TECHNISCHE UNIVERSITÄT DRESDEN

Compute-and-Forward in Multi-User Relay Networks — Optimization, Implementation, and Secrecy —

Johannes Richter

von der Fakultät Elektrotechnik und Informationstechnik
der Technischen Universität Dresden

zur Erlangung des akademischen Grades

D O K T O R I N G E N I E U R
(Dr.-Ing.)

genehmigte Dissertation

Vorsitzende:	Prof. Dr.-Ing. habil. Renate Merker
Gutachter:	Prof. Dr.-Ing. Eduard A. Jorswieck
	Prof. Dr.-Ing. Aydin Sezgin
	Prof. Ing. CSc. Jan Sýkora
Tag der Einreichung:	06. 03. 2017
Tag der Verteidigung:	26. 04. 2017

Abstract

In this thesis, we investigate physical-layer network coding in an $L \times M \times K$ relay network, where L source nodes want to transmit messages to K sink nodes via M relay nodes. We focus on the information processing at the relay nodes and the compute-and-forward framework. Nested lattice codes are used, which have the property that every linear combination of codewords is a valid codeword. This property is essential for physical-layer network coding.

Because the actual network coding occurs on the physical layer, the network coding coefficients are determined by the channel realizations. Finding the optimal network coding coefficients for given channel realizations is a non-trivial optimization problem. In this thesis, we provide an algorithm to find network coding coefficients that result in the highest data rate at a chosen relay. The solution of this optimization problem is only locally optimal, i.e., it is optimal for a particular relay. If we consider a multi-hop network, each potential receiver must get enough linear independent combinations to be able to decode the individual messages. If this is not the case, outage occurs, which results in data loss. In this thesis, we propose a new strategy for choosing the network coding coefficients locally at the relays without solving the optimization problem globally. We thereby reduce the solution space for the relays such that linear independence between their decoded linear combinations is guaranteed. Further, we discuss the influence of spatial correlation on the optimization problem.

Having solved the optimization problem, we combine physical-layer network coding with physical-layer secrecy. This allows us to propose a coding scheme to exploit untrusted relays in multi-user relay networks. We show that physical-layer network coding, especially compute-and-forward, is a key technology for simultaneous and secure communication of several users over an untrusted relay. First, we derive the achievable secrecy rate for the two-way relay channel. Then, we enhance this scenario to a multi-way relay channel with multiple antennas.

We describe our implementation of the compute-and-forward framework with software-defined radio and demonstrate the practical feasibility. We show that it is possible to use the framework in real-life scenarios and demonstrate a transmission from two users to a relay. We gain valuable insights into a real transmission using the compute-and-forward framework. We discuss possible improvements of the current implementation and point out further work.

Zusammenfassung

In dieser Arbeit untersuchen wir Netzwerkcodierung auf der Übertragungsschicht in einem Relay-Netzwerk, in dem L Quellen-Knoten Nachrichten zu K Senken-Knoten über M Relay-Knoten senden wollen. Der Fokus dieser Arbeit liegt auf der Informationsverarbeitung an den Relay-Knoten und dem Compute-and-Forward Framework. Es werden Nested Lattice Codes eingesetzt, welche die Eigenschaft besitzen, dass jede Linearkombination zweier Codewörter wieder ein gültiges Codewort ergibt. Dies ist eine Eigenschaft, die für die Netzwerkcodierung von entscheidender Bedeutung ist.

Da die eigentliche Netzwerkcodierung auf der Übertragungsschicht stattfindet, werden die Netzwerkcodierungskoeffizienten von den Kanalrealisierungen bestimmt. Das Finden der optimalen Koeffizienten für gegebene Kanalrealisierungen ist ein nicht-triviales Optimierungsproblem. Wir schlagen in dieser Arbeit einen Algorithmus vor, welcher Netzwerkcodierungskoeffizienten findet, die in der höchsten Übertragungsrate an einem gewählten Relay resultieren. Die Lösung dieses Optimierungsproblems ist zunächst nur lokal, d. h. für dieses Relay, optimal. An jedem potentiellen Empfänger müssen ausreichend unabhängige Linearkombinationen vorhanden sein, um die einzelnen Nachrichten decodieren zu können. Ist dies nicht der Fall, kommt es zu Datenverlusten. Um dieses Problem zu umgehen, ohne dabei das Optimierungsproblem global lösen zu müssen, schlagen wir eine neue Strategie vor, welche den Lösungsraum an einem Relay soweit einschränkt, dass lineare Unabhängigkeit zwischen den decodierten Linearkombinationen an den Relays garantiert ist. Außerdem diskutieren wir den Einfluss von räumlicher Korrelation auf das Optimierungsproblem.

Wir kombinieren die Netzwerkcodierung mit dem Konzept von Sicherheit auf der Übertragungsschicht, um ein Übertragungsschema zu entwickeln, welches es ermöglicht, mit Hilfe nicht-vertrauenswürdiger Relays zu kommunizieren. Wir zeigen, dass Compute-and-Forward ein wesentlicher Baustein ist, um solch eine sichere und simultane Übertragung mehrerer Nutzer zu gewährleisten. Wir starten mit dem einfachen Fall eines Relay-Kanals mit zwei Nutzern und erweitern dieses Szenario auf einen Relay-Kanal mit mehreren Nutzern und mehreren Antennen.

Die Arbeit wird abgerundet, indem wir eine Implementierung des Compute-and-Forward Frameworks mit Software-Defined Radio demonstrieren. Wir zeigen am Beispiel von zwei Nutzern und einem Relay, dass sich das Framework eignet, um in realen Szenarien eingesetzt zu werden. Wir diskutieren mögliche Verbesserungen und zeigen Richtungen für weitere Forschungsarbeit auf.

Contents

Abbreviations

AF	amplify-and-forward
API	application programming interface
ARQ	automatic repeat request
AWGN	additive white Gaussian noise
BC	broadcast channel
CDMA	code-division multiple access
CF	compute-and-forward
CQIP	convex quadratic integer programming
CVP	closest-vector problem
DDSS	distributed data storage systems
DF	decode-and-forward
DMC	discrete memoryless channel
DMS	discrete memoryless source
DWTC	degraded wiretap channel
FDMA	frequency-division multiple access
i.i.d.	independent and identically distributed
MAC	multiple-access channel
MIMO	multiple-input multiple-output
MISO	multiple-input single-output
ML	maximum-likelihood
MMSE	minimum mean square error
MRT	maximum-ratio transmission
PLNC	physical-layer network coding
RRC	root raised cosine
SDR	software-defined radio

SIC	successive interference cancellation
SIMO	single-input multiple-output
SINR	signal-to-interference-plus-noise ratio
SISO	single-input single-output
SNR	signal-to-noise ratio
SVP	shortest-vector problem
TCP	transmission control protocol
TDMA	time-division multiple access
USRP	universal software radio peripheral
VNR	volume-to-noise ratio

Notation

Mathematical Operators and Functions

x'	transpose of a matrix or vector
$\|x\|$	ℓ_2-norm of a vector
$\log_2^+(x)$	$\max\{\log_2(x), 0\}$
$+$	addition over the reals
$\oplus$	addition over the finite field
$\sum$	summation over the reals
$\bigoplus$	summation over the finite field
I_n	identity matrix of size n
e_i	unity vector with a one at the i-th position, zeros elsewhere
(a_{ij})	matrix consisting of the elements a_{ij}, where i is the row index and j is the column index
$\{a_k\}$	set with elements a_k
$\mathcal{P}(\mathcal{X})$	power set of a set $\mathcal{X}$
$\Re(x)$	real part of a complex number x
$\Im(x)$	imaginary part of a complex number x
$\mathrm{diag}(x)$	function that takes a vector x and returns a matrix with the elements of x on the diagonal, zeros elsewhere
$\mathrm{sgn}(x)$	function that returns -1 if $x < 0$, 0 if $x = 0$, and 1 if $x > 0$
$\sup(\mathcal{S})$	supremum of a set $\mathcal{S}$
$\mathrm{cl}(\mathcal{S})$	closure of a set $\mathcal{S}$

Probability Theory

$\Pr(A)$	probability of event A
$\mathrm{E}[X]$	expectation of a random variable X
$\mathcal{N}(\mu, \sigma^2)$	normal or Gaussian distribution with mean μ and variance σ^2
$\mathcal{U}(\mathcal{S})$	uniform distribution over the set $\mathcal{S}$

Number Sets

$\mathbb{N}$	set of positive integers
$\mathbb{Z}$	set of integers
$\mathbb{Q}$	set of rational numbers
$\mathbb{R}$	set of real numbers
$\mathbb{C}$	set of complex numbers
$\mathbb{F}_{p^m}$	finite field of size p^m, where p is prime and $m \in \mathbb{N}$

Lattice Theory

Λ	lattice
$Q(x)$	nearest neighbor quantizer with respect to the Euclidean norm
$\mathcal{V}_0, \mathcal{V}$	fundamental Voronoi region, Voronoi region
$\mathrm{Vol}\,(\Lambda)$	volume of a lattice Λ
$\Delta(\Lambda)$	density of a lattice Λ
$G(R)$	normalized second moment of a region R
$\gamma_S(R)$	shaping gain of a region R
$\gamma_C(\Lambda)$	coding gain of a lattice Λ
$\mathcal{L}$	nested lattice code

Information Theory

$H(X)$	entropy of a discrete random variable X
$H(X \mid Z)$	conditional entropy of a discrete random variable X when Z is known
$h(X)$	entropy of a continuous random variable X
$h(X \mid Z)$	conditional entropy of a continuous random variable X when Z is known
$I(X;Y)$	mutual information between a pair of random variables
$I(X;Y \mid Z)$	conditional mutual information between a pair of random variables X and Y when Z is known
$\delta(n)$	a function of n that goes to zero when n goes to infinity

Part I

Preliminaries

Chapter 1

Introduction

1.1 Motivation

Wireless communication is a very important topic today. Mobile telecommunication, Internet access with mobile phones, car-to-car communication, device-to-device communication, sensor networks, etc. are just a few examples where wireless communication is used. Although each application has its own demands and specifications, they have one thing in common: data exchange in a wireless network. Up to now, the designers of such communication networks have focused on each hop in the network separately. Consequently, it could be a point-to-point channel with one transmitter and one receiver, a multiple-access channel (MAC) with several transmitters and one receiver, or a broadcast channel (BC) with one transmitter and several receivers. In those models, interference was an unwanted enemy, and the communication protocols were designed to avoid interference. Techniques such as time-division multiple access (TDMA), frequency-division multiple access (FDMA) or code-division multiple access (CDMA) are used to achieve this. A couple of years ago, the way of thinking about interference changed with the introduction of network coding [1] and physical-layer network coding (PLNC) [49]. For wired networks, we can see a paradigm shift from packet switched networks, where each data packet is routed individually through the network, to code-centric networks, where data packets are combined by network coding, while transported through the network. This resolves bottlenecks on frequently used routes. Network coding has been introduced for multicast networks [1], but is no longer restricted to that [11]. With the introduction of random network coding [8], network coding has been spread to a large number of research fields. But network coding is not limited to wired networks. The concept can be applied to wireless networks, where interference is no longer a disadvantage but can be exploited together with multipath data distribution. With the help of PLNC approaches, it is possible to significantly increase the achievable throughput in a network. An example for the achievable sum-rate with different relaying strategies in a two-way relay channel is shown in Fig. 1.1. The mathematical background can be found in Appendix B. For the special parameter set we used, we can see that the sum-rate

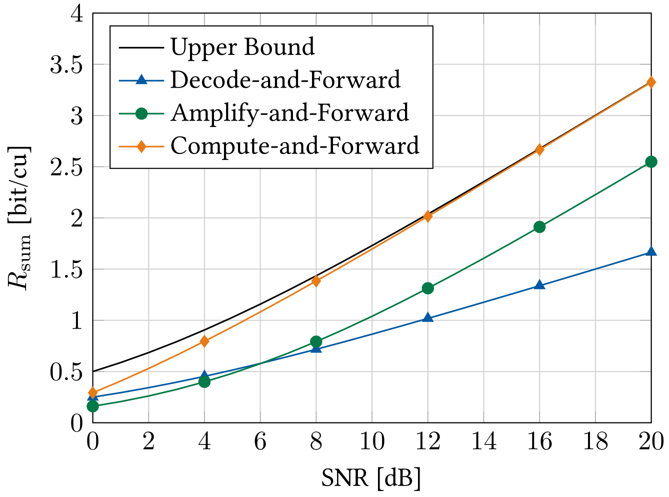


Figure 1.1: Achievable sum-rate in a two-way relay channel with unit channel coefficients (details see Appendix B).

with the compute-and-forward relaying strategy (as a special case of PLNC) converges to the upper bound in the high-signal-to-noise ratio (SNR) regime. This bound is the sum of the single-user rates that are achievable if the users transmit without interference in different time slots or frequency bands. With compute-and-forward, both users can simultaneously transmit at this rate. In general, the actual gain of PLNC depends on the system parameters and the channel properties. Therefore, PLNC approaches have to be designed properly. Within this thesis, we tackle some of those design criteria in Part II.

Network coding not only results in a possible increase in throughput, it can also be advantageous for the confidentiality of the system. Since intermediate nodes in a network do not need to decode each data packet individually, they only receive a superposition of data packets, from which they might not be able to calculate the original data. Of course, it needs some additional effort to ensure that intermediate nodes will definitely not be able to get the original data. But network coding and especially PLNC provide a kind of confidentiality that would not be possible without it. One example is the communication in a wireless network where the intermediate nodes are not trustworthy. We assume that the intermediate nodes are honest but curious, which means that they comply to the communication protocol but try to eavesdrop the communication. With PLNC, it is possible to use an untrusted node as a relay and still ensure that this node will not get any of the original data. This would not be possible without PLNC. We introduce this technique in Part III.

1.2 State of the Art

Network coding has been a promising topic in communications since it was introduced by Ahlswede et al. in [1]. It was shown that network coding can improve the throughput of a network and achieves the multicast capacity. Further, the authors of [10] proved that linear codes are sufficient. For static wired networks, network coding is very well investigated, and some frameworks are developed that provide tools to design a network code [5, 9, 17–19]. The development of random linear network coding in [8] made it more applicable in practical setups, and first implementations for industrial usage are emerging [4, 13]. Some examples are shown in Fig. 1.2 including the improvement of the transmission control protocol (TCP) [15], device-to-device communication [7], 5G [6, 16] and cloud computing [14].

Wireless networks are still subject of intensive research. The properties of the wireless channel allow network coding on different layers. Practical network coding on the forwarding layer has been proposed in [38]. The superposition property of the wireless channel also allows network coding on the physical layer, where the actual network coding occurs in the channel. This is called PLNC. Since the introduction of PLNC [49], different schemes have been developed and presented, e.g., in [36, 37, 41, 48].

A framework that has drawn a lot of attention is compute-and-forward (CF) [40]. It is based on structured codes like lattice codes that achieve the additive white Gaussian noise channel capacity with lattice decoding instead of maximum-likelihood decoding [24]. CF uses lattice codes and exploits their structure to allow the decoding of a superposition of codewords without decoding the codewords themselves. This enables the relaying nodes to decode the superposition at higher rates than it would be possible by decoding the individual messages as it is done by other approaches like decode-and-forward. A further important advantage compared to approaches like amplify-and-forward is the possibility to avoid noise accumulation, which reduces the performance in large networks. Although PLNC has a lot of advantages, and high rates are

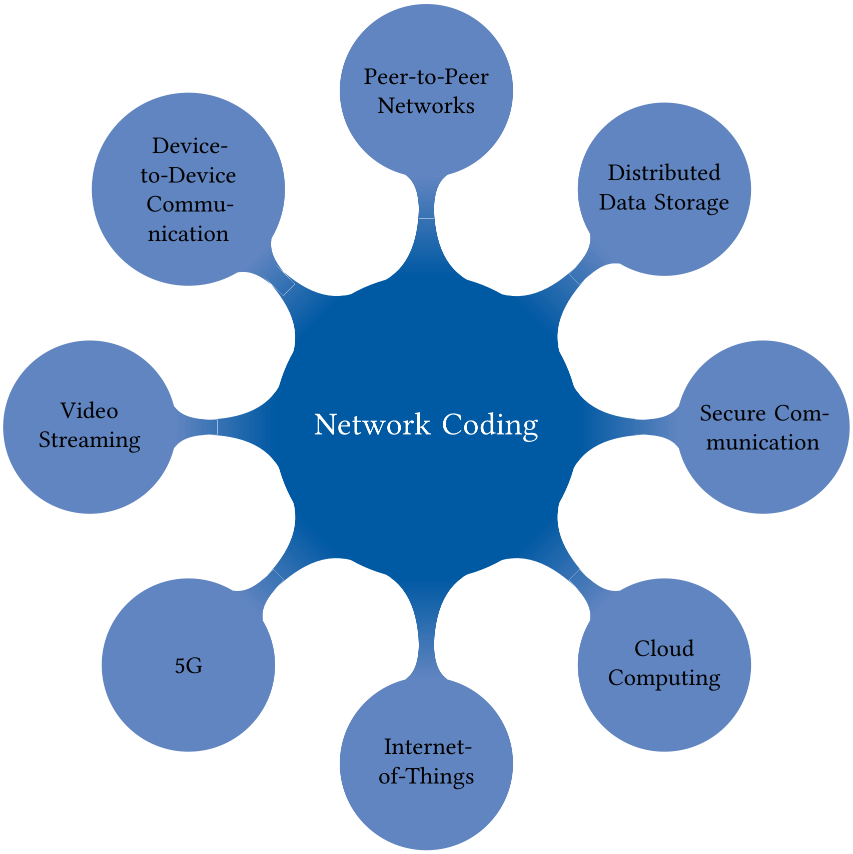


Figure 1.2: Network coding applications.

achievable, it comes with the need for network diversity. The final destination needs to collect enough independent observations to be able to jointly decode the transmitted data.

Over the past years, there has been some effort to apply PLNC in a practical environment. First attempts have been presented in the context of analog network coding [84–86, 90, 96, 102]. A first design and implementation approach for PLNC was proposed in [95]. For the two-way relay channel, an analysis of the performance of practical PLNC was published in [92]. The authors of [93, 94] went even further and showed a real-time implementation of PLNC. A little later, a practical implementation of PLNC for cloud computing was proposed in [89].

After the introduction of the compute-and-forward framework, there has been some effort to make it applicable to practical systems [97]. Since the compute-and-forward framework is based on lattice codes, it is essential that the corresponding coding schemes can be efficiently implemented. Over the last decades, there has been some effort to achieve this goal. In [31], the authors designed and implemented lattice coding schemes using digital signal processing techniques. Ten years later, the leech lattice was combined with the IEEE 802.11a WLAN standard [33]. Over the last decade, there has been some research to apply known techniques from channel codes to lattice codes, and low-density lattice codes [32] as well as polar lattices [27] have been proposed. Lattice codes are no longer just a theoretical topic. This is shown by ongoing studies to bring lattice codes to the future 5G standard [26].

Beside reliable communication, secrecy demands have become more and more important over recent years. Nowadays, the world is connected over networks, and communication can be easily overheard. Therefore, protecting the confidentiality of communication in networks is an important topic. In cryptographically protected systems, the legitimate users have a secret key and are able to encode and decode the secured messages in an easy manner. In practical systems, the confidentiality is based on the time and complexity needed by an adversary, who does not possess the secret key, to decrypt the messages. If the time and complexity at the eavesdropper is sufficiently high in terms of currently available computational power, the message is assumed to be secure. This kind of security will become weaker with increasing computational power in the years to come. Security which uses means of information-theoretic secrecy on the physical layer offers the possibility that an eavesdropper cannot get any information about the exchanged messages. Shannon introduced the notion of perfect secrecy in his seminal paper [72] for the case in which an eavesdropper has direct access to the sent codeword. Almost thirty years later, the theoretical basis for the ongoing research on information-theoretic security in wireless communication systems was introduced first by Wyner [77] and then by Csiszár and Körner [55], who proved in two seminal papers that there exist codes for noisy communication channels that guarantee both reliability and a prescribed degree of data confidentiality. The extension to continuous input and output alphabets was developed in [62]. It took more than twenty years before transceiver structures were available to support these wiretap setups. Extensive analysis and designs have been conducted, parts of their results are reported in [52, 63, 66] and recent tutorial papers [71, 73].

It has early been observed that decoding a linear combination of source messages at one network node does not automatically allow this node to decode each message individually [67]. This makes network coding interesting for communication via nodes that are not trustworthy. The achievable secrecy rate in the presence of an untrusted relay was studied in several papers with slightly different scenarios. An early work considering a relay channel with a direct connection between source and destination, where the relay is also the wiretapper, was introduced in [69]. In [61], Huang et al. investigated different secure transmission schemes in a similar scenario. The question whether an untrusted relay is helpful if a direct connection between source and destination exists was investigated in [57]. In [59], a Gaussian two-hop network was considered where source and destination do not have a direct connection. The destination node can help the source node by jamming the relay node with a random signal. This model was extended to a multi-hop line network in [58].

The two-way wiretap channel, in which two nodes can only exchange messages via an untrusted relay, was first considered in [74, 75]. It was shown that cooperative jamming, i.e., jamming with controlled interference between codewords, could reduce the eavesdropper's SNR and hence improve the level of secrecy. In [70], it was shown that this result also holds for a stronger notion of secrecy.

A lot of secrecy schemes are based on random codes, but there is also research in the field of secrecy that can be achieved by structured codes, namely lattice codes. Achievable rates for the lattice coded Gaussian wiretap channel were derived in [54]. Theorem 1 in [54] proposes a lattice code construction that achieves the weak secrecy capacity. Lattice codes for the Gaussian wiretap channel were intensively studied by Oggier et al. in [68] and references therein. They introduced the secrecy gain as a design criterion for good lattice codes for wiretap channels in [51]. It was shown by Ling et al. that lattice codes can

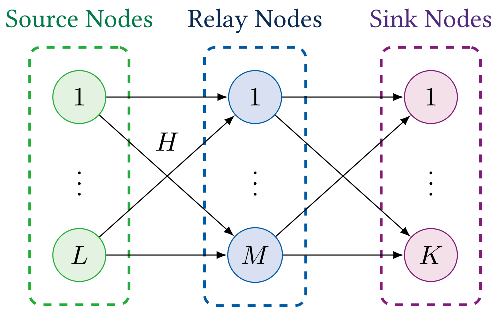


Figure 1.3: System model of a $L \times M \times K$ relay network.

achieve strong secrecy over the mod-Λ Gaussian channel [64]. In [65], Ling et al. introduced the flatness factor [21] as the main tool to prove that nested lattice codes can achieve semantic security and strong secrecy over the Gaussian wiretap channel. Compute-and-forward network coding together with strong physical-layer security based on universal hash functions were investigated in [59]. All these authors focused either on a wiretap channel or on a two-hop relay network, where a source node transmits via an untrusted relay to a destination. The destination may help by jamming but does not transmit a secure message itself. In [76], Vatedka et al. considered a two-way relay network with an untrusted relay where two nodes simultaneously transmit one message each via an untrusted relay. They provided an achievable power-rate region with perfect secrecy as well as strong secrecy by applying the compute-and-forward strategy at the relay.

1.3 System Model

The most general system model that is investigated in this thesis is the $L \times M \times K$ relay network consisting of L source nodes, M relay nodes, and K sink nodes as depicted in Fig. 1.3. Each sink node wants to get an estimate of the transmitted messages from all source nodes. We call this a multicast network. The first hop from the source nodes to the relays is of special importance because this is the part where the PLNC takes place. This hop can be characterized by the following channel model.

1.1 Definition (Channel Model). Each relay receives a noisy linear combination $y_m \in \mathbb{R}^n$ with $m \in \{1, 2, \dots, M\}$ of the signals $x_\ell \in \mathbb{R}^n$ with $\ell \in \{1, 2, \dots, L\}$ transmitted over the wireless channel, i.e.,

$$y_m = \sum_{\ell=1}^{L} h_{m\ell} x_\ell + z_m, \tag{1.1}$$

where $h_{m\ell} \in \mathbb{R}$ are the channel coefficients, and $z_m \in \mathbb{R}^n$ is white Gaussian noise with $z_m \sim \mathcal{N}(0, I_n)$. Let $h_m = (h_{m1}, \dots, h_{mL})'$ denote the vector of channel coefficients to relay m, and let $H = (h_{m\ell})$ with $m \in \{1, 2, \dots, M\}$ and $\ell \in \{1, 2, \dots, L\}$ denote the entire channel matrix. With this notation, the m-th row in H is h'_m. ◁

Each source node ℓ chooses a length-k_ℓ message vector $w_\ell \in \mathbb{F}_p^{k_\ell}$ from a uniform distribution over the index set $\{1, 2, \dots, 2^{\lfloor nR_\ell \rfloor}\}$, where R_ℓ is the message rate of source node ℓ and given by

$$R_\ell = \frac{k_\ell}{n} \log_2 p \quad \text{[bit/cu]}. \tag{1.2}$$

Usually, all messages from all L source nodes are zero-padded to a common length $k \triangleq \max_\ell k_\ell$. Therefore, we will drop the index ℓ in the following and assume length-k messages at all source nodes.

In order to ensure reliable communication the message is encoded by an encoder $\mathcal{E}_\ell$.

1.2 Definition (Encoders). Each source node is equipped with an encoder

$$\mathcal{E}_\ell : \mathbb{F}_p^k \to \mathbb{R}^n \tag{1.3}$$

that maps length-k messages w_ℓ in the finite field $\mathbb{F}_p^k$ to length-n real-valued codewords x_ℓ, i.e., $x_\ell = \mathcal{E}_\ell(w_\ell)$. Each codeword is subject to the power constraint $\|x_\ell\|^2 \leq nP$. ◁

Since there are no direct links between the source nodes and the sink nodes, the communication is supported by several relays. The relays can apply several relaying strategies, which are common to all relays and a system design parameter.

Throughout the thesis, we will investigate parts or special cases of the general model in Fig. 1.3.

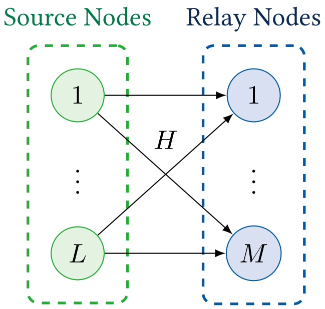


Figure 1.4: System model of an interference channel.

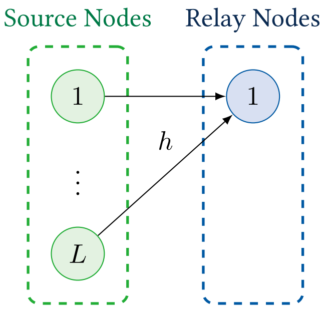


Figure 1.5: System model of a multiple-access channel.

- *Interference Channel (Fig. 1.4):* This model is the main building block for the application of PLNC and compute-and-forward. Therefore, it is used in Chapter 3 to introduce the compute-and-forward framework.
- *Multiple-Access Channel (Fig. 1.5):* This well-known model can be obtained by ignoring the last hop and concentrating on one relay node. It is used in Chapter 5 to solve the optimization problem of finding the best network coding coefficient vector for a certain channel realization.
- *Multi-Way Relay Channel (Fig. 1.6):* This model can be obtained by using only one relay. Further, the source and sink nodes are physically identical and therefore have side-information about their sent messages. This model is used in Chapter 8 to investigate the communication via an untrusted relay. The special case with only two source and sink nodes is called *Two-Way Relay Channel* and is used in Chapter 7 to provide an example for the more general case in Chapter 8.

1.4 Overview and Contribution

This thesis provides contributions to the field of PLNC, especially to the compute-and-forward protocol. The preliminaries on network coding are closely related to the lecture series “Netzwerkkodierungstheorie” (Network Coding Theory) from the “Professur für Theoretische Nachrichtentechnik” (Communications Theory Group), and for which the author of this thesis developed and

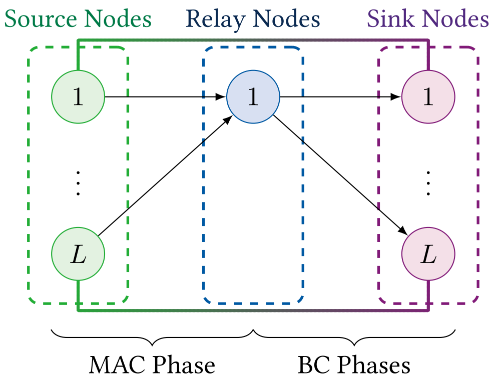


Figure 1.6: System model of a multi-way relay channel.

presented the corresponding exercises, problems, and solutions. This lecture series also comprises an introduction to compute-and-forward. Those lectures were prepared and taught by the author of this thesis.

Within this thesis, several techniques are applied and combined to enhance PLNC as shown in Fig. 1.7. We solve and discuss some open problems for the multi-hop relay network in combination with compute-and-forward in Part II. This includes solving the optimization problem of finding the optimal network coding coefficient vector. We show that the local optimization at the network nodes can lead to significant outage in the network, especially when the channels are correlated. We provide a solution to this problem without global optimization.

In Part III, physical-layer secrecy is additionally considered to provide secure communication over untrusted relay channels. We show that PLNC is the key technology to enable a secure data transmission via untrusted nodes and provide an achievable secrecy rate region. We see that adding multiple antennas to the source nodes can significantly increase the secrecy rate.

In order to show that compute-and-forward is not only a theoretical framework, an implementation with software-defined radio (SDR) is described in Part IV.

Chapter 1 In this chapter, we introduce the motivation for this work and provide a summary of the state of the art. Further, we introduce the system model and summarize the content of this thesis.

Chapter 2 In this chapter, we provide a short introduction to classical network coding. A more detailed description can be found in:

C. Fragouli and E. Soljanin. *Network Coding Fundamentals.* Now Publishers, 2007

Chapter 3 In this chapter, we introduce the compute-and-forward framework and provide the basic definitions and notation. Detailed explanations can be found in the journal paper by Nazer and Gastpar:

B. Nazer and M. Gastpar. "Compute-and-Forward: Harnessing Interference Through Structured Codes". In: *IEEE Transactions on Information Theory* 57.10 (Oct. 2011), pp. 6463–6486

Chapter 4 In this chapter, we introduce the concept of physical-layer secrecy and provide the definitions used in later chapters. The definitions are taken from:

M. R. Bloch and J. Barros. *Physical-Layer Security. From Information Theory to Security Engineering.* Cambridge University Press, 2011

Chapter 5 In this chapter, we solve the problem of finding the optimal network coding coefficient vector for the compute-and-forward relaying strategy. We solve a convex quadratic integer programming (CQIP) problem and provide a branch-and-bound algorithm. This chapter is based on the following paper:

J. Richter, C. Scheunert, and E. A. Jorswieck. "An efficient branch-and-bound algorithm for compute-and-forward". In: *23rd IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC).* Second Workshop on Network Coding in Wireless Relay Networks. Sept. 2012, pp. 77–82

Chapter 6 In this chapter, we discuss the problem of linear dependent superpositions in a multi-hop network. We show that channel correlation is a crucial problem and provide an alternative compute-and-forward strategy to overcome this problem. This chapter is based on the following paper:

J. Richter, J. Hejtmánek, E. A. Jorswieck, and J. Sýkora. “Non-Cooperative Compute-and-Forward Strategies in Gaussian Multi-Source Multi-Relay Networks”. In: *IEEE 82nd Vehicular Technology Conference (VTC Fall)*. Sept. 2015, pp. 1–5

Chapter 7 In this chapter, we combine the compute-and-forward framework with physical-layer secrecy to provide a secure communication over a two-way untrusted relay channel. This chapter is based on the following paper:

J. Richter, C. Scheunert, S. Engelmann, and E. A. Jorswieck. “Secrecy in the Two-Way Untrusted Relay Channel with Compute-and-Forward”. In: *International Conference on Communications (ICC)*. June 2015, pp. 4357–4362

Chapter 8 In this chapter, we extend the setup described in the previous chapter by introducing multiple users and multiple antennas per user. It is based on the following paper:

J. Richter, C. Scheunert, S. Engelmann, and E. A. Jorswieck. “Weak Secrecy in the Multiway Untrusted Relay Channel with Compute-and-Forward”. In: *IEEE Transactions on Information Forensics and Security* 10.6 (June 2015), pp. 1262–1273

Chapter 9 In this chapter, we describe an implementation of the compute-and-forward framework with SDR.

The following publications provide also a contribution to the field of physical-layer network coding. However, they have not been included in this thesis.

[110] In this paper, we make a transition from classical wired network coding to wireless network coding. We discuss and describe achievable rate regions in multicast networks.

J. Richter, A. Wolf, and E. A. Jorswieck. “Achievable rate regions in multiple-antenna networks with linear network coding”. In: *12th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. 2011, pp. 541–545

[103] In this paper, we deal with the optimization of the beamforming vectors at the relays in a multi-hop multi-antenna X-channel, where the compute-and-forward framework is utilized. The optimization problem is solved and the resulting rate region is characterized.

E. A. Jorswieck and J. Richter. "Compute-and-Forward in the Two-Hop Multi-Antenna X-Channel". In: *5th International Symposium on Communications Control and Signal Processing (ISCCSP)*. May 2012, pp. 1–4

[105] + **[104]** In these two papers, we compare and discuss physical-layer secrecy and packet-layer security in combination with network coding.

J. Richter, E. Franz, S. Engelmann, S. Pfennig, and E. A. Jorswieck. "Physical layer security vs. network layer secrecy: Who wins on the untrusted two-way relay channel?" In: *18th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. Sept. 2013, pp. 164–168

S. Pfennig, E. Franz, J. Richter, C. Scheunert, and E. A. Jorswieck. "Confidential Network Coding: Physical Layer vs. Network Layer". In: *IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB)*. Oct. 2015, pp. 1–5

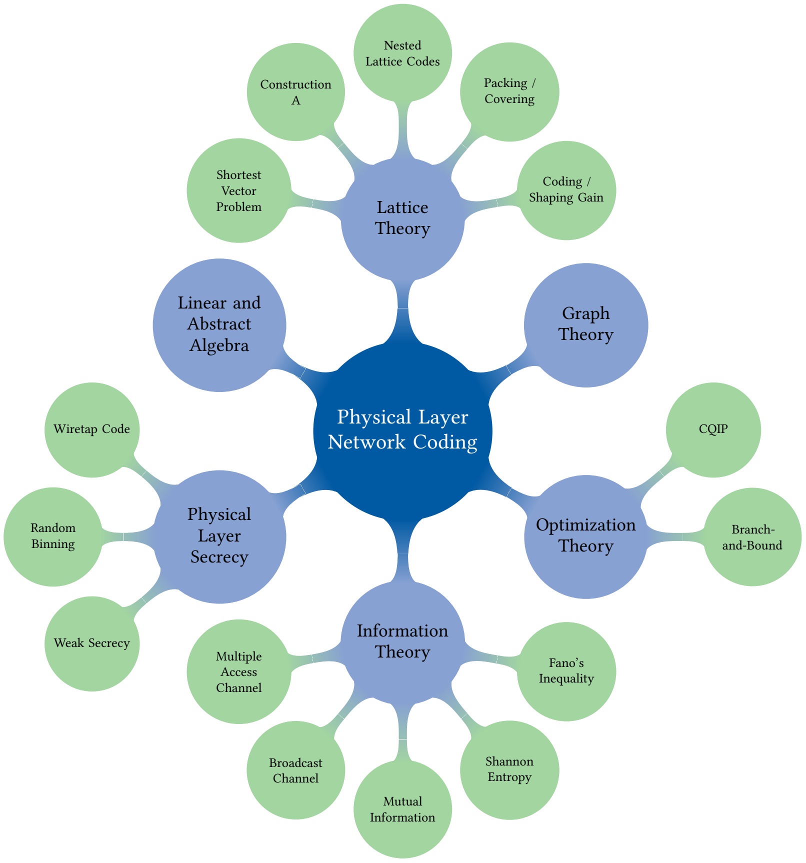


Figure 1.7: Tools and techniques applied in this thesis.

Network Coding

Network coding has been a promising topic in communications since it was introduced by Ahlswede et al. [1]. It was shown that network coding can improve the throughput of a network and achieves the multicast capacity. For static wired networks, network coding is very well investigated, and some frameworks are developed that provide tools to design a network code [5, 9, 17–19]. The development of random linear network coding [8] made it more applicable in practical setups, and first implementations for industrial usage are emerging [4, 13].

Nevertheless, there is still ongoing research in the field of network coding for wireless networks. The properties of wireless channels provide the possibility for network coding on different layers. Practical network coding on the forwarding layer has been proposed in [38]. The superposition property of the wireless channel also allows network coding on the physical layer [41], where the actual network coding is provided by the channel.

In this chapter, we introduce the basic theory of network coding. The main theorem and the most important results for network coding on the packet layer can be found in Section 2.2. In Section 2.3, we discuss applications and extensions of network coding. Since the theory of network coding relies on graph theory, we introduce the most important definitions in Section 2.1.

2.1 Graph Theory

2.1 Definition (Graph). A Graph is a tuple $G = (V, E)$ with $E \subseteq V \times V$. The set V is a finite set of nodes, and the set E is a finite set of edges. ◁

2.2 Definition (Directed Graph). A directed graph $G(V, E)$ is a graph with directed edges, i.e., every edge $e \in E$ is an ordered tuple $e = (u, v)$ of nodes. ◁

2.3 Definition (Parallel Edges). Two edges (u, v) and (x, y) in a directed graph are called parallel if $x = u$ and $y = v$. ◁

2.4 Definition (Incoming Edges). Let $G(V, E)$ be a directed graph. The set of incoming edges of a node $v \in V$ is defined as $\text{In}(v) \triangleq \{(u, v) \in E, u \in V\}$. ◁

2.5 Definition (Outgoing Edges). Let $G(V, E)$ be a directed graph. The set of outgoing edges of a node $v \in V$ is defined as $\mathrm{Out}(v) \triangleq \{(v, u) \in E, u \in V\}$.$\triangleleft$

2.6 Definition (Path). Let $G = (V, E)$ be a directed Graph. A path is an alternating sequence $P = (v_0, e_1, v_1, \dots, e_k, v_k)$ of nodes and edges with $k > 0$ and $e_i = (v_{i-1}, v_i) \in E$. Every path starts and ends with a node. $\triangleleft$

2.7 Definition (Edge-disjoint Path). Let $P = (v_0, e_1, v_1, \dots, e_k, v_k)$ be a path. Then, P is called edge-disjoint if $e_i \neq e_j$ for $i \neq j$ and $0 \leq i, j \leq k$. $\triangleleft$

2.8 Definition (Node-disjoint Path). Let $P = (v_0, e_1, v_1, \dots, e_k, v_k)$ be a path. Then, P is called node-disjoint if $v_i \neq v_j$ for $i \neq j$ and $0 \leq i, j \leq k$. $\triangleleft$

2.9 Definition (Acyclic Graph). An acyclic graph $G(V, E)$ is a directed graph without cycles, i.e., there is no path $(v, \dots, v)$ for every node $v \in V$. $\triangleleft$

2.10 Definition (Flow). Let $G = (V, E)$ be a directed graph. Further, let $c(e)$ be a map $c : E \to \mathbb{R}$ that represents the capacity of the edge e. If $(u, v) \notin E$, then $c(u, v) = 0$. We select a node s as source and a node t as sink. A flow f on a graph G is a real function $f : V \times V \to \mathbb{R}$ with the following three properties for all nodes $u, v \in V$:

- *Capacity constraints:* $f(u, v) \leq c(u, v)$. The flow along an edge cannot exceed its capacity.
- *Skew symmetry:* $f(u, v) = -f(v, u)$. The flow from u to v must be the opposite of the flow from v to u.
- *Flow conservation:* $\sum_{w \in V} f(u, w) = 0$, unless $u = s$ or $u = t$. The flow to a node is zero, except for the source, which “produces” flow, and the sink, which “consumes” flow. $\triangleleft$

2.11 Definition (Cut). Let $G(V, E)$ be a directed graph. We choose two nodes $S, R \in V$. A cut between S and R is a subset of E, whose removal from E disconnects S from R. The value of a cut is the sum of the capacities of the edges in the cut. $\triangleleft$

2.12 Definition (Min-Cut). Let $G = (V, E)$ be a directed graph. A min-cut between two nodes $S, R \in V$ is a cut with the smallest value. $\triangleleft$

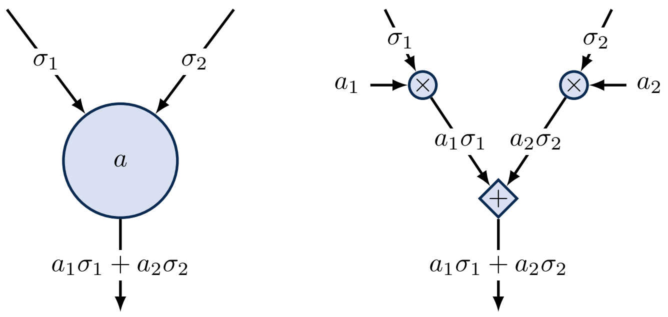


Figure 2.1: Linear network coding.

2.2 Network Coding Fundamentals

This section follows the presentation in [5]. When Ahlswede et al. introduced the concept of network coding in 2000, they assumed acyclic wired networks. Based on fundamental tools of graph theory and linear algebra, they developed a framework that allows to multicast packets from several source nodes to all receivers at a rate equal to what is possible if one sender is transmitting alone. The theory of network coding is based on the following theorem.

2.13 Theorem (Min-Cut Max-Flow Theorem). Consider a graph $G(V, E)$ with unit capacity edges, a source node S, and a receiver R. If the min-cut between S and R equals L, then the maximum flow from S to R is L. Equivalently, there exist exactly L edge-disjoint paths between S and R. $\triangleleft$

From that theorem, Ahlswede et al. derived the following theorem, which summarizes the main result of their paper.

2.14 Theorem (The Main Network Coding Theorem). Let $G = (V, E)$ be a directed acyclic graph with unit capacity edges, L unit rate sources located on the same vertex of the graph and K receivers. Assume that the value of the min-cut to each receiver is L. Then, there exists a multicast transmission scheme over a large enough finite field $\mathbb{F}_p$, in which intermediate network nodes linearly combine their incoming information symbols over $\mathbb{F}_p$, that delivers the information from the sources simultaneously to each receiver at a rate equal to L. $\triangleleft$

The main idea is that intermediate nodes in a network linearly combine their incoming packets and transmit that combination to the following nodes as illustrated in Fig. 2.1. Let $\mathrm{Out}(v) \subset E$ denote the set of outgoing edges of a node $v \in V$ and let $N = |\mathrm{In}(v)|$ represent the number of incoming edges of a node v. We associate a *local encoding vector* $a(e)$ with each edge $e \in \mathrm{Out}(v)$, i.e.,

$$a(e) = (a_1(e), \dots, a_N(e))' \tag{2.1}$$

with $a_i(e) \in \mathbb{F}_p$ for all $i \in \{1, 2, \dots, N\}$. The symbols forwarded to edge e are multiplied by this vector. The components $a_i(e)$ of each local encoding vector are called *network coding coefficients* and are collected in the set $\{a_k\}$. In large networks, this linear combination will be performed several times. The symbols $\sigma_1, \dots, \sigma_L$ will be multiplied by a *global encoding vector* $c(e) = (c_1(e), \dots, c_L(e))'$ on their way through the network. Namely, the symbol that is transported over some edge $e \in E$ in a graph $G(V, E)$ is given by

$$c'(e)\sigma = (c_1(e), \dots, c_L(e)) \begin{pmatrix} \sigma_1 \\ \vdots \\ \sigma_L \end{pmatrix} . \tag{2.2}$$

Therein, every component $c_i(e)$ with $i \in \{1, 2, \dots, L\}$ is a function of the network coding coefficients in the set $\{a_k\}$. For simplicity, we refer to the global encoding vector as the *coding vector* of an edge e, which is the vector with coefficients of the linear combination of source symbols on edge e.

Important is the flow through the last edge of a path to the receiver R_k. Assume ρ_ℓ^k is the symbol on the last edge of the path $(s_\ell, \dots, R_k)$ and $A_k \in \mathbb{F}_p^{L \times L}$ is a matrix with the coding vector of the last edge of the path $(s_\ell, \dots, R_k)$ in the ℓ-th row, i.e.,

$$A_k = \begin{pmatrix} c'(e_1) \\ \vdots \\ c'(e_L) \end{pmatrix} \tag{2.3}$$

with $e_1, \dots, e_L \in \mathrm{In}(R_k)$. Since the components of the coding vectors depend on the network coding coefficients $\{a_k\}$, the matrix A_k can be expressed in terms of the network coding coefficients. Then, the receiver R_k has to solve the following system of linear equations, i.e.,

$$\left(\rho_1^k, \dots, \rho_L^k\right)' = A_k \left(\sigma_1, \dots, \sigma_L\right)' \tag{2.4}$$

to get the source symbols $\sigma_1, \dots, \sigma_L$. In order to solve the system of linear equations uniquely, A_k has to be of full rank for all $1 \leq k \leq K$. The network code design problem is to select values for the coefficients $\{a_k\}$, such that all matrices A_k have full rank. Theorem 2.14 can be expressed in algebraic language as follows.

2.15 Theorem (Algebraic Statement). In linear network coding, there exist values in some large enough finite field $\mathbb{F}_p$ for the components $\{a_k\}$ of the local coding vectors such that all matrices A_k with $1 \leq k \leq K$, which define the information that the receivers observe, are full rank. ◁

The requirement that all matrices A_k with $1 \leq k \leq K$ have full rank is equivalent to the requirement that the product of the determinants of A_k is different from zero, i.e.,

$$f(\{a_k\}) \triangleq \prod_{k=1}^{K} \det(A_k) \neq 0. \tag{2.5}$$

The following lemma states that it is possible to find coefficients a_k over a large enough field that fulfill this requirement.

2.16 Lemma (Sparse Zeros Lemma). Let $f(a_1, \ldots, a_\eta)$ be a multivariate polynomial in variables $a_1, \ldots, a_\eta$ with maximum degree in each variable of at most d. Then, in every finite field $\mathbb{F}_p$ of size $p > d$, on which $f(a_1, \ldots, a_\eta)$ is different from zero, there exist values $q_1, \ldots, q_\eta$ such that $f(a_1 = q_1, \ldots, a_\eta = q_\eta) \neq 0$.◁

There are many assumptions in Theorem 2.14. Some of them are not restrictive and can be dropped. The following list provides an overview of the assumptions and whether they are restrictive or not.

- *Unit capacity edges — not restrictive.* We can relax this assumption to rational numbers and allow parallel edges.
- *Co-located sources — not restrictive.* We can add an artificial common source vertex to the graph and connect it to the L source vertices through unit capacity edges.
- *Directed graph — restrictive.* We can construct an example where the main theorem does not hold.
- *Zero delay — not restrictive.* From a theoretical point of view, delay introduces a relationship between network coding and convolutional codes.

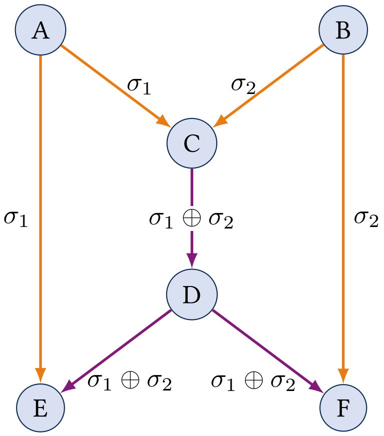

A
B
C
D
E
F
σ_1
σ_2
σ_1
$\sigma_1 \oplus \sigma_2$
σ_2
$\sigma_1 \oplus \sigma_2$
$\sigma_1 \oplus \sigma_2$

Figure 2.2: Network coding example: Butterfly network.

- *Acyclic graph — not restrictive.* There exist network configurations where we need to deal with the underlying cyclic graphs by taking special action, for example by introducing memory.
- *Same min-cut values — restrictive.* If the receivers have different min-cuts, we can multicast at the rate equal to the minimum of the min-cuts, but cannot always transmit to each receiver at the rate equal to its min-cut.

2.17 Remark. There is a connection between network codes and channel codes: The global encoding functions in a linear network are equal to the rows of the generator matrix of a linear channel code. However, every encoding function has to fulfill the law of information preservation (linear span). The choice of the generator matrix for a linear channel code is arbitrary. ◁

2.3 Network Coding Extensions

Inter-Flow vs. Intra-Flow

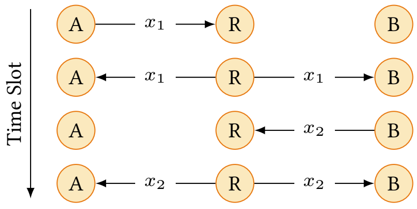
In the first view of network coding as introduced above, we talk about *inter-flow* network coding. This means, we have several source nodes that multicast their messages to several receivers. However, it is also possible to use network coding with a single source node. The packets of the source are encoded with each other and transmitted as a combination of a certain number of packets. This is called *intra-flow* or *intra-session* network coding. It was shown that this is especially useful in combination with the TCP because it significantly reduces the latency [15].

Network Coding for Wireless Networks

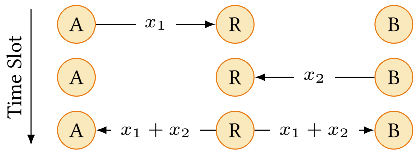
In a wireless setup, network coding can be applied on several levels in the protocol stack. Especially, it is possible to use the superposition property of the wireless channel to obtain a superposition of the transmitted signals. The benefit of exploiting this property is shown in Fig. 2.3 for the two-way relay channel. Because the "network coding" is done on the physical layer, we call those schemes that exploit this channel property *physical-layer network coding* schemes. Sometimes, one can find the alternative name *analog network coding* in literature. However, this usually refers to uncoded, analog signals and their superposition [90].

Distributed Data Storage

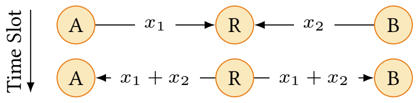
Network coding has been shown to be an enabling technique for distributed data storage systems (DDSS). The work by Dimakis et al. [2, 3] showed that there is a trade-off between storage and repair bandwidth. The authors introduced the concept of functional repair, which does not re-create the damaged data but creates a new node such that the data can still be retrieved from any k nodes. This is a known concept from network coding theory. An overview and a summary on DDSS can be found in [12].



(a) Time division.



(b) Traditional network coding.



(c) Physical-layer network coding.

Figure 2.3: Comparison of network coding schemes for the two-way relay channel.

Chapter 3

Compute-and-Forward

Physical-layer network coding has been an intensive field of research since the introduction by Zhang et al. [49]. One of the main challenges is to provide reliable communication over a noisy MAC. In 2011, Nazer and Gastpar showed several ways to obtain that in [41]. At the same time, they developed a framework in [40] that they called *compute-and-forward*. This framework uses well-known tools from lattice theory and linear algebra to transform noisy Gaussian channels into noiseless channels over finite fields (see Fig. 3.1). In this chapter, we provide an introduction to the compute-and-forward framework, where we follow the presentation in [40]. since lattice theory is the main tool and heavily used in the sequel of this thesis, we provide a short summary of the most important results in lattice theory in Appendix A.

3.1 Introduction

The wireless transmission of data from source to destination usually needs the help of intermediate relays because of physical distance, interference or other obstacles that reduce the signal quality. Whereas the relaying for one source - destination pair is rather easy, it gets more complicated if there are several source nodes and destination nodes involved, especially in a multicast setup. From classical network coding (Chapter 2), we already know that it is optimal to use network coding in a multicast network. In a wireless network, this leaves us with several options for the relaying scheme.

In a *decode-and-forward* system, the received data is fully decoded and re-encoded prior to re-transmission. This means that each users' data has to be decoded individually and then combined to form a superposition, which is then forwarded to the next node in the network. This forms the well-known model of the MAC and the achievable rate is limited by the MAC capacity region, which is also the main disadvantage of this strategy. The relay is usually not interested in the individual data. This scheme suffers from a loss in performance because of the individual decoding.

In an *amplify-and-forward* system, the relay just amplifies the received signal and forwards it to the next node in the network. While this is very simple

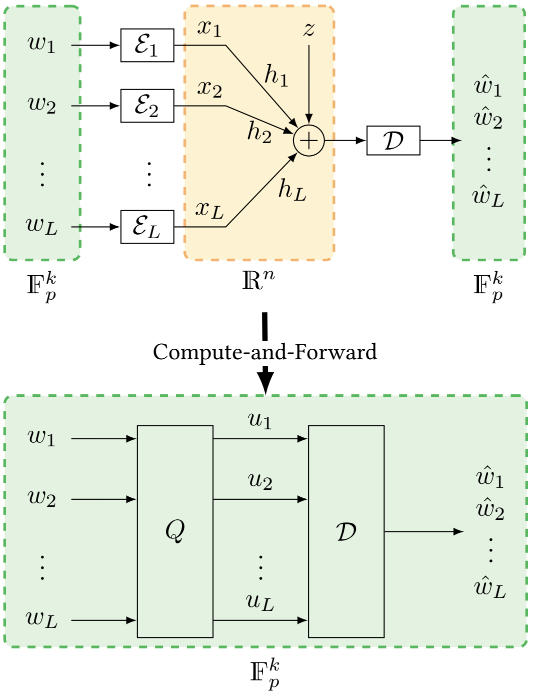


Figure 3.1: Compute-and-forward transforms noisy Gaussian channels into noiseless channels over a finite field.

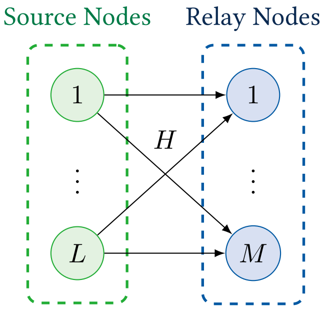


Figure 3.2: System model of the first hop of a relay network.

and needs no decoding and encoding, the relay not only amplifies the users' signals but also noise and interference. Especially in large networks, this can be a problem because noise and interference are accumulated, which results in a bad signal-to-interference-plus-noise ratio (SINR) at the receivers.

In order to overcome the drawbacks of decode-and-forward as well as amplify-and-forward, a new scheme was proposed in [40], called *compute-and-forward*. It uses lattice codes (nested lattice codes to be precise) that have a certain structure. This allows the decoding of a superposition of codewords without the need to decode the individual codewords, and a gain can be seen in the achievable rate. Therefore, it combines the advantages of decode-and-forward and amplify-and-forward without their drawbacks.

3.2 Encoding / Decoding

Compute-and-forward is a relaying scheme that allows to decode linear combinations of messages without the need to decode the individual messages. The main advantage is the gain in achievable rate without noise accumulation. The system model that is investigated is depicted in Fig. 3.2. Let the channel model be as defined in Definition 1.1.

A lattice with generator matrix B and Voronoi region $\mathcal{V}_C$ is denoted by Λ_C. Further, we define a lattice Λ_F with generator matrix G and Voronoi region $\mathcal{V}_F$. The lattice Λ_F is constructed such that $\Lambda_C \subseteq \Lambda_F$. This can be accomplished for example by construction A as explained in detail in Appendix A.3.1. We call Λ_C the coarse lattice and Λ_F the fine lattice. Based on the definitions in

Appendix A.3, we define the nested lattice code $\mathcal{L} = \Lambda_F \cap \mathcal{V}_C$. Each source node encodes its messages by an encoder $\mathcal{E}_\ell$ that is defined as follows.

3.1 Definition (Encoders). Each source node is equipped with an encoder,

$$\mathcal{E}_\ell : \mathbb{F}_p^k \to \Lambda_F \cap \mathcal{V}_C, \tag{3.1}$$

that maps length-k messages over the finite field to length-n real-valued codewords of a lattice code $\mathcal{L} = \Lambda_F \cap \mathcal{V}_C$, i.e., $x_\ell = \mathcal{E}_\ell(w_\ell)$. Each codeword is subject to the power constraint $\|x_\ell\|^2 \leq nP$. Therefore, Λ_C is chosen such that the second moment of Λ_C equals P. $\triangleleft$

3.2 Remark. We assume that each source node uses the same nested lattice code. However, this is not restrictive. Without loss of generality, we can assume the message lengths are sorted in descending order, i.e., $k_1 \geq k_2 \geq \cdots \geq k_L$. Thus, we can build a nested lattice code chain according to the different message lengths, i.e., $\Lambda_C \subseteq \Lambda_L \subseteq \cdots \subseteq \Lambda_1$. Using the nested lattice code $\Lambda_\ell = \Lambda_\ell \cap \mathcal{V}_C$ with the non-padded message is the same as using the nested lattice code $\Lambda_1 = \Lambda_1 \cap \mathcal{V}_C$ with the zero-padded message. Therefore, we call the finest lattice in the lattice chain fine lattice Λ_F and assume that all source nodes use the resulting nested lattice code. $\triangleleft$

3.3 Definition (Desired Equations). The goal of each relay is to reliably recover a linear combination of the messages, i.e.,

$$u_m = \bigoplus_{\ell=1}^{L} q_{m\ell} w_\ell, \tag{3.2}$$

where $q_{m\ell}$ are coefficients taking values in $\mathbb{F}_p$. $\triangleleft$

3.4 Definition (Lattice Equation). A lattice equation v is an integral combination of lattice codewords $t_\ell \in \mathcal{L}$ modulo the coarse lattice, i.e.,

$$v = \left[\sum_{\ell=1}^{L} a_\ell t_\ell\right] \bmod \Lambda_C \tag{3.3}$$

for some coefficients $a_\ell \in \mathbb{Z}$. We call $a = (a_1, \ldots, a_L)'$ the lattice coefficient vector. $\triangleleft$

Let the function g denote the map between the prime-sized finite-field $\mathbb{F}_p$ and the corresponding subset of the integers $\{0, 1, \dots, p-1\}$. This is essentially an identity map except for a change of alphabet. The inverse function is denoted by g^{-1}. If g or its inverse are used on a vector or matrix, the operation is taken element-wise.

3.5 Lemma (Encoder Function). Let w_ℓ be a message of length k_ℓ that is zero-padded to length k. Each symbol of w_ℓ is taken from $\mathbb{F}_p$ and therefore $w_\ell \in \mathbb{F}_p^k$. The function

$$\Phi(w_\ell) = \big[Bp^{-1}g(Gw_\ell)\big] \bmod \Lambda_C \tag{3.4}$$

is a one-to-one map between the set of such messages and the elements of the nested lattice code $\mathcal{L} = \Lambda_F \cap \mathcal{V}_C$. ◁

Further, the coefficients of the desired equation and the lattice equation are linked by the following equation.

$$q_{m\ell} = g^{-1}([a_{m\ell}] \bmod p). \tag{3.5}$$

3.6 Definition (Decoder). Each relay is equipped with a decoder $\mathcal{D}_m : \mathbb{R}^n \to \mathbb{F}_p^k$ that maps the received channel output y_m to an estimate $\hat{u}_m = \mathcal{D}_m(y_m)$ of the equation u_m. ◁

3.7 Lemma (Decoder Function). Let $u = \bigoplus_\ell q_\ell w_\ell$ be the desired equation for some coefficients $q_\ell \in \mathbb{F}_p$ and messages $w_\ell \in \mathbb{F}_p^{k_\ell}$ zero-padded to length k. Assume the messages are mapped to nested lattice codewords $t_\ell = \Phi(w_\ell)$, and let $v = [\sum_\ell a_\ell t_\ell] \bmod \Lambda_C$ denote the lattice equation for some $a_\ell \in \mathbb{Z}$ such that $q_\ell = g^{-1}([a_\ell] \bmod p)$. Then, the desired equation can be obtained using $u = \Phi^{-1}(v)$, where

$$\Phi^{-1}(v) = (G'G)^{-1}G' \cdot g^{-1}(p[B^{-1}v] \bmod \mathbb{Z}^n). \tag{3.6}$$

◁

The complete encoding and decoding operations are depicted in Fig. 3.3. Each encoder $\mathcal{E}_\ell$ uses the encoder function in Eq. (3.4) to obtain a lattice point t_ℓ in the lattice code. Further, it dithers its lattice point with a dither vector d_ℓ, which is generated independently according to a uniform distribution over the Voronoi region $\mathcal{V}_C$. All dither vectors are made available to each relay. After dithering, the encoder $\mathcal{E}_\ell$ takes the lattice point modulo Λ_C and transmits the resulting vector x_ℓ, i.e.,

$$x_\ell = [t_\ell - d_\ell] \bmod \Lambda_C. \tag{3.7}$$

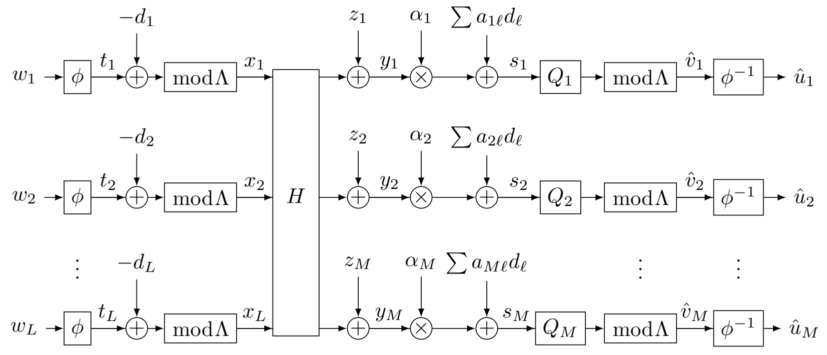


Figure 3.3: System diagram of the lattice encoding and decoding operations.

3.8 Remark (Dithering [40, Sec. V]). When a relay attempts to decode an integral combination of the lattice points, it faces two sources of noise. One is the channel noise z. The other comes from the codewords themselves because the channel coefficients are usually not exactly equal to the desired equation coefficients. This is sometimes referred to as "self-noise" or "quantization noise" (because the real-valued channel coefficients are quantized to integral values). In order to overcome this issue, the transmitters will dither their lattice points using common randomness that is also known to the relays. This dithering makes the transmitted codewords independent from the underlying lattice points. It can be shown that at least one set of good fixed dither vectors exists. This means that no common randomness is actually necessary. The proof can be found in [40, Appendix C] $\triangleleft$

Relay m receives a noisy superposition of the transmitted codewords, i.e.,

$$y_m = \sum_{\ell=1}^{L} h_{m\ell} x_\ell + z_m. \tag{3.8}$$

It scales the channel output with a scaling factor α_m and removes the dither, i.e.,

$$s_m = \alpha_m y_m + \sum_{\ell=1}^{L} a_{m\ell} d_\ell. \tag{3.9}$$

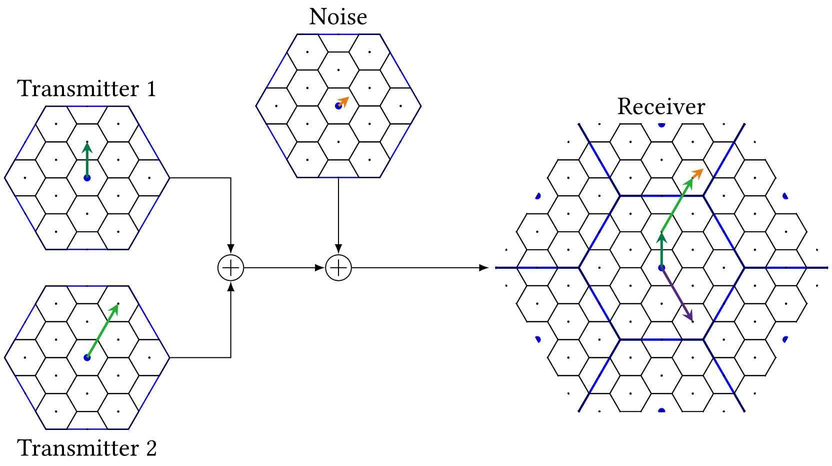


Figure 3.4: Compute-and-forward transmission over an AWGN channel without fading.

In order to get an estimate of the lattice equation v_m, the vector s_m is quantized onto the fine lattice Λ_F modulo the coarse lattice, i.e.,

$$\hat{v}_m = [Q_{\Lambda_F}(s_m)] \bmod \Lambda_C. \tag{3.10}$$

The relay uses the decoder function in Eq. (3.6) with the estimated lattice equation to get an estimate of the linear combination of the transmitted messages, i.e., the desired equation u_m.

An example of a lattice encoded transmission used by compute-and-forward is shown in Fig. 3.4 for an additive white Gaussian noise (AWGN) channel without fading. Although, the example simplifies the transmission and ignores the dithering, it shows the individual operations.

3.3 Achievable Rate Region

The following theorems are the main results of [40], and the proofs can be found therein. We assume that the noise power is normalized to one.

3.9 Theorem. For real-valued AWGN networks with channel coefficient vectors $h_m \in \mathbb{R}^L$ and equation vectors $a_m \in \mathbb{Z}^L$, the following computation rate region is achievable:

$$\mathcal{R}(h_m, a_m) = \max_{\alpha_m \in \mathbb{R}} \frac{1}{2} \log_2^+ \left(\frac{P}{\alpha_m^2 + P \|\alpha_m h_m - a_m\|^2} \right). \tag{3.11}$$

◁

From Eq. (3.11), one can already see that there are two noise terms: channel noise and quantization noise. The first one is introduced by the channel and the second one is due to the fact that we try to approximate the real-valued channel coefficients with integer values. For this approximation, we need a scaling factor α_m, which tries to steer the channel vector closer to a certain lattice coefficient vector. However, there is a trade-off because this scaling factor also amplifies the channel noise. The optimal α_m can be easily obtained by a derivation of Eq. (3.11), which provides the following theorem.

3.10 Theorem. The computation rate given in Eq. (3.11) is uniquely maximized by choosing α_m to be the minimum mean square error (MMSE) coefficient

$$\alpha_{\text{MMSE}} = \frac{P h_m' a_m}{1 + P \|h_m\|^2}, \tag{3.12}$$

which results in a computation rate region of

$$\mathcal{R}(h_m, a_m) = \frac{1}{2} \log_2^+ \left(\left(\|a_m\|^2 - \frac{P (h_m' a_m)^2}{1 + P \|h_m\|^2} \right)^{-1} \right). \tag{3.13}$$

◁

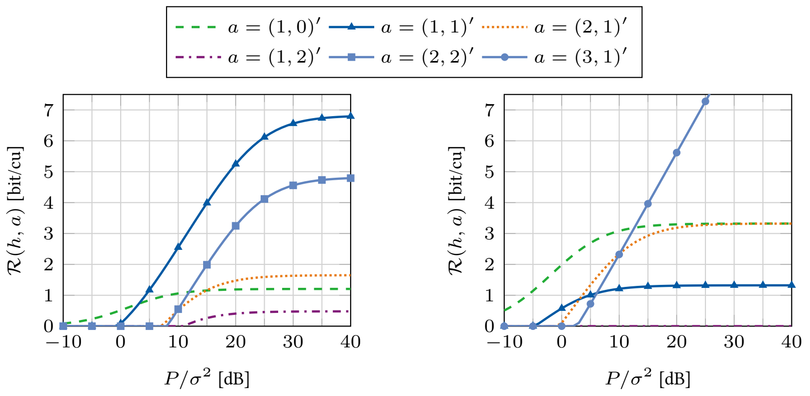
The relays can simultaneously decode linear combinations with lattice coefficient vector a_m as long as the involved messages' rates are within the computation rate region.

3.11 Definition (Computation Rate). We say that the computation rate region $\mathcal{R}(h_m, a_m)$ is achievable, if for any $\varepsilon > 0$ and n large enough, there exist encoders and decoders such that all relays can recover their desired equations with average probability of error ε as long as the underlying message rates R_ℓ satisfy

$$\forall \ell \in \{1, \ldots, L\} : R_\ell < \min_{m : a_{m\ell} \neq 0} \mathcal{R}(h_m, a_m). \tag{3.14}$$

◁

It is obvious that the achievable rate depends on the chosen lattice coefficient vector. However, this depends on the channel coefficients. The resulting optimization problem is not trivial and subject to ongoing research. An overview and an algorithm to solve this problem is presented in Chapter 5.



(a) Computation rate for channel $h = (0.8, 0.7)'$ and different lattice coefficient vectors.

(b) Computation rate for channel $h = (2.1, 0.7)'$ and different lattice coefficient vectors.

Figure 3.5: Computation rate for a two-user MAC with different channel vectors and lattice coefficient vectors.

While the typical optimization problem tries to find the optimal lattice coefficient vector depending on the channel coefficients, it is also worth looking at the achievable computation rate formula as a whole. As a starting point, one can rewrite Eq. (3.13) as

$$\mathcal{R}(h_m, a_m) = \frac{1}{2}\log_2^+\left(\frac{1 + P(h_m' a_m)^2}{\|a_m\|^2 + P\|h_m\|^2\|a_m\|^2 - P(h_m' a_m)^2}\right). \tag{3.15}$$

From Eq. (3.15), one can see that the computation rate reaches its maximum if

- $\|h_m\|^2\|a_m\|^2$ is equal to $(h_m' a_m)^2$, which is achieved if $h_m = a_m$, and
- $\|a_m\|$ is as small as possible, i.e., $a_m = (1, 1, \ldots, 1)'$, assuming that all messages must be included in the decoded linear combination.

We can already conclude that the highest computation rates are achieved if the channel coefficients are equal and close to one.

Further, we plot the computation rate for two examples of channel vectors and some lattice coefficient vectors in Fig. 3.5. For low SNR, we can see that it is best to decode only one message, i.e., for example $a = (1, 0)'$. When the

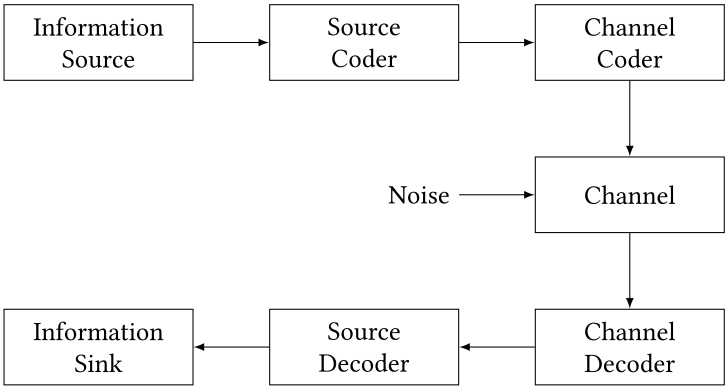


Figure 3.6: Block diagram of data transmission.

channel coefficients are close to each other, the optimal coefficient vector is $a = (1, 1)'$ as we have already seen from Eq. (3.15).

We see a very interesting example in Fig. 3.5b. In this case, the channel coefficients are multiples of each other. This allows the compute-and-forward framework to exactly steer the channel coefficients to the coefficient vector $a = (3, 1)'$ by a scaling factor. This is only optimal for relatively high SNR because the scaling amplifies the noise, too. However, if the SNR is high enough, the channel noise does not play a crucial role, and quantization noise does not occur in this case. In every other case, the computation rate saturates at a certain point because of the quantization noise and the channel noise combined.

3.4 Lattice Points as Analog Signals

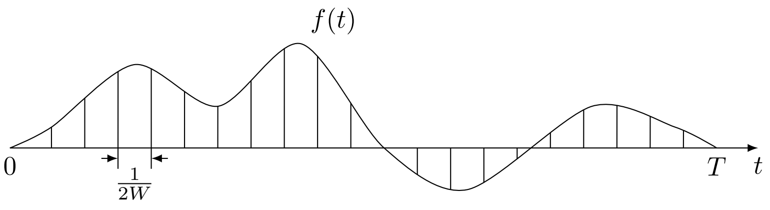
The following statements follow the description in [22, Chapter 3] and can be found in more detail therein. We consider the simple system model of a communication chain in Fig. 3.6. Messages are produced by an information source, e.g., a person, an orchestra or a computer. The source coder converts those messages into a digital form. In general, this means sampling and quantizing, which is known as analog-to-digital conversion. The source produces messages at a certain rate and wants to transmit those messages over a noisy channel. Although the channel is not error-free, the source wants to transmit its messages reliably and efficiently. In order to ensure this, we need a channel coder that exchanges the original messages from the source with signals that are easily distinguishable even with additional noise. The signals that have this property are called codewords of a code. Depending on the code and the amount of noise on the channel, the transmitted signals can be recovered error-free with a certain probability. Typically, one uses a digital model of the channel, which includes digital-to-analog conversion, modulation, etc., when designing a channel code. However, it is also possible to directly design the transmitted waveforms. This might not be as convenient as working in the digital domain, but the theory is similar.

As already mentioned, sampling plays an important role in the communication chain. An important result is the *Sampling Theorem*, which states that if $f(t)$ is a signal, i.e., a function of the time t and does not contain a frequency larger than W Hz, then f is completely specified by its samples

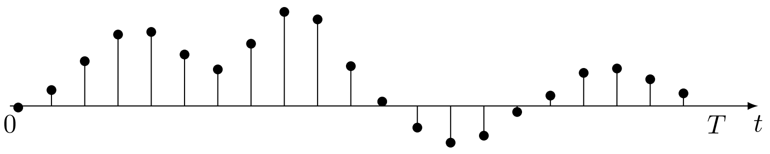
$$\dots, f\left(-\frac{1}{2W}\right), f(0), f\left(\frac{1}{2W}\right), f\left(\frac{2}{2W}\right), \dots, \tag{3.16}$$

which are produced every $1/(2W)$ seconds. We can group n such samples into a vector that is an element of the vector space $\mathbb{R}^n$. This means that we can describe the transmit signal by a sequence of vectors. In Fig. 3.7, there is an example of the different abstraction levels.

As stated above, it is crucial to design the transmit signal properly to cope with a noisy channel. Therefore, we will not use arbitrary signals but signals with certain properties. One way to do so is to apply lattice codes that use only a certain subset of $\mathbb{R}^n$ as transmit signals. Compute-and-forward exploits that and uses the mathematical notation of lattices to create a simpler Gaussian channel model.



(a) Analog signal.



(b) Samples.

$$\left(f(0), f\left(\frac{1}{2W}\right), f\left(\frac{2}{2W}\right), \ldots, f\left(\frac{n-1}{2W}\right)\right)$$

(c) Sampling vector.

(d) Sampling point.

Figure 3.7: Arbitrary analog signal with sampling intervals. Samples are taken every $1/(2W)$ seconds from a function f. Within T seconds, a number of $n = 2TW$ samples are taken and considered as the coordinates of a point in the n-dimensional space $\mathbb{R}^n$.

Chapter 4

Physical-Layer Secrecy

Security is an omnipresent topic nowadays. It is crucial to protect data and systems from malicious attacks and unauthorized access. One very vulnerable part of every system is the communication between different parts. Especially in wireless communication, eavesdropping a communication is easy due to the broadcast nature of the wireless channel. Therefore, protection mechanisms are necessary to prevent an eavesdropper from obtaining the transmitted data. The standard practice is to add authentication and encryption techniques to the existing protocols at various layers in the communication stack. Those cryptographical approaches rely on the computational power that an attacker has at its disposal. Some decades ago, a new security paradigm has been introduced, which is embedded at the physical layer. This so called *physical-layer secrecy* exploits the randomness of noisy communication channels to ensure that the malicious eavesdropper does not obtain any information about the sent messages.

In this chapter, we introduce the fundamental concepts and notations of physical-layer secrecy. Thereby, we follow the presentation in [53].

4.1 Perfect, Strong, and Weak Secrecy

In 1949, the concept of information-theoretic secure communication was introduced by Shannon [72]. Shannon's model of secure communication, which is

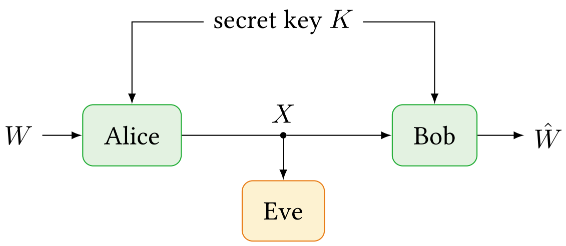


Figure 4.1: Shannon's cipher system.

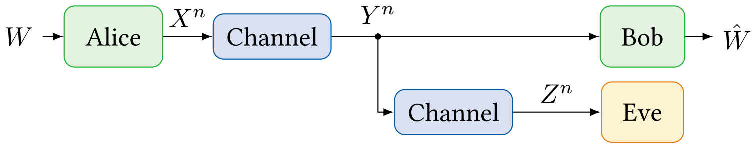


Figure 4.2: Wyner's wiretap channel model.

often called *Shannon's cipher system*, is illustrated in Fig. 4.1. It considers the transmission from a transmitter to a legitimate receiver over a noiseless channel, while an eavesdropper overhears all signals sent over the channel. In order to prevent the eavesdropper from retrieving any information, the transmitter encodes its messages $W \in \mathcal{W}$ into codewords $X \in \mathcal{X}$. This encoding is done with a secret key $K \in \mathcal{K}$, which is known by the transmitter and the legitimate receiver but unknown by the eavesdropper. We assume that K is independent of W. The encoding function is denoted by $f : \mathcal{W} \times \mathcal{K} \to \mathcal{X}$, the decoding function is denoted by $g : \mathcal{X} \times \mathcal{K} \to \mathcal{W}$, and the pair (f, g) is called a *coding scheme*.

In order to measure information-theoretic secrecy, we consider the conditional entropy $H(W \mid X)$, which is called the eavesdropper's *equivocation*. The equivocation represents the eavesdropper's uncertainty about the message after intercepting the codewords. A coding scheme is said to achieve *perfect secrecy* if

$$H(W \mid X) = H(W) \quad \text{or, equivalently,} \quad I(W; X) = 0. \tag{4.1}$$

The quantity $I(W; X)$ is called *leakage* of information to the eavesdropper. In other words, perfect secrecy is achieved if the codewords X are statistically independent of the messages W.

Random noise is an intrinsic element of almost all physical communication systems. In order to understand the role of noise in the context of secure communication, Wyner introduced the *wiretap channel* model as illustrated in Fig. 4.2. The main differences to Shannon's original secrecy system are that

- the legitimate transmitter encodes a message W into a codeword X^n consisting of n symbols, which is sent over a noisy channel to the legitimate receiver;
- the eavesdropper observes a noisy version Z^n of the signal Y^n available at the receiver.

Additionally, Wyner suggested a new definition for the secrecy condition. It is convenient to replace the requirement of exact statistical independence between the message W and the eavesdropper's observation Z^n by asymptotic statistical independence as the codeword length n goes to infinity. In principle, this asymptotic independence can be measured in terms of any distance defined on the set of joint probability distributions on $\mathcal{W} \times \mathcal{Z}^n$. Most often the Kullback-Leibler divergence is used, which results in the following requirement

$$\lim_{n\to\infty} I(W;Z^n) = 0. \tag{4.2}$$

This condition is called *strong secrecy condition* and requires the amount of information leaked to the eavesdropper to vanish.

Further, it is also convenient to consider the condition

$$\lim_{n\to\infty} \frac{1}{n} I(W;Z^n) = 0, \tag{4.3}$$

which requires the rate of information leaked to the eavesdropper to vanish. This condition is weaker than strong secrecy since it is satisfied as long as $I(W;Z^n)$ grows at most sub-linearly with n. It is called *weak secrecy condition.*

It can be shown that there exist channel codes that asymptotically guarantee both an arbitrarily small probability of error at the intended receiver and a prescribed level of secrecy. Such codes are known as *wiretap codes.* The supremum of the transmission rates that are achievable under these premises is called *secrecy capacity.*

4.2 Degraded Wiretap Channel

In 1975, Wyner introduced the wiretap channel [77], where a transmitter, Alice, wants to send a confidential message to a receiver, Bob, in the presence of an eavesdropper, Eve. This system model corresponds to wired transmissions, where the eavesdropper is wiretapping the signal at the receiver and gets only a degraded version of Bob's receive signal. This channel model is depicted in Fig. 4.2 and is called degraded wiretap channel (DWTC).

A *discrete memoryless DWTC* $(\mathcal{X}, p_{Z|Y}p_{Y|X}, \mathcal{Y}, \mathcal{Z})$ consists of the finite input alphabet $\mathcal{X}$, the two finite output alphabets $\mathcal{Y}$ and $\mathcal{Z}$, and the transition probabilities of two discrete memoryless channels (DMCs), such that

$$\begin{aligned}&\forall n \geq 1 \quad \forall (x^n, y^n, z^n) \in \mathcal{X}^n \times \mathcal{Y}^n \times \mathcal{Z}^n : \\ &p_{Y^n Z^n|X^n}(y^n, z^n|x^n) = \prod_{i=1}^{n} p_{Y|X}(y_i|x_i) p_{Z|Y}(z_i|y_i).\end{aligned} \tag{4.4}$$

The DMC $(\mathcal{X}, p_{Y|X}, \mathcal{Y})$, which is characterized by the marginal transition probabilities $p_{Y|X}$, is referred to as the *main channel*, while the DMC $(\mathcal{X}, p_{Z|X}, \mathcal{Z})$, which is characterized by the marginal transition probabilities $p_{Z|X}$, is referred to as the *eavesdropper's channel*.

Randomness in the encoding process enables secure communication. It is convenient to represent this randomness by the realization of a discrete memoryless source (DMS) $(\mathcal{R}, p_R)$, which is independent of the channel and of the transmitted messages. Because the source is available to Alice but neither to Bob nor to Eve, we call it a source of *local randomness*.

4.1 Definition (Wiretap Code). A $(2^{nR_s}, n)$ code $\mathcal{C}_n$ of a DWTC consists of

- a message set $\mathcal{W} = \left\{1, \dots, 2^{nR_s}\right\}$,
- a source of local randomness $(\mathcal{R}, p_R)$ at the encoder,
- an encoding function $f : \mathcal{W} \times \mathcal{R} \to \mathcal{X}^n$, which maps a message w and a realization of the local randomness r to a codeword x^n, and
- a decoding function $g : \mathcal{Y}^n \to \mathcal{W} \cup \{e\}$, which maps each channel observation y^n to a message $\hat{w} \in \mathcal{W}$ or an error message e. $\triangleleft$

The reliability of a code $\mathcal{C}_n$ is measured in terms of its average probability of error

$$\mathbf{P}_e(\mathcal{C}_n) \triangleq \Pr(\hat{W} \neq W \mid \mathcal{C}_n). \tag{4.5}$$

The secrecy performance of a code $\mathcal{C}_n$ is measured either in terms of the leakage of information to the eavesdropper

$$\mathbf{L}(\mathcal{C}_n) \triangleq I(W; Z^n \mid \mathcal{C}_n) \tag{4.6}$$

or, equivalently, in terms of the equivocation, i.e., the uncertainty of the eavesdropper

$$\mathbf{E}(\mathcal{C}_n) \triangleq H(W \mid Z^n \mathcal{C}_n). \tag{4.7}$$

4.2 Definition. A weak rate-equivocation pair (R_s, R_e) is achievable for the DWTC if there exists a sequence of $(2^{nR_s}, n)$ codes $\{\mathcal{C}_n\}_{n\geq 1}$ such that

$$\lim_{n\to\infty} \mathbf{P}(\mathcal{C}_n) = 0 \qquad \text{(reliability condition)}, \tag{4.8}$$

$$\lim_{n\to\infty} \frac{1}{n}\mathbf{E}(\mathcal{C}_n) \geq R_e \quad \text{(weak secrecy condition)}. \tag{4.9}$$

The weak rate-equivocation region of a DWTC is

$$\mathcal{R}_s \triangleq \mathrm{cl}\{(R_s, R_e) : (R_s, R_e) \text{ is achievable}\}, \tag{4.10}$$

and the weak secrecy capacity of a DWTC is

$$C_s \triangleq \sup\{R_s : (R_s, R_s) \in \mathcal{R}_s\}. \tag{4.11}$$

◁

4.3 Untrusted Relays

The models which were introduced so far assume a set of legitimate users and one (or possibly more) eavesdroppers. When we assume a two- or multi-way relay channel, the relay might be the eavesdropper. Despite being part of the communication chain, the relay might not be trustworthy. This means that a coding scheme is needed that prevents the relay from obtaining the transmitted messages, but still allows the legitimate users to communicate with the help of the relay. This might sound impossible at first sight, but with the help of network coding theory, we can develop a coding scheme that fulfills the requirements. It is already obvious that we can achieve secret message transmission only if we apply a relaying scheme where the relay does not need to decode messages individually. Further, we need to achieve a transmission rate tuple which is not inside the MAC capacity region, otherwise the relay would be able to decode the individual messages, for example by successive interference cancellation (SIC). This is a necessary but not a sufficient condition. A relaying strategy that allows to fulfill these requirements is compute-and-forward (see Chapter 3). In Part III, we build a coding scheme based on compute-and-forward and wiretap codes to allow reliable and secret message transmission via an untrusted relay.

In the following, we will formally describe the multi-way untrusted relay channel and the respective secrecy criterion. Assume that there are L legitimate users as illustrated in Fig. 4.3. Each user ℓ with $\ell \in \{1, 2, \ldots, L\}$ transmits a

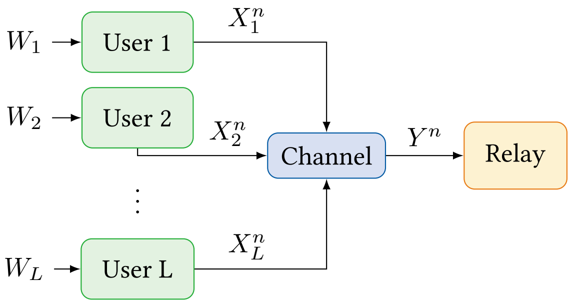


Figure 4.3: L user untrusted relay channel.

message W_ℓ to a relay, which is additionally considered as an eavesdropper. The weak secrecy condition in Eq. (4.3) needs to be extended to L users. For any $\mathcal{W}_p \in \mathcal{P}(\{W_1, \dots, W_L\})$, it can be formulated as

$$\lim_{n\to\infty} \frac{1}{n} I(\mathcal{W}_p; Y_r) = 0, \tag{4.12}$$

where $\mathcal{P}$ denotes the power set. By using the chain rule for mutual information, we see that

$$I(W_1, \dots, W_L; Y_r) - I(W_1, \dots, W_m; Y_r) = \sum_{\ell=m}^{L} I(W_\ell; Y_r \mid W_1, \dots, W_{\ell-1}).$$

Due to the non-negativity of mutual information, we have

$$I(W_1, \dots, W_L; Y_r) - I(W_1, \dots, W_m; Y_r) \geq 0 \tag{4.13}$$

and therefore

$$\forall m < L : I(W_1, \dots, W_L; Y_r) \geq I(W_1, \dots, W_m; Y_r). \tag{4.14}$$

Hence, the following condition implies Eq. (4.12)

$$\lim_{n\to\infty} \frac{1}{n} I(W_1, \dots, W_L; Y_r) = 0. \tag{4.15}$$

We call Eq. (4.15) the weak secrecy condition for the untrusted relay channel.

4.3 Definition (Weak Secrecy Rate). We say that a secrecy rate R_{s_ℓ} is achievable for user ℓ if each user m with $m \in \{1, 2, \ldots, L\}$ gets the correct estimate $\hat{w}_\ell$ of message w_ℓ with an arbitrarily small probability of error and if the weak secrecy condition in Eq. (4.15) is fulfilled. The weak secrecy rate of user ℓ is defined as

$$R_{s_\ell} = \lim_{n\to\infty} \tfrac{1}{n} H(W_\ell). \tag{4.16}$$

◁

Part II

Multi-Hop Relay Channels

Chapter 5

Finding the Best Equation

We introduced the compute-and-forward framework in Chapter 3 and showed that the achievable rates depend on the channel coefficients and the decoded linear combination. It is the task of the relays to optimize their achievable rates by choosing the best linear combination. With our information-theoretic view, best linear combination means the lattice coefficient vector that results in the highest achievable computation rate. In general, this is an np-hard problem because the lattice coefficient vector takes only elements in the set of integers. However, it is possible to exploit the special structure of the optimization problem to reduce the complexity. Up to the writing of this thesis, several algorithms have been proposed to solve this optimization problem [109, 42, 44, 45, 50]. The algorithms differ not only in speed and complexity but also in optimality. The work in [45, 50] provides the fastest algorithms so far but does not necessarily find the optimal coefficient vector. The algorithm proposed in [42] is faster than the branch-and-bound algorithm in [109] and finds the optimal coefficient vector. However, the branch-and-bound algorithm in [109] makes it very easy to add additional constraints to the solutions such as non-zero coefficients. That makes this algorithm especially valuable for enhanced compute-and-forward relaying as it will be introduced in Chapter 6. Therefore, we provide a detailed description of this algorithm in this chapter.

5.1 Problem Statement

We recall the computation rate of compute-and-forward from Eq. (3.13), i.e.,

$$\mathcal{R}(h, a) = \frac{1}{2} \log_2^+ \left(\left(\|a\|^2 - \frac{P(h'a)^2}{1 + P\|h\|^2} \right)^{-1} \right). \tag{5.1}$$

The goal is to find an efficient method to calculate the coefficient vector $a \neq 0$ such that $\mathcal{R}(h, a)$ is maximized for a given channel vector h. Because a is the only variable for the optimization problem, we define

$$\mathcal{R}_h(a) \triangleq \mathcal{R}(h, a). \tag{5.2}$$

The optimization problem can then be written as

$$\max_{a\in\mathbb{Z}^n\setminus\{0\}} \mathcal{R}_h(a). \tag{5.3}$$

Before we start to investigate solutions of Eq. (5.3), we will transform it to an equivalent standard problem. Therefore, at the first stage we define

$$x \triangleq \sqrt{\frac{P}{1+P\|h\|^2}}\, h. \tag{5.4}$$

Then, $0 < \|x\| < 1$ so that we may call x the normalized channel vector. Next, let A and B two matrices with

$$A \triangleq \operatorname{diag}\left[\operatorname{sgn}(x_1),\dots,\operatorname{sgn}(x_n)\right], \tag{5.5}$$

$$B \triangleq \text{permutation such that } |Bx_1| \geq \cdots \geq |Bx_n| \tag{5.6}$$

Then, the vector ABx is called standard normalized channel vector. Without loss of generality, we always assume x to be of this type, i.e.,

$$x_1 \geq \cdots \geq x_n \geq 0. \tag{5.7}$$

For details regarding this assumption see Section 5.2.1. Now, with respect to x we can write

$$\mathcal{R}_x(a) \triangleq \frac{1}{2}\log^+\left(\left(\|a\|^2 - (x'a)^2\right)^{-1}\right). \tag{5.8}$$

Then, $\mathcal{R}_x = \mathcal{R}_h$, and we have

$$\max_{a\in\mathbb{Z}^n\setminus\{0\}} \mathcal{R}_h(a) = \max_{a\in\mathbb{Z}^n\setminus\{0\}} \mathcal{R}_x(a). \tag{5.9}$$

As a final stage of transformation, we consider

$$\|a\|^2 - (x'a)^2 = a'Xa, \tag{5.10}$$

where $X \triangleq I - xx'$ is symmetric and positive definite. Then, the equivalent standard problem is given by

$$\operatorname*{argmax}_{a\in\mathbb{Z}^n\setminus\{0\}} \mathcal{R}_x(a) = \operatorname*{argmin}_{a\in\mathbb{Z}^n\setminus\{0\}} a'Xa. \tag{5.11}$$

5.2 Characterization of Solutions

Here, we will analyze properties of the solutions of Eq. (5.11). Since we are only interested in positive rates, i.e. $\mathcal{R}_x = \mathcal{R}_h > 0$, and X is positive definite, we must have

$$0 < a'Xa = \|a\|^2 - (x'a)^2 < 1, \quad a \in \mathbb{Z}^n \setminus \{0\}. \tag{5.12}$$

Hence, first we have to make sure that such a solution exists.

5.1 Theorem. Problem Eq. (5.11) always has a solution. Moreover, if a^* is a solution then $-a^*$ is a solution as well. $\triangleleft$

Proof. From $0 < \|x\|$ and $x_1 \geq \cdots \geq x_n \geq 0$, we observe that $a = (1, 0, \ldots, 0)$ is always feasible for Eq. (5.11), as

$$0 < \|a\|^2 - (x'a)^2 = 1 - x_1^2 < 1. \tag{5.13}$$

Further, from [40, Lemma 1] and Eq. (5.4) we have

$$\|a\| < \frac{1}{\sqrt{1 - \|x\|^2}}. \tag{5.14}$$

Hence, both relations Eq. (5.13) and Eq. (5.14) make sure that the feasible set of Eq. (5.11) is non-empty, bounded, and finite. Therefore, a solution can be found by complete search.

Now assume a^* to be a solution of Eq. (5.11). Because of

$$(x'a^*)^2 = (-1)^2(x'a^*)^2 = [x'(-a^*)]^2, \tag{5.15}$$

$-a^*$ is also a solution. $\square$

5.2.1 Solution Properties of the Equivalent Standard Problem

Next, we will investigate if the solutions of the original problem Eq. (5.3) and the equivalent standard problem Eq. (5.11) considering Eq. (5.7) are truly the same.

5.2 Lemma. Let $x = (x_1, \ldots, x_n)'$ be an arbitrary normalized channel vector and $a = (a_1, \ldots, a_n)'$ an optimal coefficient vector of Eq. (5.11). Then, either $\operatorname{sgn}(a_i) = \operatorname{sgn}(x_i)$ or $\operatorname{sgn}(a_i) = -\operatorname{sgn}(x_i)$ holds for all $i \in \{1, 2, \ldots, n\}$. $\triangleleft$

Proof. Assume that the statement does not hold. Then, there exist indices $k, l \in \{1, 2, \ldots, n\}$ with $k \neq l$ such that $\operatorname{sgn}(a_k) = \operatorname{sgn}(x_k)$ and $\operatorname{sgn}(a_l) = -\operatorname{sgn}(x_l)$.

At the first stage, assume

$$\operatorname{sgn}\left(\sum_{i \neq k,l} x_i a_i\right) \geqq 0. \tag{5.16}$$

Due to $\operatorname{sgn}(a_k) = \operatorname{sgn}(x_k)$, this implies

$$\operatorname{sgn}\left(\sum_{i \neq l} x_i a_i\right) \geq 0. \tag{5.17}$$

Next, let $b = (b_1, \ldots, b_n)'$ such that

$$b_i \triangleq \begin{cases} -a_l & \text{if } i = l \\ a_i & \text{otherwise} \end{cases}. \tag{5.18}$$

Then, we have $\|b\| = \|a\|$. The optimality of a yields

$$(x'b)^2 \leq (x'a)^2 \tag{5.19}$$

$$\Rightarrow \quad \left(\sum_{i \neq l} x_i a_i - x_l a_l\right)^2 \leq \left(\sum_{i \neq l} x_i a_i + x_l a_l\right)^2 \tag{5.20}$$

$$\Rightarrow \quad 0 \leq 4 x_l a_l \left(\sum_{i \neq l} x_i a_i\right) \tag{5.21}$$

$$\Rightarrow \quad 0 \leq \operatorname{sgn}(x_l) \operatorname{sgn}(a_l), \tag{5.22}$$

where the implications follow from Eq. (5.18), the binomial theorem, as well as Eq. (5.17) together with $r = \operatorname{sgn}(r)|r|$ for any real-valued number r. This result however contradicts $\operatorname{sgn}(a_l) = -\operatorname{sgn}(x_l)$.

Hence, consider the alternative to Eq. (5.16), i.e., assume

$$\operatorname{sgn}\left(\sum_{i \neq k,l} x_i a_i\right) < 0. \tag{5.23}$$

With $\operatorname{sgn}(a_l) = -\operatorname{sgn}(x_l)$, this clearly implies

$$\operatorname{sgn}\left(\sum_{i\neq k} x_i a_i\right) < 0. \tag{5.24}$$

Now, let $b = (b_1, \dots, b_n)'$ such that

$$b_i \triangleq \begin{cases} -a_k & \text{if } i = k \\ a_i & \text{otherwise} \end{cases}. \tag{5.25}$$

Then, for the same reasons as before we find

$$(x'b)^2 \leq (x'a)^2 \quad \Rightarrow \quad 0 \leq -\operatorname{sgn}(x_k)\operatorname{sgn}(a_k), \tag{5.26}$$

which contradicts $\operatorname{sgn}(a_k) = \operatorname{sgn}(x_k)$. □

5.3 Remark. Lemma 5.2 ensures both the solutions of Eq. (5.3) and Eq. (5.11) to be the same even if the sign of the normalized channel vector was changed. Consequently, a solution of Eq. (5.11) with respect to Ax instead of x gives an optimal coefficient vector a^*, which multiplied by A or $-A$ gives also optimal coefficient vectors of Eq. (5.3). ◁

5.4 Lemma. Let $x = (x_1, x_2, \dots, x_n)'$ be an arbitrary normalized channel vector and $a = (a_1, a_2, \dots, a_n)'$ an optimal coefficient vector of Eq. (5.11). If $x_1 > x_2 > \cdots > x_n > 0$, then either $a_1 \geq a_2 \geq \cdots \geq a_n \geq 0$ or $-a_1 \geq -a_2 \geq \cdots \geq -a_n \geq 0$ holds. ◁

Proof. Due to Lemma 5.2, we have to consider two cases regarding the sign of a.

First, consider $\operatorname{sgn}(a_i) = \operatorname{sgn}(x_i)$ for all $i \in \{1, 2, \dots, n\}$. Then, we have to show that

$$x_1 > x_2 > \cdots > x_n > 0 \quad \Rightarrow \quad a_1 \geq a_2 \geq \cdots \geq a_n \geq 0 \tag{5.27}$$

holds. Assume that this statement does not hold. Then, there exist indices $k, l \in \{1, 2, \dots, n\}$ with $k < l$ such that $a_k < a_l$. Let $b = (b_1, \dots, b_n)'$ with

$$b_i \triangleq \begin{cases} a_l & \text{if } i = k \\ a_k & \text{if } i = l \\ a_i & \text{otherwise} \end{cases}. \tag{5.28}$$

Then, we have $\|b\| = \|a\|$. The optimality of a yields

$$\begin{aligned}
(x'b)^2 \leq (x'a)^2 \quad &\Rightarrow \quad x'b \leq x'a && (5.29)\\
&\Rightarrow \quad x_k a_l + x_l a_k \leq x_k a_k + x_l a_l && (5.30)\\
&\Rightarrow \quad a_l(x_k - x_l) \leq a_k(x_k - x_l) && (5.31)\\
&\Rightarrow \quad a_l \leq a_k, && (5.32)
\end{aligned}$$

which contradicts $a_k < a_l$.

Now, consider $\operatorname{sgn}(a_i) = -\operatorname{sgn}(x_i)$ for all $i \in \{1, 2, \ldots, n\}$. Then, we have to show the implication

$$x_1 > x_2 > \cdots > x_n > 0 \quad \Rightarrow \quad -a_1 \geq -a_2 \geq \cdots \geq -a_n \geq 0. \qquad (5.33)$$

Again, assume that this statement does not hold. Then, there exist indices $k, l \in \{1, 2, \ldots, n\}$ with $k < l$ such that $a_k > a_l$. Again, let b be given by Eq. (5.28). From $\|b\| = \|a\|$ and the optimality of a, we get

$$\begin{aligned}
(x'b)^2 \leq (x'a)^2 \quad &\Rightarrow \quad x'b \geq x'a && (5.34)\\
&\Rightarrow \quad x_k a_l + x_l a_k \geq x_k a_k + x_l a_l && (5.35)\\
&\Rightarrow \quad a_l(x_k - x_l) \geq a_k(x_k - x_l) && (5.36)\\
&\Rightarrow \quad a_l \geq a_k, && (5.37)
\end{aligned}$$

which contradicts $a_k > a_l$. □

5.5 Remark. Lemma 5.4 ensures both the solutions of Eq. (5.3) and Eq. (5.11) to be the same even if the elements of the normalized channel vector have been sorted. That is, a solution of Eq. (5.11) with respect to Bx instead of x gives an optimal coefficient vector a^*, which multiplied by B' or $-B'$ gives also optimal coefficient vectors of Eq. (5.3). ◁

5.6 Remark. Lemma 5.4 assumes $x_i \neq x_j$ for $i \neq j$. If otherwise, there exist $i, j \in \{1, \ldots, n\}$ with $i \neq j$ such that $h_i = h_j$. Then, the optimal coefficient vector a^* as well as a^{**} with

$$a_i^{**} = \begin{cases} a_l^* & \text{if } i = k \\ a_k^* & \text{if } i = l \\ a_k^* & \text{otherwise} \end{cases} \qquad (5.38)$$

give the same solution. ◁

5.7 Remark. Lemma 5.4 assumes $x_i > 0$ for all $i \in \{1, \dots, n\}$. If $x_n = 0$, then problem Eq. (5.11) can be reduced with respect to $x = (x_1, \dots, x_{n-1})'$ as a_n must also be zero. If otherwise, a would not be optimal as

$$\|a\|^2 - (x'a)^2 = \sum\nolimits_{i=1}^{n} a_i^2 - \left[\sum\nolimits_{i=1}^{n-1} x_1 a_i\right]^2 \tag{5.39}$$

$$\geq \sum\nolimits_{i=1}^{n-1} a_i^2 - \left[\sum\nolimits_{i=1}^{n-1} x_1 a_i\right]^2 . \tag{5.40}$$

$\triangleleft$

5.2.2 Geometric Interpretation of the Problem

From Eq. (5.11) and the properties of X follows that we have to find the smallest ellipsoid

$$E(y) = \{u \in \mathbb{R}^n : u'Xu = y\}, \tag{5.41}$$

which includes a non-zero integer point. In order to find such a point, we may use properties of the ellipsoid. First, the eigenvectors of X point in the direction of the semiaxes of the ellipsoid. Second, the eigenvalues μ_i correspond to the quadratic inverse of the length ℓ_i of the semiaxes,

$$\ell_i = \frac{1}{\sqrt{\mu_i}}, \quad i = 1, \dots, n. \tag{5.42}$$

Because of the special structure of the matrix X, we already know the eigenvectors and eigenvalues. All eigenvalues will be one except for one eigenvalue, which is smaller than one, given by

$$\mu_1 = 1 - \|x\|^2, \tag{5.43}$$

with the corresponding eigenvector

$$v_1 = \frac{x}{\|x\|}. \tag{5.44}$$

The eigenvectors v_i with $i \in \{1, 2, \dots, n\}$ are linearly independent. It follows that the ellipsoid is more like a sphere that is baggy in the direction of v_1. The set of possible ellipsoids is bounded by $E(1)$, which follows from Eq. (5.12).

5.3 Solving the Optimization Problem

5.3.1 Complete Search

In order to achieve a computation rate $\mathcal{R}_h(a) > 0$, the authors of [40] gave an upper bound on the set of possible coefficient vectors a by [40, Lemma 1]

$$\|a\| < \sqrt{1 + P\|h\|^2}. \tag{5.45}$$

This condition is necessary but not sufficient and corresponds to all integer points inside a sphere

$$S(r) \triangleq \{u : \|u\| = r\}. \tag{5.46}$$

The relation in Eq. (5.45) with respect to the normalized channel vector is given by Eq. (5.14). Since the right-hand side of Eq. (5.14) equals the largest semiaxes (there is only one different from one) of the ellipsoid $E(1)$, the sphere $S(\ell_1)$ upper bounds the ellipsoid $E(1)$. Since the volume of the sphere is much larger than the volume of the ellipsoid, searching over all points inside the sphere can be very inefficient. It would be more efficient to search only over all points inside the ellipsoid. However, finding all integer points inside an ellipsoid is a non-trivial task. There are some publications available addressing the number of lattice / integer points inside an ellipsoid, but as far as we know there are non addressing the issue of finding the actual points. Since we do not need to search over the whole ellipsoid in our case, this task might get even more complicated. In the following, we provide a branch-and-bound algorithm that performs much better than the search over all points inside the sphere.

5.3.2 Branch-and-Bound Algorithm

In [79], the authors proposed an efficient algorithm to solve CQIP problems

$$\begin{aligned} \min \quad & f(a) \triangleq a'Xa + L'a + c \\ \text{s.t.} \quad & a \in \mathbb{Z}^n \cap B, \end{aligned} \tag{5.47}$$

where $X \in \mathbb{R}^{n\times n}$ is positive definite, $L \in \mathbb{R}^n$, $c \in \mathbb{R}$, and $B \subseteq \mathbb{R}^n$ is a convex set. This corresponds to the optimization problem Eq. (5.11) with $L = 0$ and $c = 0$.

Let $\bar{X}_d$ be the submatrix of X given by rows and columns $d+1, \ldots, n$. The suggested algorithm is very fast because the matrices $\bar{X}_d$ and $\bar{X}_d^{-1}$ can be

Algorithm 1 : Modified branch-and-bound algorithm.

```
Input : the normalized channel vector $x$
Output : a vector $a \in \mathbb{Z}^n \setminus \{0\}$ minimizing $f(a) = a'Xa$
1  let $\tilde{x} = ABx$ (see Eq. (5.5) and Eq. (5.6));
2  calculate matrix $X = I - \tilde{x}\tilde{x}'$;
3  let $\bar{X}_d$ be the submatrix of $X$ given by rows and columns $d+1, \dots, n$;
4  compute the inverse matrices $\bar{X}_d^{-1}$ for $d = 1, \dots, n$;
5  set $d \triangleq 1$, $ub \triangleq X_{11}$, $r = r^* \triangleq (1, 0, \dots, 0)'$;
6  while $d \geq 1$ do
7      define $\bar{f} : \mathbb{R}^{n-d} \to \mathbb{R}$ by $\bar{f}(a) \triangleq f((r_1, \dots, r_d, a_1, \dots, a_{n-d})')$;
8      compute $\bar{L}$ and $\bar{c}$ s.t. $\bar{f}(a) = a'\bar{X}_d a + \bar{L}'a + \bar{c}$;
       // compute lower bound
9      compute the continuous minimum $\bar{a} \triangleq -\frac{1}{2}\left(\bar{X}_d^{-1}\bar{L}\right) \in \mathbb{R}^{n-d}$ of $\bar{f}$;
10     set $lb \triangleq \bar{f}(\bar{a})$;
       // compute upper bound
11     set $r_j \triangleq \lfloor \bar{a}_{j-d} \rceil$ for $j = d+1, \dots, n$ to form $r \in \mathbb{Z}^n$;
       // update solution
12     if $\bar{f}((r_{d+1}, \dots, r_n)') < ub$ then
13         set $r^* \triangleq r$;
14         set $ub \triangleq \bar{f}((r_{d+1}, \dots, r_n)')$;
15     end
       // prepare next node
16     if $lb < ub$ then
17         set $d \triangleq d + 1$;
18     else
19         set $d \triangleq d - 1$;
20         if $d > 0$ then
21             increment $r_d$ according to (2) or (3) in [79]
22         end
23     end
24 end
25 set $a = (AB)'r^*$;
```

precomputed. Regarding the special structure of X, in particular the matrix inversion of $\bar{X}_d$ can be efficiently achieved by

$$\bar{X}_d^{-1} = -\frac{1}{1 - \sum_{i=d}^{n} \tilde{x}_i^2} \left(\bar{X}_d - \left(2 - \sum_{i=d}^{n} \tilde{x}_i^2 \right) I \right) . \tag{5.48}$$

The basic idea of the branch-and-bound algorithm is to compute the real-valued minimum of the objective function as a lower bound while fixing the first d components of the vector a to integers. An upper bound is obtained with the rounded version of the real-valued minimum. As long as the upper bound is greater than the lower bound, the current component is fixed to its rounded integer value, and the computation continues with the next component. If otherwise, the previous component will be incremented.

Without modification of the algorithm given by [79], the solution would be zero, since $L = 0$ and $c = 0$. Without the constraint $a \neq 0$, this will be the optimal solution. However, with this additional constraint present, the first component of a has to be at least one by Lemma 5.2 and Lemma 5.4. Hence, the vector $r = (1, 0, \ldots, 0)'$ is a possible solution, which also gives $X_{11} = 1 - x_1^2$ as an upper bound. Instead of searching in all directions, it is sufficient to search only in positive direction, because we can transform the optimization problem to standard form Eq. (5.11).

5.8 Example. Consider the normalized channel vector $x = (0.95, 0.3)'$. The branch-and-bound algorithm will need four iterations to find the optimal coefficient vector $a = (3, 1)'$. All steps are given in Table 5.1 with the following variables:

Variable	Description
k	number of iterations
d	current vector component
$\bar{a}$	the continuous minimum with the first d components fixed to integers
lb	the current lower bound
r	the current coefficient vector
$\bar{f}(r)$	defined as in line 7 of Algorithm 1
ub	current upper bound

Table 5.1: Detailed iterations for normalized channel vector $x = (0.95, 0.3)'$.

k	d	$\bar{a}$	lb	r	$\bar{f}(r)$	ub
0	1	-	-	$(1,0)'$	-	0.0975
1	1	$(1,0.3132)'$	0.0082	$(1,0)'$	0.0975	0.0975
2	1	$(2,0.6264)'$	0.0330	$(2,1)'$	0.1600	0.0975
3	1	$(3,0.9396)'$	0.0742	$(3,1)'$	0.0775	0.0975
4	1	$(4,1.2527)'$	0.1319	$(4,1)'$	0.1900	0.0775

In this setting, the matrix $X = I - xx'$ is given by

$$X = \begin{pmatrix} 0.0975 & -0.2850 \\ -0.2850 & 0.9100 \end{pmatrix}. \tag{5.49}$$

Iteration 0 initializes the algorithm with vector $r = (1, 0)'$ and the corresponding upper bound, which is given by X_{11}. Iteration 1 calculates the continuous minimum with the first component fixed to one. Then, the vector r is given by rounding the continuous minimum to the next integer, which results again in $r = (1, 0)'$. So the same upper bound is obtained. Since this bound is not better than the previous one and the algorithm is already at the last component, the previous component is increased by one. Iteration 2 calculates again the continuous minimum with the first component fixed to two. The vector r is obtained as before and results in an higher upper bound, so the upper bound is not updated and the previous component is increased by one. Iteration 3 gets now a smaller upper bound and updates it. Since the algorithm is at the last component, it increases the previous component. Iteration 4 now gets a lower bound which is larger than the upper bound and therefore the algorithm stops.$\triangleleft$

5.4 Simulation Results

In this section, we provide simulation results to compare the proposed branch-and-bound algorithm with the complete search inside the sphere $S(\ell_1)$ as well as inside the ellipsoid $E(1)$. All values are averages taken over 1000 randomly chosen vectors.

Fig. 5.1 shows that the number of iterations does not increase for $\|x\| \leq 0.7$. This is due to the fact that vectors with small norm correspond to the low-SNR regime at the relay or a bad channel. This results in an optimal coefficient vector

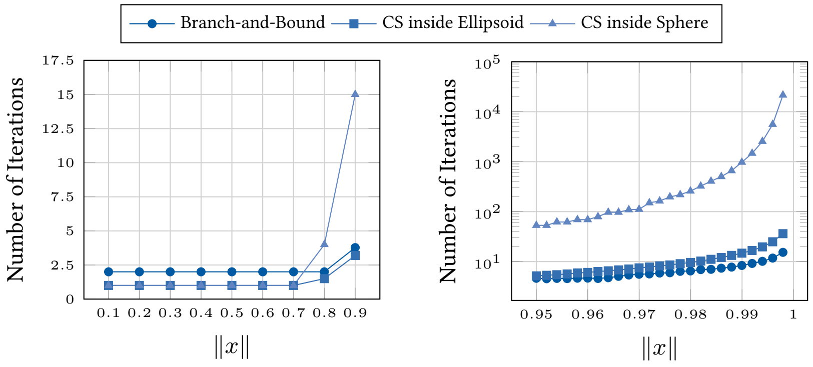


Figure 5.1: Number of iterations for the branch-and-bound algorithm (BnB) and the complete search (CS) with $n = 4$.

$a = (1, 0, \dots, 0)'$, which equals the initial vector r. Hence, only two iterations are required to find the solution of the problem, one to test the vector and one to stop the algorithm. With increasing $\|x\|$, the feasible set of points inside $E(1)$ increases.

We can see in Tables 5.3a and 5.3b that the proposed branch-and-bound algorithm (BnB) needs less iterations than a complete search over the ellipsoid (CS-EL). However, from Table 5.3c follows that the branch-and-bound algorithm performs worse when $n > 5$. This is due to the fact that the volume of the ellipsoid is decreasing. Nevertheless, the branch-and-bound algorithm performs significantly better than a complete search over the sphere (CS-SP), especially for $\|x\|$ close to 1.

Fig. 5.3a shows the best coefficient vector a in terms of computation rate for a grid of normalized channel vectors $x = (x_1, x_2)'$. Fig. 5.3b shows the computation rate corresponding to the best choice of coefficient vector a. These results were obtained by trying all different vectors a between $(0, 0)'$ and $(4, 4)'$ and choosing the vector which results in the highest rate. As you can see, the plot is symmetric in terms of the vector components, which illustrates Lemma 5.4. It is interesting to observe that for small $\|x\|$ the coefficient vector is either $(1, 0)'$ or $(0, 1)'$, which means that only a single message is decoded. Therefore, network coding with compute-and-forward is only good for $\|x\|$ close to one. Remember that a norm of x close to one means either high SNR or good channels to the relay.

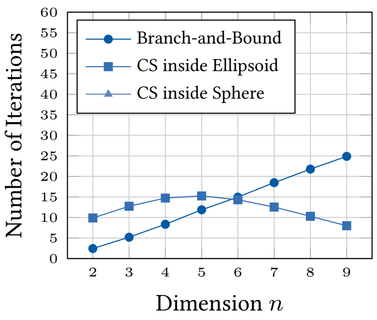
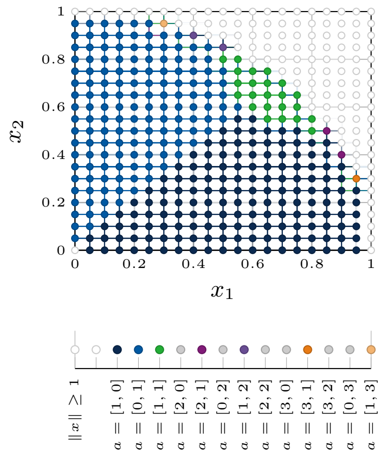
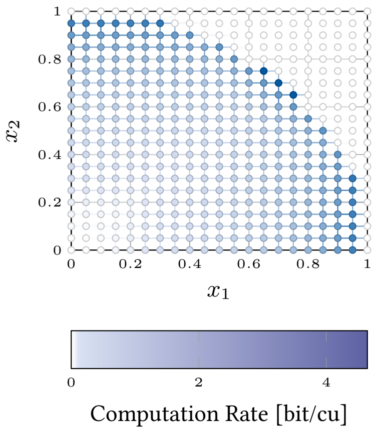


Figure 5.2: Number of iterations for the branch-and-bound algorithm (BnB) and the complete search (CS) with $\|x\| = 0.99$.



(a) Coefficient vector a resulting in the highest computation rate for normalized channel vector $x = (x_1, x_2)'$.



(b) Highest computation rate for normalized channel vector $x = (x_1, x_2)'$.

Figure 5.3: Best choices of the coefficient vector a and the respective achievable rates for $n = 2$.

Table 5.2: Average number of iterations.

(a) $n = 2$.

$\|x\|$	BnB	CS-EL	CS-SP
0.1	2	1	1
0.7	2	1	1
0.8	2	1.718	2
0.9	2	2.437	5
0.92	2	2.87	5
0.94	2	3.375	6
0.96	2	4.389	9
0.98	2.165	6.616	20
0.99	2.44	9.883	40

(b) $n = 4$.

$\|x\|$	BnB	CS-EL	CS-SP
0.1	2	1	1
0.7	2	1	1
0.8	2	1.501	4
0.9	3.777	3.194	15
0.92	4.211	3.781	24
0.94	4.457	4.564	31
0.96	4.662	6.061	69
0.98	6.486	9.597	258
0.99	8.348	14.732	972

(c) $n = 9$.

$\|x\|$	BnB	CS-EL	CS-SP
0.1	2	1	1
0.7	2	1	1
0.8	2	1.007	9
0.9	3.136	1.19	180
0.92	5.806	1.333	320
0.94	8.346	1.704	938
0.96	9.964	2.549	3857
0.98	16.41	4.891	58023
0.99	24.888	7.995	8.6287e+05

Chapter 6

Non-Cooperative Compute-and-Forward Strategies

The performance of the compute-and-forward framework highly depends on the alignment of the channel coefficients with the coefficients of the decoded equation. Several algorithms have been proposed to find the optimal equation in terms of achievable computation rate (see Chapter 5 for an overview). Most of these algorithms perform a local optimization. They ignore the network structure and the possible linear dependence of the equations at the sink nodes. This results in outages since a sink node is not able to decode the messages if it has not enough linear independent equations. This is a very serious problem in real applications, especially in large networks with several hops because re-transmissions from all source nodes to the sink nodes are not applicable. The delay might be extremely large, and the necessary signaling will pollute the network. This problem can be overcome by allowing cooperation between nodes [44]. In realistic setups, this might not be applicable due to the large signaling overhead, which would be necessary to allow this kind of cooperation. Further, it is not very robust against channel state changes. Therefore, the question arises if it is possible to exploit the network structure such that no cooperation between nodes is needed, where the network initialization phase is the only exception.

In this chapter, we introduce some non-cooperative schemes that enforce linear independence of the decoded equations at the sink nodes. We compare the performance of these schemes in multi-source multi-relay networks and show that correlation between channel coefficients plays a crucial role. Our simulation results show that correlation can decrease the achievable sum-rate of the classical compute-and-forward scheme by $1.5\,\mathrm{bit/cu}$, whereas our proposed schemes are robust against correlation and can achieve twice the sum-rate.

The results in this chapter were developed in collaboration with Jan Hejtmánek and Jan Sýkora at “České vysoké učení technické v Praze” (Czech Technical University in Prague). They have been published in [106] and presented at VTC-Fall in 2015.

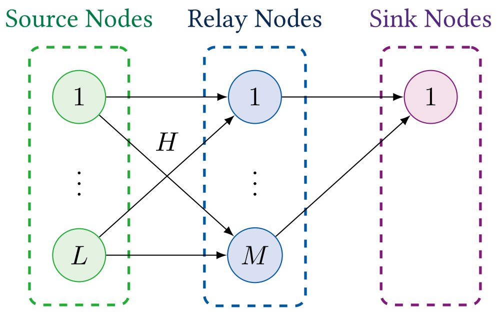


Figure 6.1: System model of an $L \times M \times 1$ relay network.

6.1 System Model

We investigate an $L \times M \times 1$ relay network consisting of L source nodes, M relay nodes, and one sink node as depicted in Fig. 6.1. Each source node ℓ has a message w_ℓ that has to be transmitted to the sink node. The communication has to be supported by several relay nodes since there are no direct links between the source nodes and the sink node. For simplicity, we assume that all M relay nodes are used. The system designer can choose between several relaying schemes, which are explained in detail in Section 6.2. All relaying strategies are based on the compute-and-forward framework as introduced in Chapter 3 and differ only in the optimization problem regarding the lattice coefficient vector a_m. The optimization problem for the classical CF strategy is explained in detail in Chapter 5. All relay nodes use the same relaying strategy, and we will investigate which relaying strategy is the best for certain system parameters.

6.1 Remark. We want to stress that we assume no cooperation between the relay nodes since this would drastically increase the complexity in realistic scenarios. The cost for network providers in terms of infrastructure complexity is much less when the nodes act independently. For example, we have less signaling overhead without cooperation. Further, node independence increases the robustness of the network against topology changes. $\triangleleft$

In this thesis, we will focus on the communication between source and relay nodes and the decoding at the relay nodes. There might be other layers or networks after the relay nodes, which are beyond the scope of our investigation.

Therefore, we assume that the channels between the relay nodes and the sink node are bit-pipes, whose capacity is large enough. They are error-free and do not interfere with each other. This might be achieved by FDMA or similar techniques.

The sink node receives M linear combinations of the lattice codewords from the relay nodes, i.e.,

$$y_D = Ax, \tag{6.1}$$

where $x = (x_1, \dots, x_L)'$ is the vector of transmitted lattice codewords from all source nodes, and $A = (a_{ij})$ with $i \in \{1, 2, \dots, M\}$ and $j \in \{1, 2, \dots, L\}$ is the coefficient matrix of the linear combinations. Please note that the m-th row in A is equal to the coefficient vector a_m of the linear combination decoded at relay m. The sink node can solve the system of linear equations in Eq. (6.1) via matrix inversion if the coefficient matrix A has full rank, i.e., $\operatorname{rank}(A) = m$. Otherwise, an outage occurs.

6.2 Definition (Outage Probability). The outage probability P_{out} is the probability that the sink node is not able to decode each single codeword x_ℓ transmitted by source node ℓ with $\ell \in \{1, 2, \dots, L\}$. $\lhd$

6.3 Definition (Achievable Sum-rate). The achievable sum-rate is defined as the goodput of all source nodes to the sink node, i.e.,

$$R_{\text{sum}} = (1 - P_{\text{out}}) \sum_{\ell=1}^{L} R_\ell, \tag{6.2}$$

where P_{out} is the outage probability. $\lhd$

6.2 Relaying Strategies

In this section, we introduce the relaying strategies. All of them are based on CF as introduced in Chapter 3. Each relay m with $m \in \{1, 2, \dots, M\}$ decodes a linear combination of lattice codewords with coefficient vector a_m. The achievable computation rate at relay m is given by

$$\mathcal{R}(h_m, a_m) = \frac{1}{2} \log_2^+ \left(\frac{1}{a_m' G_{\text{CF}} a_m} \right) \tag{6.3}$$

with

$$G_{\text{CF}} = I_L - \frac{P}{1 + P\|h_m\|^2} h_m h_m'. \tag{6.4}$$

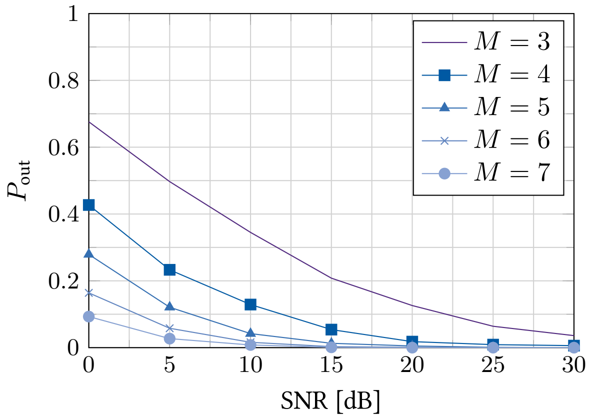


Figure 6.2: Probability of rank failure with the classical compute-and-forward scheme for 3 source nodes and M relay nodes.

The CF strategy is able to achieve a high computation rate due to the fact that the relay nodes decode a linear combination of messages and not each single message like for example the decode-and-forward strategy. However, the network structure is not taken into account. In a large network with multiple hops, we get outages due to linear dependent equations at the sink node. Even for a single hop with multiple relay nodes, the outage probability is very high (see Fig. 6.2). This motivates us to modify the optimization problem of finding the best equation such that the network structure is exploited. Therefore, we introduce some modified CF schemes, which differ in the way the lattice coefficient vector a_m is chosen.

6.2.1 Classical Compute-and-Forward

With the term classical CF, we refer to the unaltered scheme introduced in Chapter 3. In the classical CF scheme, each relay solves the following optimization problem individually to maximize the achievable computation rate

$$R_{\text{CF},m}^{\text{CCF}} = \max_{a_m \in \mathbb{Z}^L \setminus \{0\}} \mathcal{R}(h_m, a_m). \tag{6.5}$$

This can be solved by several algorithms, which are presented for example in [109, 42, 44].

Since we assume no cooperation between the relay nodes, it is not guaranteed that the relay nodes decode linearly independent combinations of codewords. Therefore, the sink node cannot always decode all codewords. In this case, the outage probability P_{out} is the probability that the coefficient matrix A does not have full rank, i.e.,

$$P_{\text{out}} \triangleq \Pr\{\text{rank}(A) < L\}. \tag{6.6}$$

A possible method to reduce the outage probability is to decode only non-zero equations with coefficient vector a_m [40, Sec. X]. However, this will significantly reduce the achievable rate and does not guarantee zero outage probability.

6.4 Remark. In real-world applications, we have to ensure that the transmitted data is decodable at the sink node. In a point-to-point channel with outages, this is usually done by an automatic repeat request (ARQ) protocol. In the CF framework, one would have to re-transmit the codewords from all source nodes. In multi-hop networks this is usually not applicable due to large delays and large overhead. ◁

6.2.2 Single-User Decoding

With the term single-user decoding, we refer to a special case of CF, where relay m decodes the linear combination $a_m = e_m$. This means that each relay decodes only a single codeword and treats all other signals as noise. Since each relay decodes a different codeword, it is guaranteed that the coefficient matrix A at the sink node has full rank. Therefore, we do not have outages, i.e., $P_{\text{out}} = 0$. However, this comes at the cost of achievable computation rate because we choose a suboptimal coefficient vector in Eq. (6.5). The achievable computation rate at relay m is given by

$$R_{\text{CF},m}^{\text{SU}} = \mathcal{R}(h_m, e_m). \tag{6.7}$$

6.2.3 Subspace Compute-and-Forward

In the following, we introduce the subspace CF scheme. We reduce the feasible set of possible coefficient vectors in the optimization problem in Eq. (6.5) for each relay such that the subsets are linearly independent. We get the following optimization problem at relay m:

$$R_{\text{CF},m}^{\text{SCF}} = \max_{a_m \in \mathcal{S}_m} \mathcal{R}(h_m, a_m) \tag{6.8}$$

with $\mathcal{S}_m \subset \mathbb{Z}^L$. For the construction of the subsets, we restrict ourselves to the case $L = M$, i.e., the number of relay nodes equals the number of source nodes. In the case of $M > L$, it is obvious that we get $M - L$ linear dependent equations at the sink nodes, which are not necessary for decoding. This becomes a problem of relay selection and is beyond the scope of this thesis.

Let $B \in \mathbb{Z}^{L\times L}$ be a basis matrix for $\mathbb{Z}^L$. We define B_m to be a sliced version of B containing only the first m rows of B. The subset $\mathcal{S}_m$ is then constructed by

$$\mathcal{S}_m = \{B_m' \beta_m : \beta_m \in \mathbb{Z}^m, \beta_{mm} \neq 0\}. \tag{6.9}$$

The achievable computation rate at relay m is the solution of the following optimization problem

$$R^{\text{SCF}}_{\text{CF},m} = \max_{\substack{\beta_m \in \mathbb{Z}^m \\ \beta_{mm} \neq 0}} \mathcal{R}(h_m, B_m' \beta_m), \tag{6.10}$$

which can be written as

$$R^{\text{SCF}}_{\text{CF},m} = \max_{\substack{\beta_m \in \mathbb{Z}^m \\ \beta_{mm} \neq 0}} \frac{1}{2} \log_2^+ \left(\frac{1}{\beta_m' G_{\text{SCF}} \beta_m} \right) \tag{6.11}$$

with

$$G_{\text{SCF}} = B_m B_m' - \frac{P}{1 + P\|h_m\|^2} B_m h_m h_m' B_m'. \tag{6.12}$$

6.5 Remark. Please note that the optimization problem has the same structure as the one for classical CF (see Eq. (6.5)). If a sorted channel vector h_m is assumed, the constraint sets are also equivalent (see Chapter 5 for details). Therefore, we can use the same algorithms to solve this optimization problem. ◁

6.2.4 Hierarchical Compute-and-Forward

Hierarchical CF has been introduced independently and from different points of view in [39, 43]. This scheme fixes the equations that are decoded at each relay independently from the channel realizations. Therefore, the equation coefficients can be chosen such that linear independence at the sink node is guaranteed. This comes at the prize of a lower performance since the channel coefficients and the equation coefficients are not well aligned in general. The key technique used to improve the performance is interference cancellation. The decoder at relay m decodes an auxiliary equation with coefficient vector

$\tilde{a}_m$ and subtracts this from the received signal to create a new virtual channel $\tilde{h}_m = h_m - \eta \tilde{a}_m$.

The auxiliary equation has to be chosen such that the decoding performance of the desired equation with coefficient vector a_m is increased for this virtual channel. The coefficient $\eta \in \mathbb{R}$ has to be chosen such that the mean squared error between the virtual channel $\tilde{h}_m$ and the desired coefficients a_m is minimized. In order to obtain the optimal auxiliary equation, one has to solve the following optimization problem at relay m:

$$R_{\text{CF},m}^{\text{HCF}} = \max_{\tilde{a} \in \mathcal{A}_m} \min\{\mathcal{R}(h_m, \tilde{a}_m), \mathcal{R}(\tilde{h}_m, a_m)\}, \tag{6.13}$$

where $\mathcal{A}_m = \{\tilde{a}_m : \mathcal{R}(h_m, \tilde{a}_m) > \mathcal{R}(h_m, a_m)\}$ is the set of all auxiliary equations that improve the performance. So far the optimization problem in Eq. (6.13) lacks an efficient algorithm to solve this problem. It is basically a combinatorial problem, which can be solved by an exhaustive search with high complexity.

6.6 Remark. Please note that the hierarchical CF scheme can be combined with the single-user decoding scheme as well as subspace CF scheme to improve the performance of the respective scheme. However, one has to keep in mind that this will dramatically increase the computational complexity for solving the respective optimization problems. The performance increase might not be worth the effort. $\triangleleft$

6.3 Simulation Results

In this section, we provide simulation results showing the performance of the relaying schemes introduced above. We use Algorithm 1 to obtain the coefficients for classical and subspace CF. Further, we use an exhaustive search for hierarchical CF. Please note that we measure the performance solely in terms of achievable sum-rate. Some practical concerns about signaling overhead and delay have already been raised at the beginning of the chapter. In addition to the non-cooperative schemes, we also plot the achievable sum-rate for CF with cooperation between the relay nodes as an upper bound. For that, we use the algorithm provided in [44].

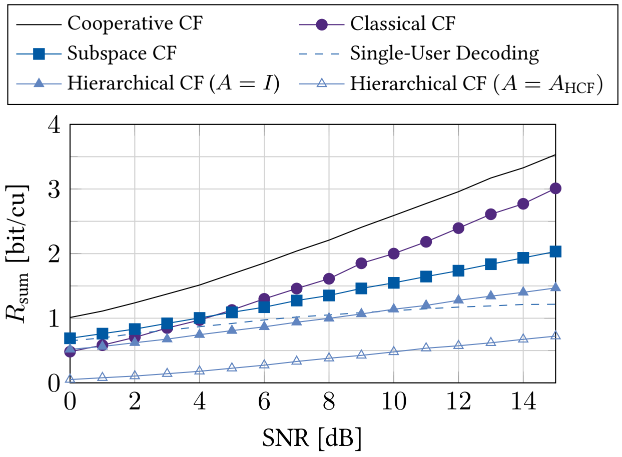


Figure 6.3: Achievable sum-rate for i.i.d. Gaussian channel coefficients. 10 000 channel matrices per SNR value.

6.3.1 No Correlation

In the following, we compare the achievable sum-rate of the different schemes in a $3 \times 3 \times 1$ relay network, where the channel coefficients are independent and identically distributed (i.i.d.) according to a standard Gaussian distribution, i.e., $h_{m\ell} \sim \mathcal{N}(0, 1)$. As one can see in Fig. 6.3, the classical non-cooperative CF scheme achieves approximately half a bit less sum-rate than the cooperative scheme. The subspace CF strategy performs well for low SNR but looses its performance benefit in the high-SNR regime.

The performance of hierarchical CF mainly depends on the chosen equations as one can see in Fig. 6.3, where we plotted the achievable sum-rate for two different coefficient matrices, which are $A = I_3$ and

$$A = A_{\text{HCF}} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}. \tag{6.14}$$

A good choice for the coefficient matrix is $A = I_M$. One might think that this should result at least in the same achievable sum-rate as single-user decoding. Unfortunately, this is not true as one can see from Eq. (3.14). The achievable computation rate of relay m only constrains the rate of the source nodes whose codewords are decoded with a non-zero coefficient. If relay m decodes with $a_m = e_i$, it will only affect the achievable rate of source node i. If relay m additionally decodes an auxiliary equation with more non-zero entries, the computation rate of that auxiliary equation affects more relay nodes. Since each source node is constrained by the minimum achievable computation rate of all relay nodes, it is possible that the sum-rate is reduced by using an auxiliary equation at a relay although the individual computation rate at the relay benefits from using an auxiliary equation. This is one side-effect of having no cooperation between the relay nodes.

6.7 Remark. For hierarchical CF, deriving the optimal desired coefficient matrix A for a given channel distribution is still an open problem and needs to be investigated in future research. ◁

6.3.2 Spatial Correlation

In this section, we want to show the influence of spatial correlation on the achievable sum-rate. Therefore, we model the channel between source nodes and relay nodes by the Kronecker model, i.e.,

$$H = R_R^{\frac{1}{2}} W R_T^{\frac{1}{2}}, \tag{6.15}$$

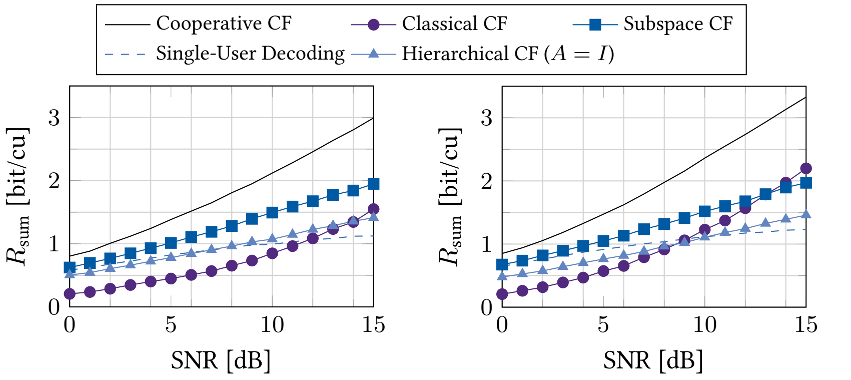
where R_R and R_T are the receive and transmit correlation matrices, respectively, and the entries $w_{m\ell}$ with $m \in \{1, 2, \dots, M\}$ and $\ell \in \{1, 2, \dots, L\}$ of the matrix $W = (w_{m\ell})$ are chosen i.i.d. from a standard Gaussian distribution, i.e., $w_{m\ell} \sim \mathcal{N}(0, 1)$. For the simulations, we generate 10 000 channel matrices per SNR value and assume a $3 \times 3 \times 1$ relay network with different correlation scenarios:

- strong correlation between two source nodes as well as two relay nodes:

$$R_R = \begin{pmatrix} 1.0 & 0.9 & 0.1 \\ 0.9 & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 \end{pmatrix} \; R_T = \begin{pmatrix} 1.0 & 0.9 & 0.1 \\ 0.9 & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 \end{pmatrix}, \tag{6.16}$$

- weak correlation between all three source nodes and strong correlation between two relay nodes:

$$R_R = \begin{pmatrix} 1.0 & 0.9 & 0.1 \\ 0.9 & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 \end{pmatrix} \; R_T = \begin{pmatrix} 1.0 & 0.1 & 0.1 \\ 0.1 & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 \end{pmatrix}. \tag{6.17}$$



(a) Strong receive and transmit correlation between source 1 and 2 as well as relay 1 and 2 according to Eq. (6.16).

(b) Strong receive and weak transmit correlation according to Eq. (6.17).

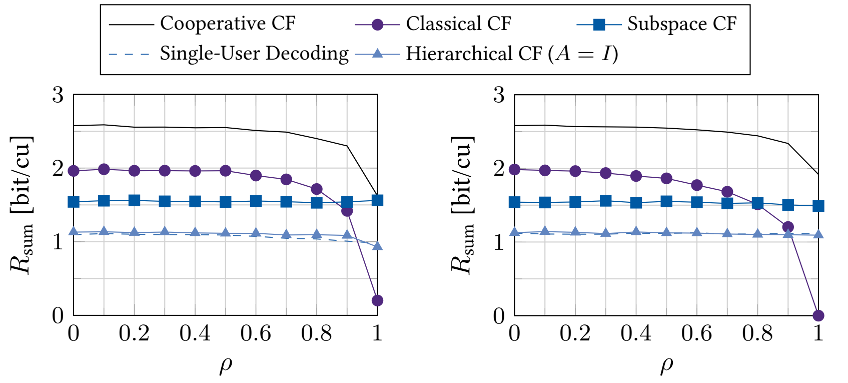
Figure 6.4: Achievable sum-rate in a correlated environment.

As one can see in Fig. 6.4, the correlation in the network plays an important role. Strong correlation even between only two source or relay nodes significantly decreases the achievable sum-rate of the classical CF scheme compared to the case without correlation in Fig. 6.3. This is due to the fact that a strong correlation increases the probability that the relay nodes will decode a linear dependent equation. Subspace CF and single-user decoding are robust against correlation and do not show a decrease in performance. In fact, subspace CF shows a superior performance for high correlation.

In order to get a better impression on the dependency between achievable sum-rate and correlation, we plot one in dependence of the other in Fig. 6.5. Again, we assume a $3 \times 3 \times 1$ relay network and generate 10 000 channel matrices per correlation value at an SNR of 10 dB. We consider two scenarios:

- receive correlation between relay 1 and relay 2, i.e.,

$$R_R = \begin{pmatrix} 1.0 & \rho & 0.1 \\ \rho & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 \end{pmatrix} \quad R_T = \begin{pmatrix} 1.0 & 0.1 & 0.1 \\ 0.1 & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 \end{pmatrix}, \tag{6.18}$$



(a) Achievable sum-rate with transmit correlation ρ according to Eq. (6.19).

(b) Achievable sum-rate with receive correlation ρ according to Eq. (6.18).

Figure 6.5: Achievable sum-rate at SNR = 10 dB depending on the correlation ρ.

- transmit correlation between source 1 and source 2, i.e.,

$$R_R = \begin{pmatrix} 1.0 & 0.1 & 0.1 \\ 0.1 & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 \end{pmatrix} \quad R_T = \begin{pmatrix} 1.0 & \rho & 0.1 \\ \rho & 1.0 & 0.1 \\ 0.1 & 0.1 & 1.0 \end{pmatrix}. \tag{6.19}$$

One can see in Fig. 6.5 that a large correlation factor decreases the performance of classical CF drastically. Furthermore, it can be seen that the non-cooperative schemes show a stable performance.

One can see from the figures that the choice of the lattice equation that is decoded at the relay nodes is essential for the performance of the complete network. Especially, correlated signals have a huge impact on the achievable sum-rate. The results can be summarized as follows:

- In networks with several hops, where re-transmissions from source to sink nodes are not applicable, one should use subspace CF. The decrease of achievable sum-rate compared to classical CF is approximately 1 bit/cu for an SNR of 10 dB.

- Hierarchical CF can increase the performance of subspace CF as well as single-user decoding in the high-SNR regime but has high computational complexity.

- Spatial correlation plays an important role on the performance of classical CF and can reduce the achievable sum-rate to $0\,\mathrm{bit/cu}$. Subspace CF, hierarchical CF as well as single-user decoding are robust against correlation.
- If a high spatial correlation between several nodes in the network occurs, subspace CF will show a superior performance.

We can conclude that subspace CF should be preferred over the other schemes if spatial correlation is an issue. It has no increase in computational complexity compared to classical CF. It is robust against spatial correlation and has a good performance in almost all scenarios.

Part III

Untrusted Relay Channels

Chapter 7

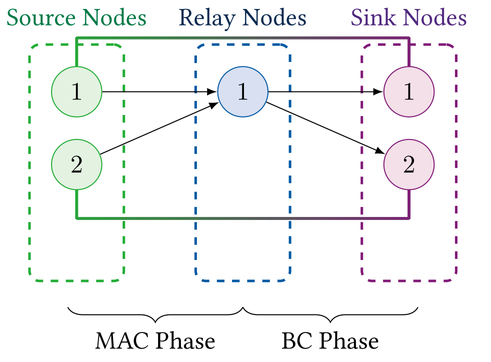
Two-Way Untrusted Relay Channel

In this chapter, we investigate the two-way relay channel as depicted in Fig. 7.1. The two-way relay channel can be derived from the general system model introduced in Section 1.3. Physically, there are only three nodes. Source node 1 and sink node 1 are physically the same, and therefore sink node 1 has side information about the transmitted message. The same holds for source node 2 and sink node 2. Since the source and sink nodes are identical, we omit the suffix "source" or "sink" in the following. All nodes allow half-duplex transmission.

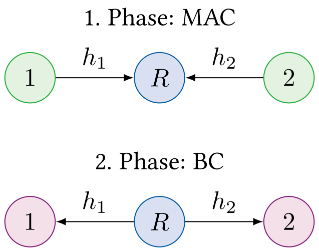
Nodes 1 and 2 have messages for each other but have no direct connection. They transmit messages with the help of a relay in two phases. In practice, it often happens that such a transmission between two nodes has to use a relay that cannot be trusted. Therefore, the messages have to be encoded at nodes 1 and 2 such that the relay cannot decode the messages separately. Since the relay is the intended receiver of the messages and the eavesdropper at the same time, we need a relaying strategy which does not decode the messages separately. Further, we need to achieve a transmission rate larger than the MAC capacity, otherwise the relay might be able to decode the single messages. A relaying scheme which provides the necessary features is compute-and-forward as introduced in Chapter 3. In the following, we assume that the relay is an honest but curious eavesdropper. It does not interfere with the data transmission and sticks to the communication protocol but tries to eavesdrop the data sent by the source nodes.

7.1 Channel Model

In the first phase, the relay receives a superposition $y_r \in \mathbb{R}^n$ of both signals $x_1 \in \mathbb{R}^n$ and $x_2 \in \mathbb{R}^n$ from nodes 1 and 2, respectively, and tries to decode a linear combination of the original messages. In the second phase, this linear combination is encoded with a capacity achieving code and sent to nodes 1 and 2 simultaneously. The links are AWGN channels with fading coefficients h_1 and h_2. We assume that the channels are reciprocal and constant over both phases. Each node has a transmit power constraint $\|x_j\|^2 \leq nP$, where n is the number of channel uses and $j \in \{1, 2, r\}$. The noise is distributed according to



(a) System model derived from the general model in Section 1.3.



(b) Classical system model representation.

Figure 7.1: System model of a two-way relay channel.

a Gaussian distribution with $z_j \sim \mathcal{N}(0, I_n)$ and $j \in \{1, 2, r\}$. We can write the channel model for the first phase as follows

$$y_r = h_1 x_1 + h_2 x_2 + z_r. \tag{7.1}$$

And for the second phase, it is given by

$$y_1 = h_1 x_r + z_1, \tag{7.2a}$$
$$y_2 = h_2 x_r + z_2. \tag{7.2b}$$

In order to ensure reliable transmission, we have to fulfill

$$\lim_{n\to\infty} \Pr(\hat{W}_i \neq W_i) = 0, \quad i = 1, 2 \tag{7.3}$$

at nodes 1 and 2, where W_i is the random variable modeling the transmitted message from node i, and $\hat{W}_i$ is the random variable modeling the estimate of the former at the receiver. The received signal at the relay will be modeled by the random variable Y_r. The secrecy requirement follows from Eq. (4.15), i.e.,

$$\lim_{n\to\infty} \frac{1}{n} I(W_1, W_2; Y_r) = 0. \tag{7.4}$$

For the analysis of the secrecy condition, we will also use the following alternative representation of Eq. (7.4)

$$\lim_{n\to\infty} H(W_1, W_2) = \lim_{n\to\infty} H(W_1, W_2 \mid Y_r). \tag{7.5}$$

The achievable secrecy rate R_s can be obtained by choosing $H(W_1) = 2^{nR_s}$ and $H(W_2) = 2^{nR_s}$ such that

$$\lim_{n\to\infty} \frac{1}{n} \log_2 H(W_1) = R_s, \tag{7.6a}$$

$$\lim_{n\to\infty} \frac{1}{n} \log_2 H(W_2) = R_s. \tag{7.6b}$$

7.2 Encoding

Each node i with $i \in \{1, 2\}$ chooses a message $w_i \in \mathbb{F}_p^k$ i.i.d. from a uniform distribution over the index set $\{1, 2, \dots, 2^{\lfloor nR_s \rfloor}\}$. For simplicity, we assume an equal message length k at all nodes. Messages with length smaller k will be zero-padded.

Each message is mapped to a lattice codeword in $\mathcal{L} = \Lambda_F \cap \mathcal{V}_C$, where the second moment of Λ_C equals P, i.e., the power constraint is satisfied. In order to fulfill the secrecy requirement in Eq. (7.4), some additional effort is required. Each node i with $i \in \{1, 2\}$ uses the same codebook $\mathcal{L} = \Lambda_F \cap \mathcal{V}_C$ with $|\Lambda_F \cap \mathcal{V}_C| = 2^{\lfloor n(R_s+R_d) \rfloor}$. Like wiretap codes, this codebook is randomly binned into several bins, where each bin contains $2^{\lfloor nR_d \rfloor}$ codewords. The secret message w_i is mapped to the bins. The actual transmitted codeword t_i is chosen from that bin according to a uniform distribution.

Further, we add some dither u_i that is uniformly distributed over $\mathcal{V}_C$ and known by the relay. This dithering provides stochastic properties of the transmitted signal that are needed to achieve the compute-and-forward rate [40]. It was shown in [40, Appendix C] that this dither might be chosen in a deterministic way. In order to ensure that the transmitted signal fulfills the power constraint, we build the modulo with respect to the coarse lattice. We get the following n-dimensional transmit vector at node i

$$x_i = [t_i + u_i] \bmod \Lambda_C. \tag{7.7}$$

With this encoding, we obtain the following rates:

- R_d is the rate of the randomly chosen messages within a bin,
- R_s is the secret message rate, and
- $R_s + R_d = \frac{1}{n} \log_2 \frac{\operatorname{Vol}(\mathcal{V}_C)}{\operatorname{Vol}(\mathcal{V}_F)}$ is the transmit rate of the user nodes.

7.3 Relay Strategy

The relay receives the random vector

$$y_r = h_1 x_1 + h_2 x_2 + z_r, \tag{7.8}$$

where $z_r \sim \mathcal{N}(0, I_n)$ is AWGN. It tries to decode a linear combination of transmitted lattice points with coefficient vector $a = (a_1, a_2)'$ using the compute-and-forward framework, i.e.,

$$\hat{y}_r = [a_1 t_1 + a_2 t_2] \bmod \Lambda_C. \tag{7.9}$$

The decoding is successful if the transmission rate of nodes 1 and 2 is below the computation rate [40, Theorem 2], i.e.,

$$R_{CF} = \frac{1}{2} \log_2^+ \left(\left(\|a\|^2 - \frac{(h'a)^2 P}{1 + P\|h\|^2} \right)^{-1} \right) \tag{7.10}$$

with $h = (h_1, h_2)'$. We assume that the relay will choose the coefficient vector a such that the computation rate is maximized. This can be done by several algorithms [109, 44].

For the first phase, we get the following rate constraint

$$R_s + R_d \leq R_{CF}. \tag{7.11}$$

In the second phase, the relay maps the decoded lattice point to an index of the set $\{1, 2, \ldots, 2^{nR_r}\}$ and uses a capacity achieving code to encode and transmit to nodes 1 and 2 with rate R_r.

7.4 Decoding

Nodes 1 and 2 receive an index of the message set $\{1, 2, \ldots, 2^{nR_r}\}$. They can decode as long as the transmission rate from the relay to the destination is less than the point-to-point capacity of the channels, which is a special case of [81],

$$R_r \leq \min \left\{ \tfrac{1}{2} \log_2(1 + P h_1^2), \tfrac{1}{2} \log_2(1 + P h_2^2) \right\}. \tag{7.12}$$

If nodes 1 and 2 decode successfully, they know the lattice point $\hat{y}_r \in \Lambda_F \cap \mathcal{V}_C$ transmitted by the relay. Each user can obtain the lattice point of the other user by subtracting its own lattice point. A lattice point uniquely determines the underlying message. Hence, each user gets the message of the other user.

From phases one and two, we obtain a rate constraint for nodes 1 and 2 given by

$$R_s + R_d \leq \min\{R_{CF}, R_r\} = R_{CF}. \tag{7.13}$$

This results in the following constraint for the secure communication rate R_s

$$R_s \leq R_{CF} - R_d. \tag{7.14}$$

We get the same constraint for both nodes because both use the same nested lattice chain. In order to ensure that the relay will not get any information about individual messages, the rate R_d has to be chosen appropriately. This will be addressed in the next section.

7.5 Achievable Secrecy Rate Region

In this section, we provide an achievable secrecy rate region and give the proof of achievability.

7.1 Theorem (Achievable Secrecy Rate). Assume a two-way relay channel as shown in Fig. 7.1 with fading coefficients h_1 and h_2. Each node has a transmit power constraint $\|x_j\|^2 \leq nP$ with $j \in \{1, 2, r\}$. Then, the weak secrecy rate region is given by

$$2R_s \leq \max\left\{0, 2R_{CF} - \tfrac{1}{2}\log_2(1 + Ph_1^2 + Ph_2^2)\right\},$$

where

$$R_{CF} = \max_{a \neq 0} \frac{1}{2} \log_2^+ \left(\left(\|a\|^2 - \frac{(h'a)^2 P}{1 + P\|h\|^2} \right)^{-1} \right)$$

is the computation rate of compute-and-forward. ◁

Proof. An overview of all random variables is provided in Table 7.1. For the achievability of the secrecy rate region, we must show that the weak secrecy condition in Eq. (7.4) holds, i.e.,

$$\lim_{n\to\infty} \tfrac{1}{n} I(W_1, W_2; Y_r \mid U_1, U_2) = 0. \tag{7.15}$$

Table 7.1: Variable overview.

variable	distribution	comment
W_i	$\sim \mathcal{U}(\{1,2,\dots,2^{nR_s}\})$	message of length k
T_i	$\sim \mathcal{U}(\Lambda_F \cap \mathcal{V}_C)$	encoded message in $\mathcal{L}$
U_i	$\sim \mathcal{U}(\mathcal{V}_C)$	dither
X_i	$\sim \mathcal{U}(\mathcal{V}_C)$	transmit vector of node i
Z_r	$\sim \mathcal{N}(0, I_n)$	AWGN
Y_r	continuous	received vector at relay

For the ease of readability, we omit the condition on U_1 and U_2 in the following, since it is present for every mutual information and entropy and does not change the equations. The condition in Eq. (7.15) is equivalent to

$$\lim_{n\to\infty} \frac{1}{n} H(W_1, W_2) = \lim_{n\to\infty} \frac{1}{n} H(W_1, W_2 \mid Y_r). \tag{7.16}$$

We can explicitly write the left-hand side and get

$$2 \cdot R_s = \lim_{n\to\infty} \tfrac{1}{n} H(W_1, W_2 \mid Y_r). \tag{7.17}$$

Now, we need a lower bound on the right-hand side.

$$\begin{aligned}
&\lim_{n\to\infty} \tfrac{1}{n} H(W_1, W_2 \mid Y_r) && (7.18)\\
&= \lim_{n\to\infty} \tfrac{1}{n} [H(W_1, W_2 \mid X_1, X_2, Y_r) && (7.19)\\
&\qquad + H(X_1, X_2 \mid Y_r)\\
&\qquad - H(X_1, X_2 \mid W_1, W_2, Y_r)]\\
&\geq \lim_{n\to\infty} \tfrac{1}{n} [H(X_1, X_2 \mid Y_r) && (7.20)\\
&\qquad - H(X_1, X_2 \mid W_1, W_2, Y_r)]\\
&\overset{a)}{\geq} \lim_{n\to\infty} \tfrac{1}{n} [H(X_1, X_2 \mid Y_r) - n\delta(n)] && (7.21)\\
&= \lim_{n\to\infty} \tfrac{1}{n} [H(X_1, X_2 \mid Y_r) && (7.22)\\
&\qquad - H(X_1, X_2) + H(X_1, X_2) - n\delta(n)]\\
&= \lim_{n\to\infty} \tfrac{1}{n} [H(X_1, X_2) - I(X_1, X_2; Y_r) - n\delta(n)] && (7.23)
\end{aligned}$$

$$\overset{b)}{\geq} \lim_{n\to\infty} \tfrac{1}{n}[H(X_1, X_2) \\ - n \cdot \tfrac{1}{2}\log_2(1 + Ph_1^2 + Ph_2^2) - n\delta(n)] \tag{7.24}$$

$$\overset{c)}{=} 2 \cdot (R_s + R_d) - \tfrac{1}{2}\log_2(1 + Ph_1^2 + Ph_2^2) \tag{7.25}$$

We used the following arguments:

a) We used Fano's inequality to bound the last term. The size of each bin is kept small enough such that the eavesdropper can determine X_1, X_2 from the received signals for given W_1, W_2.

b) We rewrite the mutual information in terms of entropy and obtain

$$I(X_1, X_2; Y_r) = h(Y_r) - h(Y_r \mid X_1, X_2). \tag{7.26}$$

In the following, we use the fact that the conditional entropy is smaller than the unconditioned entropy. Furthermore, the normal distribution maximizes the entropy for an average power constraint. This yields

$$\begin{aligned} h(Y_r) &= \sum_{k=1}^{n} h(Y_{r,k} \mid Y_{r,k-1}, Y_{r,k-2}, \ldots, Y_{r,1}) \\ &\leq \sum_{k=1}^{n} h(Y_{r,k}) \\ &\leq n \cdot \tfrac{1}{2}\log_2(2\pi e(Ph_1^2 + Ph_2^2 + 1)). \end{aligned} \tag{7.27}$$

If the transmitted signals X_1 and X_2 are known, the only remaining uncertainty in the received signal Y_r is due to the noise Z_r. The components of the noise vector are i.i.d. according to a normal distribution, which results in the following entropy

$$\begin{aligned} h(Y_r \mid X_1, X_2) &= h(Z_r) \\ &= \sum_{k=1}^{n} h(Z_{r,k}) \\ &= n \cdot h(Z_{r,k}) \\ &= n \cdot \tfrac{1}{2}\log_2(2\pi e). \end{aligned} \tag{7.28}$$

When we combine Eqs. (7.27) and (7.28), we get an upper bound on the mutual information $I(X_1, X_2; Y_r) \leq n \cdot \frac{1}{2}\log_2(1 + Ph_1^2 + Ph_2^2)$.

c) We used the fact that X_1 and X_2 are i.i.d.

Since all rates are symmetric and $R_s + R_d \leq R_{\text{CF}}$, we get the following weak secrecy rate

$$2 \cdot R_s \leq 2 \cdot R_{CF} - \tfrac{1}{2}\log_2(1 + Ph_1^2 + Ph_2^2). \tag{7.29}$$

This concludes the proof. □

7.6 Discussion

In this section, we discuss and illustrate Theorem 7.1. It is interesting to note that the achievable secrecy rate is the difference between the achievable compute-and-forward sum-rate and the sum-capacity of the MAC. This means that we get a secrecy rate greater than zero if the compute-and-forward rate region is larger than the MAC capacity region. This is illustrated in Fig. 7.2, where we plotted the achievable rate regions for the channel coefficients $(0.9, 0.8)'$ and a transmit SNR of $P/\sigma^2 = 10\,\text{dB}$. The dots in Fig. 7.2 mark the corner points of the achievable compute-and-forward rate regions for different coefficient vectors. The solid line illustrates the border of the MAC capacity region. As one can see, a rate greater than the MAC sum-capacity is only achievable for a single coefficient vector, in this example $a = (1, 1)'$. In general, there is at most one point or there are several linear dependent points outside the MAC capacity region. Otherwise, the relay would be able to decode the single messages by solving a linear equation system, which contradicts the MAC capacity definition. From this illustration, we see that it is only possible to transmit secure messages via the relay, if nodes 1 and 2 transmit at a higher rate than the MAC capacity. This ensures that the relay cannot decode the single messages but a linear combination if the compute-and-forward rate is higher than the MAC sum-capacity.

The achievable computation rate mainly depends on the channel coefficients because the compute-and-forward framework tries to approximate the real-valued channel coefficients with integer-valued network coding coefficients. This means that there does not always exist a network coding coefficient vector with an achievable computation rate which is outside of the MAC capacity region. In Figs. 7.3 and 7.4, we show the achievable secrecy rate for different channel realizations. One can see that the achievable secrecy rate reaches its highest values if the channel coefficients are equal. This is not surprising because the computation rate of compute-and-forward reaches its highest values for

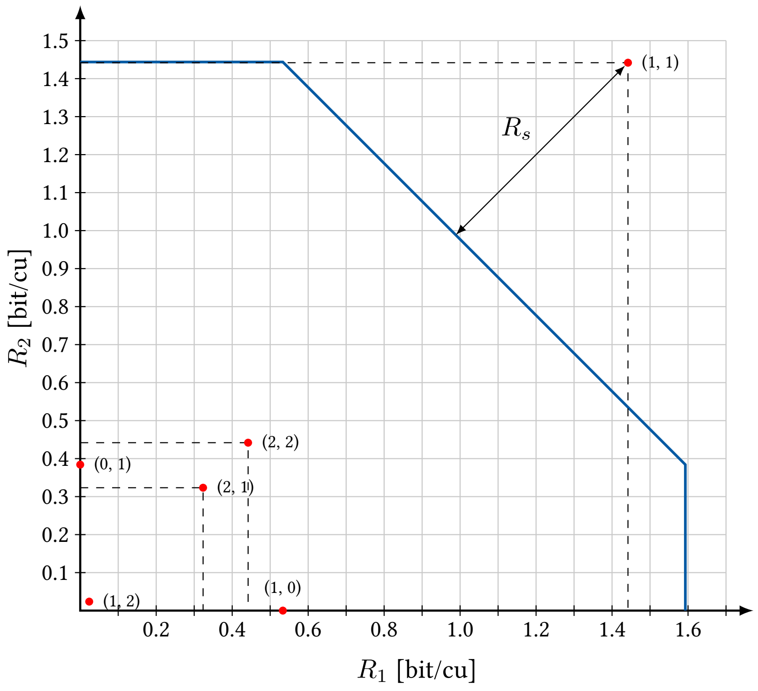


Figure 7.2: MAC capacity region (blue line) with achievable compute-and-forward rates for different coefficient vectors a (red dots) for $h = (0.9, 0.8)'$ and $P/\sigma^2 = 10\,\mathrm{dB}$.

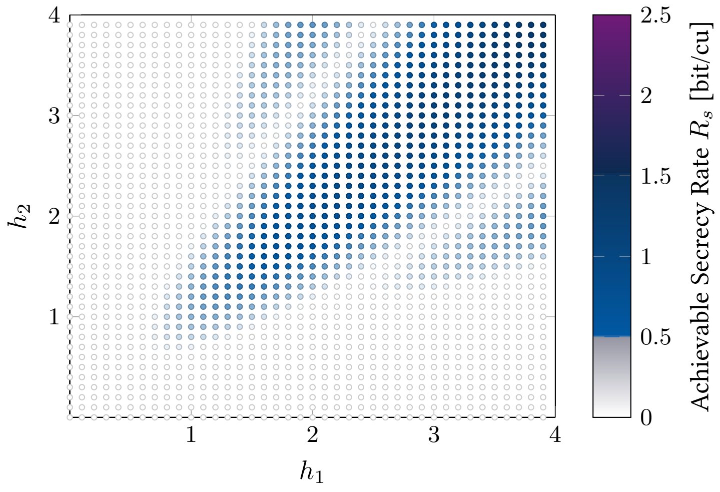


Figure 7.3: Existence of a compute-and-forward rate tuple outside of the MAC capacity region. $P/\sigma^2 = 5\,\text{dB}$.

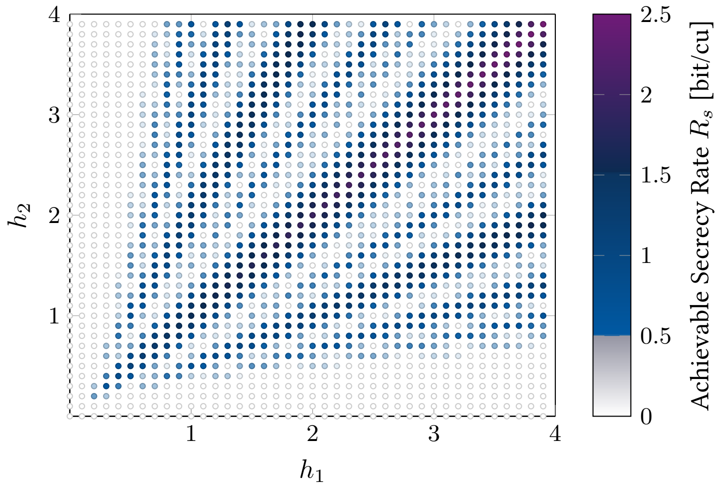


Figure 7.4: Existence of a compute-and-forward rate tuple outside of the MAC capacity region. $P/\sigma^2 = 20\,\text{dB}$.

equal channel coefficients (see Section 3.3). One can also see that small channel coefficient values do not achieve positive secrecy rates. This behavior can be compensated in part by adjusting the power allocation such that the effective channel coefficients are close to each other. Unfortunately, if we assume that the channel coefficients are distributed according to a standard Gaussian distribution, i.e., $h_i \sim \mathcal{N}(0,1)$, then small channel coefficients occur with higher probability. This might be a depressing result, but we can improve the performance by optimizing the transmit power and introducing multiple antennas. We will discuss that in a more general setup in Chapter 8.

In the following, we compare the performance of our scheme to schemes presented in the literature. The two-way relay channel is often modeled as a wiretap channel where the second user is helping the first user by jamming the eavesdropper, i.e., the relay [56, 60]. This differs from our work because we assume that both users transmit a secure message simultaneously. In literature, the two-way relay channel is often investigated without fading, and those results cannot directly be extended to fading channels. However, in order to be able to compare our scheme to existing schemes, we relax our model to match the one assumed in [56]. In this case, user 1 transmits a secure message to user 2 via an untrusted relay. User 2 helps by sending a jamming signal to the relay. The main difference to our original model is the fact that the message of the second user does not need to be secure. Further, we assume that the channel coefficients are equal and set to 1. With this relaxed model, we get the following corollary, which follows directly from Theorem 7.1.

7.2 Corollary. Assume a two-way relay channel with fading coefficients $h_1 = h_2 = 1$ and equal power constraint P. The following secrecy rate is achievable with a cooperative jammer

$$R_s \le \max\left\{0, \frac{1}{2}\log_2\left(\frac{1}{2}+P\right) - \frac{1}{2}\log_2\left(1+\frac{P}{1+P}\right)\right\} \triangleq R_s^{\mathrm{CF}}. \tag{7.30}$$

◁

We compare this result to several results available in the literature. The first approach is a scheme by He and Yener, which provides a weak secrecy result [56], i.e.,

$$R_s \le \max\{0, \tfrac{1}{2}\log_2(\tfrac{1}{2}+P) - 1\} \triangleq R_s^{\mathrm{HS}}. \tag{7.31}$$

This result is extended in [59] for strong secrecy. The achievable secrecy rate is shown to be the same as for weak secrecy by utilizing a hash function. Please note that we can extend our result as well with a hash function to get a result

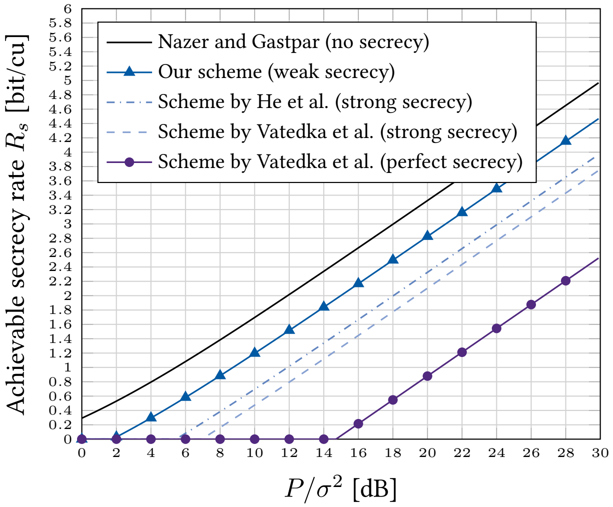


Figure 7.5: Achievable secrecy rate of the two-way relay channel with $h = (1,1)'$ and $\sigma^2 = 1$ for different schemes. The second user works as a cooperative jammer.

for strong secrecy. The second scheme is provided by Vatedka et al. in [76]. The achievable secrecy rates are

$$R_s \leq \max\{0, \tfrac{1}{2}\log_2(P) - 1 - \log_2(e)\} \triangleq R_s^{\mathrm{VP}} \tag{7.32}$$

and

$$R_s \leq \max\{0, \tfrac{1}{2}\log_2(\tfrac{1}{2} + P) - \log_2(2e)\} \triangleq R_s^{\mathrm{VS}} \tag{7.33}$$

for perfect and strong secrecy, respectively. Observe that $R_{CF} \geq R_s^{\mathrm{CF}} \geq R_s^{\mathrm{HS}} \geq R_s^{\mathrm{VS}} \geq R_s^{\mathrm{VP}}$. The comparison is visualized in Fig. 7.5.

Chapter 8

Multi-Way Untrusted Relay Channel

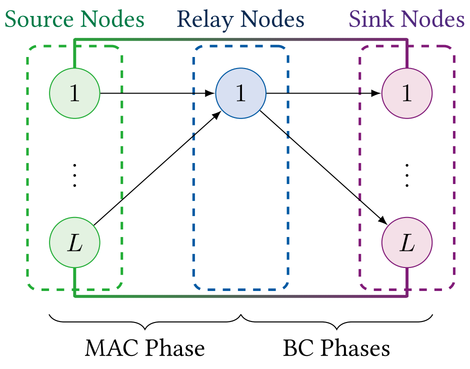
In this chapter, we extend the model of the two-way relay channel from Chapter 7 to the L-user relay channel as depicted in Fig. 8.1a. Each user node ℓ with $\ell \in \{1, 2, \ldots, L\}$ has a message $w_\ell \in \mathbb{F}_p^k$, which it wants to broadcast to all other user nodes. The user nodes do not have direct connections and their communication is supported by relay R. They exchange the messages within L phases. In the first phase, all user nodes transmit their messages to the relay, which tries to decode $L - 1$ linear combinations of the messages (MAC phase). The relay sends these linear combinations in the remaining $L - 1$ phases to the user nodes (BC phases). After all L phases, the user nodes have $L - 1$ linear combinations and their own message. If we assume that these linear combinations are linearly independent, the user nodes can solve a system of linear equations to get the estimates $\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_L$ of all messages.

In order to ensure a secret communication between all nodes, the weak secrecy condition in Eq. (4.15) has to be fulfilled. The achievable secrecy rate is defined in Definition 4.3.

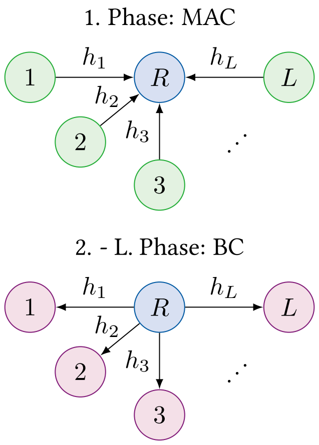
Further, we allow the nodes to have multiple antennas. Depending on the number of antennas at the nodes, we differentiate between three scenarios:

- *single-input multiple-output (SIMO):* single antenna at the source nodes and multiple antennas at the relay,
- *single-input single-output (SISO):* single antenna at the source nodes and the relay, and
- *multiple-input single-output (MISO):* multiple antennas at the source nodes and single antenna at the relay.

First, we will focus on the SIMO case because we will show later that the results for the SISO and the MISO case can be derived from the results of the SIMO case. We will not investigate the multiple-input multiple-output (MIMO) case in this thesis. This is left for further research.



(a) System model derived from the general model in Section 1.3.



(b) Classical system model representation.

Figure 8.1: System model of a multi-way relay channel.

8.1 System Model

In the following, we investigate the SIMO channel, i.e., only the relay is equipped with multiple antennas. The system model for the first phase, where the user nodes transmit their messages to the relay, is shown in Fig. 8.2. Each user node ℓ has a message w_ℓ, which it maps with an encoder

$$\mathcal{E}_\ell : \mathbb{F}_p^k \to \mathbb{R}^n \tag{8.1}$$

to a real-valued vector signal, which is transmitted over the channel. The relay uses a pre-processing matrix B to get $L-1$ different signals $\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_{L-1}$ from its $\eta_R \geq L-1$ antennas. Each signal $\tilde{y}_\ell$ is processed by a decoder $\mathcal{D}_\ell$ to get an estimate of a linear combination of lattice codewords. We will elaborate on the details in the following subsections.

8.1.1 MAC Phase

Let $x_\ell \in \mathbb{R}^n$ denote the transmit signal of user node ℓ with $\ell \in \{1, 2, \dots, L\}$, in the first phase. Each user node has a transmit power constraint $\|x_\ell\|^2 \leq nP$. We

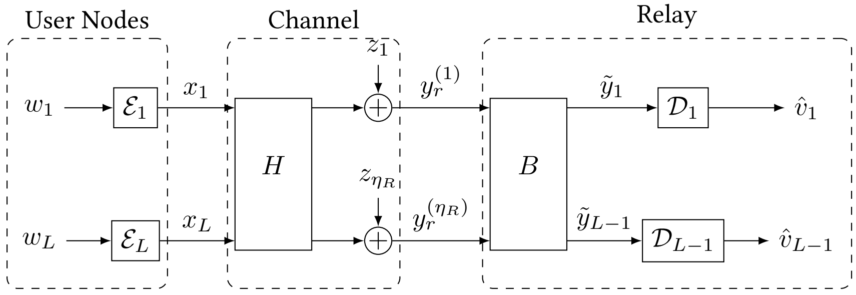


Figure 8.2: System model of a SIMO L-user relay channel (MAC phase).

assume that the relay is equipped with η_R antennas and therefore receives η_R signals $y_r^{(1)}, y_r^{(2)}, \dots, y_r^{(\eta_R)}$. The channel is an AWGN channel with quasi-static block flat fading and is characterized by the channel matrix $H \in \mathbb{R}^{\eta_R \times L}$, whose entries h_{ij} are the fading coefficients from the j-th user to the i-th antenna at the relay. Note that the i-th row of H, denoted by h_i, represents the channel from user i to the relay. The channel model for the first phase is then given by

$$Y = HX + Z, \tag{8.2}$$

where $X \in \mathbb{R}^{L \times n}$ is a matrix whose ℓ-th row is the transpose transmit vector x'_ℓ of user ℓ. Further, $Y \in \mathbb{R}^{\eta_R \times n}$ is a matrix whose i-th row represents the received data stream $y_r^{(i)}$ at the i-th antenna, and $Z \in \mathbb{R}^{\eta_R \times n}$ is white Gaussian noise. The rows of Z are denoted by z_i and are i.i.d. according to a normal distribution with zero mean and unit variance, i.e., $z_i \sim \mathcal{N}(0, I_n)$.

The relay decodes $L-1$ linear combinations of the original messages and encodes them with a capacity achieving code. Then, the $L-1$ codewords are sent to all users in the remaining $L-1$ phases simultaneously. For these phases, we have a BC and we assume reciprocal channels, which are constant over all L phases. Therefore, the rate constraints for the last $L-1$ phases are given by the capacity of the individual point-to-point channels. Due to the uplink-downlink duality for reciprocal channels and equal power constraints [83, Chapter 10.3], these are always larger or equal to the MAC rate region, and the first phase will be the limiting one. Therefore, we will focus on that phase for deriving the achievable rate of the whole system.

8.1.2 BC Phases

Let $x_r^{(1)}, x_r^{(2)}, \dots, x_r^{(L)} \in \mathbb{R}^n$ denote the transmit signals of the relay in phases 1 to L, respectively. The relay has a transmit power constraint $||x_r^{(i)}||^2 \leq nP$ for all $i \in \{1, 2, \dots, L\}$. Please note that the relay does not send anything in the first phase, and therefore $x_r^{(1)} = 0$. We assume that all channels are reciprocal and constant over all L phases. Each user node ℓ receives a signal

$$y_\ell^{(i)} = \omega_r' h_\ell x_r^{(i)} + z_\ell^{(i)} \tag{8.3}$$

in the i-th phase, where ω_r is the beamforming vector at the relay, and $z_\ell^{(i)} \sim \mathcal{N}(0, I_n)$ is white Gaussian noise. Please note that the user nodes do not receive anything in the first phase, and therefore $y_\ell^{(1)} = 0$ because we assume no direct connection between the user nodes.

8.1.3 Channel State Information

The relay needs to know all channels between itself and the user nodes. The user nodes only need to know the effective channel between themselves and the relay, i.e., user ℓ needs to know $\omega_r' h_\ell$.

8.2 Encoding

Each user node ℓ chooses a message $w_\ell \in \mathbb{F}_p^k$ i.i.d. from a uniform distribution over the index set $\{1, 2, \dots, 2^{\lfloor nR_s \rfloor}\}$. For simplicity, we assume an equal message length k for all users, which implies an equal secrecy rate for all users. If this is not the case, we would have to extend the nested lattice code chain as described in [40]. Each message is mapped to a lattice codeword in $\mathcal{L} = \Lambda_F \cap \mathcal{V}_C$, where the second moment of Λ_C equals P, i.e., the power constraint is satisfied. Each user node ℓ uses the same codebook $\mathcal{L} = \Lambda_F \cap \mathcal{V}_C$ with $|\Lambda_F \cap \mathcal{V}_C| = 2^{\lfloor n(R_s+R_d) \rfloor}$. Similar to wiretap codes, this codebook is randomly binned into $2^{\lfloor nR_s \rfloor}$ bins, where each bin contains $2^{\lfloor nR_d \rfloor}$ codewords. The secret message w_ℓ is mapped to the bins, and the lattice codeword t_ℓ is chosen from that bin according to a uniform distribution. Further, we add some dither u_ℓ that is uniformly distributed over $\mathcal{V}_C$ and known by the relay. This dithering provides the statistical properties of the transmitted signal that are needed to achieve the compute-and-forward rate [40]. In order to make sure that the transmit signal fulfills the power constraint, we build the modulo with respect to the coarse lattice. We get the following n-dimensional transmit vector at node ℓ

$$x_\ell = [t_\ell + u_\ell] \bmod \Lambda_C. \tag{8.4}$$

With this encoding scheme, we get the following rates:

- R_d is the rate of the randomly chosen messages within a bin,
- R_s is the secret message rate, and
- $R_s + R_d = \frac{1}{n}\log_2 \frac{\mathrm{Vol}(\mathcal{V}_C)}{\mathrm{Vol}(\mathcal{V}_F)}$ is the transmit rate of the user nodes.

Note that compute-and-forward implies the same rate at all L user nodes because a maximum rate equal to the computation rate is achievable. It will turn out by the secrecy analysis that the information leakage is symmetric for all L user nodes. Therefore, $R_s = R_{s_1} = R_{s_2} = \cdots = R_{s_L}$.

8.3 Relay Strategy

The relay uses compute-and-forward [40] as relaying strategy and tries to decode $L-1$ linear combinations $\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_{L-1}$ of the transmitted lattice codewords as shown in Fig. 8.2. It uses a preprocessing matrix B to get the optimal signals prior to decoding. The achievable computation rate is then given by [40, 46]

$$R(H, a_\ell, b_\ell) = \frac{1}{2}\log_2\left(\frac{P}{\|b_\ell\|^2 + P\|H'b_\ell - a_\ell\|^2}\right), \tag{8.5}$$

where a_ℓ is the coefficient vector for the ℓ-th linear combination, and b_ℓ is the preprocessing vector corresponding to the ℓ-th row in B. The optimal preprocessing matrix for a given coefficient matrix $A = (a'_1, a'_2, \ldots, a'_{L-1})'$ and the resulting rates have been derived in [46]. In the following, we use these results and provide them for completeness.

The optimal preprocessing matrix is given by

$$B = AH'(HH' + \tfrac{1}{P}I_{\eta_R})^{-1}. \tag{8.6}$$

When we plug Eq. (8.6) into Eq. (8.5), we get

$$R(H, a_\ell) = -\frac{1}{2}\log_2(a'_\ell V D V' a_\ell), \tag{8.7}$$

where $V \in \mathbb{R}^{L\times L}$ is the right eigenmatrix of H, and $D \in \mathbb{R}^{L\times L}$ is a diagonal matrix with elements

$$D_{ii} = \begin{cases} \frac{1}{P\lambda_i+1} & i \leq \operatorname{rank}(H) \\ 1 & i > \operatorname{rank}(H) \end{cases}, \tag{8.8}$$

where λ_i is the i-th eigenvalue of $H'H$.

All $L-1$ linear combinations have to be decodable at the relay in order to allow all L users to decode all original messages. Therefore, the resulting achievable rate of all L users is

$$R_{CF} = \min_{\ell\in\{1,2,\ldots,L-1\}} R(H, a_\ell). \tag{8.9}$$

In order to get the highest possible rates, we need to find a set of full-rank coefficient matrices $A_\ell \in \mathbb{Z}^{L-1\times L}$. Since all L users need to be able to decode all messages, we additionally require the rows of A_ℓ to be linearly independent of any vector e_ℓ, where e_ℓ is the unit vector with a one at the ℓ-th position and zeros elsewhere. This results in the following optimization problem

$$\begin{aligned} &\max_{A_\ell} \min_{\ell\in\{1,2,\ldots,L-1\}} \left(-\frac{1}{2}\log_2(a_\ell' V D V' a_\ell)\right) \\ &\text{s.t. } \operatorname{rank}(A_\ell) = L \text{ for all } \ell \in \{1,2,\ldots,L\} \end{aligned} \tag{8.10}$$

with

$$A_\ell = \begin{pmatrix} a_1' \\ \vdots \\ a_{L-1}' \\ e_\ell' \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,L} \\ \vdots & \vdots & \ddots & \vdots \\ a_{L-1,1} & a_{L-1,2} & \cdots & a_{L-1,L} \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

There are several algorithms which find the best coefficient vector, e.g., [109, 42, 44, 45]. These algorithms need to be extended to fulfill the following additional constraints:

- for all $\ell \in \{1,2,\ldots,L\}$, all $L-1$ coefficient vectors need to be linearly independent of e_ℓ.
- we need to find the $L-1$ best coefficient vectors and not only the best one.

For example, the algorithm of [44] can be easily extended to solve Eq. (8.10). We used this algorithm to provide the simulation results in Section 8.7.

In [40], it was shown that reliable communication can be achieved as long as

$$R_s + R_d \leq R_{CF}. \tag{8.11}$$

In the second phase, the relay maps all decoded linear combinations $\hat{v}_\ell \in \Lambda_F \cap \mathcal{V}_C$ to an index of the set $\{1, 2, \ldots, 2^{\lfloor nR_r \rfloor}\}$ and uses a capacity achieving code to encode these messages. Further, it uses an optimal beamforming vector to transmit to the user nodes with rate

$$R_r = \max_{\|\omega_r\|^2 \leq 1} \min_{\ell \in \{1,2,\ldots,L\}} \tfrac{1}{2} \log_2(1 + P(\omega_r' h_\ell)^2), \tag{8.12}$$

where ω_r is the multicast beamforming vector at the relay. An efficient way to obtain the optimal beamforming vector can be found in [80].

8.4 Decoding at the User Nodes

In each of the $L-1$ phases, each user node receives an index of the index set $\{1, 2, \ldots, 2^{\lfloor nR_r \rfloor}\}$. The user nodes can decode as long as the transmission rate from the relay to the user is less than the point-to-point capacity of the channels, which is given in Eq. (8.12). If they decode successfully, they know the lattice point $\hat{v}_\ell \in \Lambda_F \cap \mathcal{V}_C$ that was transmitted by the relay.

User ℓ can decode the original lattice points from the other users by solving the following system of linear equations

$$A_\ell T = \tilde{V}_\ell \tag{8.13}$$

with

$$\tilde{V}_\ell = \begin{pmatrix} \hat{v}_{1,1} & \hat{v}_{1,2} & \cdots & \hat{v}_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{v}_{L-1,1} & \hat{v}_{L-1,2} & \cdots & \hat{v}_{L-1,n} \\ t_{\ell,1} & t_{\ell,2} & \cdots & t_{\ell,n} \end{pmatrix} \tag{8.14}$$

and

$$T = \begin{pmatrix} t_{1,1} & t_{1,2} & \cdots & t_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ t_{L,1} & t_{L,2} & \cdots & t_{L,n} \end{pmatrix}. \tag{8.15}$$

User ℓ already knows t_ℓ because this is its own message. Please note that the users get the lattice points without the dither because the relay already subtracted it. User ℓ obtains an estimate of all lattice points t_i for all $i \in \{1, 2, \dots, L\}$ by

$$\hat{T}_\ell = A_\ell^{-1} \tilde{V}_\ell, \tag{8.16}$$

where

$$\hat{T}_\ell = \begin{pmatrix} \hat{t}'_1 \\ \vdots \\ \hat{t}'_L \end{pmatrix} \tag{8.17}$$

contains all estimates of the transmitted lattice codewords received by user ℓ.

Having solved this with respect to $\hat{T}_\ell$, each user knows all L lattice points. If a lattice point is known, the message and the bin are known, too. Therefore, each user gets all L messages.

The rate constraint for the user nodes, given by

$$R_d + R_s \leq \min\{R_{CF}, R_r\} = R_{CF}, \tag{8.18}$$

ensures reliable communication within all L phases. From the lattice code construction in Section 8.2 and the constraint in Eq. (8.18), we get the following constraint for the weak secrecy rate R_s

$$R_s \leq R_{CF} - R_d. \tag{8.19}$$

In order to ensure that the relay will not get any information about individual messages, the rate R_d has to be chosen appropriately. This will be addressed in the next section.

8.5 Achievable Secrecy Rate Region

In this section we provide an achievable secrecy rate region for the following cases:

- *SIMO:* single antenna at the source nodes and multiple antennas at the relay,
- *SISO:* single antenna at the source nodes and the relay, and
- *MISO:* multiple antennas at the source nodes and single antenna at the relay.

The result for the SISO case can directly be derived from the SIMO case. In Section 8.5.3, we show that the result for the MISO case can be derived from the SISO case.

8.5.1 SIMO

8.1 Theorem (Achievable Secrecy Rate). Consider a multi-way relay channel with L users and η_R antennas at the relay. The channel is characterized by the matrix H whose entries h_{ij} represent the fading coefficient from the j-th user to the i-th antenna at the relay. All nodes can only communicate via the relay and have no direct links. Each node has a transmit power constraint $\|x_i\|^2 \leq nP$ for all $i \in \{1, 2, \dots, L, r\}$. Then, the weak secrecy rate region is given by

$$LR_s \leq \max\left\{0, LR_{CF} - \tfrac{1}{2}\log_2 \det(I_{\eta_R} + PHH')\right\},$$

where

$$R_{CF} = \min_{i \in \{1,2,\dots,L-1\}} R(H, a_i)$$

is the achievable computation rate. The coefficient vector a_i is chosen according to the programming problem in Eq. (8.10). ◁

8.2 Remark. We provide a detailed interpretation and discussion of the theorem in Section 8.7. ◁

Every user node has a power constraint $\|x_\ell\|^2 \leq nP$. However, it is not always optimal to send with full power. Therefore, the transmit power for each user node needs to be optimized. We define a diagonal matrix $P_{\text{tr}} = \operatorname{diag}(\sqrt{P_1}, \sqrt{P_2}, \dots, \sqrt{P_L})$ which contains the square roots of the individual transmit powers of the user nodes. Further, we define the effective channel as $\tilde{H} = HP_{\text{tr}}$. This results in the following optimization problem

$$\max_{P_\ell \leq P} LR_{CF} - \tfrac{1}{2}\log_2 \det(I_{\eta_R} + HP_{\text{tr}}P_{\text{tr}}'H'), \tag{8.20}$$

where

$$R_{CF} = \min_{\ell \in \{1,2,\dots,L-1\}} -\tfrac{1}{2}\log_2(a_\ell' VDV' a_\ell) \tag{8.21}$$

is the achievable computation rate. The matrix V is the right eigenmatrix of $\tilde{H}$, and D is a diagonal matrix with elements

$$D_{ii} = \begin{cases} \frac{1}{\lambda_i + 1} & i \leq \operatorname{rank}(\tilde{H}) \\ 1 & i > \operatorname{rank}(\tilde{H}) \end{cases}, \tag{8.22}$$

where λ_i is the i-th eigenvalue of $\tilde{H}'\tilde{H}$.

This problem is hard to solve because of its non-linearity and non-convexity. The scope of this thesis does not include an analytic result nor the design of an efficient algorithm. For the simulations, we used a grid search algorithm to obtain the optimal power allocation.

8.5.2 SISO

The results for the SISO case can directly be derived from Theorem 8.1, where the channel matrix reduces to a vector.

8.3 Corollary. Consider a multi-way relay channel with L users and single antennas at all nodes. The channel from user ℓ to the relay is characterized by the coefficient $h_\ell \in \mathbb{R}$ with $h = (h_1, h_2, \dots, h_L)'$. All nodes can only communicate via the relay and have no direct links. Each node has a transmit power constraint $\|x_i\|^2 \le nP$ for all $i \in \{1, 2, \dots, L, r\}$. Then, the weak secrecy rate region is given by

$$LR_s \le \max\left\{0, LR_{CF} - \tfrac{1}{2}\log_2(1 + \|\tilde{h}\|^2)\right\},$$

where

$$R_{CF} = \min_{i\in\{1,2,\dots,L-1\}} \log_2^+\left(\left(\|a_i\|^2 - \frac{(\tilde{h}'a_i)^2}{1+\|\tilde{h}\|^2}\right)^{-1}\right)$$

is the achievable computation rate. Further, $\tilde{h} = \operatorname{diag}(\sqrt{P_1}, \sqrt{P_2}, \dots, \sqrt{P_L}) \cdot h$ is the effective channel, and $P_\ell \le P$ is the transmit power of user ℓ. $\triangleleft$

8.5.3 MISO

For the MISO case, we assume no cooperation at the source nodes in order to choose the beamforming vectors. Therefore, it is optimal to use maximum-ratio transmission (MRT) in the direction of the channel. This leaves us with a reduced optimization problem, where we only have to choose the optimal power allocation. The effective channel is

$$\tilde{h} = \operatorname{diag}(\sqrt{P_1}, \sqrt{P_2}, \dots, \sqrt{P_L}) \cdot (h_1'\omega_1, h_2'\omega_2, \dots, h_L'\omega_L)', \tag{8.23}$$

where h_ℓ is the channel vector from user ℓ to the relay, and ω_ℓ is the beamforming vector with $\|\omega_\ell\| \le 1$ for user ℓ. If we use MRT, we get

$$\omega_\ell = \tfrac{h_\ell}{\|h_\ell\|} \text{ and } h_\ell'\omega_\ell = \|h_\ell\|. \tag{8.24}$$

Table 8.1: Overview of variables and symbols.

variable	distribution	comment
W_i	$\sim \mathcal{U}(\{1, 2, \ldots, \lceil 2^{nR_s} \rceil\})$	secret message of length k
U_i	$\sim \mathcal{U}(\mathcal{V}_C)$	dither at node i
X_i	$\sim \mathcal{U}(\mathcal{V}_C)$	transmit vector at node i
Z_j	$\sim \mathcal{N}(0, I_n)$	AWGN at relay antenna j
Y_j	continuous	received vector at relay antenna j

The secrecy rate is now equivalent to the one in the SISO case, and we get the following corollary.

8.4 Corollary. Consider a multi-way relay channel with L users and η_T antennas at the user nodes. The relay is equipped with a single antenna. The channel from user ℓ to the relay is characterized by the vector $h_\ell \in \mathbb{R}^{\eta_T}$. All nodes can only communicate via the relay and have no direct links. Each node has a transmit power constraint $\|x_i\|^2 \leq nP$ for all $i \in \{1, 2, \ldots, L, r\}$. Then, the weak secrecy rate region is given by

$$LR_s \leq \max\left\{0, LR_{CF} - \tfrac{1}{2}\log_2(1 + \|\tilde{h}\|^2)\right\},$$

where

$$R_{CF} = \min_{i \in \{1,2,\ldots,L-1\}} \log_2^+\left(\left(\|a_i\|^2 - \frac{(\tilde{h}' a_i)^2}{1 + \|\tilde{h}\|^2}\right)^{-1}\right)$$

is the achievable computation rate. Further, the effective channel is

$$\tilde{h} = \operatorname{diag}(\sqrt{P_1}, \sqrt{P_2}, \ldots, \sqrt{P_L}) \cdot (\|h_1\|, \|h_2\|, \ldots, \|h_L\|)', \tag{8.25}$$

and $P_\ell \leq P$ is the transmit power of user ℓ. $\triangleleft$

8.6 Proof of the Achievable Secrecy Rate Region

In this section, we provide the proof of Theorem 8.1. An overview of all random variables is provided in Table 8.1.

Proof. For the achievability of the secrecy rate region, we must show that the weak secrecy condition in Eq. (4.15) holds for all signals $Y_1, Y_2, \ldots, Y_{\eta_R}$ received at all η_R relay antennas in the first phase. Since the relay knows the dither, Eq. (4.15) transforms to

$$\begin{aligned}&\lim_{n\to\infty} \tfrac{1}{n} H(W_1, W_2, \ldots, W_L \mid U_1, U_2, \ldots, U_L)\\ &= \lim_{n\to\infty} \tfrac{1}{n} H(W_1, W_2, \ldots, W_L \mid Y_1, Y_2, \ldots, Y_{\eta_R}, U_1, U_2, \ldots, U_L).\end{aligned} \tag{8.26}$$

For the ease of readability, we define the following vectors of random variables:

- $X = (X_1, X_2, \ldots, X_L)$,
- $U = (U_1, U_2, \ldots, U_L)$,
- $W = (W_1, W_2, \ldots, W_L)$,
- $Y = (Y_1, Y_2, \ldots, Y_{\eta_R})$.

With this notation, we can write Eq. (8.26) as

$$\lim_{n\to\infty} \tfrac{1}{n} H(W \mid U) = \lim_{n\to\infty} \tfrac{1}{n} H(W \mid Y, U). \tag{8.27}$$

Please note that W is independent of U, and therefore

$$H(W \mid U) = H(W). \tag{8.28}$$

The messages $W_1, W_2, \ldots, W_L$ are i.i.d. according to a uniform distribution over a discrete set with 2^{nR_s} items. Thus, we can rewrite the left-hand side of Eq. (8.27) and get

$$LR_s = \lim_{n\to\infty} \tfrac{1}{n} H(W \mid Y, U). \tag{8.29}$$

Now, we need a lower bound on the right-hand side. We define $X_\Sigma \triangleq HX$ and notice that X, X_Σ, and Y form a Markov chain $X \to X_\Sigma \to Y$. The idea to define a new random variable for the received signal without noise is inspired by the proof of Theorem 1 in [74]. We start to express $H(W \mid Y, U)$ as follows

$$\begin{aligned}&H(W \mid Y, U)\\ &\quad = H(W \mid U) - I(W; Y \mid U)\\ &\quad = H(W \mid U) - I(W; Y \mid U) + I(W; Y \mid X_\Sigma, U)\\ &\quad = H(W \mid U) - h(Y \mid U) + h(Y \mid W, U)\\ &\qquad\quad + h(Y \mid X_\Sigma, U) - h(Y \mid W, X_\Sigma, U)\end{aligned}$$

$$= H(W \mid U) - I(X_\Sigma; Y \mid U) + I(X_\Sigma; Y \mid W, U). \tag{8.30}$$

First, we calculate $I(X_\Sigma; Y \mid W, U) = H(X_\Sigma \mid W, U) - H(X_\Sigma \mid Y, W, U)$. When the messages W are known, the only uncertainty that remains in X is due to the binning. Since each user node $\ell \in \{1, 2, \ldots, L\}$ independently chooses a codeword out of a bin of size 2^{nR_d} equally likely for each secret message, we have

$$H(X_\Sigma \mid W, U) = nLR_d. \tag{8.31}$$

Further, we obtain

$$H(X_\Sigma \mid Y, W, U) \leq n\delta(n) \tag{8.32}$$

if $LR_d \geq \frac{1}{2} \log \det(I + PHH')$ due to Fano's inequality.

Next, we calculate $I(X_\Sigma; Y \mid U)$, which can be expressed in terms of entropy, i.e.,

$$I(X_\Sigma; Y \mid U) = h(Y \mid U) - h(Y \mid X_\Sigma, U). \tag{8.33}$$

We use the fact that the normal distribution maximizes the entropy for an average power constraint in order to get an upper bound on the first term. From [82, Section 3.2], we know: If X is distributed according to a normal distribution with zero-mean and covariance $\mathrm{E}[xx'] = PI_L$, then $Y = HX + Z$ is also distributed according to a normal distribution with zero-mean and covariance $\mathrm{E}[yy'] = PHH' + I_{\eta_R}$. This yields

$$\begin{aligned} h(Y \mid U) &= h(Y_{1,1}, \ldots, Y_{1,n}, \ldots, Y_{\eta_R,1}, \ldots, Y_{\eta_R,n} \mid U) \\ &= h(Y_{1,1}, \ldots, Y_{\eta_R,1}, \ldots, Y_{1,n}, \ldots, Y_{\eta_R,n} \mid U) \\ &\leq \sum_{i=1}^{n} h(Y_{1,i}, \ldots, Y_{\eta_R,i} \mid U) \\ &\leq n \cdot \tfrac{1}{2} \log_2((2\pi e)^L \det(PHH' + I_{\eta_R})). \end{aligned} \tag{8.34}$$

If X_Σ is known, the only uncertainty in the received signals $Y_1, Y_2, \ldots, Y_{\eta_R}$ is due to the noise $Z_1, Z_2, \ldots, Z_{\eta_R}$. The noise is i.i.d. according to a normal distribution, which results in the following entropy:

$$\begin{aligned} h(Y \mid X_\Sigma, U) &= h(Z_1, Z_2, \ldots, Z_{\eta_R}) \\ &= \sum_{i=1}^{n} h(Z_{1,i}, Z_{2,i}, \ldots, Z_{\eta_R,i}) \\ &= n \cdot \tfrac{1}{2} \log_2((2\pi e)^L \det(I_{\eta_R})). \end{aligned} \tag{8.35}$$

Putting Eq. (8.34) and Eq. (8.35) into Eq. (8.33) yields

$$\begin{aligned} I(X_\Sigma; Y \mid U) &\leq n \cdot \tfrac{1}{2} \log_2((2\pi e)^L \det(PHH' + I_{\eta_R})) \\ &\qquad - n \cdot \tfrac{1}{2} \log_2((2\pi e)^L \det(I_{\eta_R})) \\ &= n \cdot \tfrac{1}{2} \log_2 \frac{(2\pi e)^L \det(PHH' + I_{\eta_R})}{(2\pi e)^L \det(I_{\eta_R})} \\ &= n \cdot \tfrac{1}{2} \log_2 \det(I_{\eta_R} + PHH'). \end{aligned} \tag{8.36}$$

Putting all together, we obtain the following lower bound:

$$\begin{aligned} &\lim_{n\to\infty} \tfrac{1}{n} H(W \mid Y, U) \\ &= \lim_{n\to\infty} \tfrac{1}{n} [H(W \mid U) - I(X_\Sigma; Y \mid U) + I(X_\Sigma; Y \mid W, U)] \\ &\geq L(R_s + R_d) - \tfrac{1}{2} \log_2 \det(I_{\eta_R} + PHH'). \end{aligned} \tag{8.37}$$

With Eq. (8.37), we have a lower bound on the right-hand side of Eq. (8.29). We can use this bound to obtain the following rate constraint:

$$LR_s \leq L(R_s + R_d) - \tfrac{1}{2} \log_2 \det(I_{\eta_R} + PHH'). \tag{8.38}$$

We know from Eq. (8.18) that in order to have reliable communication, we have to ensure that $R_s + R_d \leq R_{CF}$. Therefore, we get the following weak secrecy rate

$$LR_s \leq LR_{CF} - \tfrac{1}{2} \log_2 \det(I_{\eta_R} + PHH'). \tag{8.39}$$

This concludes the proof. □

8.7 Discussion

In this section, we discuss and illustrate Theorem 8.1 and Corollaries 8.3 and 8.4. A lot of the characteristics of the achievable secrecy rate region have already been discussed for the special case of the two-way relay channel in Section 7.6. In this section, we want to extend the insight to more than two users and multiple antennas.

In Fig. 7.2, we saw an example of the achievable secrecy rate for a two-way relay channel. In that case, it is optimal to transmit at full power. In Fig. 8.3, we show that this is not always the case. Therefore, we plotted the achievable rate regions for the channel coefficients $(0.8, 0.5)'$ and a transmit power constraint

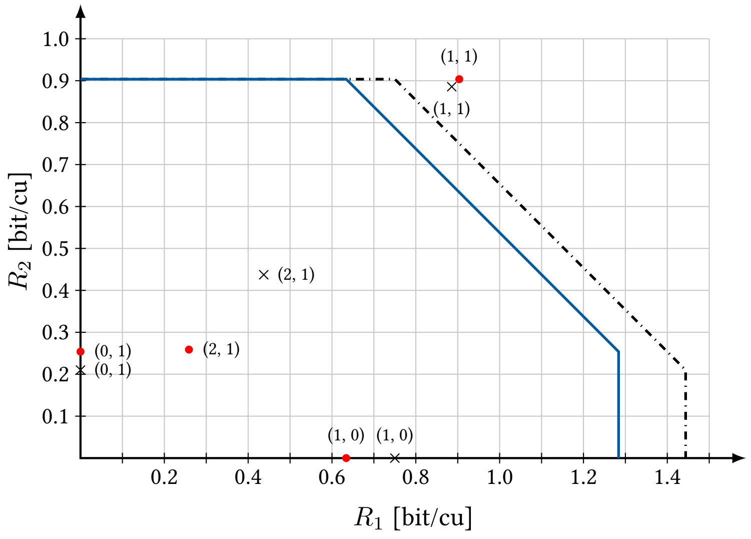


Figure 8.3: MAC capacity region with achievable compute-and-forward rates for different coefficient vectors a at full transmit power (dash-dotted line and crosses) and at optimal transmit power of $P_1/\sigma^2 = 7.7$ and $P_2/\sigma^2 = 10.0$ (blue solid line and red dots) for $h = (0.8, 0.5)'$ and $P/\sigma^2 = 10\,\text{dB}$.

of $P/\sigma^2 = 10\,\mathrm{dB}$. As one can see, optimizing the power means reducing the transmit power for one user node. Since the channel coefficients of the effective channel are then closer to each other, we achieve a higher computation rate for the coefficient vector $a = (1, 1)'$, while reducing the MAC sum-capacity at the same time. Here, we want to stress the uncommon behavior that it is possible to increase the secrecy rate by reducing the transmit power.

We already know that not all channel vectors achieve a positive secrecy rate (see Section 7.6). If we draw the channel coefficients from a normal distribution with zero mean and unit variance, the question arises what percentage of channel realizations result in a positive secrecy rate. The result for the SISO case without optimized power allocation is shown in Fig. 8.4, where we used $10\,000$ i.i.d. channel realizations from a normal distribution $\mathcal{N}(0, 1)$. One can see that we achieve positive secrecy rates for a reasonable amount of channel realizations only in the two-way relay channel. This might be a depressing result, but we can improve the performance by optimizing the transmit power and introducing multiple antennas. The result for an SNR of $5\,\mathrm{dB}$ is shown in Fig. 8.5. In the SISO case, one can see that we increase the share of the channel realizations resulting in a positive secrecy rate from $18.19\,\%$ to $52.1\,\%$ by only optimizing the power allocation. Introducing multiple antennas at the source nodes ensures that a positive secrecy rate can be achieved with high probability. Introducing multiple antennas at the relay is contra-beneficial because we give the eavesdropper more degrees of freedom.

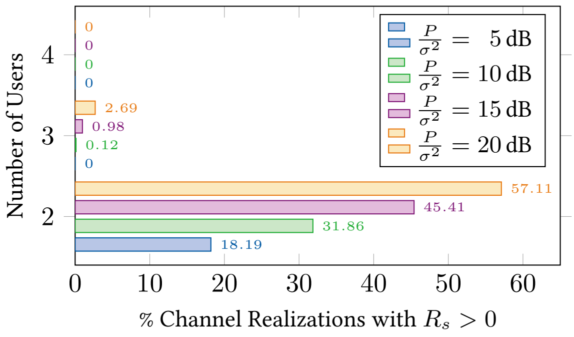
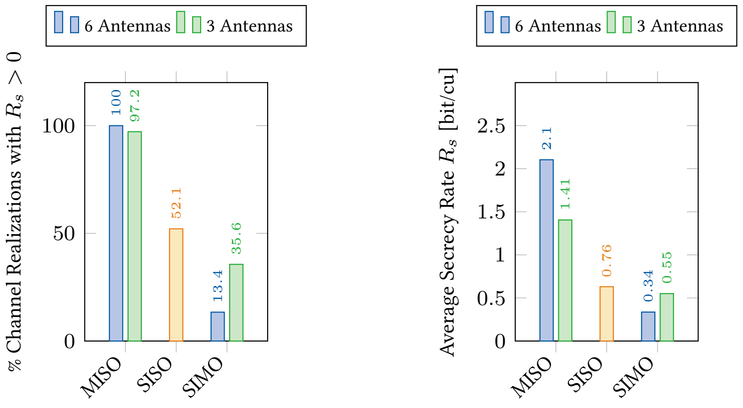


Figure 8.4: Percentage of channel realizations resulting in positive secrecy rates for different numbers of users and SNR values for the SISO multi-way relay channel without optimal power allocation.



(a) Percentage of channel realizations resulting in positive secrecy rates.

(b) Average secrecy rate over all positive rates.

Figure 8.5: Percentage of channel realizations resulting in positive secrecy rates and the according average secrecy rates for $P/\sigma^2 = 5$ dB. All schemes use optimal power allocation. 1000 channel realizations are drawn from a normal distribution $\mathcal{N}(0, 1)$.

Part IV

Compute-and-Forward Implementation

Chapter 9

Implementing PLNC – From Theory to Practice

In this chapter, we show the results of a first implementation of the compute-and-forward framework using SDR. We focus on the feasibility of a communication with lattice encoding and decoding. Therefore, we try to minimize as many sources of error as possible, e.g., synchronization errors. The first step is to make it work within a limited time period. Hence, we do not optimize algorithms or make them robust. This will be subject to further research and development. Although this leads to a very basic implementation, we gain important insights on how theory and practice go together. We pinpoint certain pitfalls and show the main challenges.

9.1 Pyncsim

Everything related to lattice coding is implemented within the python package `pyncsim`, which stands for *Python Network Simulator*. It has been developed by the author of this thesis to simulate the compute-and-forward protocol and compare it to other approaches. `Pyncsim` is embedded within the SageMath framework [101] and makes use of some of its classes, especially its linear and abstract algebra system. The main features of `pyncsim` are:

- a lattice class that provides several properties of a lattice,
- an interface to all lattices available in [29],
- a nested lattice code class with encoding and decoding algorithms,
- a compute-and-forward module with all compute-and-forward related algorithms, such as "search for optimal coefficient vector", "calculate computation rate", etc.

The content of the package is shown in Fig. 9.1, and the modules used later are highlighted. Under the supervision of the author of the thesis, part of this code has been written by Thomas Frank while he was a student assistant, by Sascha Neumann as part of a diploma thesis [98], by Hannes Ellinger as part of a student research project [87], and by Lennart Schierling as part of a diploma thesis [99].

```
pyncsim
    ├── basic_elements.py
    ├── codes
    │       └── lattice_codes.py
    ├── events.py
    ├── mathematics
    │           ├── lattice.py
    │           └── lattices.py
    ├── models
    │       ├── channels.py
    │       ├── encoder.py
    │       ├── decoder.py
    │       ├── relays
    │       │       ├── computeandforward.py
    │       │       └── decodeandforward.py
    │       └── sources.py
    ├── simple_blocks.py
    ├── traces.py
    └── utils.py
```

Figure 9.1: Content of python package `pyncsim`.

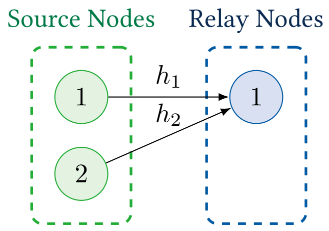


Figure 9.2: Two-user MAC model.

Table 9.1: Hardware specification of the USRPs.

Device Version	USRP2
Motherboard	N210r4
Daughterboard	CBX
Firmware Version	12.4
FPGA Version	11.1
Frequency Range	1.2 GHz to 6.0 GHz
Master Clock Rate	Fixed 100 MHz
FPGA bandwidth	25 MHz of RF BW with 16-bit samples
ADCs	14-bits 100 MS/s
DACs	16-bits 400 MS/s

9.2 Hardware Setup

In Fig. 1.3, we show the general system model that is used throughout the thesis. However, the most relevant part for compute-and-forward is the data transmission from the source nodes to the relay nodes. Therefore, the focus of the implementation was on the two-user MAC as depicted in Fig. 9.2. The communication that happens afterwards can be realized by standard schemes, which are not subject of this thesis. The setup consists of four different components as shown in Fig. 9.3. There are a host computer for the baseband signal processing, a box that provides a reference signal and a synchronization pulse, three universal software radio peripherals (USRPs), and an Ethernet switch to connect the USRPs to the host computer. The hardware specifications for the USRPs are summarized in Table 9.1.

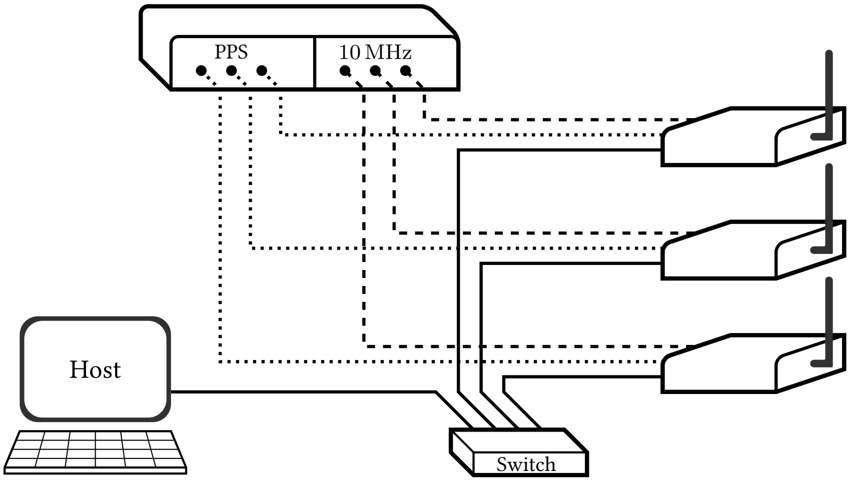


Figure 9.3: USRP setup.

Two of the three USRPs are used to transmit data, while the third USRP receives a noisy superposition of their data. The host computer deals with the baseband signal processing of all three USRPs. It sends the transmit signals via Ethernet to the two transmitters and receives the digital baseband signal also via Ethernet from the receiver.

9.1 Remark. All data has to be transported via Ethernet to the host computer. This is limiting our transmission rate and our sampling rate. We could reduce the limitation by using one host computer for each USRP. However, this does not allow cooperative processing of the data. Although that is not a requirement for the setup to work, it makes it easier to evaluate the communication process.$\lhd$

Since we want to focus on the implementation of the compute-and-forward framework, we try to remove as many sources of error as possible. One of these sources of error are the oscillators in the USRPs. In order to prevent frequency and phase shifting over time, all three USRPs get the same reference signal, i.e., a 10 MHz signal and a pulse every second. This synchronizes the oscillators of all USRPs and provides synchronized data processing at the USRPs. For a real-life application, we have to deal with synchronization errors, but this is left for further research.

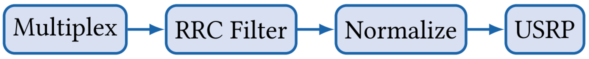


Figure 9.4: Transmitter.

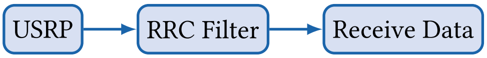


Figure 9.5: Receiver.

The baseband signal processing as well as the communication with the USRPs are done with the help of the GNU Radio Framework [100]. This framework provides the drivers to communicate with the USRPs and a lot of signal processing algorithms. However, lattice coded communication is not available in GNU Radio. Therefore, all signal processing algorithms have to be implemented except for the impulse shaping and the matched filters, which are taken from the existing GNU Radio blocks.

In order to avoid under- and overflows during the data transmission due to slow algorithms, we divide the communication into three separate phases, i.e.,

1. calculate baseband signal (encoding),
2. transmit data, and
3. decode data.

Only the second phase is done in real-time. The other two phases are done offline before and after the actual data transmission. Therefore, the GNU Radio flowgraph consists only of a few basic blocks. The transmitter as depicted in Fig. 9.4 reads the baseband signal from a file, which contains the multiplexed parts of the data as it will be described in Section 9.3. The baseband signal is processed by a root raised cosine (RRC) filter and normalized. The RRC filter is an interpolating filter with an interpolation factor of 10. This provides the impulse shaping of the transmit signal. Afterwards, it is sent to the USRP and transmitted.

The receiver as depicted in Fig. 9.5 is as simple as the transmitter. The received signal is processed by the matching RRC filter and saved to a file for later processing.

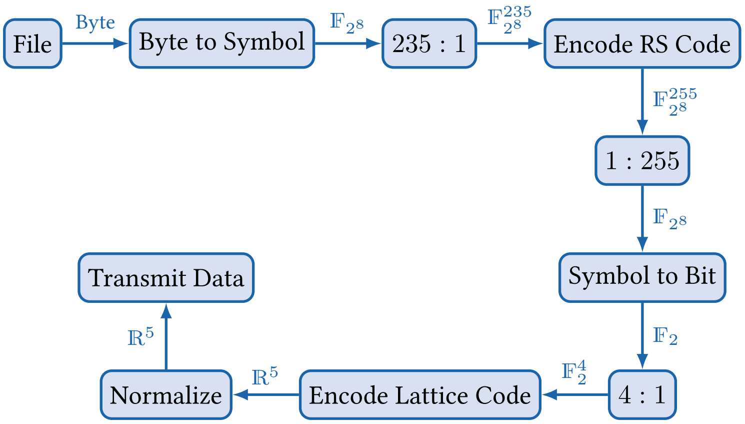


Figure 9.6: File encoding.

9.3 Encoding

In this section, we describe the encoding process of an input file that has to be transmitted via the compute-and-forward protocol. The encoding process is shown in Fig. 9.6.

9.2 Remark. The terminology of dimension and code length might be a bit confusing. As typical in coding theory, the length of the messages is the *dimension* of a code, whereas the codeword length is the *length* of the code. However, for a lattice code we use lattices of a certain dimension. The code length is therefore the dimension of the lattice, which should not be confused with the dimension of the code. ◁

The data that should be transmitted is read from a file and encoded with a python script. The following python modules are used:

```python
import numpy as np
from sage.all import GF, matrix, vector, ceil, codes, dumps

from pyncsim.mathematics import lattices
from pyncsim.codes import lattice_codes
from grncsim.utils import conversion, io
```

Now, we can create the inner and outer code that are used to encode the data. We use a Reed-Solomon code as outer code because there is an implementation included in SageMath. The inner code is a lattice code, namely a nested lattice code as introduced in [40]. In order to create the lattice code, we need a coarse lattice (here we choose A5) and a code generator matrix. With these two ingredients, the nested lattice code is created with the help of construction A (see Appendix A.3.1). The details are hidden from the end user to provide a clean application programming interface (API).

```python
# Inner Lattice Code
p, n, k = 2, 5, 4
F = GF(p)

G = matrix(F,k,n,[
    [1,0,0,0,1],
    [0,1,0,0,1],
    [0,0,1,0,1],
    [0,0,0,1,1]
])
C = codes.LinearCode(G)
lattice = lattices.create_lattice("A5")
inner_code = lattice_codes.NestedLatticeCodeNazer(lattice, C)

# Outer RS Code: (255,235,21)-Code
F1 = GF(2**8, 'a')
outer_code = codes.GeneralizedReedSolomonCode(F1.list()[1:], 235)
```

The choice of the parameters is more or less arbitrary and should be subject to further investigation. However, we choose the dimension of the lattice code such that the algorithms for decoding run within an acceptable time. Further, we use the Reed-Solomon Code over $\mathbb{F}_{2^8}$, because this aligns very well with the bytes we read from the input file. This also justifies the field size of 2 for the elements of the messages for the lattice code, because we can easily convert from $\mathbb{F}_{2^8}$ to $\mathbb{F}_2$. The message length of the lattice code is chosen to be 4, because this aligns perfectly with the symbol size of the outer code, i.e., two messages of the inner code represent one symbol of a codeword from the outer code.

9.3 Remark. Although the choice of the parameters and codes are arbitrary, it is favorable to design all parts such that the lengths of the codewords and messages in each stage of the encoding process align. This avoids zero-padding and the corresponding overhead. $\triangleleft$

Having created the codes, we can read and encode the data. Therefore, we iterate over the read bytes, which are already converted to elements in $\mathbb{F}_{2^8}$. We group an appropriate number of symbols corresponding to the dimension of the outer Reed-Solomon code and encode it to get the codeword of the outer code. The symbols of the codeword of the outer code are converted to elements in $\mathbb{F}_2$. These symbols provide the input for the inner lattice code. Again, we iterate over groups of these symbols, where the number of symbols matches the dimension of the inner code.

```python
outer_dimension = outer_code.dimension()
inner_dimension = inner_code.dimension()

coded_input = []

# read symbols in GF(2**8) from file
symbol_array = io.read_from_file(source_file, output="symbol")

# encode outer code
cws = len(symbol_array) / outer_dimension
for x in xrange(cws):
  pointer_o = x*outer_dimension

  wo = symbol_array[pointer_o:pointer_o+outer_dimension]
  co = outer_code.encode(vector(wo))

  bit_array = []
  for s in co:
    bit_array += conversion.symbol_to_bit(s)

  # encode inner/lattice code
  cwb = len(bit_array) / inner_dimension
  for y in xrange(cwb):
    pointer_i = y*inner_dimension

    wi = vector(bit_array[pointer_i:pointer_i+inner_dimension])
    ci = inner_code.encode(wi)
    coded_input += ci.list()
```

The encoded data is now available as a numpy array with real-valued entries. Basically, this is the baseband signal that will be transmitted. However, the USRPs accept only signal amplitudes less or equal to one. Higher signal amplitudes will be cut off. Therefore, we normalize the baseband signal according to the largest amplitude of any lattice point in the lattice code.

9.4 Remark. We normalize all codewords using the same factor. This might be a disadvantage for codewords that have only small amplitudes. However,

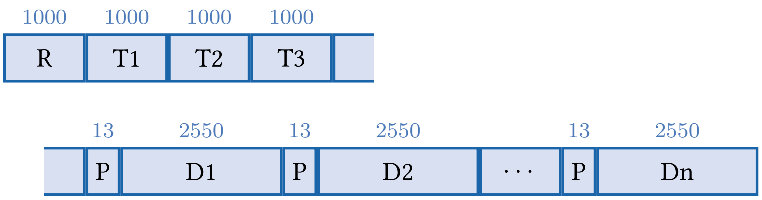


Figure 9.7: Global frame structure.

Table 9.2: Parts of the data transmission.

Part	Explanation
R	random elements from $\{-1, +1\}$ by user 1 and user 2
T1	training sequence from user 1 and zeros from user 2
T2	training sequence from user 2 and zeros from user 1
T3	zeros from both users
P	preamble from user 1 and user 2
Di	payload data from user 1 and user 2, $i \in \{1, \ldots, n\}$

it allows us to easily revoke the normalization at the receiver because the normalization factor depends only on the code and not on the sent codeword. For further research, it might be of interest to develop nested lattice codes such that the codewords are uniquely determined by the relative values of the samples and not the absolute values. $\triangleleft$

Having encoded the input file, we need to multiplex the data to a global frame structure because the transmit data consists not only of the encoded input file but also of training sequences, preambles, etc. The frame structure is depicted in Fig. 9.7, where the numbers represent the samples within each part. A summary of all parts is shown in Table 9.2.

R This part consists of 1000 random samples chosen from the set $\{-1, +1\}$. This is used to adjust and train the receive amplifier. It can be randomly chosen for each transmission, and the receiver does not need to know it. This is different to the training sequence, which is chosen once and needs to be known by the receiver.

T1 In this time slot, the first user sends its training sequence, whereas the second user is silent. The training sequence is generated by a pseudo-random number generator over the set $\{-1, +1\}$. The goal is to have a sequence that provides good correlation properties such that it can easily be found within the data stream. A pseudo-random number generator provides a sequence with these properties. However, the training sequence is chosen prior to transmission and needs to be known by the receiver.

T2 In this time slot, the second user sends its training sequence, whereas the first user is silent. For simplicity, the second user uses the same training sequence as the first user.

T3 In this time slot, both users are silent. This allows the measurement of the noise power at the receiver.

P This is the preamble, which is sent right before the payload. The preamble is a Barker code of length 13 and is used to mark the start of the actual data.

Di Those time slots contain the transmit data. One time slot contains exactly one codeword from the outer code. Because we have an outer code length of $255\,\mathrm{symbols}$ with a symbol size of $8\,\mathrm{bit}$, we get the following number of samples:

$$255 \cdot 8 \cdot \frac{5}{4} = 2550, \tag{9.1}$$

where the inner code length is $5\,\mathrm{samples}$, and we encode $4\,\mathrm{bit}$ into one codeword.

9.4 Transmission

The data transmission is done with the tools of GNU Radio. Hence, we need to create a flowgraph that represents the communication chain. This is realized by creating a class and inheriting a GNU Radio top block, which can be run later.

```python
import sys, os
from gnuradio import gr
from grncsim.blocks.transceiver import TransceiverFactory

class LatticeMacTransmission(gr.top_block):

  def __init__(self, sample_rate, sps, channel,
               directory, pre_wait_time=1.0):
```

```
    gr.top_block.__init__(self, "Lattice MAC Transmission")

    # create transmitter and receiver
    transceiver = TransceiverFactory(sample_rate, sps, debug=False)
    tx1 = transceiver.create_transmitter(
            os.path.join(directory, "transmit1.data"), 0, 2)
    tx2 = transceiver.create_transmitter(
            os.path.join(directory, "transmit2.data"), 1, 2)
    rx  = transceiver.create_receiver(
            os.path.join(directory, "receive.data"))

    # connect flow graph
    self.connect(tx1, (channel,0))
    self.connect(tx2, (channel,1))
    self.connect(channel, rx)

    # Start USRPs with delay
    channel.set_tx_delay(pre_wait_time)
```

We keep the flowgraph as simple as possible because we define a transmitter class and a receiver class that can be created by a transceiver factory. Having created the flowgraph, we are almost ready to start the transmission. Before we can do that, we need to load the settings from a configuration file. After that, we create a new channel and a new top block with the flowgraph of our communication chain.

```
from grncsim.blocks.channels import UsrpMacChannel
from grncsim.settings import settings

# Create new channel
channel = UsrpMacChannel(
  int(settings['HARDWARE']['sample_rate']),
  int(settings['HARDWARE']['carrier_frequency']),
  settings['HARDWARE']['ip_terminal_0'],
  settings['HARDWARE']['ip_terminal_1'],
  settings['HARDWARE']['ip_relay'],
)

tb = LatticeMacTransmission(
  int(settings['HARDWARE']['sample_rate']),
  int(settings['FILTER']['sps']),
  channel,
  args.output,
  pre_wait_time=0.5
)
```

```

tb.start()
transmitted_samples = 0
while transmitted_samples < transmit_samples:
  time.sleep(0.5)
  transmitted_samples = tb.tx1.samples_transmitted()
tb.stop()
tb.wait()
```

The samples are filtered by an interpolating RRC filter and then sent to the USRPs. The receiver gets a superposition of the transmitted samples and filters them using the matching RRC filter. The interpolation of the transmit filter is not revoked at the receiver. The received and filtered samples are saved to a numpy file for further processing.

9.5 Decoding and Analysis

The signal processing at the receiver consists of the following steps:

1. find the training sequences,
2. estimate the channel coefficients,
3. find optimal network coding coefficients,
4. find preambles,
5. decode payload.

We use the following python modules in the sequel of this section.

```
import numpy as np
from scipy import signal
from scipy import optimize
from scipy import linalg
```

Find Training Sequences

We start with the search for the training sequences. This can be realized by correlating the received signal with the expected training sequence. Special care needs to be taken because we search for several occurrences of the training sequence. Therefore, we do not only care for the maximum of the cross-correlation but also for the second largest peak. This peak can be positioned before the maximum peak or after the maximum peak depending on the channels.

```python
def find_training_sequences(recv_data, search_data, number=1):

  # calculate cross-correlation
  corr = np.abs(signal.fftconvolve(recv_data,
                                   search_data[::-1], mode='valid'))

  index        = np.argmax(corr)
  indices      = [index,]
  length       = len(search_data)
  skip_range   = [ int(index-length/2), int(index+length/2) ]
  found_number = 1

  while ( (found_number < number) ):

    before  = np.argmax(corr[skip_range[0]-length:skip_range[0]])
    before += skip_range[0]-length
    after   = np.argmax(corr[skip_range[1]:skip_range[1]+length])
    after  += skip_range[1]

    if corr[before] > corr[after]:
      skip_range[0] = int(before-length/2)
      indices.append(before)
    else:
      skip_range[1] = int(after+length/2)
      indices.append(after)

    found_number += 1

  return indices
```

An example of the cross-correlation is shown in Fig. 9.8. We can see that the start of the two training sequences can be obtained clearly. This also justifies the use of a pseudo-random sequence as training sequence, which obviously provides good correlation properties.

Estimate Channel Coefficients

Having obtained the training sequences from each user, we can use them to estimate the channel coefficients. Although many different channel estimation procedures can be applied, we restrict ourselves to maximum-likelihood (ML) channel estimation. The advantage is that this procedure requires neither knowledge of the noise variance nor any statistical information about the channel (see [91] for a summary of channel estimation techniques). The optimization problem for ML channel estimation is given by

$$\min_h \|y - hx_t\|^2, \tag{9.2}$$

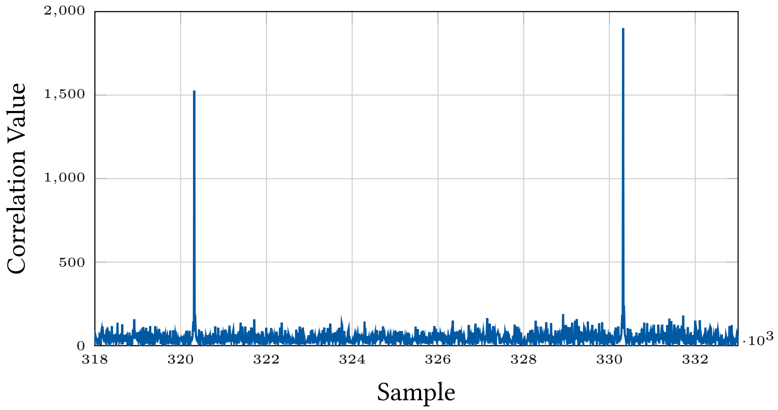


Figure 9.8: Cross-correlation of the training sequence and the received signal.

where y is the received signal, x_t is the transmitted signal, i.e., the training sequence, and h is the channel coefficient. We use a real representation of the complex signal to perform the optimization, i.e.,

$$\min_x \|Ax + b\|^2, \tag{9.3}$$

where A is a 2000×2 matrix containing the transmitted training sequence, b is a vector of length 2000 containing the received training sequence, and x is a length-2 vector containing the real and imaginary part of the channel coefficient. Explicitly, that is

$$A = \begin{pmatrix} -\Re(x_t) & \Im(x_t) \\ -\Im(x_t) & -\Re(x_t) \end{pmatrix}, \quad b = \begin{pmatrix} \Re(y) \\ \Im(y) \end{pmatrix}, \quad x = \begin{pmatrix} \Re(h) \\ \Im(h) \end{pmatrix}. \tag{9.4}$$

The problem is a continuous quadratic programming problem, and we use the optimize package from scipy to perform the optimization.

```python
def f(x):
  return linalg.norm(np.dot(A,x) + b)

res = optimize.fmin(f,x0=np.array([0,0]))
```

```
channel_real = res[0]
channel_imag = res[1]
```

Find Optimal Network Coding Coefficients

From the channel coefficients, we can calculate the optimal network coding coefficients as described in Chapter 5. Further, we can calculate the optimal scaling coefficient α according to the channel coefficients and the network coding coefficients, i.e.,

$$\alpha = \frac{Ph'a}{1 + P\|h\|^2}. \tag{9.5}$$

Find Preambles

Now, we have all the meta information that we need to decode the actual data. We correlate the remaining received signal with the transmitted preamble to find the start of the payload. As for the training sequences, we also have multiple occurrences of the preamble. Therefore, we search for peaks in the correlation function that are above a certain threshold. This threshold can be an absolute value or an value that is relative to the maximum peak.

When we return all indices of correlation values that are above the threshold, we get consecutive indices because the peaks are not narrow enough. There are basically two reasons for that. Firstly, we correlate against the interpolated signal from the RRC filter. Therefore, we stretch the peaks by the interpolation factor. Secondly, it depends on the threshold value we use. If we choose it too high, we get narrower peaks but might also miss a peak. This highly depends on the channel quality. In order to cope with this, all consecutive indices need to be reduced to the index where the correlation function has its local maximum.

```
def find_signal_multiple(recv_data, search_data, tolerance=0.3,
                         relative_tolerance=True):

  # calculate cross-correlation
  corr = signal.fftconvolve(recv_data, search_data[::-1], mode='valid')

  # tolerance relative to maximum correlation value
  if relative_tolerance:
    index      = np.argmax(corr)
    index_inv  = np.argmin(corr)
    if np.abs(corr[index]) < np.abs(corr[index_inv]):
      inverted = True
      max_corr = corr[index_inv]
      indices  = np.where(corr <= max_corr - tolerance*max_corr)[0]
      else:
        inverted = False
        max_corr = corr[index]
        indices  = np.where(corr >= max_corr - tolerance*max_corr)[0]

    # tolerance relative to expected correlation
    else:
      max_corr = sum(map(lambda x: x**2, search_data))
      indices  = np.where(corr >= max_corr - tolerance*max_corr)[0]

    maxcorr_index = []

    if len(indices) > 0:

      # split consecutive maxima
      prev = indices[0]
      temparray = [prev,]
      for index in indices[1:]:
        if index - prev <= 3:
          temparray.append(index)
        else:
          corr_array = np.asarray([corr[i] for i in temparray])
          maxcorr_index.append(temparray[np.argmax(corr_array)])
          temparray = [index,]
        prev = index

      # last entry
      corr_array = np.asarray([corr[i] for i in temparray])
      maxcorr_index.append(temparray[np.argmax(corr_array)])

    return maxcorr_index
```

Decode Payload

Now, we can split the data into several parts as depicted in Fig. 9.7 and obtain multiple payload blocks. Since we do not revoke the interpolation of the transmit filter, each block should have a length of 25 500 samples, which is the length at the encoder multiplied by the interpolation factor of 10. The decoding of each payload block follows the encoding procedure in reverse order. Firstly, we revoke the normalization according to the maximum amplitude of all codeword samples in the lattice code. Secondly, we scale the payload with the optimal scaling factor α, which is provided in Eq. (9.5). Thirdly, we take every tenth sample to revoke the interpolation, starting at the preamble index obtained in the previous step. This yields an estimate of the superposition of the lattice codewords that the two users have sent.

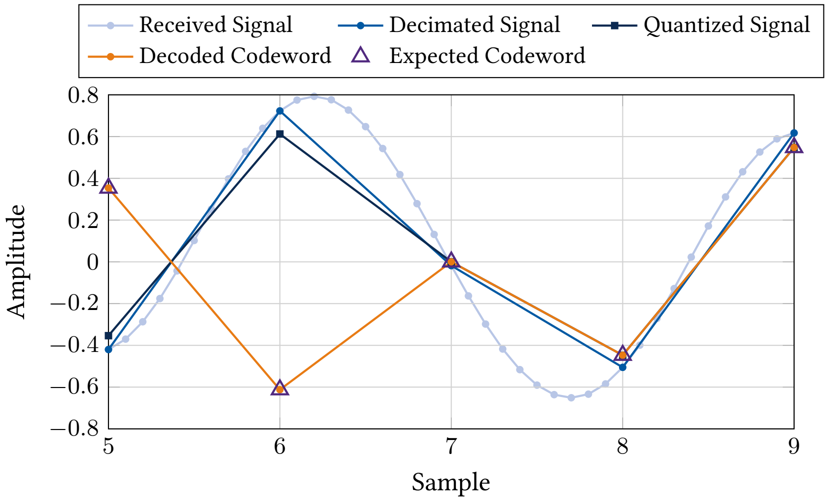


Figure 9.9: Decoding stages of a lattice codeword.

Now, we divide the payload block into blocks whose size corresponds to the inner code length, i.e., five samples in the current implementation. We iterate over these blocks to decode the lattice codewords and obtain the original bits. First of all, we need an estimate of the lattice codeword, which is obtained by the following equation

$$\hat{v} = [Q_{\Lambda_F}(\alpha y)] \bmod \Lambda_C. \tag{9.6}$$

Therefore, the scaled receive signal is quantized with respect to the fine lattice and then taken modulo by the coarse lattice. The individual steps are shown for the second lattice codeword in an example payload block in Fig. 9.9.

9.5 Remark. The quantization as well as the modulo operation require to solve a closest-vector problem (CVP). It has been shown [23, 28] that this is in general an np-complete problem. An overview of different algorithms can be found for example in [25]. Our implementation uses the algorithm proposed by Agrell et al. [20]. So far, this is a substantial bottleneck for the performance of our decoding procedure. Choosing a certain lattice, where the CVP can efficiently be solved by an adapted algorithm, could be subject to further research. ◁

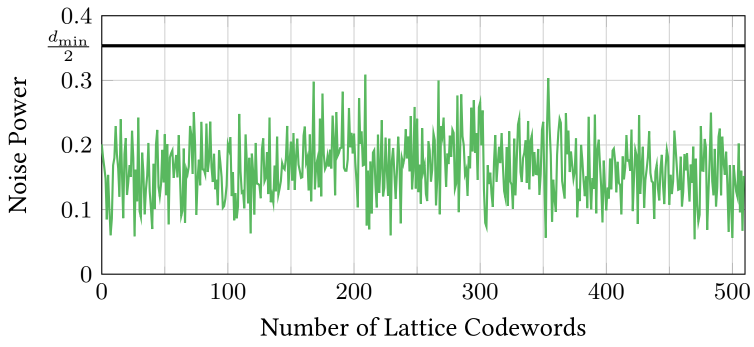


Figure 9.10: Effective noise for each lattice codeword over time.

```
decoded_inner = []
ilength = inner_code.length()
c_inner = len(payload) / ilength
for x in xrange(c_inner):
  c  = payload[x*ilength:x*ilength+ilength]
  qc = inner_code.fine_lattice.quantize(c)
  mc = inner_code.coarse_lattice.mod(qc)
  w  = inner_code.decode(mc)

  decoded_inner += w.list()
```

A correct lattice codeword is obtained as long as the noise does not leave the Voronoi region of the fine lattice, i.e., the quantization results in a correct lattice point. As typical for block codes, the minimal distance between any two codewords is a figure-of-merit on how much noise is acceptable for error-free decoding. The same holds for lattice codes, where the minimal distance corresponds to half of the packing radius. We show the distance of the received lattice codewords from the expected lattice codewords of one payload block in Fig. 9.10. The distance corresponds to the noise power. As long as the noise power is less than half of the minimal distance of the nested lattice code, the lattice codewords are obtained without error. We can see in Fig. 9.10 that all lattice codewords are correctly obtained in this particular example.

Having decoded the lattice codewords, we have a sequence of bits. Those bits will be re-packed to symbols in $\mathbb{F}_{2^8}$ and then can be decoded by the outer Reed-Solomon code.

```
# bit to symbols
symbols = []
c_symbols = len(decoded_inner) / 8
F = outer_code.base_ring()
for x in xrange(c_symbols):
  l = decoded_inner[x*8:x*8+8]
  symbols.append(F(l))

# decode outer code
symbols = sagelib.vector(symbols)
message = outer_code.decode_to_message(symbols)
```

9.6 Challenges and Pitfalls

A lot of challenges occurred during the implementation. Some of them have been solved but others are still open. We want to stress that the goal of this chapter is a proof of concept. That means, we want to show that it is possible to realize a communication with compute-and-forward. Therefore, we leave the optimization of algorithms and the evaluation of parameter sets to further research. In the following, we explain the most important challenges.

Maximum Amplitude

The hardware we used accepts only samples with a maximum amplitude of one. Higher amplitudes are cut off. Therefore, the signal needs to be normalized.

There are basically two ways to normalize the amplitude. We can either normalize each codeword according to its highest amplitude, or we can normalize each codeword according the highest amplitude of the code. Normalizing each codeword separately has the advantage that the maximal possible amplitude of that codeword is utilized. This is beneficial in terms of SNR because the whole amplitude range is utilized. However, the normalization has to be revoked at the receiver. Therefore, we would need to guess the normalization factor somehow from the received codeword. This is not possible with the current encoding scheme. One might overcome this issue by choosing a lattice coding scheme such that the normalization factor can be recovered from the relative sample values. However, the easiest solution is to normalize each codeword by the same normalization factor, which is chosen to be the largest possible amplitude for all codewords in the codebook. This will possibly waste SNR for codewords that only have samples with small amplitudes, but it has the advantage that the receiver knows the normalization factor and can revoke it. However, the receiver would not even need to revoke the normalization because the normalization factor is equal for all codewords and could be interpreted as part of the channel. Since we estimate the channel values from the training sequences, which are not normalized by the same factor, we have to revoke the normalization at the receiver in our setup.

Hardware Parameters

There are many hardware parameters that can be adjusted and need to be set correctly. The most important ones are the carrier frequency and the sample rate, which we choose to be 2.45 GHz and 625 kHz, respectively. The carrier frequency needs to be within a range that is supported by the USRPs (see Table 9.1). However, the sample rate has to be chosen according to several hardware parameters. Since all three USRPs are connected to the host computer by Ethernet, they share the bandwidth of the Ethernet interface, which is approximately 1 Gbit/s. Further, there are some hardware limitations as stated in the Application Notes on the Ettus Research website:

> It is important to understand that strictly-integer decimation and interpolation are used within USRP hardware to meet the requested sample-rate requirements of the application at hand. That means that the desired sample rate must meet the requirement that master-clock-rate / desired-sample-rate be an integer ratio. Further, it is strongly desirable for that ratio to be even.
>
> There are further constraints on the desired sample rate, such that if the required decimation or interpolation exceeds 128, then the resulting decimation must be evenly divisible by 2, and that if the required decimation exceeds 256, the resulting decimation must be evenly divisible by 4. [88]

The complex signal that is sent to the USRPs consists of 32-bit samples (16 bit for the real part and 16 bit for the imaginary part). With a sample rate of 625 kHz, we get a data rate of 20 Mbit/s. That is the data rate without package overhead for one device that has to be provided by the host computer. If the host computer is not able to provide that data rate because of limited Ethernet bandwidth or its own limited computational performance, underflow errors will occur. Since we have three devices, we get a total data rate of 60 Mbit/s over the Ethernet interface. This is below the maximum Ethernet bandwidth. But keep in mind that there will be additional overhead for packaging the samples. We find that a sample rate of 625 kHz is the maximum in our hardware setup in order to avoiding overflows and underflows. It also fulfills the requirement that the decimation is evenly divisible by two, since we have

$$\frac{100\,\mathrm{MHz}}{625\,\mathrm{kHz}} = 160. \tag{9.7}$$

But sample rate and carrier frequency are not the only parameters that need to be adjusted. Two very important parameters are the transmit and receive gain, which can be chosen from $0\,\mathrm{dB}$ to $30\,\mathrm{dB}$. An obvious choice is to chose the transmit gain as high as possible to provide a good receive SNR. However, the path loss in our setup is not very high. If we set the transmit gain to the maximum value, the receiver get into a state of overdrive because both transmit signals will get superposed and their powers add up.

Floating Point Arithmetic

Due to the fact that most decimal fractions cannot be represented by binary fractions, all decimal fractions are stored as approximations in the system. In most cases, the approximation is close enough to not bother the programmer. It becomes interesting for algorithms that rely on comparisons. In our case, the CVP falls in this category. When we design the nested lattice code, it is possible that lattice points in the fine lattice lay on the border of the Voronoi region of the coarse lattice. This is valid but the lattice points have to be assigned to the Voronoi regions in a systematic manner. A systematic manner is already provided by the algorithm, which compares the distances of the lattice points always in the same order and manner. However, depending on the previous calculations and the floating point approximations therein, we will not get a lattice point that exactly lays on the border of a Voronoi region. A very small floating point approximation error will result in a serious decoding error. This can be overcome by limiting the number of significant digits for the comparisons or using different python modules such as `decimal` or `fraction`.

Time Consumption

The focus of the current implementation is not on efficiency. As already mentioned, the used algorithms for encoding and decoding are not optimized. The bottleneck in performance is the lattice quantization and the modulo operation. Both require to solve a CVP, which is np-complete in general. Therefore, the implementation is relatively slow. A possible improvement is the optimization of the algorithm for a certain lattice code. However, that is left for further research.

Since we use very short lattice codes, we need quite a lot of iterations to encode a frame with the inner code. This results in a large total time for the lattice encoding. As an example, we encode a bitmap with 22 400 pixels. The average time needed for encoding one lattice codeword is 1.05 ms. However, we do this 148 410 times, which results in a total time of 156.48 s. The average encoding time for a codeword of the Reed-Solomon code is 3.00 ms. Due to the larger field size and the larger code dimension, it takes only 291 iterations to encode all frames, which results in a total time of 0.87 s. Although increasing the dimension of the lattice code might reduce the number of iterations needed for encoding, the time needed by the encoding algorithm will increase in a non-linear way. Therefore, there will be a trade-off between the number of iterations and the code dimension. One promising parameter is the field size. Increasing it will reduce the number of iterations. However, there will be additional overhead due to more complex operations over the larger field, which will result in a higher time consumption, and there will be a trade-off again.

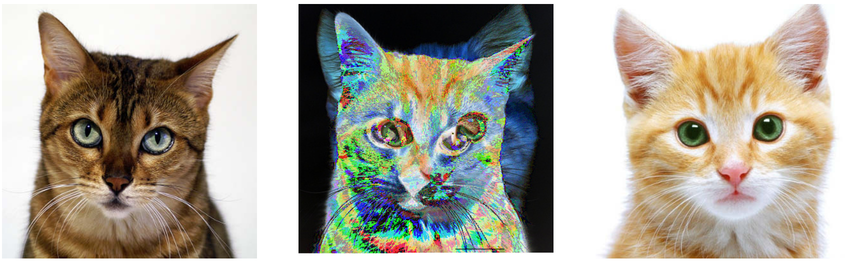
Further, finding the optimal network coding coefficients is also very slow for a large amount of users. However, this needs to be done only at the beginning of a transmission if we assume that the channel is constant for a certain amount of time.

We also find that the overhead introduced by SageMath is significant. It is a very convenient tool for developing and testing algorithms, because it provides a lot of functionality like vector and matrix operations, etc. However, it is not designed to handle real-time applications. In order to bring the signal processing closer to real-world applications, a native implementation of all the signal processing steps is necessary.

We measured the encoding and decoding rates on an Intel i5 Quad-Core CPU with 3.1 GHz. However, our implementation is not designed for multithreading. Therefore, it uses only one core. With these preconditions, we get an encoding rate of approximately 3.4 kbit/s and a decoding rate of approximately 1.5 kbit/s.

9.7 Example Transmission

In the following, we show an example transmission of two pictures to a relay and the received superposition at the relay. The respective pictures are shown in Fig. 9.11. We do not transmit the files as a whole but use only the pixel data of the pictures. Therefore, we do not have to deal with an unreadable file if the meta or header information of the file are not decoded correctly. Further, we can directly see the superposition of the pixels, which would not be possible if



(a) Picture transmitted by user 1.

(b) Picture received by the relay.

(c) Picture transmitted by user 2.

Figure 9.11: Decoded superposition of two pictures.

the meta information of the files are also superposed because this would result in an invalid file.

The files are bitmaps with 8 bit RGB encoding, which fits perfectly in our encoding scheme since we use a symbol size of 2^8 for the outer code. The pixel data is sequentially read line by line. Since we have a multi-band image, we flatten the input stream such that every third symbol contains the value of the R band, followed by the G band and the B band. In order to avoid zero-padding in the outer code, we use a $(255, 231)$ Reed-Solomon code instead of a $(255, 235)$ code. However, any code with a dimension divisible by three is a good choice.

In the example transmission, the channel coefficients are measured to be 0.131 and 0.133 for user 1 and user 2, respectively. Since the channel coefficients are almost identical, the relay can decode a superposition with coefficients $(1, 1)$. This equals an XOR of the bits, and we can see the result in Fig. 9.11b.

9.6 Remark. We chose the encoding such that we can see the result as an XOR combination of the image data. If we choose different finite fields for the inner and outer codes, the result would be different and not as nicely visible as shown in Fig. 9.11. Further, we could treat the image file as a binary file and transmit its data. Again, we would not be able to visualize the superposition as nicely.$\triangleleft$

As already mentioned, a lot of time is spent for encoding and decoding the data. We show the 30 most time consuming python methods in Figs. 9.12 and 9.13. For better comparison, we normalize the time to the most time consuming method. We can see that the lattice encoding and decoding take the most time. However, a very significant amount of time is spent in methods that just provide data models for finite fields, vector and matrix arithmetic, etc. Therefore, we can conclude that it is not sufficient to optimize the algorithms, we additionally need a more efficient implementation of the methods provided by SageMath.

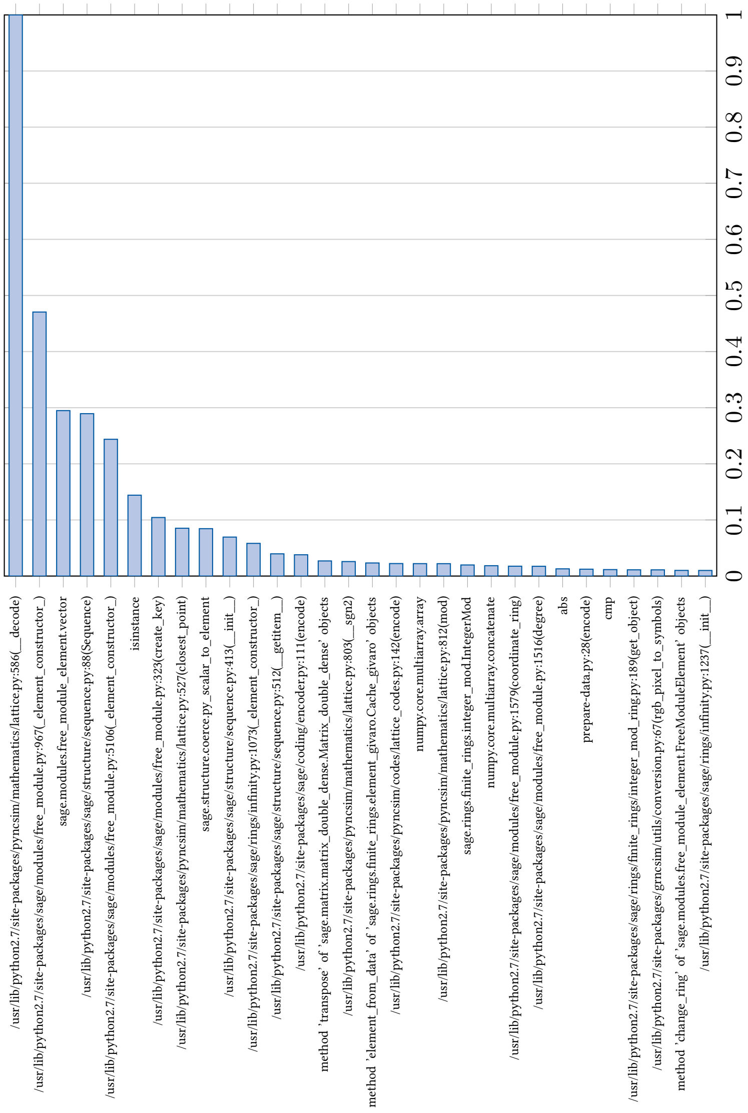


Figure 9.12: Normalized time consumption of the top 30 most time consuming methods (encoding).

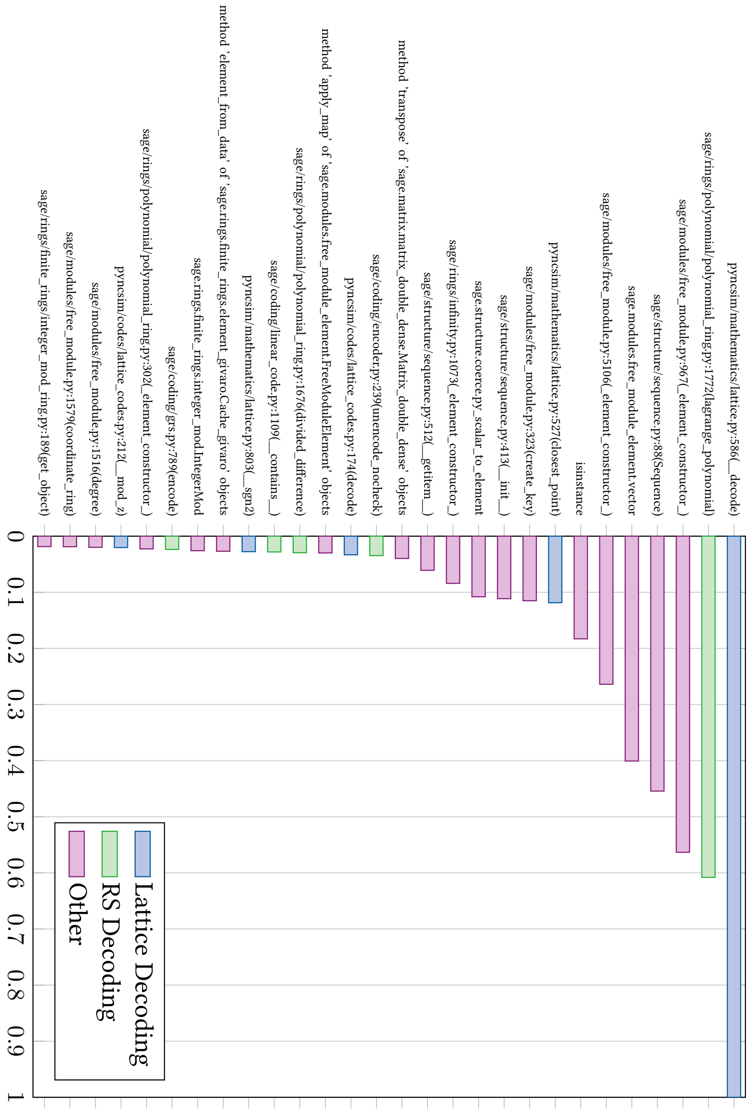


Figure 9.13: Normalized time consumption of the top 30 most time consuming methods (decoding).

Part V

Open Topics and Conclusion

Open Topics

In this thesis, we combine different techniques to solve problems related to physical-layer network coding. During the research for this thesis, we came across some problems and interesting topics that we did not investigate. In the following sections, we describe some of those topics to provide a starting point for further research. We also discuss possible improvements and enhancements for the topics we described in this thesis.

10.1 Optimizing Network Coding Coefficients

The problem of finding the optimal network coding coefficients for a lattice encoded superposition has been discussed in Part II. Thereby, we focused on the lattice equations to find the optimal coefficients. But in the end, we are interested in a linear combination in the message space, which is usually a finite field. The requirement of linear independent lattice equations is only a necessary but not a sufficient condition to fulfill the requirement of linear independent equations in the message space. Because two linear independent vectors with elements in $\mathbb{Z}$ are not necessarily independent if the elements are transformed to elements in $\mathbb{F}_p$. Take the example of messages from $\mathbb{F}_2^k$, where the coefficients are within $\mathbb{F}_2$. Two lattice equations with coefficients $(1,1) \in \mathbb{Z}^2$ and $(1,3) \in \mathbb{Z}^2$ are linearly independent. However, both vectors transform to $(1,1)$ in $\mathbb{F}_2^2$ and are consequently linearly dependent. If the field size is large enough, the probability that this happens vanishes. For practical systems however, the field size cannot be arbitrarily large. Therefore, studies are needed that investigate the decoding error probability on the message level. Further, the field size could be an additional constraint for the algorithms that optimize the coefficient vector. Since a finite field contains only a finite number of elements, it will be more efficient to solve the shortest-vector problem (SVP) over the finite field instead of solving it over $\mathbb{Z}$. However, the objective function of the optimization problem has to be adjusted because the computation rate depends on the lattice coefficients and not on the message coefficients.

Another question arises regarding the optimization criterion. All algorithms optimize the coefficient vector with respect to the maximal computation rate.

However, this computation rate is derived using infinite block length and will not be achieved in practical systems. Therefore, one might ask if this is the right optimization criterion for real-life applications. Another criterion could be the outage probability.

10.2 Untrusted Relay Channel

In Part III, we discussed untrusted relay channels and the achievable secrecy rate. Therein, we looked at multi-antenna scenarios but did not investigate the MIMO case. From the results for the SIMO case, we concluded that multiple antennas at the relay decrease the secrecy rate since the eavesdropper has more degrees of freedom. However, providing the legitimate users with an equal or higher amount of antennas might be beneficial again. This could be subject to further investigations.

10.3 Two-Way Relay Channel with External Eavesdropper

It is also possible to extend the scheme proposed in Chapter 7 to two-way relay channels with an external eavesdropper. This means that the relay is a trustworthy node but somewhere in the communication range there is a malicious node, which eavesdrops the communication. This eavesdropper has η_e antennas and is passive, which means it does not interfere with the communication. It receives the following signals in the MAC and BC phase, respectively,

$$y_{e1} = g_1 x_1 + g_2 x_2 + z_{e1}, \tag{10.1}$$

$$y_{e2} = g_3 x_r + z_{e2}, \tag{10.2}$$

where $g_1, g_2, g_3 \in \mathbb{R}^{\eta_e}$ and $z_{e1}, z_{e2} \sim \mathcal{N}(0, I_{\eta_e})$. In the first phase, the eavesdropper has two decoding options:

- aiming for the individual messages, or
- aiming for the linear combination of the messages.

In the second phase, it might get the linear combination that the relay decoded in the first phase. If the eavesdropper can decode two different linear combinations in the two phases, it is able to obtain the individual messages. The first phase can be protected by following the approach in Chapter 7. In the second phase, the relay sees a classical wiretap channel, for which the secrecy capacity is well known [55, 77] and given by

$$R_{\mathrm{BC},s}^{\mathrm{CF}} \triangleq R_{\mathrm{BC}}^{\mathrm{CF}} - \tfrac{1}{2}\log_2(1 + \|g_3\|^2 P). \tag{10.3}$$

Therefore, there are two possible protection strategies:

- We could allow the eavesdropper to decode a linear combination in the MAC phase and use a wiretap code in the BC phase to ensure that the eavesdropper does not get the linear combination decoded by the relay.
- We could alter the wiretap coding scheme in the MAC phase such that the eavesdropper is not able to decode a linear combination nor the individual messages. Thus, it is not necessary to protect the BC phase.

The interesting part is the MAC phase, where we need to secure the transmission. The following definitions correspond to the two strategies, respectively.

10.1 Definition (Weak Joint Secrecy). A transmission of two messages w_1, w_2 from two users to a relay is said to be weakly secure for joint secrecy if the following condition is met:

$$\lim_{n\to\infty} \tfrac{1}{n} I(W_1, W_2; Y_e^n) = 0, \tag{10.4}$$

where Y_e^n is a random variable representing the received signal at the eavesdropper. ◁

10.2 Definition (Weak Sum Secrecy). A transmission of two messages w_1, w_2 from two users to a relay is said to be weakly secure for sum secrecy if the following conditions are met simultaneously:

$$\lim_{n\to\infty} \tfrac{1}{n} I(W_1, W_2; Y_e^n) = 0, \tag{10.5a}$$

$$\lim_{n\to\infty} \tfrac{1}{n} I(W_\Sigma; Y_e^n) = 0, \tag{10.5b}$$

where Y_e^n is a random variable representing the received signal at the eavesdropper, and $W_\Sigma = W_1 \oplus W_2$ is a random variable representing the sum of the transmitted messages. ◁

How to ensure weak joint secrecy can directly be derived from the proof in Section 8.6, which results in the following theorem.

10.3 Theorem. In a MAC with an external eavesdropper, the following joint secrecy sum-rate

$$R_{s_1} + R_{s_2} \le R_{\mathrm{MAC}}^{\mathrm{CF,I}} \triangleq 2R_{\mathrm{CF}}(h,a) - \tfrac{1}{2}\log_2 \det(I + G_{\mathrm{MAC}}P) \tag{10.6}$$

is achievable under an average transmit power constraint $\|x_j\|^2 \le nP$ with $j \in \{1,2,r\}$ and $G_{\mathrm{MAC}} \triangleq g_1 g_1' + g_2 g_2'$. $\triangleleft$

If the eavesdropper aims for a linear combination w_Σ in the first phase, it tries to decode a linear combination v_e of lattice codewords, from which it can determine the linear combination of messages. The linear combination of lattice codewords is given by

$$v_e = [b_1 x_1 + b_2 x_2] \bmod \Lambda_C. \tag{10.7}$$

The eavesdropper can successfully decode v_e if the communication rates of the user nodes are below the achievable computation rate at the eavesdropper, i.e.,

$$R_1 \le R_{\mathrm{CF}}(G_{\mathrm{CF}}, b), \tag{10.8a}$$

$$R_2 \le R_{\mathrm{CF}}(G_{\mathrm{CF}}, b), \tag{10.8b}$$

where

$$R_{\mathrm{CF}}(G_{\mathrm{CF}}, b) = -\frac{1}{2}\log_2(b' V D V' b) \tag{10.9}$$

with $V \in \mathbb{R}^{2\times 2}$ being the right matrix of eigenvectors of G_{CF}, and $D \in \mathbb{R}^{2\times 2}$ being a diagonal matrix with elements

$$D_{ii} = \begin{cases} \frac{1}{P\lambda_i + 1} & i \le \operatorname{rank}(G_{\mathrm{CF}}) \\ 1 & i > \operatorname{rank}(G_{\mathrm{CF}}) \end{cases}, \tag{10.10}$$

where λ_i is the i-th eigenvalue of $G_{\mathrm{CF}}' G_{\mathrm{CF}}$, and G_{CF} is the channel matrix from the users to the eavesdropper (see [46, 47] for details). Intuitively, this results in the following conjecture.

10.4 Conjecture. In a MAC with an external eavesdropper, the following secrecy sum-rate

$$R_{s_1} + R_{s_2} \le \min\left\{ R_{\mathrm{MAC}}^{\mathrm{CF,I}}, R_{\mathrm{MAC}}^{\mathrm{CF,II}} \right\} \tag{10.11}$$

is achievable under an average transmit power constraint $\|x_j\|^2 \le nP$ with $j \in \{1,2,r\}$ and

$$\begin{aligned} R_{\mathrm{MAC}}^{\mathrm{CF,I}} &\triangleq 2R_{\mathrm{CF}}(h,a) - \tfrac{1}{2}\log_2 \det(I + G_{\mathrm{MAC}}P), \\ R_{\mathrm{MAC}}^{\mathrm{CF,II}} &\triangleq 2R_{\mathrm{CF}}(h,a) - 2R_{\mathrm{CF}}(G_{\mathrm{CF}}, b). \end{aligned}$$

$\triangleleft$

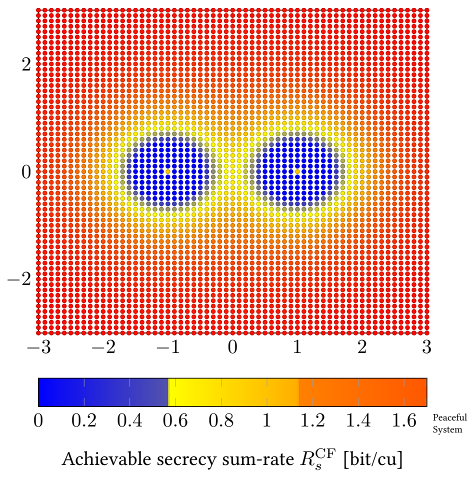
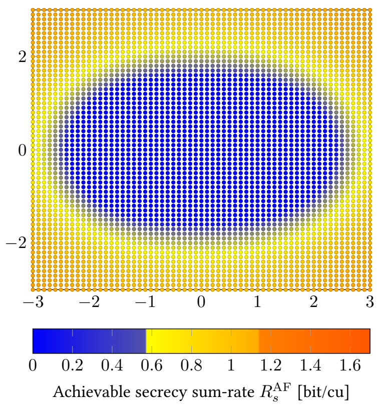


Figure 10.1: Achievable secrecy sum-rate with the proposed CF scheme for different eavesdropper locations.

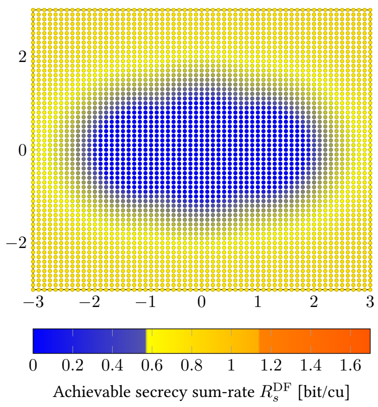
The two rates $R_{\mathrm{MAC}}^{\mathrm{CF,I}}$ and $R_{\mathrm{MAC}}^{\mathrm{CF,II}}$ correspond to the strategies of decoding the individual codewords or a linear combination of codewords at the eavesdropper, respectively. However, proving this conjecture is not trivial. Although, the steps are the same as in Section 8.6, we need to calculate the mutual information between the sum codeword X_Σ^n and the received signal Y_e^n. In order to be able to calculate this mutual information, we need the probability density function of Y_e^n, which is hard to calculate. Using a Gaussian distribution as an upper bound is not helpful because the bound is too loose. Therefore, the proof of Conjecture 10.4 remains an open topic.

If we assume that it is possible to prove Conjecture 10.4, we can investigate the achievable secrecy sum-rate depending on the positions of the nodes. This is shown in Fig. 10.1, where the user nodes and the relay are positioned in a line and the eavesdropper takes an arbitrary position. All simulations are performed with a transmit SNR equal to 10 dB for all users. The channel gains are proportional to $1/d^2$, where d denotes the distance between the users (large scale fading). The user nodes are positioned at $(-1, 0)$ and $(1, 0)$, whereas the relay is located at $(0, 0)$. The eavesdropper has one antenna.

It is interesting to note that around the user nodes there is an area where no



(a) Amplify-and-forward.



(b) Decode-and-forward.

Figure 10.2: Achievable secrecy sum-rate for different eavesdropper locations.

secure transmission is possible or only at a very low rate. In contrast, it is still possible to transmit data securely if the eavesdropper is close to the relay. This is similar to the results in Chapter 7.

In almost all cases, our simulations show that the eavesdropper is either not able to decode a linear combination or can only decode the same linear combination as the relay. Therefore, it is not necessary to protect the BC phase, and the sum secrecy constraint in the MAC phase can be removed. Only at a very few positions between the relay and the user nodes, the eavesdropper can decode a different linear combination than the relay and therefore either the BC phase needs to be protected or the sum secrecy constraint has to be used in the MAC phase such that the eavesdropper does not get a linear combination. Of course, this behavior depends on the concrete setup and the topology and has to be investigated for different positions of the user nodes and the relay.

In Fig. 10.2a, we show the achievable secrecy sum-rate with an amplify-and-forward (AF) scheme. When we compare Fig. 10.1 and Fig. 10.2a, we can see that the CF scheme outperforms the AF scheme. In Fig. 10.2b, we show the achievable secrecy sum-rate with a decode-and-forward (DF) scheme. This scheme is again outperformed by our CF scheme.

10.4 Implementation

In Part IV, we described a compute-and-forward implementation. Therein, we have already discussed a lot of possible improvements, such as algorithm optimization, implementation optimizations, etc. Nevertheless, we have a setup that allows us to evaluate certain system parameters. Especially the used codes can be varied, and their performance can be investigated. Table 10.1 lists some lattices up to dimension 5 and their properties, which are computed by the lattice implementation of `pyncsim`. Since we primarily choose the coarse lattice, which is responsible for shaping the nested lattice code, it is beneficial to choose a lattice with a shaping gain γ_S larger than one. However, the influence of the lattice properties in a real-life system needs to be evaluated.

Further, we have a concatenation of an inner and an outer code. Certainly, there will be a trade-off between the error correction properties of both codes. If there mostly are sample errors that are equally distributed over the received signal, it might be beneficial to correct those errors by the lattice code and therefore use a lattice code with a large Voronoi region. However, if the errors occur in bursts, it might be beneficial to correct them on the outer code level since the size of a symbol might be larger than the burst. Therefore, one burst error might result in only one symbol error but will affect a lot of lattice codewords. The influence of the combination of both codes should be further investigated.

Table 10.1: Some lattices and their properties.

Name	n	Δ	$d_{\min}$	Vol	det	γ_S	γ_C
A1*	1	0.5	1	1	1	1	1
A1	1	0.5	1.4142	1.4142	2	1	1
D1	1	0.5	1	1	1	1	1
Z	1	0.5	1	1	1	1	1
HZ40	2	0.7854	1	1	1	1	1
D2	2	0.7854	1.4142	2	4	1	1
D2*	2	0.7854	0.7071	0.5	0.25	1	1
Z2	2	0.7854	1	1	1	1	1
BGF.2.414	2	0.8112	2	3.873	15	0.851	1.0328
A2*	2	0.9069	1.4142	1.7321	3	1.0393	1.1547
A2	2	0.9069	1.4142	1.7321	3	1.0393	1.1547
D3*	3	0.5101	0.866	0.5	0.25	0.8296	1.1906
A3*	3	0.5101	1.7321	4	16	0.8296	1.1906
mcc	3	0.5208	1.0987	1	1	0.7644	1.2071
zcc	3	0.5236	2	6	36	0.9718	1.2114
D3	3	0.5554	1.4142	2	4	1.0583	1.2599
A3	3	0.5554	1.4142	2	4	1.0582	1.2599
Z4	4	0.3084	1	1	1	1	1
E(14)	4	0.3525	2	14	196	0.6267	1.069
S(7)	4	0.3965	1.7321	7	49	0.415	1.1339
A4*	4	0.4414	2	11.1803	125	0.7449	1.1963
QQF.4.b	4	0.4486	2	11	121	0.7664	1.206
QQF.4.f	4	0.4828	2.4495	23	529	0.5929	1.2511
A4	4	0.5517	1.4142	2.2361	5	1.0681	1.3375
D4*	4	0.6169	1	0.5	0.25	0.9376	1.4142
D4	4	0.6169	1.4142	2	4	0.9376	1.4142
F4	4	0.6169	1.4142	2	4	0.9376	1.4142
A5*	5	0.2395	2.2361	36	1296	0.6794	1.1925
A5	5	0.3561	1.4142	2.4495	6	1.0733	1.3977
D5	5	0.4362	1.4142	2	4	0.8046	1.5157

Chapter 11

Conclusion

This thesis provides a contribution to the field of physical-layer network coding. We investigated an $L \times M \times K$ relay network with L source nodes, M relay nodes, and K sink nodes, where the relay nodes perform network coding on the physical layer. We focused on the compute-and-forward framework, which provides a coding scheme based on nested lattice codes to enable physical-layer network coding. We showed that physical-layer network coding and compute-and-forward in particular are key enabling technologies to allow secure communication in the information-theoretic sense via an untrusted relay. We derived an achievable secrecy rate region and showed that this secrecy rate region is the difference between the MAC capacity region and the achievable compute-and-forward rate region. Further, it is interesting that reducing the transmit power can increase the achievable secrecy rate. We investigated the influence of multiple antennas on the achievable secrecy rate and concluded that multiple antennas at the user nodes increase the achievable secrecy rate significantly, whereas multiple antennas at the relay are counter-beneficial.

Within the compute-and-forward framework, it is crucial to find the optimal network coding coefficients, which depend on the channel realizations. In this thesis, we proposed a branch-and-bound algorithm to solve this problem. We studied the optimization problem and characterized the solution set. Although faster algorithms have been developed after the first publication of our algorithm [109], the branch-and-bound structure of our proposed algorithm makes it easy to adapt it to additional constraints on the solution set, e.g., non-zero coefficients. Therefore, we used this algorithm in our studies on the global optimization of the network coding coefficients. We described the problem of local optimization, i.e., the linear dependence of codeword combinations at the sink nodes in a larger network. We proposed different strategies that guarantee enough linear independent combinations such that the sink nodes can decode the individual codewords. We investigated the influence of spatial correlation on the outage probability of these schemes and showed that spatial correlation plays a crucial role in the performance of the different schemes. However, our proposed schemes are robust against spatial correlation and perform well under these conditions.

We implemented the compute-and-forward framework with software-defined radio and demonstrated the practical feasibility. Although the performance can significantly be enhanced if we improve the implementation, we gained valuable insights into the encoding and decoding in a realistic setup. We highlighted the most time-consuming steps in the encoding and decoding process and pointed out possible improvements.

Part VI

Appendix

Appendix A

Lattices and Lattice Codes

In this chapter, we introduce the basic definitions used in lattice theory. A very exhaustive overview of lattices and their properties is given by Conway and Sloane in [22]. A list of all known lattices can be found in [29]. There is a good introduction to lattice codes given by Zamir in his journal paper [34]. More recently, he wrote a very good book on lattice coding for signals and networks [35]. The most important definitions that are used to introduce the compute-and-forward framework can also be found in [40]. The following sections provide a summary of the previously mentioned references.

A.1 Lattices

A.1 Definition (Lattice). An n-dimensional lattice Λ is defined by a set of n basis (column) vectors $g_1, \ldots, g_n \in \mathbb{R}^m$. The lattice Λ is composed of all integral combinations of the basis vectors, i.e.,

$$\Lambda = \{G'z : z \in \mathbb{Z}^m\}, \tag{A.1}$$

where the $n \times m$ generator matrix G is given by $G = (g_1, \ldots, g_n)$. $\triangleleft$

A.2 Remark. The generator matrix G is not unique for a given lattice Λ. $\triangleleft$

Definition A.1 implies lattice properties that are important for channel codes, i.e.,

- $\Lambda \subset \mathbb{R}^n$,
- if $s, t \in \Lambda$, then $s + t \in \Lambda$,
- if $s \in \Lambda$, then $-s \in \Lambda$,
- if $\lambda \in \Lambda$, then $\Lambda = \lambda + \Lambda$, i.e., lattices are shift-invariant,
- Λ is an algebraic group, and
- $0 \in \Lambda$, i.e., the null vector is always a lattice point.

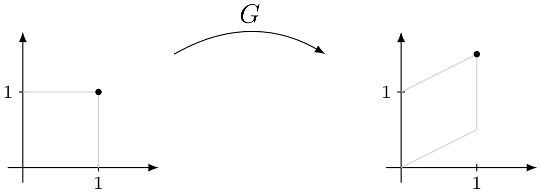


Figure A.1: Linear transformation of point $(1, 1)$.

A lattice is spanned by the n basis vectors $g_i \in \mathbb{R}^m$ as stated in Definition A.1. Therefore, a lattice is an image of the integer lattice $\mathbb{Z}^n$ according to a linear transformation of the vector space $\mathbb{R}^n$ with the linear operator G. This transformation is visualized in Fig. A.1. The parallelotope that consists of all points

$$a_1 g_1 + \cdots + a_n g_n, \text{ with } a_i \in [0, 1), \text{ for all } i \in \{1, 2, \ldots, n\} \tag{A.2}$$

is called a *fundamental parallelotope.* A fundamental parallelotope is one example of a *fundamental region* of a lattice. If such a region is repeated throughout the n-dimensional space, it covers the whole space, and each copy contains only one lattice point.

Another example of a fundamental region is the *fundamental Voronoi region.* We define a quantizer that maps all points in $\mathbb{R}$ to the nearest lattice point with respect to the Euclidean distance.

A.3 Definition (Nearest-Neighbor Quantizer). Let Λ be a lattice in n-dimensional space and $x \in \mathbb{R}^n$. The nearest-neighbor quantizer Q is defined as

$$Q(x) \triangleq \arg\min_{\lambda \in \Lambda} \|x - \lambda\|. \tag{A.3}$$

◁

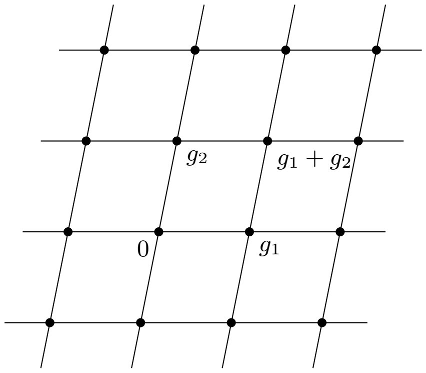
A.4 Definition (Quantization Error). The quantization error can be expressed by the modulo-Λ operation with respect to the lattice, which is defined as

$$x \bmod \Lambda = x - Q(x). \tag{A.4}$$

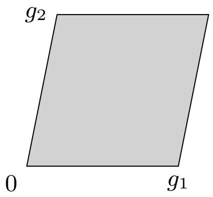
◁

For all $s, t \in \mathbb{R}^n$ and $\Lambda \subseteq \Lambda_1$, the $\bmod \Lambda$ operation satisfies:

$$[s + t] \bmod \Lambda = [[s] \bmod \Lambda + t] \bmod \Lambda, \tag{A.5}$$



(a) Lattice.



(b) Fundamental parallelotope.

Figure A.2: The plane is divided into fundamental parallelotopes of a two-dimensional lattice.

$$[Q_{\Lambda_1}(s)] \bmod \Lambda = [Q_{\Lambda_1}([s] \bmod \Lambda)] \bmod \Lambda, \tag{A.6}$$
$$[as] \bmod \Lambda = [a[s] \bmod \Lambda] \bmod \Lambda \text{ for all } a \in \mathbb{Z}, \tag{A.7}$$
$$\beta[s] \bmod \Lambda = [\beta s] \bmod \beta\Lambda \text{ for all } \beta \in \mathbb{R}. \tag{A.8}$$

Now, we can define the Voronoi region as follows.

A.5 Definition (Voronoi Region). The basic or fundamental Voronoi region of a lattice Λ is the set of all points in $\mathbb{R}^n$ closest to the zero point, i.e., $\mathcal{V}_0 = \{x : Q(x) = 0\}$. The Voronoi region associated with each $\ell \in \Lambda$ is the set of all points x such that $Q(x) = \ell$, and it is given by a shift of $\mathcal{V}_0$ by ℓ. $\triangleleft$

There are several possible ways to define a basis and a fundamental region of a lattice Λ. However, the volume of the fundamental region is unique for a given lattice. The square of the volume of the fundamental region is called the *determinant* of the lattice and is an important characterization of the lattice.

The matrix
$$M \triangleq GG' \tag{A.9}$$
is called *Gram matrix* of the lattice. The (i, j)-th entry is the inner product of the basis vectors $g_i g_j$ of the lattice. Thus, the determinant of a lattice Λ is the determinant of the matrix M,
$$\det(\Lambda) = \det(M). \tag{A.10}$$

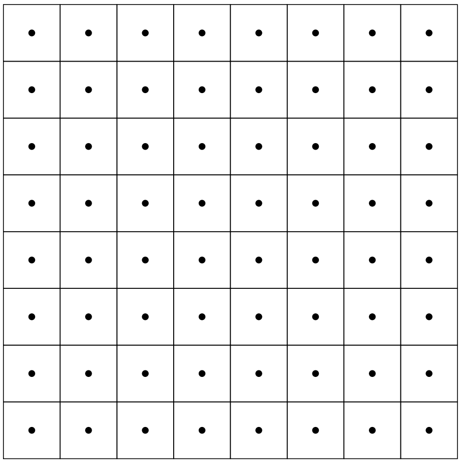


Figure A.3: Integer Lattice $\mathbb{Z}^2$.

If G is a square matrix, this simplifies to

$$\det(\Lambda) = [\det(G)]^2. \tag{A.11}$$

A.6 Example (Integer Lattice $\mathbb{Z}^2$). The integer lattice $\mathbb{Z}^2$ has the identity matrix as a generator matrix, i.e.,

$$\Lambda_{\mathbb{Z}^2} = \{I_2 z \mid z \in \mathbb{Z}^2\}. \tag{A.12}$$

Therefore, the determinant is

$$\det(\Lambda_{\mathbb{Z}^2}) = [\det(I_2)]^2 = 1, \tag{A.13}$$

and the volume of the fundamental region is also 1. ◁

A.7 Example (Root lattice A_2). The root lattice A_2 has the following generator matrix

$$G_{\Lambda_{A_2}} = \begin{pmatrix} 1 & 0 \\ \frac{1}{2} & \frac{1}{2}\sqrt{3} \end{pmatrix}. \tag{A.14}$$

The Gram matrix is

$$M = GG' = \begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & 1 \end{pmatrix}, \tag{A.15}$$

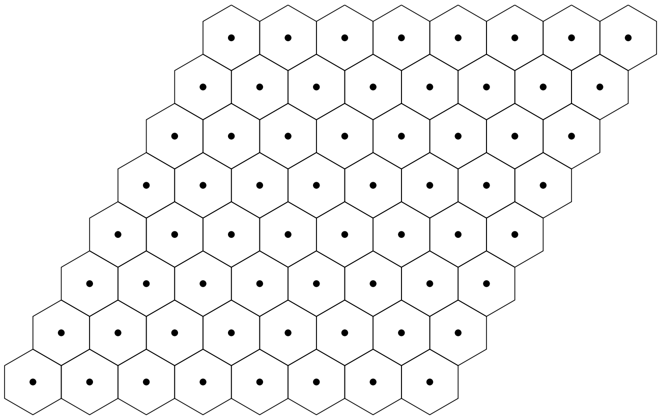


Figure A.4: Root lattice A_2.

and the determinant is

$$\det(\Lambda_{A_2}) = [\det(G_{\Lambda_{A_2}})]^2 = \tfrac{3}{4}. \tag{A.16}$$

Therefore, the volume of the fundamental region is $\sqrt{\frac{3}{4}}$. ◁

The generator matrix in Example A.7 looks not very nice, and sometimes it is beneficial to use the following generator matrix:

$$\tilde{G}_{\Lambda_{A_2}} = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{pmatrix}. \tag{A.17}$$

The Gram matrix is then given by

$$\tilde{M} = \tilde{G}\tilde{G}' = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}, \tag{A.18}$$

and the determinant is now $\det(\Lambda_{A_2}) = 3$. Both Eqs. (A.14) and (A.17) describe the hexagonal lattice A_2 but with different coordinates and a different scaling. Equation (A.17) describes a two-dimensional lattice in a three-dimensional space that lies in the plane $x + y + z = 0$. We call both lattices *equivalent*. It can be shown that any lattices generated by generator matrices G_1 and G_2 are equivalent if and only if there exist matrices W and U such that

$$G_2 = \left(\frac{V_2}{V_1}\right)^{\frac{1}{n}} W G_1 U, \tag{A.19}$$

where all elements of W are integers, W has determinant ± 1, and U is orthonormal. V_1 and V_2 represent the volume of the lattices generated by G_1 and G_2, respectively. The coefficient $(V_2/V_1)^{1/n}$ takes care of scaling, W of basis change, and U of rotation and / or reflection.

A.8 Definition (Packing). The set $\Lambda + r\mathcal{B}$ is called a packing for a given radius r if

$$(x + r\mathcal{B}) \cap (y + r\mathcal{B}) = \emptyset \tag{A.20}$$

holds for all lattice points $x, y \in \Lambda$ with $x \neq y$, where $\mathcal{B}$ is the unit ball in Euclidean space. $\triangleleft$

This means that the balls do not overlap.

A.9 Definition (Packing Radius). The packing radius r_Λ^{pack} of a lattice is defined as

$$r_\Lambda^{\text{pack}} \triangleq \sup\{r : \Lambda + r\mathcal{B} \text{ is a packing}\}. \tag{A.21}$$

$\triangleleft$

The packing radius r_Λ^{pack} is the radius of the largest n-dimensional ball in Euclidean space that lays completely within the Voronoi region $\mathcal{V}_0$.

A.10 Definition (Covering). The set $r\mathcal{B}$ consisting of balls centered at a lattice point is called a covering of Euclidean space if

$$\mathbb{R} \subseteq \Lambda + r\mathcal{B}. \tag{A.22}$$

$\triangleleft$

This means that every point in space is covered by at least one ball.

A.11 Definition (Covering Radius). The covering radius r_Λ^{cov} of a lattice is defined as

$$r_\Lambda^{\text{cov}} \triangleq \min\{r : \Lambda + r\mathcal{B} \text{ is a covering}\} \tag{A.23}$$

$\triangleleft$

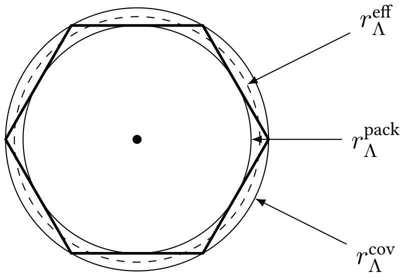


Figure A.5: Lattice point with Voronoi region and the respective radii.

The covering radius r_Λ^{cov} is the radius of the smallest n-dimensional ball in Euclidean space that covers the complete Voronoi region $\mathcal{V}_0$.

A.12 Definition (Effective Radius). We call r_Λ^{eff} the effictive radius of a lattice if

$$V_{\mathcal{B}}(r_\Lambda^{\text{eff}}) = \text{Vol}\,(\mathcal{V})\,, \tag{A.24}$$

where $V_{\mathcal{B}}(r)$ is the volume of a ball with radius r. ◁

There are several figure-of-merits to characterize a lattice. The following are related to the Voronoi region.

- The *volume* $\text{Vol}\,(\Lambda)$ of a lattice Λ is defined to be the volume of the fundamental region of the lattice and can be calculated by

$$\text{Vol}\,(\Lambda) = \sqrt{\det(GG')} = \sqrt{\det(\Lambda)}. \tag{A.25}$$

- The *density* $\Delta(\Lambda)$ of a lattice Λ is the proportion of space that is occupied by the balls that form a packing and can be calculated by

$$\Delta(\Lambda) = \frac{\text{volume of one ball}}{\text{volume of the fundamental region}}. \tag{A.26}$$

- The *packing efficiency* is the ratio of the packing radius and the effective radius, i.e.,

$$\rho_{\text{pack}}(\Lambda) \triangleq \frac{r_\Lambda^{\text{pack}}}{r_\Lambda^{\text{eff}}}. \tag{A.27}$$

  It follows that $0 \le \rho_{\text{pack}}(\Lambda) \le 1$.

- The *covering efficiency* is the ratio of the covering radius and the effective radius, i.e.,

  $$\rho_{\mathrm{cov}}(\Lambda) \triangleq \frac{r_\Lambda^{\mathrm{cov}}}{r_\Lambda^{\mathrm{eff}}}. \tag{A.28}$$

  It follows that $\rho_{\mathrm{cov}}(\Lambda) \geq 1$.

A.2 Lattice Constellations

Lattices are infinite. For coding however, we need a finite number of codewords that meet the power constraint. Therefore, we cut a finite number of lattice points out of the lattice and form a lattice constellation.

A.13 Definition (Lattice Constellation). A lattice constellation $C(\Lambda, R)$ is a countable number of points of a lattice Λ (or a shifted version $\Lambda + x$ with $x \in \mathbb{R}^n$) which lay within a closed and bounded region R of the n-dimensional space, i.e.,

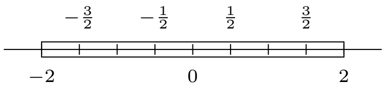
$$C(\Lambda, R) = (\Lambda + x) \cap R,\ x \in \mathbb{R}^n. \tag{A.29}$$

◁

A.14 Remark. The region R is called *shaping region.* ◁

A.15 Example (4-PAM). The 4-ary pulse amplitude modulation is an example of a lattice constellation with the following values:

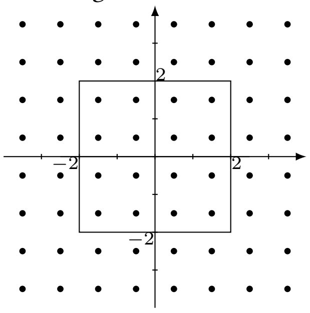
$$\Lambda + \lambda = \mathbb{Z} + \tfrac{1}{2},$$
$$R = [-2, 2].$$



◁

A.16 Example (16-QAM). The 16-ary quadrature amplitude modulation is an example for a lattice constellation with the following values:

$$\Lambda + \lambda = \mathbb{Z}^2 + \left(\tfrac{1}{2}, \tfrac{1}{2}\right),$$
$$R = [-2, 2]^2.$$



◁

A.17 Definition (Minimal Distance). The minimal distance $d_{\min}$ of a lattice constellation C is the minimal distance between any two points in the constellation, i.e.,

$$d_{\min} = \min_{\substack{x \in C \\ y \in C}} \|x - y\|. \tag{A.30}$$

◁

Let $x \in C$ be a point in the lattice constellation C, and let $|C|$ be the cardinality of C. The average energy of all constellation points is

$$\mathrm{E}\big[\|x\|^2\big] = \frac{1}{|C|}\|x\|^2. \tag{A.31}$$

We assume a uniform distribution of the lattice points within the shaping region R, i.e.,

$$\Pr(x) = \begin{cases} \frac{1}{\operatorname{Vol}(R)} & x \in R \\ 0 & \text{otherwise} \end{cases}, \tag{A.32}$$

where $\operatorname{Vol}(R)$ specifies the volume of the shaping region R. Then, we can calculate the average energy per dimension of the region R as

$$P(R) = \int_R \underset{\substack{\uparrow \\ \text{probability distribution}}}{\frac{1}{\operatorname{Vol}(R)}} \frac{\overset{\substack{\text{energy of } x \\ \downarrow}}{\|x\|^2}}{\underset{\substack{\uparrow \\ \text{normalize by dimension}}}{n}} \, \mathrm{d}x. \tag{A.33}$$

This is equivalent to the second moment of the region. Therefore, it is often denoted by σ^2. We define the normalized second moment of a shaping region R as follows

$$G(R) = \frac{P(R)}{\operatorname{Vol}(R)^{2/n}}. \tag{A.34}$$

A.18 Remark. Assume that the shaping region is scaled by α. Then, $P(R)$ scales with α^2 and $\operatorname{Vol}(R)$ with α^n. It follows that $G(R)$ is independent of the scaling. ◁

A.19 Example. Lets have a look at the 1-dimensional space and the region $R = [-1, 1]$. The volume of this region is $\mathrm{Vol}(R) = 2$ and the average energy per dimension or the second moment of the region can be calculated as follows

$$P(R) = \int_{-1}^{1} \frac{1}{2} x^2 \mathrm{d}\, x = \left[\frac{1}{6} x^3\right]_{-1}^{1} = \frac{1}{3}. \tag{A.35}$$

The normalized second moment is given by

$$G([-1,1]) = \frac{\frac{1}{3}}{2^2} = \frac{1}{12}. \tag{A.36}$$

◁

A.20 Remark. The normalized second moment of a m-cube that is centered at the origin is independent from its scaling and namely

$$G([-1,1]^m) = \frac{1}{12}. \tag{A.37}$$

◁

With the previous definitions, we can define two figure-of-merits of a lattice constellation, which are especially important for coding. The first one is the *shaping gain*, which provides a statement regarding the shaping region of the constellation. It quantifies the average reduction in energy with respect to the m-cube and is defined as follows

$$\gamma_S(R) = \frac{\frac{1}{12}}{G(R)}. \tag{A.38}$$

The second figure-of-merit refers to the lattice of the constellation. The *(nominal) coding gain* indicates the increase of the density of the lattice with respect to the integer lattice $\mathbb{Z}^n$ and is defined as follows

$$\gamma_C(\Lambda) = \frac{d_{\min}^2(\Lambda)}{V(\Lambda)^{2/n}}. \tag{A.39}$$

A.21 Remark. The nominal coding gain is invariant regarding scaling. Lets assume that the lattice is scaled by α. Then, $d_{\min}$ is scaled by α and $\mathrm{Vol}(\Lambda)$ is scaled by α^n. It is easy to see that $\gamma_C(\Lambda)$ is independent of the scaling α. ◁

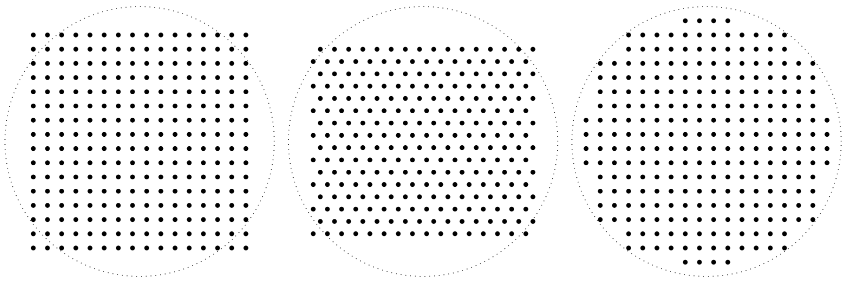


Figure A.6: A lattice constellation and two alternative constellations that illustrate the shaping and coding gain. All three constellations have the same minimal distance, and each has 256 points [78].

A.3 Nested Lattice Codes

In Appendix A.2, we showed how to use lattices for coding. So far, a lattice constellation consists of a lattice and an arbitrary shaping region. However, the shaping region can be chosen to be the fundamental Voronoi region of another lattice. This leads to the concept of nested lattices and nested lattice codes.

A.22 Definition (Nested Lattice). A pair of n-dimensional lattices (Λ_1, Λ_2) is called nested if $\Lambda_2 \subset \Lambda_1$, i.e., there exist corresponding generator matrices G_1 and G_2 such that

$$G_2 = G_1 J, \tag{A.40}$$

where $J \in \mathbb{Z}^{n\times n}$ and $\det(J) > 1$. ◁

The volumes of the Voronoi regions of Λ_1 and Λ_2 satisfy

$$V_2 = \det(J) V_1, \tag{A.41}$$

where $V_2 = \mathrm{Vol}\left(\mathcal{V}_{0,2}\right)$ and $V_1 = \mathrm{Vol}\left(\mathcal{V}_{0,1}\right)$. We call $\sqrt[n]{\det(J)} = \sqrt[n]{V_2/V_1}$ the *nesting ratio.*

A.23 Definition (Coset Leader). The points of the set

$$\Lambda_1 \bmod \Lambda_2 \triangleq \Lambda_1 \cap \mathcal{V}_{0,2} \tag{A.42}$$

are called the coset leaders of Λ_2 relative to Λ_1. ◁

For each $v \in \{\Lambda_1 \bmod \Lambda_2\}$, the shifted lattice $\Lambda_{2,v} = v + \Lambda_2$ is called a *coset* of Λ_2 relative to Λ_1. There are $V_2/V_1 = \det(J)$ different cosets.

A.24 Definition (Nested Lattice Code). A nested lattice code $\mathcal{L}$ is the set of all points of a fine lattice Λ_1 that are within the fundamental Voronoi region $\mathcal{V}$ of a coarse lattice Λ, i.e.,

$$\mathcal{L} = \Lambda_1 \cap \mathcal{V} = \{\lambda \bmod \Lambda,\ \lambda \in \Lambda_1\}. \tag{A.43}$$

$\triangleleft$

The probability of decoding error is the probability that the noise leaves the Voronoi region of the transmitted lattice point, i.e.,

$$P_e = \Pr\{z \notin \mathcal{V}_0\}. \tag{A.44}$$

A.25 Definition (Volume-to-Noise Ratio). The volume-to-noise ratio (VNR) of a lattice at probability of error P_e is defined as

$$\mu(\Lambda, P_e) = \frac{\operatorname{Vol}(\mathcal{V})^{2/n}}{\sigma^2}, \tag{A.45}$$

where σ^2 is chosen such that Eq. (A.44) is satisfied with equality. $\triangleleft$

The minimum possible value of $\mu(\Lambda, P_e)$ over all lattices in $\mathbb{R}^n$ is denoted by $\mu_n(P_e)$. It follows from a paper by Poltyrev [30] that

$$\lim_{n\to\infty} \mu_n(P_e) = 2\pi e,\ \forall\, 0 < P_e < 1. \tag{A.46}$$

A.3.1 Construction A

There have been several algebraic constructions proposed to obtain lattice codes. The one used throughout the thesis is called *Construction A*. For more detailed information on algebraic constructions for lattices, we refer the reader to [22, Ch.8]. In the following, we provide a summary of Construction A, where we follow the presentation in [40].

1. Use a coarse lattice Λ of dimension n scaled such that its second moment is equal to P. Let $B \in \mathbb{R}^{n\times n}$ denote the generator matrix of this lattice.
2. Draw a matrix $G_L \in \mathbb{F}_p^{n\times k_L}$ with every element chosen i.i.d. according to the uniform distribution over $\{0, 1, \dots, p-1\}$. Recall that p is prime.

3. Define the codebook $\mathcal{C}_L$ as follows:

$$\mathcal{C}_L = \{G_L w : w \in \mathbb{F}_p^{k_L}\}. \tag{A.47}$$

   All operations in this step are over $\mathbb{F}_p$.

4. Form the lattice $\tilde{\Lambda}$ by projecting the codebook into the reals by

$$g : \mathbb{F}_p \to \{0, 1, \ldots, p-1\}, \tag{A.48}$$

   scale it down by a factor of p, and place a copy at every integer vector. This tiles the codebook over $\mathbb{R}^n$, i.e.,

$$\tilde{\Lambda} = p^{-1} g(\mathcal{C}) + \mathbb{Z}^n. \tag{A.49}$$

5. Rotate $\tilde{\Lambda}_L$ by the generator matrix of the coarse lattice to get the fine lattice for transmitter L, i.e.,

$$\Lambda_L = B\tilde{\Lambda}_L. \tag{A.50}$$

6. Repeat all steps for each transmitter $\ell = L-1, L-2, \ldots, 1$ by replacing G_L with G_ℓ, which is defined to be the first k_ℓ columns of G_L.

A.26 Remark. The generator matrix B of the coarse lattice is the transformation matrix of the coarse lattice regarding the integer lattice. $\triangleleft$

Appendix B

Two-Way Relay Channel Sum-Rate

In this chapter, we provide the mathematical background for Fig. 1.1. We investigate a two-way relay channel as depicted in Fig. B.1. All channel coefficients are assumed to be one as well as the noise power. The SNR, which is defined as the ratio of transmit and noise power, is equal for all nodes. The compute-and-forward scheme decodes a linear combination with coefficients $a = (1, 1)$. The achievable rates are obtained as stated below.

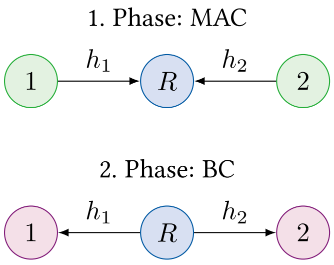


Figure B.1: Two-way relay channel.

B.1 Upper Bound

As an upper bound, we assume the interference-free communication, i.e., only one node sends at a time. It can be derived as follows.

Phase 1

$$R_1 = \frac{1}{2}\log_2(1 + h_1^2 P_1) \tag{B.1}$$

$$R_2 = \frac{1}{2}\log_2(1 + h_2^2 P_2) \tag{B.2}$$

Phase 2

$$R_{r1} = \frac{1}{2}\log_2(1 + h_1^2 P_r) \tag{B.3}$$

$$R_{r2} = \frac{1}{2}\log_2(1 + h_2^2 P_r) \tag{B.4}$$

Sum-rate

$$R_1 + R_2 = \min\{R_1, R_{r1}\} + \min\{R_2, R_{r2}\} \tag{B.5}$$

B.2 Decode-and-Forward

A relay that uses the Decode-and-forward strategy decodes both transmitted symbols in the MAC phase. Therefore, it uses SIC. The decoded messages are linearly combined to a new codeword and broadcasted to the receivers. The receivers can subtract their own messages to get the message of the other user.

Phase 1

$$R_1 = \frac{1}{2}\log_2\left(1 + \frac{h_1^2 P_1}{1 + h_2^2 P_2}\right) \tag{B.6}$$

$$R_2 = \frac{1}{2}\log_2(1 + h_2^2 P_2) \tag{B.7}$$

or

$$R_1 = \frac{1}{2}\log_2(1 + h_1^2 P_1) \tag{B.8}$$

$$R_2 = \frac{1}{2}\log_2\left(1 + \frac{h_2^2 P_2}{1 + h_1^2 P_1}\right) \tag{B.9}$$

Phase 2

$$R_r = \frac{1}{2}\log_2(1 + \min\{h_1^2, h_2^2\}P_r) \tag{B.10}$$

Sum-rate

$$R_1 + R_2 = \frac{\min\{R_1, R_r\} + \min\{R_2, R_r\}}{2} \tag{B.11}$$

B.3 Amplify-and-Forward

A relay that uses the Amplify-and-forward strategy forwards the received signal after amplifying it. The user nodes can subtract their own signal and decode the signal of the other user. The noise at the relay is also amplified and forwarded.

Phase 1

$$y_r = h_1 x_1 + h_2 x_2 + z_r \tag{B.12}$$

$$\alpha = \sqrt{\frac{P_r}{h_1^2 P_1 + h_2^2 P_2 + n_r}} \quad \text{(Amplification Factor)} \tag{B.13}$$

Phase 2

$$y_1 = h_1\, \alpha\, y_r + z_1 \tag{B.14}$$

$$y_2 = h_2\, \alpha\, y_r + z_2 \tag{B.15}$$

Sum-rate

$$y_1 = h_1\, \alpha\, (h_1 x_1 + h_2 x_2 + z_r) + z_1 \tag{B.16}$$

$$\hat{x}_2 = h_1\, \alpha\, h_2 x_2 + h_1\, \alpha\, z_r + z_1 \tag{B.17}$$

$$\mathrm{SNR}_1 = \frac{h_1^2 h_2^2 \alpha^2 P_2}{h_1^2 \alpha^2 n_r + n_1} \tag{B.18}$$

$$\mathrm{SNR}_2 = \frac{h_1^2 h_2^2 \alpha^2 P_1}{h_2^2 \alpha^2 n_r + n_2} \tag{B.19}$$

$$R_1 + R_2 = \frac{\frac{1}{2}\log_2(1 + \mathrm{SNR}_1) + \frac{1}{2}\log_2(1 + \mathrm{SNR}_2)}{2} \tag{B.20}$$

B.4 Compute-and-Forward

A relay that uses the Compute-and-forward strategy decodes a linear combination of the transmitted symbols with the help of lattice codes. It does not need to decode the individual symbols.

Phase 1

$$R_{CF} = \frac{1}{2}\log_2^{+}\left(\left(\|a\|^2 - \frac{P(h'a)^2}{1 + P\|h\|^2}\right)^{-1}\right) \tag{B.21}$$

Phase 2

$$R_r = \min\left\{\frac{1}{2}\log_2(1 + h_1^2 P_r), \frac{1}{2}\log_2(1 + h_2^2 P_r)\right\} \tag{B.22}$$

Sum-rate

$$R_1 + R_2 = \frac{\min\{2R_{CF}, 2R_r\}}{2} \tag{B.23}$$

Appendix C

Bibliography

Network Coding

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. “Network information flow”. In: *IEEE Transactions on Information Theory* 46.4 (July 2000), pp. 1204–1216.

[2] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran. “Network Coding for Distributed Storage Systems”. In: *IEEE Transactions on Information Theory* 56.9 (Sept. 2010), pp. 4539–4551.

[3] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh. “A Survey on Network Codes for Distributed Storage”. In: *Proceedings of the IEEE* 99.3 (Mar. 2011), pp. 476–489.

[4] F. H. P. Fitzek, J. Heide, M. V. Pedersen, and M. Katz. “Implementation of Network Coding for Social Mobile Clouds [Applications Corner]”. In: *IEEE Signal Processing Magazine* 30.1 (Jan. 2013), pp. 159–164.

[5] C. Fragouli and E. Soljanin. *Network Coding Fundamentals.* Now Publishers, 2007.

[6] J. Hansen, D. E. Lucani, J. Krigslund, M. Médard, and F. H. P. Fitzek. “Network coded software defined networking: Enabling 5G transmission and storage networks”. In: *IEEE Communications Magazine* 53.9 (Sept. 2015), pp. 100–107.

[7] N. J. Hernández Marcano, J. Heide, D. E. Lucani, and F. H. P. Fitzek. “Throughput, energy and overhead of multicast device-to-device communications with network-coded cooperation”. In: *Transactions on Emerging Telecommunications Technologies* (2016).

[8] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. “A Random Linear Network Coding Approach to Multicast”. In: *IEEE Transactions on Information Theory* 52.10 (Oct. 2006), pp. 4413–4430.

[9] R. Koetter and M. Médard. “An algebraic approach to network coding”. In: *IEEE/ACM Transactions on Networking* 11.5 (Oct. 2003), pp. 782–795.

[10] S.-Y. R. Li, R. W. Yeung, and N. Cai. “Linear network coding”. In: *IEEE Transactions on Information Theory* 49.2 (Feb. 2003), pp. 371–381.

[11] M. Médard, F. H. P. Fitzek, M.-J. Montpetit, and C. Rosenberg. “Network coding mythbusting: Why it is not about butterflies anymore”. In: *IEEE Communications Magazine* 52.7 (July 2014), pp. 177–183.

[12] F. Oggier and A. Datta. “Coding Techniques for Repairability in Networked Distributed Storage Systems”. In: *Foundations and Trends in Communications and Information Theory.* Vol. 9. 4. now Publishers, 2012, pp. 383–466.

[13] A. Paramanathan, M. V. Pedersen, D. E. Lucani, F. H. P. Fitzek, and M. Katz. “Lean and mean: Network coding for commercial devices”. In: *IEEE Wireless Communications* 20.5 (Oct. 2013), pp. 54–61.

[14] M. Sipos, J. Heide, D. E. Lucani, M. V. Pedersen, F. H. P. Fitzek, and H. Charaf. “Adaptive Network Coded Clouds: High Speed Downloads and Cost-Effective Version Control”. In: *IEEE Transactions on Cloud Computing* (2015).

[15] J. K. Sundararajan, D. Shah, M. Médard, S. Jakubczak, M. Mitzenmacher, and J. Barros. “Network Coding Meets TCP: Theory and Implementation”. In: *Proceedings of the IEEE* 99.3 (Mar. 2011), pp. 490–512.

[16] P. Torres Compta, F. H. P. Fitzek, and D. E. Lucani. “Network coding is the 5G Key Enabling Technology: Effects and strategies to manage heterogeneous packet lengths”. In: *Transactions on Emerging Telecommunications Technologies* 26.1 (2015), pp. 46–55.

[17] R. W. Yeung. *Information Theory and Network Coding.* Springer, 2008.

[18] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang. *Network Coding Theory Part I: Single Source.* Now, 2005.

[19] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang. *Network Coding Theory Part II: Multiple Source.* Now, 2005.

Lattice Codes

[20] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. “Closest point search in lattices”. In: *IEEE Transactions on Information Theory* 48.8 (2002), pp. 2201–2214.

[21] J.-C. Belfiore. “Lattice codes for the compute-and-forward protocol: The flatness factor”. In: *IEEE Information Theory Workshop (ITW).* Oct. 2011, pp. 1–4.

[22] J. H. Conway and N. J. A. Sloane. *Sphere Packings, Lattices and Groups.* 3rd ed. Springer, 1999.

[23] P. van Emde Boas. *Another NP-complete problem and the complexity of computing short vectors in a lattice.* Report 81-04. Mathematisch Instituut, Amsterdam, The Netherlands, Apr. 1981.

[24] U. Erez and R. Zamir. “Achieving 1/2 log (1+ SNR) on the AWGN channel with lattice encoding and decoding”. In: *IEEE Transactions on Information Theory* 50.10 (2004), pp. 2293–2314.

[25] G. Hanrot, X. Pujol, and D. Stehlé. “Algorithms for the Shortest and Closest Lattice Vector Problems”. In: *Proceedings of the Third International Conference on Coding and Cryptology.* IWCC’11. June 2011, pp. 159–190.

[26] L.-J. Hu, C. Duan, D.-F. Zhao, and X. Liao. “Study Status and Prospect of Lattice Codes in Wireless Communication”. In: *Fifth International Conference on Instrumentation and Measurement, Computer, Communication and Control (IMCCC).* Sept. 2015, pp. 593–598.

[27] L. Liu and C. Ling. “Polar Codes and Polar Lattices for Independent Fading Channels”. In: *IEEE Transactions on Communications* PP.99 (2016), pp. 1–1.

[28] D. Micciancio. “The hardness of the closest vector problem with preprocessing”. In: *IEEE Transactions on Information Theory* 47.3 (Mar. 2001), pp. 1212–1215.

[29] G. Nebe and N. J. A. Sloane. *A Catalogue of Lattices.* 2016. URL: `http://www.math.rwth-aachen.de/~Gabriele.Nebe/LATTICES/` (visited on 12/28/2016).

[30] G. Poltyrev. “On coding without restrictions for the AWGN channel”. In: *IEEE Transactions on Information Theory* 40.2 (Mar. 1994), pp. 409–417.

[31] J. A. Sheppard and A. G. Burr. “The design and implementation of lattice codes using digital signal processing techniques”. In: *Sixth International Conference on Digital Processing of Signals in Communications.* 1991, pp. 256–261.

[32] N. Sommer, M. Feder, and O. Shalvi. “Low-Density Lattice Codes”. In: *IEEE Transactions on Information Theory* 54.4 (Apr. 2008), pp. 1561–1585.

[33] N. Wang and J. D. Gibson. “Leech lattice coding and modulation for IEEE 802.11a WLAN”. In: *IEEE Emerging Technologies Symposium on BroadBand Communications for the Internet Era Symposium digest.* 2001, pp. 38–42.

[34] R. Zamir. “Lattices are everywhere”. In: *Information Theory and Applications Workshop.* Feb. 2009, pp. 392–421.

[35] R. Zamir. *Lattice Coding for Signals and Networks.* Cambridge University Press, 2014.

Physical Layer Network Coding

[36] Z. Chen, B. Xia, Z. Hu, and H. Liu. “Design and Analysis of Multi-Level Physical-Layer Network Coding for Gaussian Two-Way Relay Channels”. In: *IEEE Transactions on Communications* 62.6 (June 2014), pp. 1803–1817.

[37] C. Feng, D. Silva, and F. R. Kschischang. “An Algebraic Approach to Physical-Layer Network Coding”. In: *IEEE Transactions on Information Theory* 59.11 (Nov. 2013), pp. 7576–7596.

[38] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft. “XORs in the Air: Practical Wireless Network Coding”. In: *IEEE/ACM Transactions on Networking* 16.3 (June 2008), pp. 497–510.

[39] B. Nazer. “Successive Compute-and-Forward”. In: *Proceedings of the 22nd Biennial International Zurich Seminar on Communication.* Mar. 2012. URL: `http://iss.bu.edu/bobak/bn_izs12.pdf`.

[40] B. Nazer and M. Gastpar. “Compute-and-Forward: Harnessing Interference Through Structured Codes”. In: *IEEE Transactions on Information Theory* 57.10 (Oct. 2011), pp. 6463–6486.

[41] B. Nazer and M. Gastpar. “Reliable Physical Layer Network Coding”. In: *Proceedings of the IEEE* 99.99 (2011), pp. 438–460.

[42] S. Sahraei and M. Gastpar. “Compute-and-Forward: Finding the Best Equation”. In: *52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton).* Sept. 2014, pp. 227–233.

[43] J. Sýkora and J. Hejtmánek. “Joint and Recursive Hierarchical Interference Cancellation with Successive Compute & Forward Decoding”. In: *COST IC1004.* May 2014.

[44] L. Wei and W. Chen. "Compute-and-Forward Network Coding Design over Multi-Source Multi-Relay Channels". In: *IEEE Transactions on Wireless Communications* 11.9 (Sept. 2012), pp. 3348–3357.

[45] J. Wen, B. Zhou, W. H. Mow, and X.-W. Chang. "Compute-and-Forward Protocol Design Based on Improved Sphere Decoding". In: *IEEE International Conference on Communications (ICC)*. Oct. 2015, pp. 1631–1636.

[46] J. Zhan, B. Nazer, U. Erez, and M. Gastpar. "Integer-forcing linear receivers". In: *IEEE International Symposium on Information Theory Proceedings (ISIT)*. 2010, pp. 1022–1026.

[47] J. Zhan, B. Nazer, U. Erez, and M. Gastpar. "Integer-Forcing Linear Receivers". In: *IEEE Transactions on Information Theory* 60.12 (Dec. 2014), pp. 7661–7685.

[48] S. Zhang and S. C. Liew. "Applying Physical-Layer Network Coding in Wireless Networks". In: *EURASIP Journal on Wireless Communications and Networking* 2010 (2010), pp. 1–12.

[49] S. Zhang, S. C. Liew, and P. P. Lam. "Hot topic: Physical-Layer Network Coding". In: *Proceedings of the 12th annual international conference on Mobile computing and networking*. Vol. 1. Node 1. 2006, pp. 358–365.

[50] B. Zhou and W. H. Mow. "A Quadratic Programming Relaxation Approach to Compute-and-Forward Network Coding Design". In: *IEEE International Symposium on Information Theory (ISIT)*. June 2014, pp. 2296–2300.

Secrecy

[51] J.-C. Belfiore and F. Oggier. "Secrecy gain: A wiretap lattice code design". In: *International Symposium on Information Theory and its Applications (ISITA)*. Oct. 2010, pp. 174–178.

[52] M. R. Bloch. "Achieving secrecy: Capacity vs. resolvability". In: *IEEE International Symposium on Information Theory Proceedings (ISIT)*. Aug. 2011, pp. 633–637.

[53] M. R. Bloch and J. Barros. *Physical-Layer Security. From Information Theory to Security Engineering*. Cambridge University Press, 2011.

[54] L.-C. Choo, C. Ling, and K.-K. Wong. “Achievable Rates for Lattice Coded Gaussian Wiretap Channels”. In: *IEEE International Conference on Communications Workshops (ICC)*. June 2011, pp. 1–5.

[55] I. Csiszár and J. Körner. “Broadcast channels with confidential messages”. In: *IEEE Transactions on Information Theory* 24.3 (1978), pp. 339–348.

[56] X. He and A. Yener. “Providing secrecy with lattice codes”. In: *46th Annual Allerton Conference on Communication, Control, and Computing*. Sept. 2008, pp. 1199–1206.

[57] X. He and A. Yener. “Cooperation With an Untrusted Relay: A Secrecy Perspective”. In: *IEEE Transactions on Information Theory* 56.8 (Aug. 2010), pp. 3807–3827.

[58] X. He and A. Yener. “End-to-End Secure Multi-Hop Communication with Untrusted Relays”. In: *IEEE Transactions on Wireless Communications* 12.1 (Jan. 2013), pp. 1–11.

[59] X. He and A. Yener. “Strong Secrecy and Reliable Byzantine Detection in the Presence of an Untrusted Relay”. In: *IEEE Transactions on Information Theory* 59.1 (2013), pp. 177–192.

[60] X. He and A. Yener. “Providing Secrecy With Structured Codes: Two-User Gaussian Channels”. In: *IEEE Transactions on Information Theory* 60.4 (Apr. 2014), pp. 2121–2138.

[61] J. Huang, A. Mukherjee, and A. L. Swindlehurst. “Secure Communication Via an Untrusted Non-Regenerative Relay in Fading Channels”. In: *IEEE Transactions on Signal Processing* 61.10 (May 2013), pp. 2536–2550.

[62] S. K. Leung-Yan-Cheong and M. E. Hellman. “The Gaussian wire-tap channel”. In: *IEEE Transactions on Information Theory* 24.4 (July 1978), pp. 451–456.

[63] Y. Liang, H. V. Poor, and S. Shamai (Shitz). *Information Theoretic Security*. Vol. 5. Foundations and Trends in Communications and Information Theory 4-5. now publishers, 2009, pp. 355–580.

[64] C. Ling, L. Luzzi, and J.-C. Belfiore. “Lattice codes achieving strong secrecy over the mod-Λ Gaussian Channel”. In: *Proc. of the International Symposium on Information Theory (ISIT)*. 2012, pp. 2306–2310.

[65] C. Ling, L. Luzzi, J.-C. Belfiore, and D. Stehlé. “Semantically Secure Lattice Codes for the Gaussian Wiretap Channel”. In: *IEEE Transactions on Information Theory* 60.10 (Oct. 2014), pp. 6399–6416.

[66] R. Liu and W. Trappe, eds. *Security Wireless Communications at the Physical Layer*. Springer, 2010.

[67] K. Lu, S. Fu, Y. Qian, and T. Zhang. "On the security performance of physical-layer network coding". In: *IEEE International Conference on Communications (ICC)*. June 2009, pp. 1–5.

[68] F. Oggier, P. Solé, and J.-C. Belfiore. "Lattice Codes for the Wiretap Gaussian Channel: Construction and Analysis". In: *IEEE Transactions on Information Theory* 62.10 (Oct. 2016), pp. 5690–5708.

[69] Y. Oohama. "Relay Channels with Confidential Messages". Mar. 2007. URL: `http://arxiv.org/abs/cs/0611125`.

[70] A. J. Pierrot and M. R. Bloch. "Strongly Secure Communications Over the Two-Way Wiretap Channel". In: *IEEE Transactions on Information Forensics and Security* 6.3 (Sept. 2011), pp. 595–605.

[71] H. V. Poor. "Information and Inference in the Wireless Physical Layer". In: *IEEE Transactions on Wireless Communications* 19.1 (Jan. 2012), pp. 40–47.

[72] C. E. Shannon. "Communication Theory Of Secrecy Systems". In: *Bell Systems Technical Journal* 28 (1949), pp. 656–715.

[73] Y.-S. Shiu, S. Y. Chang, H.-C. Wu, S. C.-H. Huang, and H.-H. Chen. "Physical Layer Security in Wireless Networks: A Tutorial". In: *IEEE Transactions on Wireless Communications* 18.2 (Feb. 2011), pp. 66–74.

[74] E. Tekin and A. Yener. "The General Gaussian Multiple-Access and Two-Way Wiretap Channels: Achievable Rates and Cooperative Jamming". In: *IEEE Transactions on Information Theory* 54.6 (June 2008), pp. 2735–2751.

[75] E. Tekin and A. Yener. "Correction to: The General Gaussian Multiple Access and Two-Way Wire-Tap Channels: Achievable Rates and Cooperative Jamming". In: *IEEE Transactions on Information Theory* 56.9 (2010), pp. 4762–4763.

[76] S. Vatedka, N. Kashyap, and A. Thangaraj. "Secure Compute-and-Forward in a Bidirectional Relay". In: *IEEE Transactions on Information Theory* 61.5 (May 2015), pp. 2531–2556.

[77] A. D. Wyner. "The Wiretap Channel". In: *Bell Systems Technical Journal* 54.8 (1975), pp. 1355–1387.

General Publications

[78] J. R. Barry, E. A. Lee, and D. G. Messerschmitt. *Digital Communications.* 3rd ed. Springer, 2004.

[79] C. Buchheim, A. Caprara, and A. Lodi. “An Effective Branch-and-Bound Algorithm for Convex Quadratic Integer Programming”. In: *Integer Programming and Combinatorial Optimization.* Vol. 6080/2010. Lecture Notes in Computer Science. Springer, 2010, pp. 285–298.

[80] R. Mochaourab and E. A. Jorswieck. “Optimal Beamforming in Interference Networks with Perfect Local Channel Information”. In: *IEEE Transactions on Signal Processing* 59.3 (2011), pp. 1128–1141.

[81] T. J. Oechtering, C. Schnurr, I. Bjelakovic, and H. Boche. “Broadcast Capacity Region of Two-Phase Bidirectional Relaying”. In: *IEEE Transactions on Information Theory* 54.1 (2008), pp. 454–458.

[82] E. Telatar. “Capacity of Multi-antenna Gaussian Channels”. In: *European Transactions on Telecommunications* 10.6 (1999), pp. 585–595.

[83] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication.* Cambridge University Press, 2005.

Practical PLNC

[84] S. Agnihotri, S. Jaggi, and M. Chen. “Analog network coding in general SNR regime”. In: *International Symposium on Information Theory (ISIT).* July 2012, pp. 2052–2056.

[85] S. Agnihotri, S. Jaggi, and M. Chen. “Analog network coding in general SNR regime: Performance of a greedy scheme”. In: *International Symposium on Network Coding (NetCod).* June 2012, pp. 137–142.

[86] A. Argyriou and A. Pandharipande. “Cooperative Protocol for Analog Network Coding in Distributed Wireless Networks”. In: *IEEE Transactions on Wireless Communications* 9.10 (Oct. 2010), pp. 3112–3119.

[87] H. Ellinger. “Analyse von Lattices für das Physical Layer Network Coding”. Studienarbeit. Technische Universität Dresden, Fakultät Elektrotechnik und Informationstechnik, Feb. 2014.

[88] Ettus Research. *General Application Notes. Sample rate notes.* 2017. URL: `http://files.ettus.com/manual/page_general.html#general_sampleratenotes` (visited on 01/10/2017).

[89] T. Hynek, D. Halls, and J. Sýkora. "Practical Implementation of Cloud Initialization Procedure for Wireless Physical Layer Network Coding Clouds". In: *20th European Wireless Conference.* May 2014, pp. 1–6.

[90] S. Katti, S. Gollakota, and D. Katabi. "Embracing Wireless Interference: Analog Network Coding". In: *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM).* Oct. 2007, pp. 397–408.

[91] G. Leus and A.-J. van der Veen. "Channel Estimation". In: *Smart Antennas – State of the art.* Ed. by T. Kaiser, A. Bourdoux, H. Boche, J. R. Fonollosa, J. B. Andersen, and W. Utschick. EURASIP Book Series on Signal Processing and Communications. Hindawi Publishing Corporation, 2005. Chap. 15, pp. 293–319.

[92] R. H. Y. Louie, Y. Li, and B. Vucetic. "Practical physical layer network coding for two-way relay channels: Performance analysis and comparison". In: *IEEE Transactions on Wireless Communications* 9.2 (2010), pp. 764–777.

[93] L. Lu, T. Wang, S. C. Liew, and S. Zhang. "Implementation of physical-layer network coding". In: *International Conference on Communications (ICC).* June 2012, pp. 4734–4740.

[94] L. Lu, L. You, Q. Yang, T. Wang, M. Zhang, S. Zhang, and S. C. Liew. "Real-time Implementation of Physical-layer Network Coding". In: *Second Workshop on Software Radio Implementation Forum (SRIF).* 2013, pp. 71–76.

[95] D. Maduike, H. D. Pfister, and A. Sprintson. "Design and implementation of physical-layer network-coding protocols". In: *Forty-Third Asilomar Conference on Signals, Systems and Computers (ASILOMAR).* Nov. 2009, pp. 781–787.

[96] I. Marić, A. Goldsmith, and M. Médard. "Analog Network Coding in the High-SNR Regime". In: *IEEE Wireless Network Coding Conference (WiNC).* June 2010, pp. 1–6.

[97] A. Mejri and G. R.-B. Othman. “Practical Physical Layer Network Coding in Multi-Sources Relay Channels via the Compute-and-Forward”. In: *Wireless Communications and Networking Conference Workshops (WCNCW).* 2013, pp. 166–171.

[98] S. Neumann. “Analyse von Lattices für das Physical Layer Network Coding”. Diploma Thesis. Technische Universität Dresden, Fakultät Elektrotechnik und Informationstechnik, Sept. 2013.

[99] L. Schierling. “Analoge Netzwerkcodierung mit GNU Radio”. Diploma Thesis. Technische Universität Dresden, Fakultät Elektrotechnik und Informationstechnik, Nov. 2015.

[100] The GNU Radio Foundation, Inc. *GNU Radio Website.* 2016. URL: `http://www.gnuradio.org` (visited on 12/13/2016).

[101] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 7.4).* 2016. URL: `http://www.sagemath.org` (visited on 12/13/2016).

[102] H.-M. Wang, X.-G. Xia, and Q. Yin. “A Linear Analog Network Coding for Asynchronous Two-Way Relay Networks”. In: *IEEE Transactions on Wireless Communications* 9.12 (Dec. 2010), pp. 3630–3637.

Publications by the Author

[103] E. A. Jorswieck and J. Richter. “Compute-and-Forward in the Two-Hop Multi-Antenna X-Channel”. In: *5th International Symposium on Communications Control and Signal Processing (ISCCSP).* May 2012, pp. 1–4.

[104] S. Pfennig, E. Franz, J. Richter, C. Scheunert, and E. A. Jorswieck. “Confidential Network Coding: Physical Layer vs. Network Layer”. In: *IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB).* Oct. 2015, pp. 1–5.

[105] J. Richter, E. Franz, S. Engelmann, S. Pfennig, and E. A. Jorswieck. “Physical layer security vs. network layer secrecy: Who wins on the untrusted two-way relay channel?” In: *18th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD).* Sept. 2013, pp. 164–168.

[106] J. Richter, J. Hejtmánek, E. A. Jorswieck, and J. Sýkora. “Non-Cooperative Compute-and-Forward Strategies in Gaussian Multi-Source Multi-Relay Networks”. In: *IEEE 82nd Vehicular Technology Conference (VTC Fall).* Sept. 2015, pp. 1–5.

[107] J. Richter, C. Scheunert, S. Engelmann, and E. A. Jorswieck. “Secrecy in the Two-Way Untrusted Relay Channel with Compute-and-Forward”. In: *International Conference on Communications (ICC).* June 2015, pp. 4357–4362.

[108] J. Richter, C. Scheunert, S. Engelmann, and E. A. Jorswieck. “Weak Secrecy in the Multiway Untrusted Relay Channel with Compute-and-Forward”. In: *IEEE Transactions on Information Forensics and Security* 10.6 (June 2015), pp. 1262–1273.

[109] J. Richter, C. Scheunert, and E. A. Jorswieck. “An efficient branch-and-bound algorithm for compute-and-forward”. In: *23rd IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC).* Second Workshop on Network Coding in Wireless Relay Networks. Sept. 2012, pp. 77–82.

[110] J. Richter, A. Wolf, and E. A. Jorswieck. “Achievable rate regions in multiple-antenna networks with linear network coding”. In: *12th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC).* 2011, pp. 541–545.