism in simulation on corridors of up to five stages. While our WARP Drive implementation also allows for practical experiments (c.f. Chapter 7), we need simulations in order to analyze the performance of our IA stage mechanism on random channels. This is not possible in our lab because our practical channels are mostly constant. In other words, practical experiments would allow us to verify the practicability of IA but would only give us insights into a very limited scenario. However, our results in Section 6.4.2 show that evaluating IA for a variety of CSI conditions is crucial. For practical experiments which show the feasibility of frequency IA on our SDR testbed, we refer the reader to Section 6.1. In the following, we present the scenarios that we use in our simulations, and discuss the results.

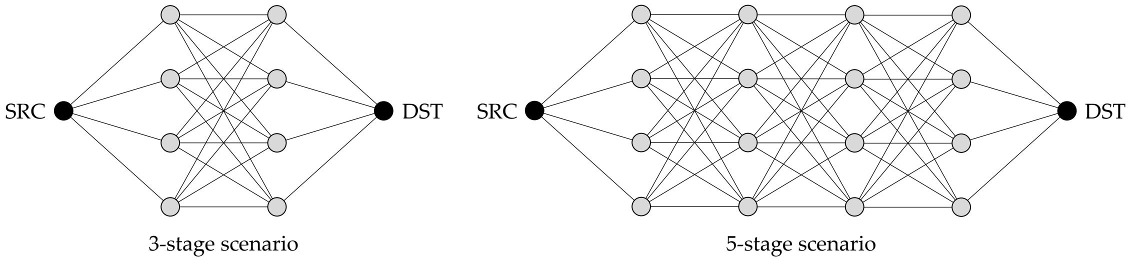


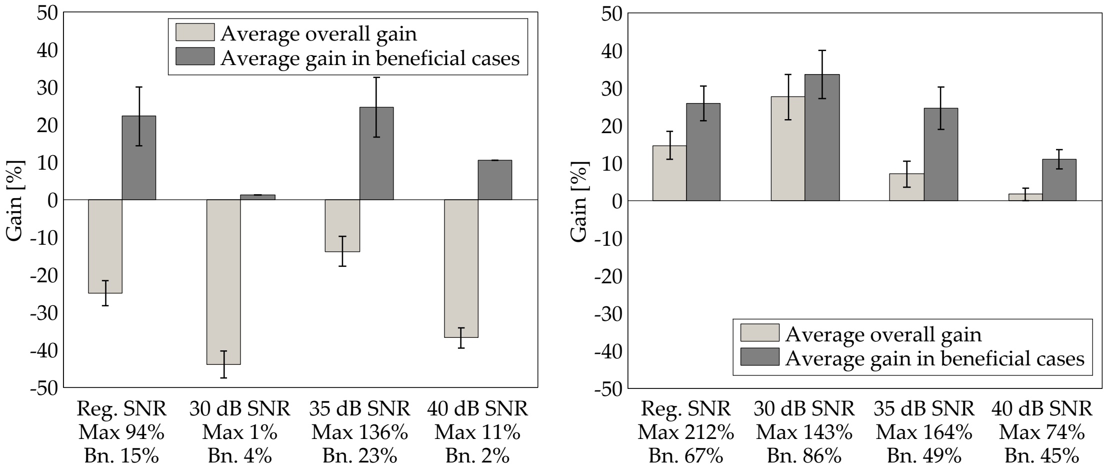
Figure 6.17: IA multi-hop simulation scenarios.

6.4.1 *Scenarios*

We perform simulations on corridors with three and five stages. Figure 6.17 shows both scenarios. In contrast to previous chapters, we do not place nodes randomly but use fixed locations. While this would have a significant impact if we would evaluate, for instance, a geographic corridor construction protocol, in our case it does not limit our results. The reason is that the channel characteristics that actually influence our IA allocation and scheduling schemes are (a) the CSI, and (b) the average SNR. For each simulation run, we generate random CSI for each link in both of our scenarios. Moreover, we investigate multiple SNR values. Hence, despite the fixed location of our nodes, our simulations allow us to obtain general insights on the performance of our IA stage mechanism. Regarding, the SNR we consider two different settings, namely, regular and homogeneous SNRs. The former are the SNR values resulting from the topologies in Figure 6.17 according to the distance among nodes. In particular, the distance among the node groups in both of our scenarios is one meter, and we assume that the SNR at a distance of one meter from a transmitter is 30 dB. Hence, the horizontal links in Figure 6.17 have an SNR of 30 dB. We use a path-loss coefficient of $\alpha = 2$ to compute the SNR on the other links. In contrast, for the case of homogeneous SNRs, we set the same SNR on all links, regardless of the geometry of our scenarios. More precisely, we consider the SNR values 30 dB, 35 dB, and 40 dB. Due to the characteristics of IA, we expect IA to perform significantly better with high SNRs. Finally, we also study both wideband and per-subchannel scheduling (c.f. Chapter 4) for the SL stage mechanism that we use as a baseline. However, we do not expect a large difference among both scheduling types since our measurements in Section 4.2.3.1 show that the estimation of the transmission time performs well in simulation. For each parameter setting, we perform on average roughly 300 simulation runs. Moreover, we use a codebook with 16 entries, which is a trade-off between the one-bit feedback of OFDMA, and the detailed feedback of IA. The underlying reason is to achieve a balance between accuracy and overhead—providing full CSI would be beneficial for IA but would incur unnecessary control data if the stage decides to use OFDMA.

6.4.2 *Results*

In the following, we present our simulation results for our IA stage mechanism. More precisely, we first discuss its performance for different SNR values in our 3-stage scenario. After that, we analyze our 5-stage scenario, and the impact of per-subchannel scheduling. We do not allow the corridor to switch among stage mechanisms, that is, in the IA experiments we always use our IA stage mechanism if $m \geqslant 3$ and $n \geqslant 3$.



(a) Throughput gain of IA over SL.

(b) Throughput gain of OFDMA over SL.

Figure 6.18: Throughput gain of IA and OFDMA compared to SL in our 3-stage scenario.

6.4.2.1 *SNR Study*

GAIN OVER SL. Figure 6.18a depicts the throughput gain of our IA stage mechanism compared to an SL baseline using wideband scheduling. For all SNR values, we observe that, on average, IA results in negative gains. In contrast to OFDMA in Figure 6.18b, always using IA to forward data along a corridor often does not pay off. However, if we only consider the beneficial cases, which are the cases that yield a positive throughput gain, IA achieves 20% average gain for some SNR values. Hence, for the beneficial cases, we achieve gains in the same order of magnitude as OFDMA in Figure 6.18b. The best IA simulation run achieves up to 136% throughput gain compared to SL, as shown in the label of the 35 dB case in Figure 6.18a. Hence, if IA is beneficial for the current channel and traffic conditions, it provides significant gains. This occurs in a much smaller fraction of the simulation runs than for OFDMA, as the percentages of beneficial cases in Figures 6.18a and Figure 6.18b show. In particular, while OFDMA is beneficial in up to 86% of the simulation runs, IA only pays off in at most 23% of the cases.

Regarding the SNR, in Figure 6.18a we observe that the regular SNR case achieves a significant throughput gain in the beneficial cases. In contrast, the cases with homogeneous SNRs on all links result in a peak at 35 dB but lower gains at both 30 dB and 40 dB. This is a clear difference to the behavior of OFDMA in Figure 6.18b. In particular, OFDMA achieves smaller gains for higher SNRs because the SL baseline suffers from less deep fades (c.f. Chapter 5). Our measurements suggest that the reason for the aforementioned IA peak is related to the number of subcarrier triplets that use IA. In particular, since the corridor width is four, we use IA+OFDM 4×4 as defined in Table 6.1. Thus, even though we always choose IA as a stage mechanism, individual triplets may resort to OFDM if IA results in a large noise impact on them. Our experiments show that the middle stage in our 3-stage scenario uses IA on more triplets at 30 dB and 40 dB than at 35 dB. This matches the behavior we observe in Figure 6.14 of our single-hop evaluation. Equivalently to the discrete-rate case in Section 6.1.5.4, Figure 6.18a suggests that at both 30 dB and 40 dB, OFDM requires a similar number of batches than IA to convey a Strider packet. Hence, the stage uses IA on a large number of triplets. In contrast, at 35 dB, OFDM requires a lower

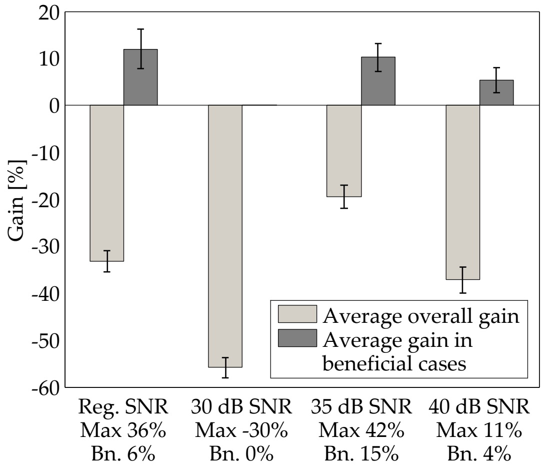


Figure 6.19: Throughput gain of IA over OFDMA.

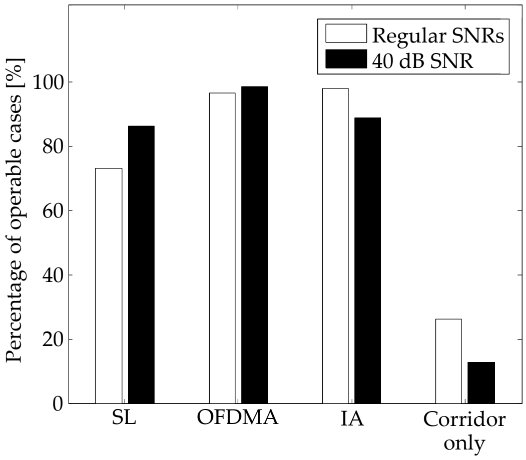
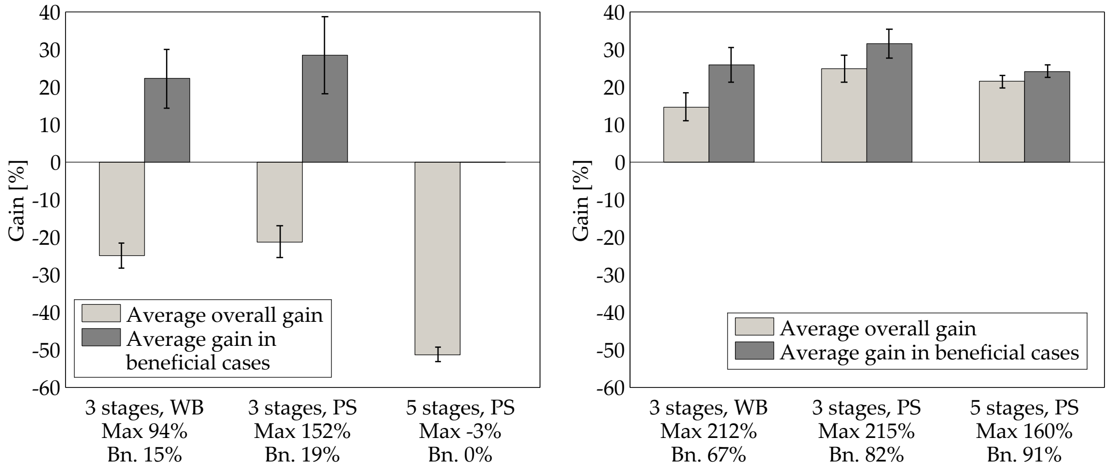


Figure 6.20: Fraction of successful experiments.

number of batches than IA. Thus, the stage selects IA only for few triplets. Moreover, our measurements reveal that roughly 30% of the triplets for which the stage uses IA require more transmission time than the average. Also, an in-depth analysis of our data shows that subsequent IA transmissions on the same triplet with identical CSI sometimes exhibit a large difference in terms of transmission time. While this also occurs for OFDM, the difference in that case is significantly lower. Most probably, the reason is that random noise variations cause unstable IA decoding for certain CSI conditions. In other words, occasionally (a) the stage chooses IA for unsuitable triplets, and (b) the CSI of some triplets yields an unstable behavior. This results in significant scheduling inefficiencies, since the checkerboard area as in Figure 4.5 becomes very large. Further, such inefficiencies are more probable the more triplets use IA. Hence, they occur more often at 30 dB and 40 dB than at 35 dB, which results in the peak effect that we observe in Figure 6.18a. A somewhat surprising result in Figure 6.18a is that the performance of IA for 30 dB is significantly worse than for regular SNRs, although the latter also features SNRs of about 30 dB. Our measurements show that the number of IA triplets is larger in the 30 dB case. This matches Reference [95], which shows that the noise impact on IA is lower the more similar the SNR of all involved links is. According to our above discussion, this increases the probability of scheduling inefficiencies, which ultimately leads to a low throughput.

GAIN OVER OFDMA. Our results in Figure 6.18 suggest that the beneficial cases of IA perform on average worse than OFDMA. This raises the question whether a stage should use IA at all if it can use OFDMA, particularly since OFDMA poses less requirements in terms of synchronization. Figure 6.19 provides the answer. We depict the gain of IA compared to OFDMA. As expected, the average throughput gain is negative. However, for the beneficial cases, IA achieves roughly 10% average throughput gain for the 35 dB case. In the best case, the gain raises up to 42%. The percentage of beneficial cases in Figure 6.19 shows that such cases are infrequent but nevertheless exist. We conclude that, under specific link and traffic conditions, IA can provide significant throughput gains compared to OFDMA. IA benefits particularly when a node (a) exhibits a stable and low noise impact according to Section 6.1.3.1 on a double capacity link, and (b) has a large amount of data to forward on that link. Hence, deciding whether to use IA or OFDMA on a certain stage is crucial to achieve large throughput gains compared to SL.



(a) Throughput gain of IA over SL.

(b) Throughput gain of OFDMA over SL.

Figure 6.21: Throughput gain of IA and OFDMA compared to SL for different scenarios.

ROBUSTNESS. Finally, we study the robustness of our stage mechanisms in terms of the fraction of experiments for which they are operable. Figure 6.20 shows our results. While SL only works in roughly 80% of the cases, both OFDMA and IA are operable in nearly 100% of our simulation runs. Hence, we conclude that the spatial diversity of the corridor allows IA to avoid poor resource combinations, which results in a significantly more robust operation than SL. The last column in Figure 6.20 highlights this behavior. It depicts the fraction of experiments for which only a corridor—IA or OFDMA—allows the source to communicate with the destination. Further, Figure 6.20 also shows the operability difference between the regular SNR case and the 40 dB case. As expected, for higher SNRs, SL and OFDMA are operable in a larger fraction of the experiments since more subchannels exhibit good performance. In contrast, IA works in less cases. From our above results, we deduce that the underlying reason is the number of triplets for which our IA stage mechanism resorts to OFDM. Most probably, the stage uses IA on more triplets for the 40 dB case than for the regular SNR case. According to our results in Figure 6.18a, this increases the probability that at least one of the triplets requires significantly more batches than expected. In the extreme case, such triplets may exceed the maximum number of batches that we consider for our system (c.f. Table 4.2), resulting in a transmission failure. Nevertheless, IA still is operable in roughly 90% of the cases.

6.4.2.2 *Scenario Study*

In the following, we study the impact of the scheduling type, and of the length of the corridor. For all of our results in this section, we use the above regular SNR setting.

SCHEDULING. We compare the throughput gain of IA and OFDMA to SL in Figures 6.21a and 6.21b, respectively. In both cases, we observe that the throughput gains improve slightly when switching from wideband to per-subchannel scheduling. As a result, IA achieves close to 30% average and 152% maximum throughput gain in the beneficial cases compared to SL. At first, this seems contradictory to our results in Section 5.4.2.2, which show that the performance of SL improves with per-subchannel scheduling. Hence, we

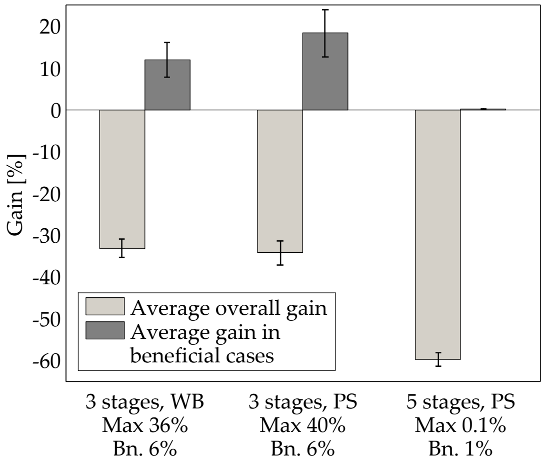


Figure 6.22: IA vs. OFDMA in multiple scenarios.

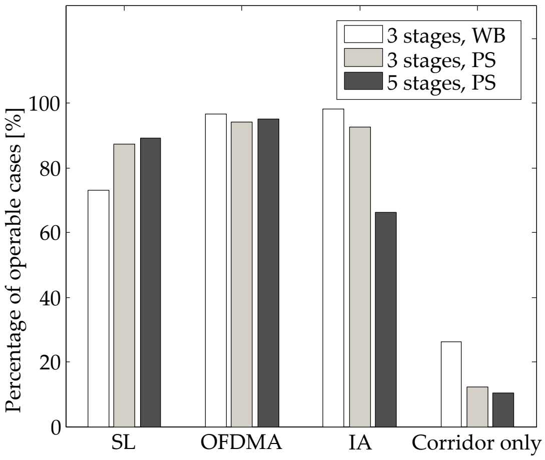
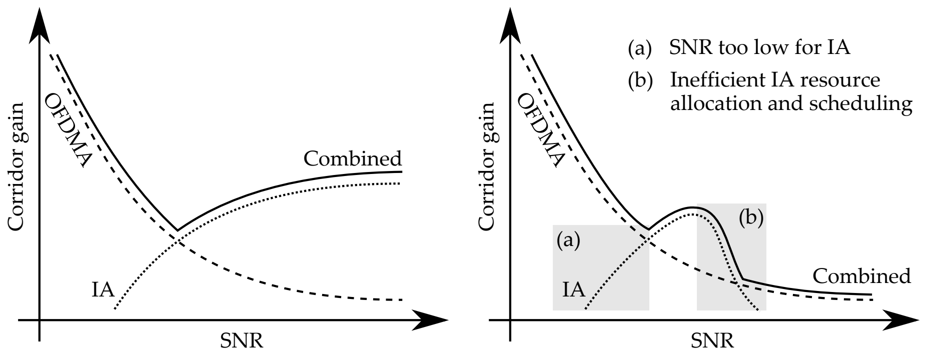


Figure 6.23: IA operability in multiple scenarios.

would expect the throughput gain to decrease in Figure 6.21. However, our measurements show that both the average SNR and the stage width have a significant impact on the improvement of SL when switching to per-subchannel scheduling. In contrast to Section 5.4.2.2, in Figure 6.21 the stage width is four, and the SNR is about 30 dB. The large stage width helps wideband scheduling to avoid deep fades. Further, the high SNR limits the advantage of per-subchannel scheduling, since most subchannels can operate at a high Strider rate also for the wideband case. For such a scenario, our experiments reveal that wideband scheduling often outperforms the per-subchannel case. As a result, we observe the aforementioned improvement of the throughput gain in Figure 6.21. In Figure 6.22, which compares IA and OFDMA, we also observe a slight improvement in the beneficial cases when switching to per-subchannel scheduling. This is surprising, since the type of scheduling only affects SL. The reason is that, by chance, our per-subchannel experiment yields slightly better beneficial cases than the wideband case. Still, the average throughput gain, which includes more experiment runs, is the same for both, which confirms that the scheduling type does not affect the gain of IA compared to OFDMA.

CORRIDOR LENGTH. Figure 6.21a shows that the throughput gain of IA drops in our 5-stage scenario, that is, in corridors with multiple consecutive IA stages. Essentially, we observe no beneficial case at all. To achieve a positive throughput gain, our previous results suggest that all IA stages of a corridor would need to feature (a) beneficial CSI for IA, and (b) a suitable data distribution at the transmitters. In our 3-stage scenario, we observe that the probability of achieving such a situation in a corridor with one IA stage is limited. Hence, the probability for multiple consecutive IA stages becomes marginal. In contrast, Figure 6.21b shows that OFDMA achieves large gains also in our 5-stage scenario. In other words, while IA is highly sensitive to CSI and traffic, OFDMA is mostly agnostic to these factors and thus provides gains in a broad range of situations. Hence, we conclude that the key to exploit IA on corridors is to use it exclusively on stages for which it actually provides gains given their CSI and traffic, and resort to schemes such as OFDMA otherwise. For further results on such a system, we refer the reader to Chapter 7.

ROBUSTNESS. The above results show that using IA on multiple subsequent stages often does not pay off. Nevertheless, IA is operable in such scenarios. In particular,



(a) Ideal combination of IA and OFDMA. (b) Current combination of IA and OFDMA.

Figure 6.24: Throughput gain characteristics of our IA and OFDMA stage mechanisms.

Figure 6.23 shows that IA successfully delivers end-to-end data in our 5-stage scenario in more than 65% of the cases. Still, we observe a percentage drop compared to the 3-stage case. Most probably, the reason is analogous to the effect we observe in Figure 6.20. Concretely, in comparison to a corridor with a single IA stage, transmitting data on multiple consecutive IA stages translates into an overall larger amount of triplets that use IA. Hence, the probability that a stage uses IA on triplets which are either unsuitable or exhibit an unstable behavior, increases (c.f. Section 6.4.2.1). Consequently, the probability of exceeding the maximum number of batches also increases. In contrast, OFDMA on the 5-stage corridor is operable in nearly 100% of the cases since OFDMA works on most CSI settings. Regarding the impact of the scheduling type, OFDMA and IA achieve roughly the same percentage of operable cases for both wideband and per-subchannel scheduling. This is expected, since they are agnostic to the scheduling of SL. The slight differences for both cases in Figure 6.23 are due to random variations in our measurements. However, we observe that SL delivers packets successfully more often for per-subchannel scheduling than for the wideband case. That is, while per-subchannel scheduling does not yield a higher SL throughput according to Figure 6.21, it improves its robustness. The reason is that per-subchannel scheduling allows SL to adapt to subchannels that exhibit an average SNR lower than the average channel SNR. Concretely, SL adapts the number of Strider batches on those subchannels to their particular SNR. In wideband scheduling, SL cannot take such subchannels into account, and thus results more often in transmission failures.

6.4.3 *Discussion*

From our above results, we conclude that IA can (a) operate on corridors, and (b) achieve significant throughput gains in selected cases. Regarding (a), our experiments show that, in beneficial cases, IA achieves roughly the same average throughput gains at the NET than at the PHY in our single-hop analysis in Section 6.1. That is, IA achieves gains also as part of a system including PHY, LNK, and NET. Regarding (b), we conclude that our current IA stage mechanism design only provides gains in a very limited number of situations. The reason is twofold. First, the performance of IA is highly unstable for certain channel conditions. Second, our allocation and scheduling mechanisms follow an effective but sometimes inefficient approach, such as in the following examples.

- Our allocation mechanism combines the two streams of an IA double capacity link to transmit a single packet segment at a higher throughput. To this end, it encodes data assuming the SNR of the worst of both streams. If the SNR difference is large, this may result in a waste of resources. To address this, we could treat each stream independently, but this would result in a more complex allocation mechanism.

- Our allocation mechanism occasionally miscalculates the performance of certain IA streams, which results in scheduling inefficiencies. The reason may be that we decide which triplets to use for IA based on the estimated number of batches instead of the noise impact metric. Directly using the noise impact metric, such as in Section 6.1.3.1, may improve the performance of the allocation mechanism.

- Our scheduling scheme may result in idle streams, as described in Section 6.3.1. In our example in Figure 6.16, IA achieves the highest throughput compared to OFDMA and SL. However, this may not always hold. To avoid this, the scheduler could switch to a different PHY on those resources, and thus achieve a higher throughput.

Due to the limited range of situations for which IA pays off, using it as the only stage mechanism in a corridor is challenging. Still, the above efficiency improvements along with identifying unstable CSI conditions would relax the requirements of IA, and thus increase the probability that a stage can benefit from IA. This would also avoid the peak effect that we observe in Figure 6.18a. That is, the throughput of our IA stage mechanism would not fall due to scheduling inefficiencies the more triplets use IA. Instead, more IA triplets would result in an increase of the throughput. We would still observe fluctuations such as in Figure 6.14, but the higher the SNR, the closer we would get to the theoretical 33% maximum IA PHY gain. Hence, we would observe the inverse behavior than for OFDMA, which according to our experiments in Chapter 5 results in smaller gains at high SNRs. In other words, OFDMA and IA would achieve large gains at low and high SNRs, respectively, as shown in Figure 6.24a. Thus, the corridor could combine them in order to adapt to a wide range of situations. However, our current system results in the behavior depicted in Figure 6.24b. That is, IA only provides gains if the SNR is large enough to enable IA operation but low enough to avoid that a large number of triplets use IA. In Chapter 7, we investigate a system which combines IA and OFDMA.

6.5 SUMMARY

In this chapter, we introduce a stage mechanism based on IA. In particular, we use IA in the frequency domain, and consider a scenario with three transmitters and three receivers. IA enables one of the transmitters to transmit two data symbols on one resource. Hence, the transmitters can send overall four data symbols on three resources, that is, they achieve a throughput gain of 33%. However, we observe that the performance of such an approach is limited in practice due to its high SNR requirements. More precisely, the decoding process may augment the noise at the receivers. Hence, we first present an approach that enables practical frequency IA for a single-hop scenario. To this end, we determine analytically the factor which augments noise. We find out that it depends solely on the channel coefficients. Moreover, we design a metric which captures this augmentation as a noise impact value. In order to minimize this value, our solution exploits diversity. Basically, the noise impact depends on which transmitters communicate with which receivers using which subcarriers. For a stage with three or more transmitters and receivers, we obtain a number of different resource combinations on top of which we can use IA. Hence, we compute the noise impact factor for each of the possible resource combinations, and choose the one with the smallest noise impact. However, the challenge is that even small stages lead to a large amount of combinations. To make this exhaustive search tractable, we use heuristics based on the characteristics of the wireless medium that reduce the search space significantly. This enables us to compute the best resource combination before the CSI feedback that we need as an input becomes outdated. Based on this design, we present the first practical frequency IA system. We address practical issues such as correcting CFO and STO at the receivers, and finding a suitable codebook size tailored to the requirements of IA for efficient CSI feedback. Our results show that choosing a resource combination which minimizes the noise impact is crucial to enable frequency IA in practice. Moreover, we achieve up to 20% throughput gain compared to SL. Further, our measurements illustrate that minimizing the noise impact factor allows us to achieve more robust and stable IA transmissions than a traditional IA scheme.

Next, we extend our IA single-hop prototype to a full-fledged stage mechanism. While in our above design we assume that transmitters always have data to send to any receiver, the amount of data at each transmitter for a stage mechanism depends on how packet segments split and join in the previous corridor stages. Hence, nodes may not be able to fully exploit their resources if they only have a limited amount of data. To minimize the transmission time, we use an allocation mechanism which takes into account the stage CSI and the amount of data at each node. We implement our IA stage mechanism on WARP Drive, and evaluate it in simulation in order to cover a broad range of CSI values. We observe that IA only provides gains in a limited number of situations. Basically, IA requires that (a) the CSI allows the transmitters to find a resource allocation with low noise impact, and (b) the node which can transmit two symbols on one resource has more data than the other transmitters. Our measurements show that this happens at most in 20% of the cases. However, in those cases, IA achieves significant gains—on average, about 20% compared to SL, and 10% compared to OFDMA. Finding such beneficial cases on multiple consecutive stages using IA is unlikely. Hence, we conclude that corridors should not use IA as a default stage mechanism but only when it actually provides throughput gains compared to other PHYs. Still, more efficient allocation and scheduling mechanisms may broaden the range of CSI and traffic situations for which IA pays off.

7

SYSTEM EVALUATION

In previous chapters, we introduce (a) the architecture of Corridor-based Routing, and (b) three different stage mechanisms based on OFDM, OFDMA, and IA, respectively. Further, we evaluate each of these contributions individually. In this chapter, we bring them together in order to investigate the performance of Corridor-based Routing as a system. In particular, the key difference to our earlier evaluations is that we allow corridors to switch among different stage mechanisms according to CSI and traffic conditions. To this end, we introduce a *decision engine* which, prior to the transmission in a stage, selects the stage mechanism which achieves the shortest transmission time. This implements the mechanism selection component of stage maintenance in Figure 3.3. We depict the operation of the decision engine in Figure 7.1. To estimate the transmission time, we use the techniques that we introduce for each individual PHY in previous chapters. Concretely, we use Strider as an error correction and detection scheme, and estimate the number of batches that a transmitter needs to convey a packet segment at a receiver. We implement our Corridor-based Routing system in WARP Drive. Hence, we first introduce the operation of WARP Drive in Section 7.1. After that, we present and discuss our system evaluation results in Section 7.2. Finally, we summarize our insights in Section 7.3.

7.1 WARP DRIVE

Rapid prototyping on SDRs is limited when it comes to cross-layer mechanisms beyond one-hop wireless communication. Although frameworks exist that facilitate implementation on SDRs, the underlying programming model often relies on offline processing. This impedes interactive protocol operation. For example, while WARPLab allows for flexible implementation in Matlab, it incurs significant delays for transferring signals to the SDR, and processes data offline.

Such an offline approach is well suited for evaluating PHY techniques on real-world channels. However, it poses a number of significant challenges to cross-layer mechanisms spanning up to the NET. First, LNK and NET mechanisms often demand interactivity, such as answering requests from other nodes. This is not possible when operating offline. Second, network size is a critical parameter in multi-hop communication. The size of SDR testbeds is typically limited, and thus researchers often resort to simulations in order to investigate larger settings. This translates into implementing the system twice, that is, once on the SDR and once for the simulator. Finally, SDR rapid prototyping frameworks are

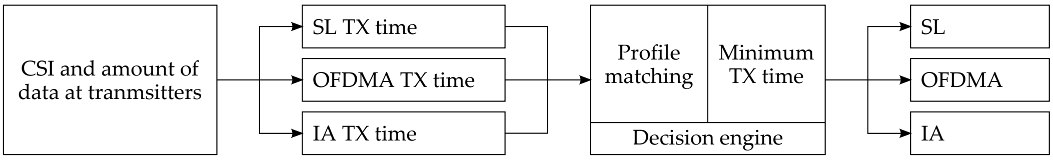


Figure 7.1: Decision engine for Corridor-based Routing. The CSI and the amount of data at each transmitter are the input data. The stage mechanism selection is the output data.

not designed for supporting multiple layers of a protocol stack. Thus, the resulting code is not modular, but needs to be highly bonded to the SDR prototyping framework itself on all layers to support cross-layer interactions. This causes a significant maintenance overhead, and hinders extensibility.

Existing frameworks that enable implementation of cross-layer mechanisms are often limited to traditional PHYs such as OFDM. Support for advanced lower layers such as OFDMA, IA, or MIMO as used in 802.11n is not available. For WARP, the alternative to the aforementioned non-real-time approach is realizing a system on its FPGA. While such a prototype is desired, the implementation is very costly in terms of time. A rapid prototyping framework which facilitates the SDR implementation of cross-layer mechanisms from the NET down to the PHY for multi-hop networks is missing. WARP Drive closes this gap. It builds on WARPLab but overcomes the aforementioned limitations using an event-driven programming model. Each transmission becomes an event and can be processed in non-real-time while still allowing for interactivity. In particular, our framework schedules these events in a *virtual timeline* and processes them sequentially. This also allows for adaptation to, for instance, channel characteristics as long as channels change slowly, such as in a lab environment. The execution of an event can generate new events which are added dynamically to the timeline, such as answering requests from other nodes. Mechanisms associated to events are encapsulated in well-defined blocks, resulting in modular code. Additionally, WARP Drive can send interchangeably via actual WARP boards or simulated channels on the same code basis. This common interface also decouples code from changes and version updates of WARPLab. In the following, we present the architecture of WARP Drive. Moreover, we provide performance results of our system in terms of processing time on third-generation WARP boards.

7.1.1 *Architecture*

The architecture of WARP Drive comprises two components, namely the modular network stack and the WARPLab interface[1]. The latter builds on the WARPLab framework provided by the WARP Project [180], which in turn interacts with the WARP hardware itself. Figure 7.2a gives an overview on all components of our architecture.

7.1.1.1 *WARPLab*

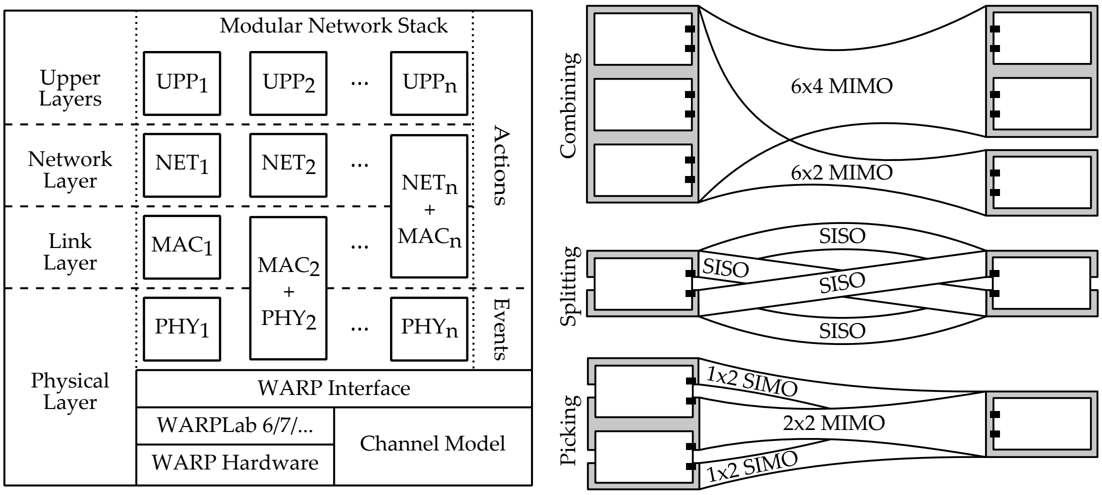
The WARPLab framework allows to interact with the WARP boards via Matlab. Essentially, a central computer processes the down-converted raw samples of all boards in the testbed. For each transmission, the computer calculates the transmit samples and sends them via Ethernet to the transmitting board. This board transmits the samples over the actual wireless medium to the receiver board, which in turn sends the received signal again via Ethernet to the central computer for further processing.

7.1.1.2 *WARPLab Interface*

The WARPLab interface follows an object-oriented approach to model the underlying hardware. It provides a fixed set of functions to the code running on top of it. Thus, new WARPLab versions do not require WARP Drive users to update their code. The current implementation of the interface supports WARPLab versions 6 and 7.

---

[1] The WARPLab interface is a contribution of Matthias Schulz (`mschulz@seemoo.de`) to WARP Drive.



(a) WARP Drive architecture.

(b) Node abstraction.

Figure 7.2: WARP Drive design. Part (b) shows how WARP Drive can combine multiple boards to form nodes with any number of antennas. The shaded areas are the resulting nodes.

NODE ABSTRACTION. The WARPLab interface abstracts from the underlying hardware. It provides node entities with one or multiple antennas which it can map to any combination of WARP antennas in the testbed. In particular, the interface can map antennas from different WARP boards according to three patterns, as depicted in Figure 7.2b.

- Antenna combining merges all antennas of two or more boards to one multi-antenna node. This enables WARP Drive to emulate, for instance, large MIMO systems.
- Antenna splitting defines a subset of the antennas of one board as one node. We use this in our mesh testbed shown in Figure 3.6b to define each antenna of a board as an individual node, and thus obtain a large network with only a few boards.
- Antenna picking joins subsets of antennas of multiple boards to form a single node, resulting in a combination of antenna combining and splitting. This allows for high flexibility, since any antenna configuration is possible.

Combining antennas of different nodes raises practical issues regarding synchronization of the underlying boards. However, the WARP Project provides means for wired synchronization among the boards forming one abstract node. This does not limit the validity of the resulting testbed experiments, since such synchronization only affects boards which in a real-world deployment would be part of one single physical node.

MODULAR DESIGN. The WARPLab interface modularizes hardware elements such as radios, boards, and the complete testbed to make code changes simple and efficient. The aforementioned node abstraction builds on these modules, allowing to change antenna mappings easily. The "WARPLab" module comprises all WARP boards involved in the experiment, triggers the transmission of samples, and handles channel simulation when running experiments offline. Next, the "board" module represents a WARP board and all its parameters, such as channel selection and the AGC. Moreover, it sets up and transfers samples to the board. Each board incorporates one "radio" module per antenna. This module controls the baseband and radio frequency amplifier gains of each radio. Finally, the "node" module maps WARP antennas to abstract nodes. It can span multiple "board" modules, as it may include antennas from any testbed board.

SWITCHING TO SIMULATIONS. As shown in Figure 7.2a, the interface can transmit samples both over simulated channels, and the actual wireless medium. Switching between both modes is seamless and does not require any code change. Simulations are not limited to one transmitter and one receiver, but are possible for any network size S. In particular, the interface models all $S \cdot (S-1)/2$ links of the network as Rayleigh channels. Each link is associated to an average SNR value which determines its quality. That is, the model is based on a fully-connected network to account for any possible interaction between nodes at the PHY. As a result, it reflects exactly the same links between the WARP boards that would compose the testbed in the practical case. We assign low SNRs to far away node pairs, or otherwise impaired links. If such a link results in transmission errors, a neighborhood discovery protocol in our modular network stack would not find the link, equivalently to the behavior of such links in the practical case.

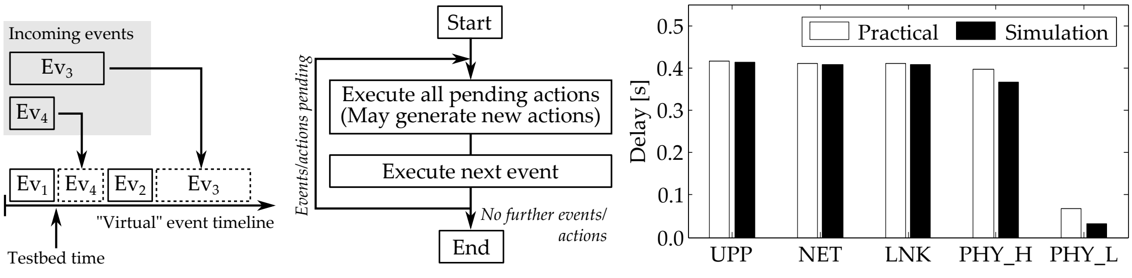
7.1.1.3 *Modular Network Stack*

In the following, we describe the architecture of our modular network stack. We design it in Matlab, based on the aforementioned node abstraction. For our system experiments in this and earlier chapters, we implement our mechanisms as part of this network stack.

ASSUMPTIONS. Similarly to most host-based SDR solutions (c.f. Section 2.3.3), WARPLab incurs significant delays for transferring samples to the aforementioned central computer, and for processing in Matlab. We cannot solve this inherent limitation but we circumvent it assuming a static scenario. That is, we consider that network changes occur at a larger timescale than the delays. While this would be a strong assumption for production systems, it is realistic for a lab environment and thus suitable for our rapid prototyping framework. Essentially, we process data and run protocols similarly to a production system but much slower. In turn we can implement protocols in Matlab, which speeds up prototyping, and allows us to get insights into the performance of WMNs quickly.

VIRTUAL TIMELINE. To allow for interactive LNK and NET protocols despite the aforementioned processing delays, we use a *virtual timeline*. Intuitively, the virtual timeline keeps track of how long transmissions would take without the processing delays that WARPLab incurs. The resulting timing values can be used for, e.g., estimating throughput. In particular, the timeline accounts for the data transmission time, that is, for the time the medium is in use. To this end, we define events and actions as follows.

- EVENTS. We call a transmission that occupies the medium an *event*. WARP Drive can schedule events on the timeline at any point in time, as long as the medium is not yet occupied. An event may involve multiple transmitters and receivers.
- ACTIONS. In contrast, we call the processing at nodes *actions*. The timeline does not account for the processing time of actions, since this is specific to the implementation. For instance, in most cases, a prototype of an algorithm in Matlab has a longer processing time than a production version on optimized hardware.



(a) Virtual timeline. (b) Main loop. (c) Cumulative processing delay at each layer.

Figure 7.3: Operation and performance of WARP Drive.

Events are related to the PHY, while actions refer to the rest of the stack. As WARP Drive processes events, the *testbed time* advances. Both actions and events can trigger new events which WARP Drive schedules in the timeline as soon as possible, either after the current testbed time, or at a given point in time. The latter allows to schedule transmissions in the future. In both cases, the timeline finds a suitable gap for the event. For instance, in Figure 7.3a Event 3 does not fit in between Event 1 and Event 2, and is thus scheduled after Event 2. WARP Drive estimates the duration of events during scheduling but can correct it later. Explicit synchronization of actions and events is not needed since both include all parameters needed to be executed individually. This fixed scheduling of events does not inherently allow for backoff-based LNK protocols such as CSMA/CA, but the modular network stack can emulate such a behavior for prototyping purposes, and schedule events accordingly.

PROCESSING LOOP. Figure 7.3b depicts the main processing loop of our modular protocol stack. Each loop iteration processes one event, that is, one transmission takes place and the testbed time advances to the point in time immediately following the transmission. Additionally, WARP Drive executes all pending actions. If there are no more pending events or actions, the experiment finishes. Both actions and events are code modules that take a certain set of parameters as inputs, and typically generate new actions or events that continue processing their output. The boundaries of each module are flexible, that is, cross-layer interactions ranging from soft information to fully merged layers are possible throughout the protocol stack. As long as modules are compatible to each other, the system can exchange them flexibly. This allows us to prototype corridors that switch among stage mechanisms at the PHY.

7.1.2 *Benchmarks*

In the following, we evaluate the performance of WARP Drive using a simple single-hop transmission. We do not include further functionalities, such as neighborhood discovery, since we aim at capturing the performance of the framework itself. In particular, we study the processing delays of WARP Drive as perceived at each layer. This is crucial since our design assumes that the coherence time of the channel is longer than the delay. In other words, this section gives an intuition on how stable channels in a lab environment must be in order to guarantee that our framework operates correctly.

7.1.2.1 *Implementation*

Our implementation of the aforementioned single-hop transmission includes essential functionality at the PHY, LNK, NET, and upper layers (UPP). We divide the PHY in "lower" and "higher"—the former transmits IQ-modulated data symbols using OFDM while the latter handles modulation and coding of binary data at a fixed rate. Moreover, the higher PHY enables unicast, multicast, and broadcast addressing, as well as frame fragmentation. The LNK triggers retransmissions, and provides adaptive modulation and coding. Finally, the NET forwards data assuming routing state is available.

7.1.2.2 *Results*

Time-critical functions typically reside at the PHY and LNK since the NET operates at a much larger timescale. Thus, we quantize the processing delay in a single-hop scenario with one transmitter and one receiver. In particular, we measure the transmission delay for both simulation and practical transmissions using third-generation WARP hardware. Figure 7.3c shows the *cumulative* delay at each layer. For instance, the value for the NET is the sum of the delays at PHY, LNK, and NET. As expected, we observe that the delay in simulation is consistently smaller than for the practical case. The reason is that, in simulation, WARP Drive does not need to send data to the WARP boards, and thus it does not incur delays for transferring samples. However, at higher layers the difference becomes smaller as simulations take more time on average—simulations use randomly generated channels and thus require retransmissions more often, whereas our practical testbed provides good channel conditions most of the times. More interestingly, Figure 7.3c shows a significant difference in delay between the lower and the higher PHY. The underlying cause is the convolutional encoder, which in our implementation is part of the higher PHY. Thus, the overall delay of about 0.4 s is to a large extent due to Matlab. The delay for transferring samples to the WARP boards is an order of magnitude smaller, at only 35.1 ms. We conclude that processing in non-real-time using WARP Drive is feasible as long as the coherence time of the channel is at least about 70 ms, which is the minimum delay incurred at the lowest layer. While further processing at higher layers requires larger coherence times, this is often feasible in lab environments.

7.2 EVALUATION

In the following, we use WARP Drive to bring together and evaluate the individual components of Corridor-based Routing that we present in previous chapters. That is, our experiments include corridor construction (c.f. Section 3), SL as a baseline (c.f. Chapter 4), and OFDMA as well as IA as stage mechanisms (c.f. Chapters 5 and 6). The key difference to our earlier evaluations is that we allow the corridor to *choose* among different stage mechanisms. To this end, we use the decision engine depicted in Figure 7.1. Hence, we evaluate our Corridor-based Routing system with respect to our goal in Section 1.2.3—allowing node subsets to use different PHYs according to channel characteristics. We expect to observe the continuous curve in Figure 6.24, which sketches the theoretical performance of a system combining OFDMA and IA. We focus on the effects that occur due to this combination, such as analyzing how often the corridor chooses a certain mechanism, and whether its decision is beneficial regarding throughput. For results on the individual performance of OFDMA and IA, we refer the reader to Chapters 5 and 6.

7.2.1 *Scenario*

Our results in Section 6.4.2 suggest that IA only provides gains under particular CSI and traffic conditions. Hence, for our evaluation in this section, we use simulations to cover a broad range of CSI values. Concretely, we run our experiments in the 3-stage and 5-stage scenarios that we introduce in Section 6.4.1, using the same simulation parameters regarding signal strength and path-loss. We list further parameters in Appendix A. We evaluate our system for two SNR settings, namely, the regular SNRs given by the distances among nodes, and homogeneous SNRs on all links. Further, we compare the performance in terms of throughput when using wideband scheduling to the per-subchannel case.

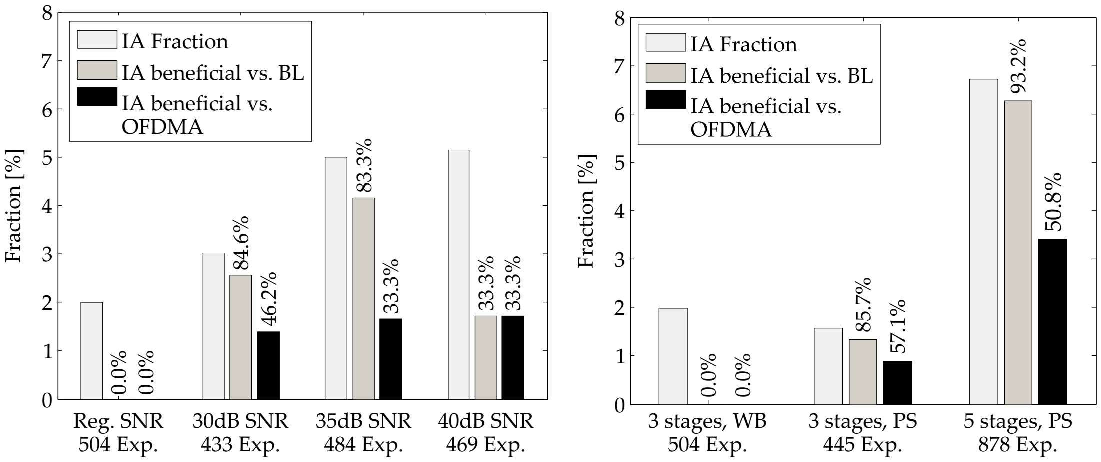
For each scenario and parameter setting, we study three corridor operating modes. The first mode is a baseline which builds an end-to-end corridor, and uses SL only. This mode stands for the performance of traditional forwarding mechanisms in WMNs which do not exploit spatial diversity. The second mode is our Corridor-based Routing system, which uses the aforementioned decision engine to choose among IA and OFDMA at each stage according to (a) the stage shape, and (b) the expected transmission time. For our 3-stage scenario, the corridor can use IA on the second stage only, since the other stages have either only one transmitter, or only one receiver. In contrast, our 5-stage corridor supports IA in the second, third, and fourth stages. If the CSI and traffic conditions of a stage are unsuitable for IA, the corridor resorts to our OFDMA stage mechanism. Since OFDMA achieves gains in most situations (c.f. Section 6.4.2), the corridor uses OFDMA most of the times on all stages, and only switches sometimes to IA. To evaluate whether choosing IA in those selected cases is correct, we introduce a third mode of operation which always uses OFDMA on all stages. This allows us to compare directly the end-to-end throughput of a corridor using IA on at least one of its stages, with the performance of the same corridor when using OFDMA always. In each experiment run, we measure the throughput for each of the three operating modes. To ensure that our three measurements are comparable, we use the same CSI for each mode. That is, we only generate new random CSI for each experiment run but not for each of the three measurements within a certain experiment. For each scenario and parameter setting, we run between 400 and 900 experiments. However, since the decision engine chooses IA rarely, the number of experiment runs that we can use to compute the average performance of IA corridors is very limited. Still, the results that we present in the following show clear trends, and match our expectations based on our insights from previous chapters.

7.2.2 *Results*

We divide our results into three studies. First, in Section 7.2.2.1 we analyze how often the corridor chooses IA and whether its decision is beneficial. After that, we focus on the throughput gain itself for different SNRs, and for different scenarios in Sections 7.2.2.2 and 7.2.2.3, respectively. Finally, we discuss our results in Section 7.2.2.4.

7.2.2.1 *Stage Mechanism Selection Study*

The light gray bars in Figure 7.4 depict the fraction of experiment runs for which the decision engine chooses IA as a stage mechanism. Moreover, the dark gray and black bars indicate how often this decision results in a larger throughput compared to SL and



(a) SNR study. (b) Corridor length and scheduling study.

Figure 7.4: Fraction of IA transmissions when using the decision engine.

OFDMA, respectively. The percentages show the later as a relative fraction. For instance, for the case of 30 dB homogeneous SNR in Figure 7.4a, the decision engine chooses IA in roughly 3% of the transmissions, out of which 84.6% result in a throughput larger than SL in the same scenario. Overall, we observe that the decision engine selects IA as a stage mechanism in, at most, 5% to 7% of the experiment runs. This matches our results in Section 6.4.2, which also suggest that IA is only beneficial for a limited number of CSI and traffic conditions. Hence, our decision engine performs well in that it does not select IA in an excessive number of cases.

COMPARISON TO SL AND OFDMA. Figure 7.4a shows that, in most scenarios, IA performs better than SL in roughly 85% of the cases where the decision engine chooses IA. This is expected, since IA has a larger DOF. However, the fraction of beneficial transmissions drops significantly for (a) the regular SNR case using wideband scheduling, and (b) the homogeneous SNR case at 40 dB. For case (a), IA is not beneficial in any of the already few cases for which the decision engine chooses it. Intuitively, this seems contradictory to Section 6.4.2, which shows that IA can provide good results in such a scenario. The key difference is that, in this case, we consider exclusively the experiment runs for which the decision engine chooses IA. That is, in case (a), the decision engine does not choose IA for any of the transmissions that would benefit from IA but only for other transmissions. In other words, it misses all the beneficial cases. Our results from previous chapters suggest two factors which may cause this behavior. First, in the regular SNR case, the SNR of each link in a stage is different, and thus the IA noise impact factor raises. As a result, the decision engine only chooses IA rarely. Second, the SL baseline in Figure 7.4a uses wideband scheduling, while IA operates on a per-subcarrier basis. Hence, while SL fully exploits its resources, the impact of idle subchannels may result in IA yielding longer transmission times than SL. Consequently, the few cases where the decision engine chooses IA do not turn out beneficial, either. The fraction of beneficial IA transmissions in the equivalent per-subchannel scheduling case in Figure 7.4b confirms this explanation. While the decision engine still chooses IA in a few cases only, it performs better than

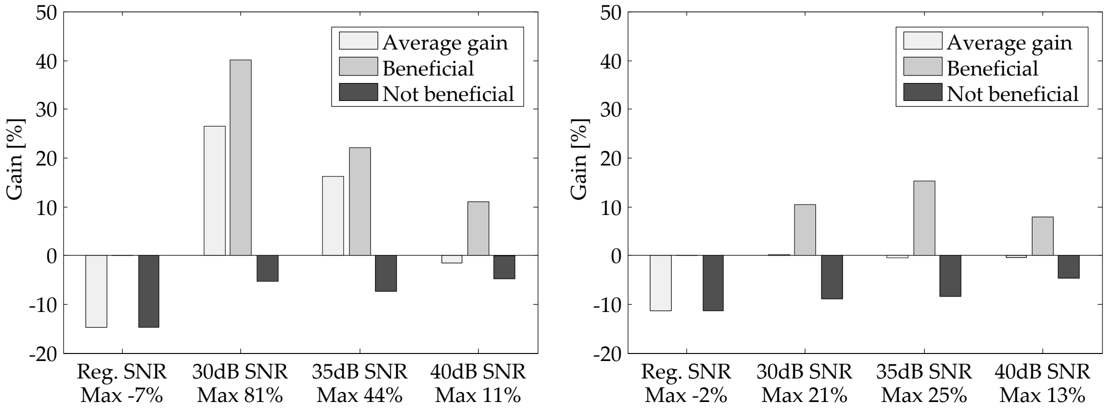
SL most of the times. The fraction of beneficial IA transmissions compared to SL also drops in case (b), that is, for a homogeneous SNR of 40 dB. In this case, the underlying reason is most probably the peak behavior that we discuss in Section 6.4.2. That is, the allocation mechanism of our IA stage mechanism decides to use IA on more subcarrier triplets due to the high SNR. Hence, the estimated transmission time decreases, and the decision engine chooses IA more often as a stage mechanism. However, a large number of IA triplets increases the probability of scheduling inefficiencies, resulting in a lower IA throughput than expected. Therefore, IA is only beneficial for the 40 dB homogeneous SNR scenario in one third of the cases where the decision engine chooses it.

In comparison to OFDMA, the fraction of beneficial IA transmissions is significantly smaller than for SL. More precisely, IA is only better than OFDMA in roughly 30% to 50% of the cases where the decision engine chooses IA. Our measurements show that, for the few cases for which the decision engine chooses IA, the advantage of IA compared to OFDMA in terms of estimated transmission time is often marginal. Due to the aforementioned scheduling inefficiencies, the slight advantage of IA may not hold for the actual transmission, resulting in the low fraction of beneficial IA transmissions that we observe in Figure 7.4a. Still, this does not mean that the throughput gain of IA is always marginal compared to OFDMA, as our results in Section 7.2.2.2 show.

SNR IMPACT. Figure 7.4a shows that the fraction of experiments for which the decision engine selects IA increases with larger SNRs. This matches theoretical work which suggests that IA requires large SNRs to operate reliably. However, the fraction of beneficial cases illustrates that the IA performance follows the aforementioned peak behavior. The key problem is that the decision engine takes the resource allocation as a basis to estimate the transmission time of IA. However, this allocation builds on a transmission time estimation which (a) may occasionally incur performance miscalculations, (b) does not capture the instability of certain CSI conditions (c.f. Section 6.4.3), and (c) does not take into account scheduling inefficiencies (c.f. Section 6.3.1). Consequently, the decision engine chooses IA in unsuitable cases. We sketch possible solutions for these issues in Section 7.2.2.4.

Further, we observe that the decision engine chooses IA more often in the homogeneous SNR scenario than in the regular SNR case, although both experience an SNR of about 30 dB. In the analogous experiment in Chapter 6, we observe that our IA stage mechanism uses IA on more triplets in the homogeneous case—most probably, the reason is that IA performs better in terms of noise impact if all participating links have similar SNRs [95]. Since the homogeneous case features more IA triplets than the regular case, we deduce that the former achieves a lower estimated transmission time than OFDMA more often than the latter. As a result, the decision engine chooses IA more often in the homogeneous SNR case. This may seem a significant impediment for practical IA since, in practice, stage links have different lengths than in the regular SNR case. However, measurements in our lab, as well as our practical IA results in Section 6.1.5.4, show that the SNR differences among indoor links of a stage are small, and not necessarily related to the link length.

CORRIDOR LENGTH IMPACT. Finally, we also study the impact of the corridor length on the fraction of experiment runs for which the decision engine chooses IA. In particular, Figure 7.4b suggests that longer corridors increase this fraction, since the decision engine chooses IA in nearly 7% of the cases for our 5-stage scenario using regular SNRs. However, this percentage includes all experiments that use IA in one or more of the three stages



(a) Throughput gain compared to SL.

(b) Throughput gain compared to OFDMA.

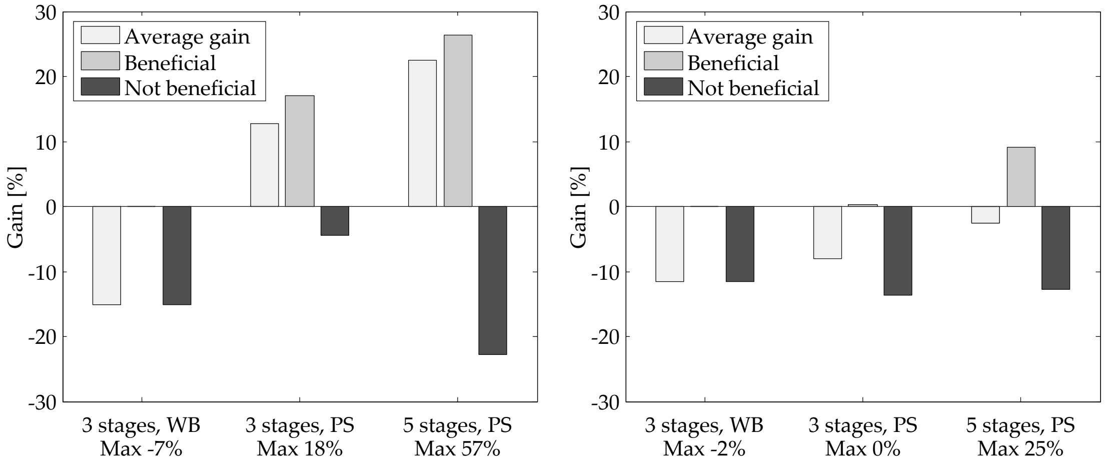
Figure 7.5: Throughput gain of IA compared to SL and OFDMA for multiple SNR values. The results consider only the transmissions for which the decision engine chooses IA.

that support IA. That is, the probability that one experiment run uses IA is three times larger compared to our 3-stage scenario. If we divide the fraction of experiments that use IA in our 5-stage scenario by three, we obtain results similar to the 3-stage case. Moreover, in contrast to our experiments in Section 6.4.2, longer corridors do not pose an additional challenge for IA. The reason is that we allow the corridor not to use IA if the CSI and traffic conditions are unsuitable, whereas in Section 6.4.2, we force the corridor to use IA on all stages. In other words, allowing the corridor to choose the PHY for each stage decouples the end-to-end throughput gain up to a certain extent from the corridor length.

7.2.2.2 *SNR Gain Study*

In Figure 7.5, we depict the throughput gains of our Corridor-based Routing system compared to SL and OFDMA for a range of SNR values. Concretely, we investigate the cases for which the decision engine chooses IA. In other words, Figure 7.5 illustrates the impact in terms of throughput gain of the fraction of experiment runs for which we use IA as depicted in Figure 7.4a. Moreover, we split the overall throughput gain into the contributions of the beneficial and the detrimental cases. That is, we show how severe the throughput loss is when the decision engine takes a wrong decision, and vice versa.

COMPARISON TO SL. Figure 7.5a shows that IA achieves significant throughput gains compared to SL. Specifically, the gain reaches up to 80% in the best case. Further, we observe that the gain decreases for increasing SNRs. This matches our above results—the higher the SNR, the more triplets use IA when the decision engine chooses our IA stage mechanism, and thus the more probable IA inefficiencies become. Further, while the throughput gain of the beneficial cases for homogeneous SNRs is always above 10%, the corresponding throughput losses in the detrimental cases are at around −5% only. Hence, the impact of unsuitable stage mechanism selections is limited. Moreover, in most cases, the average performance is better than for a traditional hop-by-hop forwarding scheme. We conclude that, despite the IA scheduling inefficiencies, IA does not undermine the throughput gain of corridors as long as the SNR on all stage links is roughly similar.



(a) Throughput gain compared to SL.

(b) Throughput gain compared to OFDMA.

Figure 7.6: Throughput gain of IA compared to SL and OFDMA for different scenarios. The results consider only the transmissions for which the decision engine chooses IA.

COMPARISON TO OFDMA. Figure 7.5b depicts the throughput gain of IA compared to OFDMA. As expected, the gain is smaller than compared to SL. However, IA still achieves roughly 10% gain on average for the beneficial cases with homogeneous SNR. Moreover, the maximum throughput gain reaches up to 25%. That is, our experiments show that IA can improve throughput significantly for individual cases, which is in line with our previous results. However, the average gain of the cases for which the decision engine chooses IA is essentially zero. This means that the wrong decisions of the decision engine undermine the correct decisions. Most probably, this occurs because of the performance miscalculations and the unstable CSI conditions that we discuss in Chapter 6. Further, we observe a peak behavior regarding the SNR, which confirms our intuition in Figure 6.24b. That is, the IA gain is initially small since OFDMA performs well at low SNRs. For middle SNRs, IA improves and OFDMA worsens, yielding a gain peak. Finally, for high SNRs, a large number of triplets use IA, and thus the probability of IA inefficiencies increases.

7.2.2.3 *Scenario Gain Study*

In the following, we perform a throughput gain study equivalent to Section 7.2.2.2 but for different scenarios. In particular, we investigate the impact of the scheduling type, and the corridor length. Figure 7.6 depicts our results. Similarly to Figure 7.5, we subdivide the average throughput gain into beneficial and detrimental contributions.

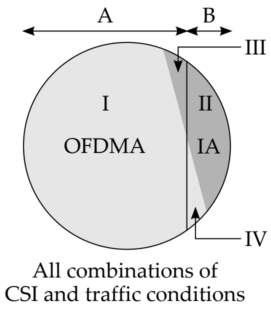
COMPARISON TO SL. In Figure 7.6a, we observe that, for the experiment runs for which the decision engine chooses SL, switching from wideband to per-subchannel scheduling results in a significant throughput gain compared to SL. This matches our explanation in Section 7.2.2.1 and our measurements in Section 6.4.2.2—since our scenario forces a stage width of four and a relatively high SNR, the performance of SL worsens when switching to per-subchannel scheduling due to the impact of idle subchannels. Moreover, the baseline and Corridor-based Routing are on a par with respect to scheduling, and thus the corridor benefits become evident. In particular, we achieve up to 18% throughput gain. Next, for the 5-stage scenario, we observe a large negative gain for the detrimental cases. This may result striking at first since the average gain is close to the gain of the beneficial cases. However, this effect is related to the frequency of the detrimental cases. As shown in Figure 7.4b, the beneficial cases occur 93.2% of the times, while the detrimental cases are rare at a fraction of only 6.8%. Hence, the negative gains only have a limited impact on the average. Moreover, our results show that Corridor-based Routing achieves an even larger gain than in the 3-stage scenario. Analogously to Figure 7.4b, the reason is that each experiment involves three instead of only one stage which is compatible with IA. As a result, the probability that the decision engine encounters CSI and traffic conditions which are beneficial for IA triples in each experiment. Otherwise, the corridor resorts to our OFDMA stage mechanism, which also yields significant gains compared to SL. This raises the question whether the gain that we observe in Figure 7.6a is exclusively due to OFDMA. However, Figure 7.6b illustrates that, also in our 5-stage scenario, IA does provide gains compared to OFDMA in some cases.

COMPARISON TO OFDMA. Figure 7.6b shows that switching from wideband to per-subchannel scheduling does not provide any improvement when comparing IA with OFDMA in the cases where the decision engine chooses IA. This is expected since both IA and OFDMA inherently operate on a per-subchannel basis. Although both are on a par in that sense, we observe that IA does not provide throughput gains compared to OFDMA in any of our experiment runs. In contrast, in our 5-stage scenario, we achieve on average 10% and up to 25% throughput gain in the beneficial cases. As discussed above, this may occur because a longer corridor increases the per-experiment probability of finding a stage with suitable CSI and traffic conditions. Moreover, since the corridor can always resort to OFDMA if needed, additional stages do not pose a challenge, in contrast to Section 6.4.2.2. Still, the average throughput gain over all cases is close to zero. From our previous experiments, we deduce two potential reasons that may cause this behavior. First, the reliability of the decision engine is limited due to the IA inefficiencies that we discuss in Section 6. Second, using a certain mechanism on one stage may result in an inconvenient data distribution among the transmitters of the next stage. In general, we expect the first reason to be the limiting factor since a corridor could always resort to SL if all other available stage mechanisms were inefficient due to the data distribution at the transmitters. Fortunately, improvements on the current transmission time estimation technique can most probably increase the reliability of the decision engine incurring only local overhead, i.e., overhead within a stage. In contrast, addressing the potentially unsuitable distribution of data at transmitters would incur costly multi-hop overhead.

7.2.2.4 *Discussion*

Based on our above results, we conclude that our decision engine captures the behavior of IA and OFDMA correctly. That is, it chooses IA rarely due to its specific CSI and traffic condition requirements, and resorts to OFDMA in most cases. We illustrate this behavior in Figure 7.7, which shows a schematic overview on how our decision engine performs. Overall, we observe the behavior that we sketch previously in Figure 6.24.

However, our results show that the decision engine does not always choose the best suited stage mechanism. In particular, it sometimes chooses IA even though OFDMA would perform better, and, vice versa, it sometimes appears to miss IA opportunities in favor of OFDMA. In Figure 7.7, area III represents the former, and area IV the latter. Although the



A	Actual fraction of experiments for which OFDMA performs best
B	Actual fraction of experiments for which IA performs best
I	Fraction for which we choose OFDMA, and which is beneficial for OFDMA
II	Fraction for which we choose IA, and which is beneficial for IA
III	Fraction for which we choose IA, but which is beneficial for OFDMA
IV	Fraction for which we choose OFDMA, but which is beneficial for IA

Figure 7.7: Schematic overview on the performance of the decision engine in our experiments.

throughput that each of our stage mechanisms can achieve is ultimately a result of the noise impact at the PHY, the challenges for improving the performance of the decision engine are related to the LNK and the NET. Concretely, our results in Sections 5.1.2 and 6.1.5 show that, most of the times, we can correctly estimate the noise impact at the PHY for both OFDMA and IA, respectively. In contrast, the LNK limits the performance of our decision engine because translating the noise impact at the PHY into an estimation of the number of batches needed to decode a packet at a receiver is challenging. Further, at the NET, the limitation stems from the idle slots that result from batch number estimation errors when deciding on which stage link to forward each packet segment. Fortunately, future work could potentially address both issues without incurring overhead. More precisely, the LNK could probably obtain a more reliable transmission time estimation directly using the CSI, which is anyhow available at the transmitters, instead of building on SNR values indirectly deduced from the CSI. Further, the NET could mitigate the impact of idle slots both a priori and a posteriori. For the former, it could schedule segments *before* the decision engine chooses the stage mechanism to avoid scheduling inefficiencies such as in Section 6.3.1. For the latter, it could dynamically use idle slots to transmit, for instance, control data, if a certain packet segment requires more batches than expected.

Finally, our experiments show that being able to choose the PHY mechanism for each stage allows the corridor to decouple the throughput gain from the corridor length. Basically, the stage mechanism selection adds a dimension to the adaptivity of the protocol. That is, if an intermediate stage in a long corridor cannot provide throughput gains even if adapting the parameters of the PHY in use, it can now switch to a different PHY and thus avoid becoming a bottleneck in the end-to-end corridor. For our experiments, this translates into corridors resorting to OFDMA for most of the stages but still being able to exploit beneficial channel conditions for IA. Hence, stage mechanisms can be categorized according to whether they are suitable for a broad range of scenarios, such as OFDMA, or only for very specific situations, such as IA. Nodes participating in a stage must implement at least one broad stage mechanism, but may additionally include specific stage mechanisms to fully exploit the capacity of the channel in particular situations.

7.3 SUMMARY

In this chapter, we evaluate our Corridor-based Routing system as a whole. That is, we consider all of its components that we introduce in earlier chapters, namely, corridor construction, the SL stage mechanism, the OFDMA stage mechanism, and the IA stage mechanism. To this end, we implement Corridor-based Routing on WARP Drive, which is a framework that builds on WARPLab to enable rapid prototyping of cross-layer protocols for wireless multi-hop networks on the WARP SDR platform. WARPLab is a reference design for WARP that allows flexible design of PHY prototypes in Matlab. It processes data offline and thus does not allow for the interactivity required by the LNK and the NET. WARP Drive avoids this limitation by using a "virtual timeline" to model the behavior of a real-time implementation using event-driven programming. Additionally, we present a modular network stack that enables protocols to span multiple layers, a modular interface for WARPLab that allows high flexibility regarding testbed configuration, and a seamless switch from practical to simulation results on the same code basis.

For our implementation on WARP Drive, we introduce a decision engine. The decision engine is a component which chooses for the current stage the stage mechanism out of SL, IA, and OFDMA that minimizes the transmission time. As an input, it uses the resource allocation that each stage mechanism would use for the current stage, and the amount of data that each transmitter needs to transmit. In our experiments, we compare the throughput performance of a corridor that can switch among OFDMA and IA to (a) a corridor that only uses OFDMA, and (b) a traditional forwarding baseline using SL. We observe that, compared to (a), our corridor achieves roughly 10% throughput gain for the stages where IA is beneficial. Compared to (b), stages using IA achieve about 25% gain on average, and up to 40% gain for the beneficial cases. Moreover, we evaluate the reliability of our decision engine, that is, whether the engine chooses the most suitable stage mechanism for each case. While our decision engine chooses the correct stage mechanism in most cases, sometimes it incurs significant throughput losses due to wrong decisions. To improve this, future work on the decision engine must (a) improve the estimation of the transmission time for IA, and (b) take into account the cases where a transmitter cannot fully exploit its resources, and thus wastes capacity.

Part III

DISCUSSION

8

DISCUSSION AND OUTLOOK

Designing advanced PHYs for WMNs involves not only the PHY but also the LNK and the NET. As a result, our Corridor-based Routing architecture must deal with issues on all three layers. In earlier chapters, we introduce a design which realizes this architecture at the PHY, LNK, and NET. In Section 8.1, we discuss the insights that we gain from this design, and in Section 8.2, we point to future work that can build on it.

8.1 DISCUSSION

The sum of our results in previous chapters illustrates the characteristics of Corridor-based Routing. In the following, we work out these characteristics, and use them to deduce the fundamental properties of corridors. More precisely, we discuss our insights on five essential topics. First, we focus on the throughput gains that our stage mechanisms can achieve. Second, we highlight the importance of performance estimation in order to choose the correct PHY for each stage. Third, we go a step further into that direction, and delve into the details of resource selection. Fourth, we draw conclusions on the compatibility of our stage mechanisms both with different types of stages, and among each other. Finally, we present guidelines on the impact of the length and width of a corridor. Table 8.1 gives an overview on our insights on these five topics.

8.1.1 *Throughput Gains*

At the NET, our experiments show that Corridor-based Routing can yield throughput gains up to 2× compared to traditional hop-by-hop forwarding. However, the PHYs that we use for our stage mechanisms only provide a much smaller gain. In particular, at typical indoor SNRs, both IA and OFDMA result in roughly 30% PHY throughput gain, as shown in Table 8.1. The reason for this significant throughput gain difference at the PHY compared to the NET is the operation of the LNK. The LNK ensures that a receiver can successfully decode the data that a transmitter transmits via a wireless link. To this end, the LNK uses discrete-rate or rateless codes. Either way, this coding translates the continuous channel quality at the PHY into gradual quality steps at the LNK. For discrete-rate codes, this refers to individual MCSs. Although rateless codes such as Strider are more fine-granular, they ultimately also transmit a discrete number of encoded batches. Such quality steps result in thresholds which may multiply PHY throughput gains. For instance, allocating a subchannel in an OFDMA stage to a link which provides a slight improvement in channel conditions may cause the SNR to pass such a threshold. If this causes the subchannel to require only two instead of three Strider batches to convey a packet at the receiver, we achieve a 50% throughput gain, albeit the gain at the PHY may be much smaller. In other words, moderate gains at the PHY may have a significant impact at the NET. For Corridor-based Routing, this means that corridors do not need to be wide to provide large gains, which in turn reduces the construction, maintenance, and coordination effort in terms of communication overhead. The overhead itself has

Characteristic	**SL**	**OFDMA**	**IA**
THROUGHPUT GAINS			
PHY throughput gain	Marginal	≈ 30%	Up to 33%
NET throughput gain	Marginal	Up to 2×	Up to 1.5×
Communication overhead	Very low	Medium	High
PERFORMANCE ESTIMATION			
Sensitivity to transmission time estimation failures	Low for WB High for PS	High	High
Impact of scheduling	None for WB Medium for PS	Medium	High
Dependence on traffic	None	Medium	Very high
RESOURCE SELECTION			
Computational overhead	Very low	Low	Very high
Selection diversity	Low	Medium	Extremely high
Robustness	Medium	High	Low at NET High at PHY
Generality	Medium	High	Very low
STAGE COMPATIBILITY			
Allows for stages with $m \neq n$	Yes	Yes	Yes if $m, n \geqslant 3$
Compatibility to other stage mechanisms	Very high	High	High
Compatibility to PHY stage characteristics	Very high	Very high	Medium
Synchronization requirements	Very low	Medium	Very high
Impact of using same coding on all OFDM subcarriers	Low/Medium[1]	High/Medium[2]	Very high
Dependence on CSI	Low	Medium	Very high
CORRIDOR LENGTH AND WIDTH			
Most suitable stage width	1	2	4
Maximum meaningful stage width for additional gains	1	4	4
Impact of corridor length as a single stage mechanism	Very low	Medium	Extremely high
Impact of corridor length in combination with other PHYs	Very low	Medium	Medium

Table 8.1: Overview on insights from our experimental results.

[1] The impact depends on the frequency selectivity of the channel—the higher, the more impact.

[2] Impact is medium if OFDMA mitigates deep fades. Otherwise, the impact is high.

again a direct influence on the throughput gain. Hence, to achieve a large throughput, we conclude that Corridor-based Routing should only provide spatial diversity at stages where it actually becomes meaningful. This kills two birds with one stone—first, we provide PHY gains at the stages which are at the edge of the aforementioned thresholds, and second, we minimize the communication overhead by limiting the stage width.

The impact of the communication overhead on the throughput gain is directly related to the coherence time (c.f. Section 3.2.1.5). For indoor scenarios, our experiments show that all of the three stage mechanism that we present result in a reasonable but different overhead impact. In particular, SL causes very low communication overhead since it only requires per-link SNR feedback, whereas OFDMA and IA result in medium and high overhead, respectively, due to CSI feedback. In contrast, we expect outdoor scenarios to result in a smaller coherence time, and thus pose a challenge to stage mechanisms such as IA. However, such scenarios could benefit from stage mechanisms based on opportunistic PHY techniques (c.f. Chapter 2). Still, our current realization of the Corridor-based Routing architecture is tailored to indoor environments.

8.1.2 *Performance Estimation*

The fundamental technique that allows Corridor-based Routing to achieve the above throughput gains is performance estimation. In particular, it must estimate the performance of each available stage mechanism with each possible resource allocation, and choose the best one. We can capture this performance with a number of metrics, such as SNR, EVM, or the amplitude of the channel coefficients $|H|$. However, our experiments show that the most relevant metric is the stage transmission time. That is, the amount of time a stage needs to deliver all the data available at the transmitters to the receivers. Ultimately, the sum of the transmission times of all stages is the actual factor which determines the end-to-end throughput. In contrast, choosing resource allocations that minimize the EVM, or maximize $|H|$, is less meaningful due to the performance thresholds that we discuss in Section 8.1.1. For instance, a resource allocation that achieves a large EVM reduction compared to an SL baseline may achieve zero gain if, as a result, none of the resources exceeds such a threshold. While an EVM metric captures this erroneously as a large improvement, a transmission time metric reflects that the resource allocation does not provide any benefit. However, in practice, estimating the transmission time proves challenging, and may undermine the throughput gain entirely. As discussed in Section 4.2.3.1, the impact is particularly evident on mechanisms that operate on a per-subchannel basis, such as OFDMA and IA. All in all, we conclude from our results that accurate transmission time estimation is crucial for Corridor-based Routing.

Our design estimates transmission times based on (a) the amount of data at each node, and (b) the resource allocation that a certain stage mechanism achieves for the current CSI. In particular, we compute the amount of packet segments per time unit that each transmitter can transmit for the given resource allocation. Using (a), this allows us to deduce the transmission time for each node. The stage transmission time is the maximum of all node transmission times. However, this does not take into account how the individual nodes schedule each packet segment on their resources. Our measurements show that, as a result of this last step, nodes often cannot fully exploit their resources, which may have a significant impact on the actual throughput. Even assuming perfect transmission time estimation and the same amount of data at each transmitter, the impact of scheduling is particularly large for OFDMA and IA, as well as SL when it uses per-subchannel scheduling. The key problem is that each subchannel may require a different number of batches to convey a packet segment, which results in inefficiencies such as the checkerboard area in Figure 5.10. For IA, this becomes even more challenging because the subcarrier triplets that the allocation mechanism assigns to one node actually affect three nodes. More precisely, the limiting factor is that the allocation mechanism chooses the resource combination for a certain subcarrier triplet based on the channel conditions of the node which limits the stage transmission time only, instead of considering the channel conditions of all three nodes that can use the given resource combination. As a result, the two nodes that the allocation mechanism does not consider may experience poor channel conditions. Hence, the scheduling mechanism may abstain from scheduling any packet segments at all for those two nodes on that subcarrier triplet since the cost outweighs the benefit. In other words, the checkerboard area as in Figure 5.10 may become very large.

The potentially uneven distribution of data at the transmitters of a stage also has an impact on performance in terms of transmission time. Thus, performance estimation must take it into account. Even if we assume that all stage links exhibit a similar link quality, the throughput of OFDMA and IA depends on the data distribution. For OFDMA, an uneven data distribution impacts performance because each node can only use an integer fraction of the available subchannels. Hence, the allocation mechanism cannot assign to each node an amount of resources exactly proportional to the amount of data that it needs to transmit. As a result, some transmitters finish transmitting earlier than others, which translates into idle subchannels and resource wastage. This issue becomes even more challenging if we take into account that, in practice, each link typically experiences a different channel quality. Further, for IA, data distribution plays a fundamental role since, by design, IA stages include one link that can transmit more data than the others. Moreover, each resource combination affects three transmitters, as discussed above. Hence, allocating to each node an amount of resources proportional to the data that it needs to transmit is significantly more difficult than for OFDMA. For instance, as a by-product of allocating a resource triplet to a certain transmitter, the allocation mechanism may assign a double capacity link to a node which barely has any data to transmit. In such a case, the double capacity link is underused, and again results in resource wastage. In contrast, for SL, the impact of the amount of data at each transmitter is virtually zero since nodes transmit sequentially. Specifically, if all links are similar, the overall stage transmission time is constant for any distribution—the distribution itself only affects the amount of time that each transmitter transmits.

8.1.3 *Resource Selection*

The above performance estimation allows Corridor-based Routing to select among different resource allocations for SL, OFDMA, and IA. Our implementation shows that this selection incurs a certain computational overhead. For SL in best link mode, the overhead is minimal since SL only needs to choose among $m \cdot n$ links. While the resource diversity is larger for OFDMA, the resulting computational overhead is still low. In particular, for 802.11-like PHY parameters and a stage of width four, OFDMA only needs to choose one of the $4 \cdot 4$ links for each of the 16 subchannels. That is, OFDMA must estimate the performance of $4 \cdot 4 \cdot 16 = 256$ resource allocations. In contrast, Section 6.2.2.1 shows that the amount of resource combinations that IA must estimate to find a suitable selection in a similar scenario is 4608 when using our IA heuristics. Hence, the resource diversity of IA is an order of magnitude larger than for OFDMA. Without the aforementioned heuristic, the number of resource combinations exceeds several millions. Moreover, computing the noise impact for IA according to Section 6.1.3.1 incurs a significantly larger computational overhead than for OFDMA, since IA must obtain the precoding vectors for each resource combination, whereas OFDMA only compares CSI values.

The diversity resulting from resource selection determines the robustness of a stage mechanism. In this context, we define robustness as the capability of a stage mechanism to transport data successfully across a stage with arbitrary CSI. Essentially, the more resource allocations a stage mechanism can choose out of, the higher the probability that it finds a working one is. Our experiments validate that, as expected, the robustness of OFDMA is larger than the one of SL. However, for IA we observe a different behavior at the PHY than at the NET. At the PHY, the large number of resource combinations out of which IA can choose result in a robust operation, as we show in Section 6.1.5.4. However, at the NET, the impact of the scheduling inefficiencies that we discuss in Section 8.1.2 results in IA only performing well for a limited range of situations in terms of CSI and traffic conditions. In particular, our measurements show that certain CSI conditions hinder a reliable transmission time estimation, as we discuss in Chapter 6. Hence, the robustness of IA at the NET is limited. This translates directly into the generality of each of our stage mechanisms—that is, the probability that a mechanism is suitable for an arbitrary stage. While OFDMA can operate on most stages, IA is very selective. We initially expected SL to be a basic stage mechanism that Corridor-based Routing could resort to if no other PHY could operate. However, our practical experiments show that SL is less general than OFDMA since it cannot avoid deep fades. All in all, we conclude that no PHY mechanism fits all stages. In other words, each PHY is most suitable for a different set of situations. Our results show that the key to achieving large throughput gains is choosing the best stage mechanism and allocation for each case. This includes transitioning among such resource selections both in time and space. That is, adapting the selection within a stage according to CSI fluctuations in time, and across different stages according to CSI variations in different spatial environments.

8.1.4 *Stage Compatibility*

The profiles of our three stage mechanisms are, in general, highly flexible. Table 8.1 includes an overview on their main characteristics. Most remarkably, all our stage mechanisms are compatible to each other. Moreover, the requirements of both SL and OFDMA regarding stage properties such as stage shape, coherence time, and coherence bandwidth are low. In contrast, IA requires at least a stage width of three, and a longer coherence time to compensate for the stage maintenance overhead. Altogether, Table 8.1 highlights the trend of increasing requirements—and thus decreasing stage compatibility—for SL, OFDMA, and IA, in that order.

At the LNK, our experiments show that per-subchannel coding influences significantly the performance of a stage mechanism. Ultimately, this may limit the compatibility of a stage mechanism to a certain stage since, without such coding, deep fades on individual subcarriers may result in persistent transmission errors. The influence of per-subchannel coding depends on (a) the stage mechanism, and (b) the frequency selectiveness of the channel. For SL, the relation is evident since the influence of per-subchannel coding increases proportionally to the frequency selectiveness—the higher, the more beneficial per-subchannel coding for SL is. For OFDMA we observe a more complex behavior. If the stage links are highly frequency selective, OFDMA avoids deep fades and thus contributes to a more flat-fading behavior of the subchannels. In that case, the impact of per-subchannel coding is limited since most subchannels allow for a similar MCS. However, if the performance of some stage links is significantly better than the average link quality in a certain stage, the channel conditions of the resulting OFDMA subchannel allocation may be highly diverse. In that case, not using per-subchannel coding results in resource wastage, and thus has a high impact. In Figure 5.7b, we observe such an effect for subcarriers 27 to 40. Finally, for IA, we expect the impact of not using per-subchannel coding to be particularly large. The reason is that the SNRs of the IA streams are highly heterogeneous due to the different noise augmentation that each stream experiences as a result of IA decoding. In our IA single-hop experiments in Section 6.1.5.4, we choose the resource combination which minimizes the overall noise impact. In that case, using homogeneous per-subchannel coding proved to be still efficient. However, our IA stage mechanism in Section 6.2 minimizes the individual node transmission times, and may thus incur the aforementioned heterogeneous stream SNRs. Hence, we deduce that using individual coding for each stream on each subchannel is crucial for our IA stage mechanism. All in all, using per-subchannel coding improves the compatibility of a stage mechanism to a particular stage.

8.1.5 *Corridor Length and Width*

Finally, we discuss our insights on the width and length of corridors. Regarding the former, we observe that each of our stage mechanisms performs best for a different stage width. For SL, our results suggest that the difference between the fixed, random, and best link modes is marginal. Most probably, the reason for this behavior is that we only compare the average SNR of each link, which may be similar for links both with and without individual deep fades. To improve this, receivers would need to send per-subchannel feedback to the transmitters, thus incurring a higher overhead. Still, for our current SL realization, a wider stage does not provide large benefits. Hence, we conclude that the most suitable stage width for SL is one, that is, no selection at all, since it does not incur any overhead. As a result, this is also the maximum meaningful stage width, which is to say that, while SL can operate also on larger stages, the improvement is marginal. For OFDMA, both our theoretical analysis and our practical testbed measurements show that we achieve the largest throughput gain improvement for a stage width of two. For wider stages, we achieve meaningful additional gains up to a stage width of four. However, this incurs further overhead and, in turn, only provides relatively small gain improvements. Thus, we conclude that the most suitable stage width for OFDMA is two. Finally, IA performs best for a stage width of four since this allows for node selection within the stage. Our IA stage mechanism can also operate on wider stages, but we expect the computational overhead required to evaluate all of the resulting resource combinations to become prohibitively large. Therefore, a stage width of four is also the maximum meaningful width—while wider stages may achieve a higher throughput, the time required to compute the resource allocation may undermine the gains. Altogether, we conclude that the intended width parameter w of corridor construction should be set to, at most, four. Larger values are (a) hard to achieve, and (b) often unprofitable.

Regarding the length of corridors, we observe that stage mechanisms fall into two categories. Namely, mechanisms that can operate individually in a multi-stage corridor, and mechanisms which Corridor-based Routing must combine with other PHYs to achieve good results. SL and OFDMA fall in the former category, and IA in the latter. Concretely, for SL, the length of the corridor has a limited impact since SL repeats the same forwarding process at each stage. In contrast, OFDMA has a stronger dependence on previous stages because it splits and merges packet segments. The problem is that the forwarding process at previous stages may result in an inconvenient distribution of data at the current stage. In the worst case, a node with poor outgoing links may receive a large amount of packet segments, resulting in a bottleneck. On average, such cases compensate with particularly beneficial cases, and thus the overall impact of the corridor length on OFDMA is medium. Next, for IA, corridors with multiple consecutive IA stages become highly challenging. The reason is its strong dependence on traffic that we discuss in Section 8.1.2, which directly results in an extremely high corridor length impact. Basically, the beneficial situations for IA in terms of CSI and traffic conditions are very limited. Hence, the longer the corridor, the more difficult it is that we achieve such conditions at each IA stage. However, our results show that Corridor-based Routing can mitigate this limitation by allowing for both IA and OFDMA, and resorting to OFDMA when IA performs poorly.

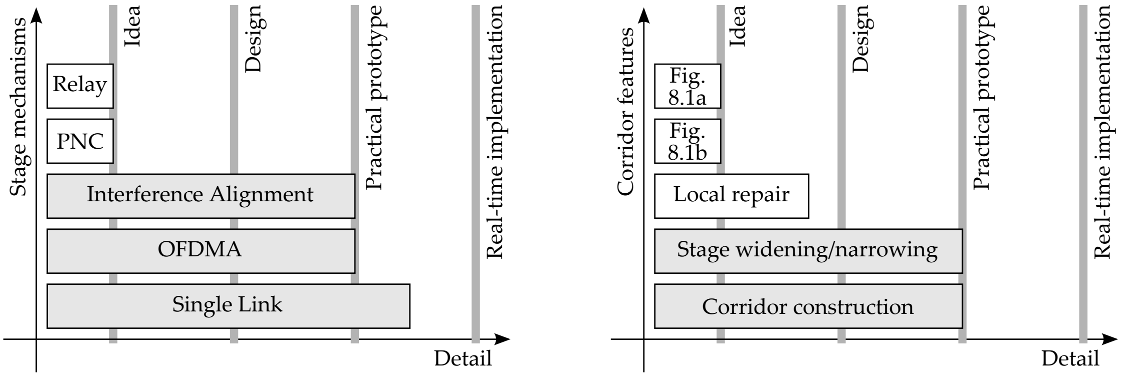
In general, the more the performance of a stage mechanism depends on the distribution of data at the transmitters, the larger the impact of corridor length is. Reference [50] addresses similar issues by allowing for multi-hop control data. For Corridor-based Routing, this would translate into inter-stage coordination. Essentially, each stage would provide coarse information to previous stages regarding the channel conditions of each link. As a result, previous stages could select resource allocations which do not result in inconvenient data distributions at any stage. However, this causes additional overhead. Hence, such an approach would need to adjust the detail of the CSI that stages exchange via inter-stage coordination to find a suitable trade-off. An alternative approach to limit the impact of corridor length is to allow for a large amount of different stage mechanisms that virtually cover the entire spectrum of potential CSI and traffic conditions at a stage. In other words, a stage would always find a beneficial stage mechanism and resource allocation for any situation. As a result, inter-stage coordination would be unnecessary, and Corridor-based Routing could operate in a fully localized manner. Increasing the number of stage mechanisms for our implementation is part of future work.

8.2 OUTLOOK

Future work on Corridor-based Routing includes both (a) additional components, and (b) extensions of our current design. The former adds new functionalities that build on the corridor architecture, while the latter further improves the mechanisms that we discuss in earlier chapters. In the following, we give an outlook on both dimensions (a) and (b) for stage mechanisms as well as for corridor construction.

8.2.1 *Stage Mechanisms*

Figure 8.1a depicts our current stage mechanisms and suggests future directions. On the horizontal axis, we show the level of detail of the implementation of each stage mechanism, while the vertical axis represents further stage mechanism designs. The



(a) Future work on stage mechanisms.

(b) Future work on corridors.

Figure 8.1: Overview on future work. The diagrams show our current Corridor-based Routing realization, and give an outlook on extensions in terms of design and implementation.

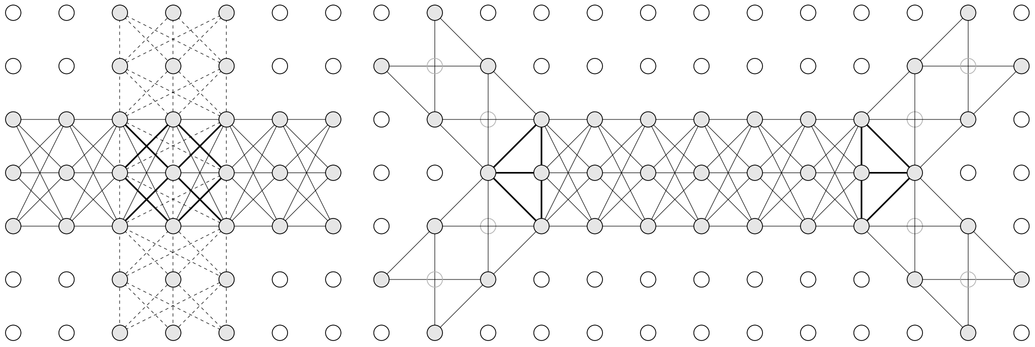
shaded areas refer to the PHYs included in our current realization of Corridor-based Routing. For both IA and OFDMA we present a design which adapts them to corridors, and a practical prototype implementation on SDRs. For SL, we are one step closer to a real-time implementation since the WARP Project provides reference designs that implement OFDM on the WARP FPGA. Still, we do not use them yet for Corridor-based Routing.

On the horizontal axis in Figure 8.1a, future work includes implementing our SL, OFDMA, and IA stage mechanisms in real-time. This would avoid the delays that our current WARPLab implementation incurs. As a result, this would allow us to measure the actual throughput that our stage mechanisms achieve, instead of resorting to extrapolations. Further, a real-time implementation opens the door to evaluating the responsiveness of a corridor stage to changes in CSI. For instance, one potential experiment would be to capture the impact of a human physically walking between the transmitters and the receivers of a stage. As the CSI changes, the stage mechanism would adapt its allocation, and thus most probably achieve a roughly constant throughput. In contrast, the throughput of a PHY mechanism which cannot adapt, such as SL in fixed link mode, would drop significantly when the movement momentarily causes deep fades. Moreover, in a further step in the same direction, future work includes full wireless synchronization among transmitters both in time and frequency. To this end, we could embed approaches such as AirSynch [10] into the design of our stage mechanisms, as well as the synchronization mechanism we present in Section 5.2.2.4. In both cases, the challenge lies in the integration with the corridor architecture, since the synchronization techniques themselves exist. Further, a fundamental part of future work deals with extending the design of our three stage mechanisms to improve their performance based on the insights that we gain from our practical experiments. For example, this includes identifying potentially unstable CSI conditions (c.f. Section 6.4.2.1), and taking into account the impact of segment scheduling when estimating the performance of a stage mechanism. As a result, the accuracy of the decision engine would increase. This is crucial to mitigate the impact of the non-beneficial cases that we observe in our current results. To address the inefficiency of the scheduler itself, improved versions of our scheduling algorithm could exploit idle slots to transmit control data and thus reduce the impact of overhead. Moreover, idle slots could also accommodate data from other nearby corridors (c.f. Section 8.2.2).

On the vertical axis in Figure 8.1a, we show new potential stage mechanisms for Corridor-based Routing. A large number of the techniques that we survey in Chapter 2 are suitable to become stage mechanisms. For instance, a technique inspired by PNC, and based on the collision-resilient characteristics of Strider, could allow for collisions within a stage. Although the LNK would require more Strider batches to convey collided packets, no transmitted batch would be in vain, and Strider would still decode data correctly. In combination with OFDMA, this simplifies resource allocation significantly. Essentially, each transmitter can select the subchannels it prefers *without* intra-stage coordination. While this means that two transmitters may select the same subchannel, the collision is uncritical since Strider can deduce the original packets at the cost of more Strider batches. Such a design would reduce the overhead of our current OFDMA stage mechanism to a large extent. Further, other stage mechanisms could, for example, build on techniques that use intermediate nodes to relay soft information [169]. Such an approach assumes that an individual node lies between the transmitters and the receivers, and helps them to forward information. In particular, even if the intermediate node cannot decode data, it can forward soft information that the receivers can combine with their own received signals to decode data successfully. In order to use such a stage mechanism, the nodes of a stage could listen for beacons from nearby nodes to find a potential relay, even if it is not part of the corridor. Alternatively, the corridor construction protocol itself could actively build stages with an intermediate relay. Stage mechanisms that use relays would include this as part of their profile. This would allow stage maintenance to identify them as potential PHYs for stages with relays. All in all, we conclude that future work on the vertical axis in Figure 8.1a is substantial since a large number of advanced PHY techniques are suitable to become stage mechanisms.

8.2.2 *Corridor Construction*

Future work on corridor construction also allows for approaches along the two aforementioned dimensions, as shown in Figure 8.1b. Our current design includes corridor construction itself, as well as support for widening and narrowing stages, in addition to stages of constant width. Along the horizontal axis in Figure 8.1b, we can extend our corridor construction protocol to include the cross-layer optimizations that we suggest in Section 3.1.2.1. Essentially, instead of using only topological criteria to choose the nodes of each stage, the corridor construction protocol could also take into account criteria which are beneficial for a particular stage mechanism. To evaluate this, future work could build on a WARPLab prototype similar to ours. However, the analysis of further extensions of our corridor construction protocol may require a real-time implementation on the WARP FPGA. For instance, this would allow us to evaluate the impact of node mobility on the performance of corridors. In particular, we could measure for different levels of node mobility whether we reach the turning point from which on the overhead for building the corridor pays off. Moreover, in combination with the local repair feature that we show on the vertical axis in Figure 8.1b, we could observe whether the corridor continues operating correctly after physically removing one corridor node. Basically, the local repair feature rearranges stages if a stage node becomes unreachable, that is, if the remaining stage nodes do not receive any beacon messages from it for a given time interval. To this end, the corridor locally rebuilds the stage using a similar algorithm to the one that we present in Section 3.2.2.1.



(a) Corridor intersection.

(b) Corridor overlapping.

Figure 8.2: Advanced corridor stages for scenarios with multiple corridors.

Moreover, future work on corridor construction also includes new mechanisms, which we represent on the vertical axis in Figure 8.1b. In particular, a crucial issue is the support for multiple simultaneous corridors, since WMNs typically forward data from multiple pairs of communicating nodes at the same time. As long as the corridors operate in different network areas, Corridor-based Routing can use our current design. However, if corridors are close to each other, we need specific solutions to deal with intersections. In particular, nodes may be part of two corridors simultaneously, and intersections may result in bottlenecks. Figure 8.2a depicts an example of a basic intersection. The continuous lines represent the links that the horizontal corridor uses, whereas the dashed lines refer to the vertical corridor. Moreover, the thick lines are the links which both corridors use. These links fall into two categories, namely, (a) links which both corridors use in the same direction, and (b) links which the corridors use in opposite directions. Links belong to (a) or (b) depending on the direction in which data flows in each corridor. However, corridors may exploit the particularities of intersections at the PHY to avoid bottlenecks, and even achieve gains. For instance, for links falling into category (b), nodes could use two-way relaying as in PNC (c.f. Chapter 2).

Figure 8.2b shows the overlapping of two corridors. While such a scenario may occur naturally, the corridor construction protocol may also induce it on purpose to exploit potential throughput gains. For example, if the individual corridors are forwarding data in opposite directions, such an overlapping may benefit from PNC. However, we expect that inducing a corridor overlapping incurs additional costs in terms of construction overhead. Hence, the corridor construction protocol must decide whether the potential throughput gain of the corridor overlapping compensates for the additional overhead. In turn, this may depend on the width and length of the overlapping corridor segment, resulting in a complex optimization problem. Future work includes designing algorithms that can take such decisions reliably. Further, corridors must support joining and splitting—the thick lines in Figure 8.2b show the corridor stages involved in both processes. The challenge lies in the operation of the nodes at the vertexes of the triangle stages since they are part of multiple corridors. All in all, we suspect that building intersecting and overlapping corridors may result in a large overhead. Hence, a further direction for corridor construction is to use mechanisms such as Sparkle [168], which exploit the capture effect to find corridor-like structures in a lightweight manner.

8.2.3 *Summary*

Our above discussion on future work shows that Corridor-based Routing is highly modular, and thus allows for a large number of extensions. This highlights that the concept underlying corridors goes far beyond a traditional WMN routing protocol. In this work, we show that Corridor-based Routing is practical, and that it provides significant benefits in terms of throughput, robustness, and adaptivity. Future work includes new stage mechanisms, as well as support for multiple corridors. We expect such work to achieve motivating and interesting insights on the three cornerstone principles of Corridor-based Routing, which are as follows.

1. **Divide and Conquer.** Divide WMNs into groups of nodes which cooperate at the timescale of the PHY within groups, and at the timescale of the NET among groups.
2. **Spatial Diversity.** Exploit the individual antennas of WMN nodes to achieve spatial diversity via cooperation, and thus allow for advanced PHY techniques.
3. **Adaptivity.** Allow nodes to (a) adapt PHY parameters, and (b) switch among PHYs according to CSI and traffic conditions to sustain high performance in any situation.

We conclude that the above principles, together with the experimental insights that we discuss in this chapter, are a good starting point for the design of future practical WMNs.

9

CONCLUSIONS

In this work, we present Corridor-based Routing, which is a cross-layer routing paradigm that enables advanced PHYs in WMNs. Our goal is to improve the practical performance of WMNs in terms of throughput, robustness, and scalability. The performance of WMNs is often limited because the underlying PHY cannot exploit the time, frequency, and spatial diversity of the wireless medium. Corridor-based Routing addresses this issue, and thus opens the door to state-of-the-art PHYs in WMNs. In particular, corridors widen traditional WMN paths in order to span multiple nodes at each hop. As a result, a hop involves a group of m transmitting nodes that cooperate at the PHY to forward data to a group of n receiving nodes. Two subsequent and fully connected groups of nodes form a corridor stage with $m \cdot n$ links. Such a topology matches the requirements of many advanced PHYs, and thus mitigates the limitations of WMNs. First, the $m \cdot n$ links of each stage provide spatial diversity, which is the key to many PHY techniques such as OFDMA, IA, and MIMO. Second, all nodes involved in a transmission are part of the same stage, and are thus direct neighbors. Hence, the WMN does not need to disseminate CSI over multiple hops, which typically carries a prohibitive cost. Finally, each stage of a corridor may use a different PHY. As a result, the corridor can adapt with high flexibility to the particular characteristics of the physical environment in which a stage operates.

We design a corridor construction protocol which builds corridors of a certain intended width using a combination of geographical and topological routing techniques. The protocol ensures that each stage is fully connected. However, it may need to narrow the corridor in sparse network areas. We observe that the control overhead that our protocol incurs pays off as long as (a) we can use the resulting corridor for multiple data packets, and (b) we use a PHY which provides end-to-end throughput gains. Essentially, the throughput gains compensate for the initial overhead after a certain number of data packets. This number depends on both the throughput gain of the PHY in use, and the network density. For our case, we reach the turning point after at most 16 packets.

Further, we implement two PHYs for Corridor-based Routing, namely, OFDMA and IA. OFDMA divides the available bandwidth into subchannels, and allows the transmitters of a stage to share them. The key benefit is that each link of a stage experiences different channel conditions on each subchannel due to spatial diversity. Hence, OFDMA can allocate each subchannel to a link for which it experiences good channel conditions, and thus improve the overall channel quality of the stage. The main challenges are (a) providing timely CSI feedback, and (b) finding suitable subchannel allocations. Regarding (a), our measurements show that one-bit feedback per subchannel is sufficient to avoid links with poor channel conditions. Hence, the resulting overhead is minimal. For (b), we propose an allocation algorithm which takes into account both the CSI, and the amount of data that each transmitter needs to transmit. Our results show that the later is crucial to avoid bottlenecks, as well as underused resources. We implement our OFDMA PHY for Corridor-based Routing on a practical SDR testbed, and in simulation. For both cases, we achieve roughly 30% average throughput gain compared to a WMN not using corridors. Moreover, we observe that OFDMA corridors do not need to be wide to yield large gains.

More precisely, we achieve gains close to the maximum already for a corridor of width two. Further, OFDMA provides robust communication in scenarios where traditional forwarding cannot operate due to deep fades. Still, we find that not all CSI and traffic conditions are beneficial for OFDMA stages. This motivates our above Corridor-based Routing design, which allows corridors to switch among different PHYs at each stage.

Our second PHY for Corridor-based Routing is IA in the frequency domain. In particular, we consider a scenario with three transmitters and three receivers. IA enables one of the transmitters to transmit two data symbols on one subchannel. Hence, the transmitters can send overall four data symbols on three subchannels, that is, they achieve a theoretical throughput gain of 33%. However, our practical measurements show that the decoding process at each receiver occasionally augments noise significantly, and thus often undermines the gains of IA entirely. We observe that the noise augmentation depends on which transmitters communicate with which receivers using which subchannels. Hence, we propose a solution that estimates the noise augmentation for all potential resource selections within a stage, and selects the most beneficial one. Still, this results in a computational challenge since the number of possible resource allocations is extremely large. To solve this, we use heuristics, which reduce the search space by exploiting the characteristics of the wireless medium. Using this design, which is suitable both for corridors as well as for infrastructure-based networks, we present the first practical frequency IA system. Our evaluation on our SDR testbed shows that our approach avoids noise augmentation efficiently, and absorbs channel fluctuations, hence improving the robustness of IA. We observe that IA is beneficial for a specific set of CSI and traffic conditions only. In comparison to OFDMA, the beneficial set of IA is more limited. Still, in comparison to a system not using corridors, we achieve about 25% average throughput gain for the beneficial scenarios, and up to 1.5× gain in the best cases. Our maximum throughput gains at the NET are larger than at the PHY due to the multiplying effect of coding techniques. Basically, a small gain at the PHY can have a large impact at the NET if it allows a transmitter to switch to the next higher MCS. Moreover, our results show that IA achieves roughly 10% throughput gain compared to OFDMA in selected situations.

Finally, we bring together our corridor construction scheme, OFDMA, and IA to a complete Corridor-based Routing system. Additionally, we include a decision engine, that is, a component which decides for each stage which PHY to use. We implement our system on a framework which allows for both practical and simulation experiments. Our results show that, as expected, the decision engine chooses IA in few cases only, and resorts to OFDMA otherwise. We observe that the impact of packet scheduling is crucial. In particular, our decision engine chooses the PHY for a stage based on the resource allocation of OFDMA and IA. Still, scheduling inefficiencies resulting from, e.g., inaccurate MCS selections may result in a worse performance than the resource allocation suggests. All in all, we conclude that Corridor-based Routing can (a) work correctly as a system, and (b) provide throughput gains in line with our results for each individual PHY.

The Corridor-based Routing architecture is highly extensible. Future work includes (a) additional features, and (b) improvements on our current realization. Regarding (a), we propose new stage mechanisms based on, e.g., PNC. Moreover, we sketch the operation of multiple simultaneous corridors. For (b), we suggest further improving the accuracy of our MCS selection, and the efficiency of our schedulers. Altogether, we expect that the principles underlying the corridor concept and the insights that we gain from our experiments serve as a good starting point for the design of future practical WMNs.

BIBLIOGRAPHY

[1] 3GPP. *TS 36.213 Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures.*

[2] Fadel Adib, Swarun Kumar, Omid Aryan, Shyamnath Gollakota, and Dina Katabi. Interference Alignment by Motion. In *Proceedings of the 19th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2013)*, 2013.

[3] Richard O. Afolabi, Aresh Dadlani, and Kiseon Kim. Multicast Scheduling and Resource Allocation Algorithms for OFDMA-Based Systems: A Survey. *IEEE Communications Surveys & Tutorials*, 15(1), 2013.

[4] Manish Agarwal, Dongning Guo, and Michael L. Honig. Limited-Rate Channel State Feedback for Multicarrier Block Fading Channels. *IEEE Transactions on Information Theory*, 56(12), 2010.

[5] Rudolf Ahlswede, Ning Cai, Shuo-Yen Robert Li, and Raymond W. Yeung. Network Information Flow. *IEEE Transactions on Information Theory*, 46(4), 2000.

[6] Jeffrey Andrews, Nihar Jindal, Martin Haenggi, Randy Berry, Syed Jafar, Dongning Guo, Sanjay Shakkottai, Robert Heath, Michael Neely, Steven Weber, and Aylin Yener. Rethinking Information Theory for Mobile Ad Hoc Networks. *IEEE Communications Magazine*, 46(12), 2008.

[7] Mustafa Y. Arslan, Jongwon Yoon, Karthikeyan Sundaresan, Srikanth V. Krishnamurthy, and Suman Banerjee. FERMI: A FEmtocell Resource Management System for Interference Mitigation in OFDMA Networks. In *Proceedings of the 17th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2011)*, 2011.

[8] Ehsan Aryafar, Mohammad Amir Khojastepour, Karthikeyan Sundaresan, Sampath Rangarajan, and Mung Chiang. MIDU: Enabling MIMO Full Duplex. In *Proceedings of the 18th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2012)*, 2012.

[9] Horia Vlad Balan, Ryan Rogalin, Antonios Michaloliakos, Konstantinos Psounis, and Giuseppe Caire. Achieving High Data Rates in a Distributed MIMO System. In *Proceedings of the 18th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2012)*, 2012.

[10] Horia Vlad Balan, Ryan Rogalin, Antonios Michaloliakos, Konstantinos Psounis, and Giuseppe Caire. AirSync: Enabling Distributed Multiuser MIMO With Full Spatial Multiplexing. *IEEE/ACM Transactions on Networking*, 21(6), 2013.

[11] Sergio Barbarossa, Massimiliano Pompili, and Georgios B. Giannakis. Channel-Independent Synchronization of Orthogonal Frequency Division Multiple Access Systems. *IEEE Journal on Selected Areas in Communications*, 20(2), 2002.

[12] Dinesh Bharadia and Sachin Katti. Full Duplex MIMO Radios. In *Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2014)*, 2014.

[13] Dinesh Bharadia, Emily McMilin, and Sachin Katti. Full Duplex Radios. In *Proceedings of the 2013 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2013)*, 2013.

[14] Vaduvur Bharghavan, Alan Demers, Scott Shenker, and Lixia Zhang. MACAW: A Media Access Protocol for Wireless LAN's. In *Proceedings of the 1994 ACM SIGCOMM Conference on Data Communication (SIGCOMM 1994)*, 1994.

[15] Apurv Bhartia, Yi-Chao Chen, Swati Rallapalli, and Lili Qiu. Harnessing Frequency Diversity in Wi-Fi Networks. In *Proceedings of the 17th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2011)*, 2011.

[16] Sanjit Biswas and Robert Morris. ExOR: Opportunistic Multi-Hop Routing for Wireless Networks. In *Proceedings of the 2005 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2005)*, 2005.

[17] Azzedine Boukerche, Begumhan Turgut, Nevin Aydin, Mohammad Z. Ahmad, Ladislau Bölöni, and Damla Turgut. Routing Protocols in Ad Hoc Networks: A Survey. *Computer Networks*, 55(13), 2011.

[18] Rasmus Brandt, Henrik Asplund, and Mats Bengtsson. Interference Alignment in Frequency-A Measurement Based Performance Analysis. In *Proceedings of the 19th International Conference on Systems, Signals and Image Processing (IWSSIP 2012)*, 2012.

[19] Guy Bresler, Abhay Parekh, and David Tse. The Approximate Capacity of the Many-to-One and One-to-Many Gaussian Interference Channels. *IEEE Transactions on Information Theory*, 56(9), 2010.

[20] Viveck Cadambe and Syed Jafar. Interference Alignment and Degrees of Freedom of the K-User Interference Channel. *IEEE Transactions on Information Theory*, 54(8), 2008.

[21] Jiannong Cao, Lifan Zhang, Guojun Wang, and Hui Cheng. SSR: Segment-by-Segment Routing in Large-Scale Mobile Ad Hoc Networks. In *Proceedings of the 3rd IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2006)*, 2006.

[22] Szymon Chachulski, Michael Jennings, Sachin Katti, and Dina Katabi. Trading Structure for Randomness in Wireless Opportunistic Routing. In *Proceedings of the 2007 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2007)*, 2007.

[23] Jieying Chen, Randall Berry, and Michael Honig. Limited Feedback Schemes for Downlink OFDMA Based on Sub-Channel Groups. *IEEE Journal on Selected Areas in Communications*, 26(8), 2008.

[24] Shan Chu, Peng Wei, Xu Zhong, Xin Wang, and Yu Zhou. Deployment of a Connected Reinforced Backbone Network with a Limited Number of Backbone Nodes. *IEEE Transactions on Mobile Computing*, 12(6), 2013.

[25] Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit Disk Graphs. *Discrete Mathematics*, 86(1-3), 1990.

[26] Thomas Clausen and Philippe Jacquet. Optimized Link State Routing Protocol (OLSR). Technical report, IETF, 2003.

[27] Reuven Cohen and Liran Katzir. Computational Analysis and Efficient Algorithms for Micro and Macro OFDMA Scheduling. In *Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM 2008)*, 2008.

[28] Marco Conti, Gaia Maselli, Giovanni Turi, and Silvia Giordano. Cross-Layering in Mobile Ad Hoc Network Design. *Computer*, 37(2), 2004.

[29] Lin Dai, Bo Gui, and Leonard Cimini. Selective Relaying in OFDM Multihop Cooperative Networks. In *Proceedings of the 2007 IEEE Wireless Communications and Networking Conference (WCNC 2007)*, 2007.

[30] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A High-Throughput Path Metric for Multi-Hop Wireless Routing. In *Proceedings of the 9th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2003)*, 2003.

[31] Rahul Dhar, Gesly George, Amit Malani, and Peter Steenkiste. Supporting Integrated MAC and PHY Software Development for the USRP SDR. In *Proceedings of the 1st IEEE Workshop on Networking Technologies for Software Defined Radio Networks (SDR 2006)*, 2006.

[32] Melissa Duarte, Chris Dick, and Ashutosh Sabharwal. Experiment-Driven Characterization of Full-Duplex Wireless Systems. *IEEE Transactions on Wireless Communications*, 11(12), 2012.

[33] Omar El Ayach, Steven W. Peters, and Robert W. Heath. The Practical Challenges of Interference Alignment. *IEEE Wireless Communications*, 20(1), 2013.

[34] Hamid Eslami, Gaurav Patel, Chitaranjan P. Sukumar, Sang V. Tran, Ahmed M. Eltawil, Raghu Rao, and Chris Dick. Demonstration of Highly Programmable Downlink OFDMA (WiMax) Transceivers for SDR Systems. In *Proceedings of the 10th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2009)*, 2009.

[35] Mohammad Fathi, Hassan Taheri, and Mehri Mehrjoo. Cross-Layer Joint Rate Control and Scheduling for OFDMA Wireless Mesh Networks. *IEEE Transactions on Vehicular Technology*, 59(8), 2010.

[36] Gregor Gaertner, Eamonn Onuallain, Andrew Butterly, Kulpreet Singh, and Vinny Cahill. 802.11 Link Quality and Its Prediction – An Experimental Study. *Lecture Notes in Computer Science*, 3260, 2004.

[37] Wojciech Galuba, Panos Papadimitratos, Marcin Poturalski, Karl Aberer, Zoran Despotovic, and Wolfgang Kellerer. Castor: Scalable Secure Routing for Ad Hoc Networks. In *Proceedings of the 29th IEEE Conference on Computer Communications (INFOCOM 2010)*, 2010.

[38] J. J. Garcia-Luna-Aceves, Hamid R. Sadjadpour, and Zheng Wang. Challenges: Towards Truly Scalable Ad Hoc Networks. In *Proceedings of the 13th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2007)*, 2007.

[39] Krishna C. Garikipati and Kang G. Shin. Measurement-Based Transmission Schemes for Network MIMO. In *Proceedings of the 15th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2014)*, 2014.

[40] Andrea Goldsmith. How Signal Processing will Revolutionize Future Wireless Network Designs, 2010.

[41] Andrea Goldsmith and Stephen Wicker. Design Challenges for Energy-Constrained Ad Hoc Wireless Networks. *IEEE Wireless Communications*, 9(4), 2002.

[42] Shyamnath Gollakota and Dina Katabi. ZigZag Decoding: Combating Hidden Terminals in Wireless Networks. In *Proceedings of the 2008 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2008)*, 2008.

[43] Shyamnath Gollakota, Samuel David Perli, and Dina Katabi. Interference Alignment and Cancellation. In *Proceedings of the 2009 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2009)*, 2009.

[44] Oscar Gonzalez, David Ramirez, Ignacio Santamaria, José A. Garcia-Naya, and Luis Castedo. Experimental Validation of Interference Alignment Techniques using a Multiuser MIMO Testbed. In *Proceedings of the 15th International ITG Workshop on Smart Antennas (WSA 2011)*, 2011.

[45] Tiangao Gou, Chenwei Wang, and Syed A. Jafar. Aiming Perfectly in the Dark - Blind Interference Alignment through Staggered Antenna Switching. In *Proceedings of the 2010 IEEE Global Telecommunications Conference (GLOBECOM 2010)*, 2010.

[46] Izrail′ Solomonovich Gradshteĭn and Iosif Moiseevich Ryzhik. *Table of Integrals, Series and Products*. Academic Press, 1980.

[47] James Gross, Marc Emmelmann, Oscar Puñal, and Adam Wolisz. Dynamic Single-User OFDM Adaptation for IEEE 802.11 Systems. In *Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM 2007)*, 2007.

[48] Aditya Gudipati and Sachin Katti. Strider: Automatic Rate Adaptation and Collision Handling. In *Proceedings of the 2011 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2011)*, 2011.

[49] Aditya Gudipati, Stephanie Pereira, and Sachin Katti. AutoMAC: Rateless Wireless Concurrent Medium Access. In *Proceedings of the 18th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2012)*, 2012.

[50] Bo Gui, Lin Dai, and Leonard Cimini. Routing Strategies in Multihop Cooperative Networks. *IEEE Transactions on Wireless Communications*, 8(2), 2009.

[51] Piyush Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory*, 46(2), 2000.

[52] Khairi Hamdi. Precise Interference Analysis of OFDMA Time-Asynchronous Wireless Ad-hoc Networks. *IEEE Transactions on Wireless Communications*, 9(1), 2010.

[53] Bo Han, Lusheng Ji, Seungjoon Lee, Bobby Bhattacharjee, and Robert R. Miller. All Bits Are Not Equal - A Study of IEEE 802.11 Communication Bit Errors. In *Proceedings of the 28th IEEE Conference on Computer Communications (INFOCOM 2009)*, 2009.

[54] Sahar Hoteit, Stefano Secci, Rami Langar, and Guy Pujolle. A Nucleolus-Based Approach for Resource Allocation in OFDMA Wireless Mesh Networks. *IEEE Transactions on Mobile Computing*, 12(11), 2013.

[55] Wei Hu, Jin Xie, and Zhenghao Zhang. Practical Opportunistic Routing in High-Speed Multi-Rate Wireless Mesh Networks. In *Proceedings of the 14th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2013)*, 2013.

[56] Peter Anthony Iannucci, Jonathan Perry, Hari Balakrishnan, and Devavrat Shah. No Symbol Left Behind: A Link-Layer Protocol for Rateless Codes. In *Proceedings of the 18th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2012)*, 2012.

[57] IEEE. Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification, 2012.

[58] International Telecommunication Union. Guidelines for Evaluation of Radio Transmission Technologies for IMT-2000. *ITU-R M.1225*, 1997.

[59] Syed A. Jafar. Blind Interference Alignment. *IEEE Journal of Selected Topics in Signal Processing*, 6(3), 2012.

[60] Mayank Jain, Jung Il Choi, Taemin Kim, Dinesh Bharadia, Siddharth Seth, Kannan Srinivasan, Philip Levis, Sachin Katti, and Prasun Sinha. Practical, Real-Time, Full Duplex Wireless. In *Proceedings of the 17th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2011)*, 2011.

[61] Kyle Jamieson and Hari Balakrishnan. PPR: Partial Packet Recovery for Wireless Networks. In *Proceedings of the 2007 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2007)*, 2007.

[62] David Johnson, Yih-Chun Hu, and David Maltz. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4. Technical report, IETF, 2007.

[63] Aaron Jow, Curt Schurgers, and Douglas Palmer. CalRadio: A Portable, Flexible 802.11 Wireless Research Platform. In *Proceedings of the 1st International Workshop on System Evaluation for Mobile Platforms (MobiEval 2007)*, 2007.

[64] Shirish Karande, Zheng Wang, Hamid R. Sadjadpour, and J.J. Garcia-Luna-Aceves. Optimal Scaling of Multicommodity Flows in Wireless Ad Hoc Networks: Beyond the Gupta-Kumar Barrier. In *Proceedings of the 5th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2008)*, 2008.

[65] Sachin Katti. *Network Coded Wireless Architecture*. PhD thesis, Massachusetts Institute of Technology, 2008.

[66] Sachin Katti, Shyamnath Gollakota, and Dina Katabi. Embracing Wireless Interference: Analog Network Coding. In *Proceedings of the 2007 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2007)*, 2007.

[67] Sachin Katti, Dina Katabi, Hari Balakrishnan, and Muriel Médard. Symbol-level Network Coding for Wireless Mesh Networks. In *Proceedings of the 2008 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2008)*, 2008.

[68] Sachin Katti, Hariharan Rahul, Wenjun Hu, Dina Katabi, Muriel Médard, and Jon Crowcroft. XORs in the Air: Practical Wireless Network Coding. In *Proceedings of the 2006 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2006)*, 2006.

[69] Douglas Kim and Murat Torlak. Optimization of Interference Alignment Beamforming Vectors. *IEEE Journal on Selected Areas in Communications*, 28(9), 2010.

[70] Seung-Jun Kim, Xiaodong Wang, and Mohammad Madihian. Optimal Resource Allocation in Multi-hop OFDMA Wireless Networks with Cooperative Relay. *IEEE Transactions on Wireless Communications*, 7(5), 2008.

[71] Didem Kivanc, Guoqing Li, and Hui Liu. Computationally Efficient Bandwidth Allocation and Power Control for OFDMA. *IEEE Transactions on Wireless Communications*, 2(6), 2003.

[72] Robin Klose. Dynamic Subchannel Allocation in OFDMA-Based Wireless Mesh Networks. Master's thesis, TU Darmstadt, 2012.

[73] Robin Klose, Adrian Loch, and Matthias Hollick. A Rapid Prototyping Framework for Practical OFDMA Systems using Software Defined Radios. In *Proceedings of the 2013 IEEE International Conference on Sensing, Communication, and Networking (SECON 2013)*, 2013.

[74] Robin Klose, Adrian Loch, and Matthias Hollick. Evaluating Dynamic OFDMA Subchannel Allocation for Wireless Mesh Networks on SDRs. In *Proceedings of the 2nd Software Radio Implementation Forum (SRIF 2013)*, 2013.

[75] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The click modular router. *ACM Transactions on Computer Systems*, 18(3), 2000.

[76] Thanasis Korakis, Michael Knox, Elza Erkip, and Shivendra Panwar. Cooperative Network Implementation Using Open-Source Platforms. *IEEE Communications Magazine*, 47(2), 2009.

[77] Alexander Kuehne, Anja Klein, Adrian Loch, and Matthias Hollick. Corridor-based Routing using Opportunistic Forwarding in OFDMA Multihop Networks. In *Proceedings of the 23rd IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2012)*, 2012.

[78] Alexander Kuehne, Anja Klein, Adrian Loch, and Matthias Hollick. Opportunistic Forwarding in Multi-Hop OFDMA Networks with Local CSI. In *Proceedings of the 17th International OFDM Workshop 2012 (InOWo 2012)*, 2012.

[79] Rafael Laufer, Henri Dubois-Ferriere, and Leonard Kleinrock. Multirate Anypath Routing in Wireless Mesh Networks. In *Proceedings of the 28th IEEE Conference on Computer Communications (INFOCOM 2009)*, 2009.

[80] Rafael Laufer, Theodoros Salonidis, Henrik Lundgren, and Pascal Le Guyadec. XPRESS: A Cross-Layer Backpressure Architecture for Wireless Multi-Hop Networks. In *Proceedings of the 17th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2011)*, 2011.

[81] Rafael Laufer, Pedro B. Velloso, Luiz Filipe M. Vieira, and Leonard Kleinrock. PLASMA: A New Routing Paradigm for Wireless Multihop Networks. In *Proceedings of the 31st IEEE Conference on Computer Communications (INFOCOM 2012)*, 2012.

[82] Ki-Dong Lee and Victor C. M. Leung. Fair Allocation of Subcarrier and Power in an OFDMA Wireless Mesh Network. *IEEE Journal on Selected Areas in Communications*, 24(11), 2006.

[83] Sung-Ju Lee, Elizabeth M. Belding-Royer, and Charles E. Perkins. Scalability Study of the Ad Hoc On-Demand Distance Vector Routing Protocol. *International Journal of Network Management*, 13(2), 2003.

[84] Sung-Ju Lee and Mario Gerla. AODV-BR: Backup Routing in Ad Hoc Networks. In *Proceedings of the 2000 Wireless Communications and Networking Confernce (WCNC 2000)*, 2000.

[85] Sung-Ju Lee, William Su, and Mario Gerla. On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks. *Mobile Networks and Applications*, 7(6), 2002.

[86] Li Erran Li, Richard Alimi, Dawei Shen, Harish Viswanathan, and Y. Richard Yang. A General Algorithm for Interference Alignment and Cancellation in Wireless Networks. In *Proceedings of the 29th IEEE Conference on Computer Communications (INFOCOM 2010)*, 2010.

[87] Tianji Li, Mi Kyung Han, Apurv Bhartia, Lili Qiu, Eric Rozner, Yin Zhang, and Brad Zarikoff. CRMA: Collision-Resistant Multiple Access. In *Proceedings of the 17th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2011)*, 2011.

[88] Soung Chang Liew, Shengli Zhang, and Lu Lu. Physical-Layer Network Coding: Tutorial, Survey, and Beyond. *Physical Communication*, 6, 2013.

[89] Kate Ching-Ju Lin, Nate Kushman, and Dina Katabi. ZipTx: Harnessing Partial Packets in 802.11 Networks. In *Proceedings of the 14th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2008)*, 2008.

[90] Yu-Jen Lin, Chen-Che Huang, and Jiun-Long Huang. PipelineOR: A Pipelined Opportunistic Routing Protocol with Network Coding in Wireless Mesh Networks.

In *Proceedings of the 71st IEEE Vehicular Technology Conference (VTC 2010 Spring)*, 2010.

[91] Yunfeng Lin, Baochun Li, and Ben Liang. CodeOR: Opportunistic Routing in Wireless Mesh Networks with Segmented Network Coding. In *Proceedings of the 2008 IEEE International Conference on Network Protocols (ICNP 2008)*, 2008.

[92] Jiajia Liu, Xiaohong Jiang, Hiroki Nishiyama, and Nei Kato. Exact Throughput Capacity under Power Control in Mobile Ad Hoc Networks. In *Proceedings of the 31st IEEE Conference on Computer Communications (INFOCOM 2012)*, 2012.

[93] Vincent Liu, Vamsi Talla, and Shyamnath Gollakota. Enabling Instantaneous Feedback with Full-Duplex Backscatter. In *Proceedings of the 20th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2014)*, 2014.

[94] Adrian Loch, Robin Klose, and Matthias Hollick. Practical OFDMA in Wireless Networks with Multiple Transmitter-Receiver Pairs. In *Proceedings of the 14th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2013)*, 2013.

[95] Adrian Loch, Thomas Nitsche, Alexander Kuehne, Matthias Hollick, Joerg Widmer, and Anja Klein. Practical challenges of IA in frequency. Technical report, TU Darmstadt, SEEMOO, 2013.

[96] David Love, Robert Heath, Vincent N. Lau, David Gesbert, Bhaskar Rao, and Matthew Andrews. An Overview of Limited Feedback in Wireless Communication Systems. *IEEE Journal on Selected Areas in Communications*, 26(8), 2008.

[97] Lu Lu and Soung Chang Liew. Asynchronous Physical-Layer Network Coding. *IEEE Transactions on Wireless Communications*, 11(2), 2012.

[98] Lu Lu, Taotao Wang, Soung Chang Liew, and Shengli Zhang. Implementation of Physical-Layer Network Coding. *Physical Communication*, 6, 2013.

[99] Desmond S. Lun, Muriel Médard, and Ralf Koetter. Network Coding for Efficient Wireless Unicast. In *Proceedings of the 2006 International Zurich Seminar on Communications (IZS 2006)*, 2006.

[100] Victoria Manfredi, Robert Hancock, and Jim Kurose. Robust Routing in Dynamic MANETs. In *Proceedings of the 2nd Annual Conference of the International Technology Alliance (ACITA 2008)*, 2008.

[101] Justin Manweiler, Naveen Santhapuri, Souvik Sen, Romit Roy Choudhury, Srihari Nelakuditi, and Kamesh Munagala. Order Matters: Transmission Reordering in Wireless Networks. In *Proceedings of the 15th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2009)*, 2009.

[102] Ivana Maric, Andrea Goldsmith, and Muriel Médard. Analog Network Coding in the High-SNR Regime. In *Proceedings of the 3rd IEEE International Workshop on Wireless Network Coding (WiNC 2010)*, 2010.

[103] Jackson W. Massey, Jonathan Starr, Seogoo Lee, Dongwook Lee, Andreas Gerstlauer, and Robert W. Heath. Implementation of a Real-Time Wireless Interference Alignment Network. In *Proceedings of the 46th Asilomar Conference on Signals, Systems and Computers (ASILOMAR 2012)*, 2012.

[104] Martin Mauve, Joerg Widmer, and Hannes Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE Network*, 15(6), 2001.

[105] Kyle Miller, Atresh Sanne, Kannan Srinivasan, and Sriram Vishwanath. Enabling Real-Time Interference Alignment. In *Proceedings of the 13th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2012)*, 2012.

[106] Andreas Molisch. *Wireless Communications*. Wiley, 2011.

[107] Michele Morelli, C.-C. Jay Kuo, and Man-On Pun. Synchronization Techniques for Orthogonal Frequency Division Multiple Access (OFDMA): A Tutorial Review. *Proceedings of the IEEE*, 95(7), 2007.

[108] Stephen Mueller, Rose P. Tsang, and Dipak Ghosal. Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges. *Lecture Notes in Computer Science*, 2965, 2004.

[109] Patrick Murphy. *Design, Implementation and Characterization of a Cooperative Communications System*. PhD thesis, Rice University, 2010.

[110] Nagesh S. Nandiraju, Deepti S. Nandiraju, and Dharma P. Agrawal. Multipath Routing in Wireless Mesh Networks. In *Proceedings of the 2006 International Conference on Mobile Adhoc and Sensor Systems (MASS 2006)*, 2006.

[111] Man Cheuk Ng, Kermin Elliott Fleming, Mythili Vutukuru, Samuel Gross, and Hari Balakrishnan. Airblue: A System for Cross-Layer Wireless Protocol Development. In *Proceedings of the 6th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS 2010)*, 2010.

[112] Dragos Niculescu and Badri Nath. Trajectory Based Forwarding and Its Applications. In *Proceedings of the 9th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2003)*, 2003.

[113] José Núñez Martínez, Jorge Baranda, and Josep Mangues-Bafalluy. Experimental Evaluation of Self-Organized Backpressure Routing in a Wireless Mesh Backhaul of Small Cells. *Ad Hoc Networks*, 24, 2015.

[114] Jose Nuñez Martinez, Josep Mangues-Bafalluy, and Jorge Baranda. Anycast Backpressure Routing: Scalable Mobile Backhaul for Dense Small Cell Deployments. *IEEE Communications Letters*, 17(12), 2013.

[115] George Nychis, Thibaud Hottelier, Zhuocheng Yang, Srinivasan Seshan, and Peter Steenkiste. Enabling MAC Protocol Implementations on Software-Defined Radios. In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2009)*, 2009.

[116] Chang Hwan Park, Hyun Il Yoo, Yeong Jun Kim, Dong Seung Kwon, and Yong Soo Cho. An Analysis of TDoA Effect for OFDMA-Based Wireless Mesh Networks. In *Proceedings of the 2011 IEEE International Conference on Communications (ICC 2011)*, 2011.

[117] Myonghee Park, Kyunbyoung Ko, Byungjoon Park, and Daesik Hong. Effects of Asynchronous MAI on Average SEP Oerformance of OFDMA Uplink Systems over Frequency-Selective Rayleigh Fading Channels. *IEEE Transactions on Communications*, 58(2), 2010.

[118] Charles Perkins, Elizabeth M. Belding-Royer, and Samir Das. Ad hoc On-Demand Distance Vector (AODV) Routing. Technical report, IETF, 2003.

[119] Steven Peters and Robert W. Heath. Orthogonalization to Reduce Overhead in MIMO Interference Channels. In *Proceedings of the 2010 International Zurich Seminar on Communications (IZS 2010)*, 2010.

[120] Steven W. Peters and Robert W. Heath. Interference Alignment via Alternating Minimization. In *Proceedings of the 34th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, 2009.

[121] Wei Pu, Chong Luo, Binxing Jiao, and Feng Wu. Natural Network Coding in Multi-Hop Wireless Networks. In *Proceedings of the 2008 IEEE International Conference on Communications (ICC 2008)*, 2008.

[122] Oscar Puñal, Humberto Escudero, and James Gross. Power Loading: Candidate for Future WLANs? In *Proceedings of the 13th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2012)*, 2012.

[123] André Puschmann, Mohamed A. Kalil, and Andreas Mitschele-Thiel. Implementation and Evaluation of a Practical SDR Testbed. In *Proceedings of the 4th International Conference on Cognitive Radio and Advanced Spectrum Management (CogART 2011)*, 2011.

[124] Božidar Radunović, Christos Gkantsidis, Peter Key, and Pablo Rodriguez. An Optimization Framework for Opportunistic Multipath Routing in Wireless Mesh Networks. In *Proceedings of the 2008 Conference on Computer Communications (INFOCOM 2008)*, 2008.

[125] Hariharan Rahul, Haitham Hassanieh, and Dina Katabi. SourceSync: A Distributed Wireless Architecture for Exploiting Sender Diversity. In *Proceedings of the 2010 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2010)*, 2010.

[126] Hariharan Shankar Rahul, Swarun Kumar, and Dina Katabi. JMB: Scaling Wireless Capacity with User Demands. In *Proceedings of the 2012 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2012)*, 2012.

[127] Peyman Razaghi and Giuseppe Caire. Interference Alignment, Carrier Pairing, and Lattice Decoding. In *Proceedings of the 8th International Symposium on Wireless Communication Systems (ISWCS 2011)*, 2011.

[128] Bogdan Roman, Frank Stajano, Ian Wassell, and David Cottingham. Multi-Carrier Burst Contention (MCBC): Scalable Medium Access Control for Wireless Networks. In *Proceedings of the 2008 IEEE Wireless Communications and Networking Conference (WCNC 2008)*, 2008.

[129] Yue Rong, Sergiy A. Vorobyov, and Alex B. Gershman. Adaptive OFDM Techniques With One-Bit-Per-Subcarrier Channel-State Feedback. *IEEE Transactions on Communications*, 54(11), 2006.

[130] Lynne Salameh, Astrit Zhushi, Mark Handley, Kyle Jamieson, and Brad Karp. HACK: Hierarchical ACKs for Efficient Wireless Medium Utilization. In *Proceedings of the 2014 USENIX Annual Technical Conference (USENIX ATC 2014)*, 2014.

[131] Thomas Schmid, Oussama Sekkat, and Mani B. Srivastava. An Experimental Study of Network Performance Impact of Increased Latency in Software Defined Radios. In *Proceedings of the 2nd ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WinTECH 2007)*, 2007.

[132] Souvik Sen, Romit Roy Choudhury, and Srihari Nelakuditi. No Time to Countdown: Migrating Backoff to the Frequency Domain. In *Proceedings of the 17th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2011)*, 2011.

[133] Andrew Sendonaris, Elza Erkip, and Behnaam Aazhang. User Cooperation Diversity - Part I: System Description. *IEEE Transactions on Communications*, 51(11), 2003.

[134] Andrew Sendonaris, Erkip Erkip, and Behnaam Aazhang. User cooperation diversity-part II: implementation aspects and performance analysis. *IEEE Transactions on Communications*, 51(11), 2003.

[135] Kibeom Seong, Mehdi Mohseni, and John Cioffi. Optimal Resource Allocation for OFDMA Downlink Systems. In *Proceedings of the 2006 IEEE International Symposium on Information Theory (ISIT 2006)*, 2006.

[136] Samat Shabdanov, Patrick Mitran, and Catherine Rosenberg. Cross-Layer Optimization Using Advanced Physical Layer Techniques in Wireless Mesh Networks. *IEEE Transactions on Wireless Communications*, 11(4), 2012.

[137] Meng Shen, Chunming Zhao, Xiao Liang, and Zhi Ding. Best-Effort Interference Alignment in OFDM Systems with Finite SNR. In *Proceedings of the 2011 IEEE International Conference on Communications (ICC 2011)*, 2011.

[138] Minqi Shen, Anders Host-Madsen, and Josep Vidal. An improved interference alignment scheme for frequency selective channels. In *Proceedings of the 2008 IEEE International Symposium on Information Theory (ISIT 2008)*, 2008.

[139] Yushi Shen and Ed Martinez. Channel Estimation in OFDM Systems. Technical report, Freescale Semiconductor, 2006.

[140] Shailendra Singh, Moloud Shahbazi, Konstantinos Pelechrinis, Karthikeyan Sundaresan, Srikanth V. Krishnamurthy, and Sateesh Addepalli. A Case for Adaptive Sub-carrier Level Power Allocation in OFDMA Networks. In *Proceedings of the 13th*

ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2012), 2012.

[141] Vineet Srivastava and Mehul Motani. Cross-layer Design: A Survey and the Road Ahead. *IEEE Communications Magazine*, 43(12), 2005.

[142] Andrej Stefanov and Elza Erkip. Cooperative Coding for Wireless Networks. *IEEE Transactions on Communications*, 52(9), 2004.

[143] Karthikeyan Sundaresan, Mohammad Khojastepour, Eugene Chai, and Sampath Rangarajan. Full-Duplex without Strings: Enabling Full-Duplex with Half-Duplex Clients. In *Proceedings of the 20th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2014)*, 2014.

[144] Kun Tan, He Liu, Jiansong Zhang, Yongguang Zhang, Ji Fang, and Geoffrey M. Voelker. Sora: High-Performance Software Radio Using General Purpose Multi-core Processors. *Communications of the ACM*, 54(1), 2011.

[145] Leandros Tassiulas and Anthony Ephremides. Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks. *IEEE Transactions on Automatic Control*, 37(12), 1992.

[146] Stefan Valentin, Thomas Freitag, and Holger Karl. Integrating Multiuser Dynamic OFDMA into IEEE 802.11 WLANs - LLC/MAC Extensions and System Performance. In *Proceedings of the 2008 IEEE International Conference on Communications (ICC 2008)*, 2008.

[147] Marzieh Veyseh, J. J. Garcia-Luna-Aceves, and Hamid R. Sadjadpour. OFDMA Based Multiparty Medium Access Control in Wireless Ad Hoc Networks. In *Proceedings of the 2009 IEEE International Conference on Communications (ICC 2009)*, 2009.

[148] Marzieh Veyseh, J. J. Garcia-Luna-Aceves, and Hamid R. Sadjadpour. Adaptive diversity based spectrum allocation in single-radio wireless ad hoc networks. In *Proceedings of the 7th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2010)*, 2010.

[149] Marzieh Veyseh, J. J. Garcia-Luna-Aceves, and Hamid R. Sadjadpour. Cross-Layer Channel Allocation Protocol for OFDMA Ad Hoc Networks. In *Proceedings of the 2010 IEEE Global Telecommunications Conference (GLOBECOM 2010)*, 2010.

[150] Marzieh Veyseh, J. J. Garcia-Luna-Aceves, and Hamid R. Sadjadpour. Multi-User Diversity in Single-Radio OFDMA Ad Hoc Networks Based on Gibbs Sampling. In *Proceedings of the 2010 Military Communications Conference (MILCOM 2010)*, 2010.

[151] Marzieh Veyseh and J.J. Garcia-Luna-Aceves. Parallel Interaction Medium Access for Wireless Ad Hoc Networks. In *Proceedings of the 17th International Conference on Computer Communications and Networks (ICCCN 2008)*, 2008.

[152] Christian Vogt, Michael Gerharz, and Christian De Waal. Corridor Routing in Mobile Ad-hoc Networks. In *Proceedings of the 3rd Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2004)*, 2004.

[153] Mythili Vutukuru. *Physical Layer-AwareWireless Link Layer Protocols*. PhD thesis, Massachusetts Institute of Technology, 2010.

[154] Mythili Vutukuru, Hari Balakrishnan, and Kyle Jamieson. Cross-Layer Wireless Bit Rate Adaptation. In *Proceedings of the 2009 ACM SIGCOMM Conference on Data Communication (SIGCOMM 2009)*, 2009.

[155] Mythili Vutukuru, Kyle Jamieson, and Hari Balakrishnan. Harnessing Exposed Terminals in Wireless Networks. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2008)*, 2008.

[156] Xin Wang and J. J. Garcia-Luna-Aceves. Collaborative Routing, Scheduling and Frequency Assignment for Wireless Ad Hoc Networks using Spectrum-Agile Radios. *Wireless Networks*, 17(1), 2011.

[157] Cheong Yui Wong, Roger S. Cheng, Khaled Ben Lataief, and Ross D. Murch. Multiuser OFDM with Adaptive Subcarrier, Bit, and Power Allocation. *IEEE Journal on Selected Areas in Communications*, 17(10), 1999.

[158] Grace R. Woo, Pouya Kheradpour, Dawei Shen, and Dina Katabi. Beyond the Bits: Cooperative Packet Recovery Using Physical Layer Information. In *Proceedings of the 13th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2007)*, 2007.

[159] Jin Xie, Wei Hu, and Zhenghao Zhang. Revisiting Partial Packet Recovery in 802.11 Wireless LANs. In *Proceedings of the 9th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys 2011)*, 2011.

[160] Hui Xu, J. J. Garcia-Luna-Aceves, and Hamid R. Sadjadpour. Exploiting the Capture Effect Opportunistically in MANETs. In *Proceedings of the 2010 Military Communications Conference (MILCOM 2010)*, 2010.

[161] Hui Xu, J.J. Garcia-Luna-Aceves, and Hamid R. Sadjadpour. Incorporating Physical-Layer Effects in Modeling of MAC Protocols Operating in MANETs. In *Proceedings of the 2010 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2010)*, 2010.

[162] Kaixin Xu, Xiaoyan Hong, and Mario Gerla. An Ad Hoc Network with Mobile Backbones. In *Proceedings of the 2002 IEEE International Conference on Communications (ICC 2002)*, 2002.

[163] Elias Yaacoub, Zaher Dawy, and Mohamad Adnan Al-Alaoui. A Novel Distributed Scheduling Scheme for OFDMA Uplink using Channel Information and Probabilistic Transmission. *Computer Communications*, 34(17), 2011.

[164] Zhe Yang, Yuanqian Luo, and Lin Cai. Network Modulation: A New Dimension to Enhance Wireless Network Performance. In *Proceedings of the 30th IEEE Conference on Computer Communications (INFOCOM 2011)*, 2011.

[165] Lei Ying, Sanjay Shakkottai, and Aneesh Reddy. On Combining Shortest-Path and Back-Pressure Routing Over Multihop Wireless Networks. In *Proceedings of the 28th IEEE Conference on Computer Communications (INFOCOM 2009)*, 2009.

[166] Jongwon Yoon, Mustafa Y. Arslan, Karthikeyan Sundaresan, Srikanth V. Krishnamurthy, and Suman Banerjee. A Distributed Resource Management Framework for Interference Mitigation in OFDMA Femtocell Networks. In *Proceedings of the 13th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2012)*, 2012.

[167] Jane Y. Yu and Peter H. J. Chong. A Survey of Clustering Schemes for Mobile Ad Hoc Networks. *IEEE Communications Surveys & Tutorials*, 7(1), 2005.

[168] Dingwen Yuan, Michael Riecker, and Matthias Hollick. Making 'Glossy' Networks Sparkle: Exploiting Concurrent Transmissions for Energy Efficient, Reliable, Ultra-Low Latency Communication in Wireless Control Networks. *Lecture Notes in Computer Science*, 8354, 2014.

[169] Georg Zeitler, Gerhard Bauch, and Joerg Widmer. Quantize-and-Forward Schemes for the Orthogonal Multiple-Access Relay Channel. *IEEE Transactions on Communications*, 60(4), 2012.

[170] Shengli Zhang, Soung Chang Liew, and Patrick P. Lam. Hot Topic: Physical-Layer Network Coding. In *Proceedings of the 12th ACM Annual International Conference on Mobile Computing and Networking (MobiCom 2006)*, 2006.

[171] Xi Zhang, Junaid Ansari, Manish Arya, and Petri Mähönen. Exploring Parallelization for Medium Access Schemes on Many-core Software Defined Radio Architecture. In *Proceedings of the 2nd ACM Workshop on Software Radio Implementation Forum (SRIF 2013)*, 2013.

[172] Xiaolu Zhang, Wenhua Jiao, and Meixia Tao. End-to-End Resource Allocation in OFDM Based Linear Multi-Hop Networks. In *Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM 2008)*, 2008.

[173] Zhongshan Zhang and Chintha Tellambura. The Effect of Imperfect Carrier Frequency Offset Estimation on an OFDMA Uplink. *IEEE Transactions on Communications*, 57(4), 2009.

[174] Pan Zhou, Guowang Miao, and Benny Bing. Cross-Layer Congestion Control and Scheduling in Multi-Hop OFDMA Wireless Networks. In *Proceedings of the 2009 IEEE Global Telecommunications Conference (GLOBECOM 2009)*, 2009.

[175] Bastian Zimmermann and Jan Weber. Wireless Synchronization - Enabling Simultaneous Transmissions. Lab report, TU Darmstadt, SEEMOO, 2014.

[176] GNU Radio. http://www.gnu.org/software/gnuradio/.

[177] USRP. The universal software radio peripheral. http://www.ettus.com/.

[178] WARP 802.11 Reference Design. http://warpproject.org/trac/wiki/802.11.

[179] WARP OFDM Reference Design. http://warpproject.org/trac/wiki/OFDMReferenceDesign.

[180] WARP Project. http://warp.rice.edu.

[181] WARPLab Reference Design. http://warpproject.org/trac/wiki/WARPLab.

Part IV

APPENDIX

A

WARP DRIVE PARAMETERS

Table A.1 summarizes the default parameters for our experiments using WARP Drive, if not stated otherwise in the description of the individual experiments.

Parameter	**Value**
Upper Layers	
Data packet size	125 kBit
Number of data packets per connection	3
Network Layer	
Minimum SNR of corridor links	10 dB
Corridor width	$w \in [2,3,4]$
Network size in simulations	$\approx$ 50 nodes
Network size in practical experiments	$\approx$ 20 nodes
Link Layer	
Maximum number of retransmissions	5
Discrete-rate bits per symbol (BPS)	BPS $\in [1,2,4,6,8]$
Discrete-rate coding rates	1/2, 2/3, 3/4, 5/6
Rateless batch size	33
Rateless bits per data packet	362 bits
Rateless number of batches	27
CRC size	16 bits
Physical Layer	
FFT size	128
Usable subcarriers	48
Subcarrier spacing	312.5 kHz
Subchannel width	3 subcarriers
CP size	25%
Number of OFDM pilot symbols per frame	10
Codebook size	16 entries
Channel and SDR Characteristics	
WARP buffer size	2^{15} samples
WARP sampling rate	40 MHz
Channel bandwidth	18 MHz
Coherence time for throughput computation	45 ms

Table A.1: Overview on default parameters for our experiments based on WARP Drive.

B

CURRICULUM VITÆ

PERSONAL INFORMATION

Name	Adrian Carlos Loch Navarro
Date of Birth	September 8, 1987
Place of Birth	Madrid, Spain
Nationality	Spanish

EDUCATION

since 2011	Technische Universität Darmstadt, Darmstadt, Germany Doctoral candidate at the Department of Computer Science
2009 – 2011	Technische Universität Darmstadt, Darmstadt, Germany Studies of Electrical Engineering and Information Technology Degree: Master of Science in Data Technology
2005 – 2011	Universidad Politécnica de Madrid, Madrid, Spain Studies of Telecommunication Engineering Degree: Ingeniero de Telecomunicación (M.Sc. equivalent)
1993 – 2005	Deutsche Schule Madrid, Madrid, Spain Degree: Abitur (German high school diploma)

WORK EXPERIENCE

since 2011	Technische Universität Darmstadt, Darmstadt, Germany Research associate at the Secure Mobile Networking Lab

AWARDS

Publication	Best Paper Award at IEEE WoWMoM 2014 Paper title: “Practical Interference Alignment in the Frequency Domain for OFDM-based Wireless Access Networks”
Publication	Best Poster Award at IEEE WoWMoM 2014 Poster title: “Corridor-based Routing: Opening Doors to PHY-Layer Advances for Wireless Multihop Networks”
Publication	Best Demo Award at IEEE WoWMoM 2013 Demo title: “Practical OFDMA in Wireless Networks with Multiple Transmitter-Receiver Pairs”
Thesis	Best Master Thesis Award 2011 in the fields of Computer Science, Mathematics and Engineering Management by Datenlotsen Informationssysteme (University-wide)

Thesis	Outstanding Master Thesis Award 2011 in the field of Computer Science by the Fakultätentag Informatik (Germany-wide)
Studies	Best Master Student Award 2011 at the Electrical Engineering and Information Technology department by CST AG
Studies	Award for Outstanding Academic Achievements 2005 at Deutsche Schule Madrid

Darmstadt, February 5, 2015

C

AUTHOR'S PUBLICATIONS

PRIMARY AUTHOR

1. Adrian Loch, Pablo Quesada, Matthias Hollick, Alexander Kuehne, and Anja Klein. Building Cross-Layer Corridors in Wireless Multihop Networks. In *Proceedings of the 11th IEEE International Conference on Mobile Ad hoc and Sensor Systems (MASS 2014)*, 2014
2. Adrian Loch, Wei-Chih Hong, and Matthias Hollick. Dynamic Curve Adaptation for Geographic Routing in Wireless Multihop Networks. In *Proceedings of the 39th IEEE Conference on Local Computer Networks (LCN 2014)*, 2014
3. Adrian Loch, Matthias Hollick, Alexander Kuehne, and Anja Klein. Practical OFDMA for Corridor-based Routing in Wireless Multihop Networks. In *Proceedings of the 39th IEEE Conference on Local Computer Networks (LCN 2014)*, 2014
4. Adrian Loch, Matthias Schulz, and Matthias Hollick. WARP Drive — Accelerating Wireless Multi-hop Cross-layer Experimentation on SDRs (Demo). In *Proceedings of the 3rd Workshop of Software Radio Implementation Forum (SRIF 2014)*, 2014
5. Adrian Loch, David Meier, and Matthias Hollick. How did you get here? PHY-Layer Path Signatures (Work-in-Progress). In *Proceedings of the 15th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2014)*, 2014
6. Adrian Loch, Matthias Hollick, Alexander Kuehne, and Anja Klein. Corridor-based Routing: Opening Doors to PHY-Layer Advances for Wireless Multihop Networks (Work-in-Progress). In *Proceedings of the 15th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2014)*, 2014
7. Adrian Loch, Matthias Hollick, Thomas Nitsche, Joerg Widmer, Alexander Kuehne, and Anja Klein. Practical Interference Alignment in the Frequency Domain for OFDM-based Wireless Access Networks. In *Proceedings of the 15th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2014)*, 2014
8. Adrian Loch, Matthias Hollick, Thomas Nitsche, Joerg Widmer, Alexander Kuehne, and Anja Klein. CSI Feedback in OFDMA Wireless Networks with Multiple Transmitter-Receiver Pairs. In *Proceedings of the 14th IEEE International Workshop on Signal Processing Advances for Wireless Communications (SPAWC 2013)*, 2013
9. Adrian Loch, Robin Klose, Matthias Hollick, Alexander Kuehne, and Anja Klein. Practical OFDMA in Wireless Networks with Multiple Transmitter-Receiver Pairs (Demo). In *Proceedings of the 14th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2013)*, 2013

10. Adrian Loch, Hannes Frey, and Matthias Hollick. Curve-based Planar Graph Routing with Guaranteed Delivery in Multihop Wireless Networks (extended version of WoWMoM 2012 paper). *Pervasive and Mobile Computing*, 11, 2014
11. Adrian Loch, and Matthias Hollick. Face the Enemy: Attack Detection for Planar Graph Routing. In *Proceedings of the 2013 Conference on Networked Systems (NetSys 2013)*, 2013
12. Hannes Frey, Matthias Hollick, and Adrian Loch. Curve-based Planar Graph Routing with Guaranteed Delivery in Multihop Wireless Networks. In *Proceedings of the 13th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2012)*, 2012
13. Hannes Frey, Matthias Hollick, and Adrian Loch. Curve-based Planar Graph Routing in Multihop Wireless Networks (Poster). In *Proceedings of the 8th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2011)*, 2011

CO-AUTHOR

14. Alexander Kuehne, Adrian Loch, Thomas Nitsche, Joerg Widmer, Matthias Hollick, and Anja Klein. BER Enhancements for Practical Interference Alignment in the Frequency Domain. In *Proceedings of the International Symposium on Wireless Communication Systems (ISWCS 2014)*, 2014
15. Matthias Schulz, Adrian Loch, and Matthias Hollick. Practical Known-Plaintext Attacks against Physical Layer Security in Wireless MIMO Systems. In *Proceedings of the Network and Distributed System Security Symposium (NDSS 2014)*, 2014
16. Robin Klose, Adrian Loch, and Matthias Hollick. Evaluating Dynamic OFDMA Subchannel Allocation for Wireless Mesh Networks on SDRs. In *Proceedings of the 2nd Workshop of Software Radio Implementation Forum (SRIF 2013)*, 2013
17. Thomas Nitsche, Joerg Widmer, Adrian Loch, and Matthias Hollick. EVM and RSSI Link Quality Measurements in Frequency Selective Fading Channels. In *Proceedings of the 14th IEEE International Workshop on Signal Processing Advances for Wireless Communications (SPAWC 2013)*, 2013
18. Robin Klose, Adrian Loch, and Matthias Hollick. A Rapid Prototyping Framework for Practical OFDMA Systems using Software Defined Radios. In *Proceedings of the 2013 IEEE International Conference on Sensing, Communication, and Networking (SECON 2013)*, 2013
19. Alexander Kuehne, Adrian Loch, Matthias Hollick, and Anja Klein. Node Selection for Corridor-based Routing in OFDMA Multihop Networks. In *Proceedings of the Wireless Communications and Networking Conference (WCNC 2013)*, 2013
20. Alexander Kuehne, Adrian Loch, Matthias Hollick, and Anja Klein. Spatial Reuse in OFDMA Multi-Hop Networks Applying Corridor-based Routing. In *Proceedings of the ITG Workshop on Smart Antennas (WSA 2013)*, 2013

21. Alexander Kuehne, Anja Klein, Adrian Loch, and M. Hollick. Corridor-based Routing Using Opportunistic Forwarding in Multi-Hop OFDMA Networks. In *Proceedings of the International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2012)*, 2012
22. Alexander Kuehne, Anja Klein, Adrian Loch, and M. Hollick. Opportunistic Forwarding in Multi-Hop OFDMA Networks with Local CSI. In *Proceedings of the 17th International OFDM-Workshop (InOWo 2012)*, 2012
23. Zhiliang Chen, Alexander Kuehne, Anja Klein, Adrian Loch, Matthias Hollick, and Joerg Widmer. Two-Way Relaying for Multiple Applications in Wireless Sensor Networks. In *Proceedings of the ITG Workshop on Smart Antennas (WSA 2012)*, 2012

D

ERKLÄRUNG LAUT §9 DER PROMOTIONSORDNUNG

Ich versichere hiermit, dass ich die vorliegende Dissertation selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmitteln verfasst habe. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch nicht zu Prüfungszwecken gedient.

Darmstadt, 5. Februar 2015

Adrian Carlos Loch Navarro