COMPUTE-AND-FORWARD RELAY NETWORKS WITH ASYNCHRONOUS, MOBILE, AND DELAY-SENSITIVE USERS

A Dissertation

by

PING-CHUNG WANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Krishna Narayanan
Committee Members,	Joseph Boutros
	Alex Sprintson
	Anxiao Jiang
Head of Department,	Miroslav Begovic

MAY 2018

Major Subject: Electrical Engineering

ABSTRACT

We consider a wireless network consisting of multiple source nodes, a set of relays and a destination node. Suppose the sources transmit their messages simultaneously to the relays and the destination aims to decode all the messages. At the physical layer, a conventional approach would be for the relay to decode the individual message one at a time while treating rest of the messages as interference. Compute-and-forward is a novel strategy which attempts to turn the situation around by treating the interference as a constructive phenomenon. In compute-and-forward, each relay attempts to directly compute a combination of the transmitted messages and then forwards it to the destination. Upon receiving the combinations of messages from the relays, the destination can recover all the messages by solving the received equations. When identical lattice codes are employed at the sources, error correction to integer combination of messages is a viable option by exploiting the algebraic structure of lattice codes. Therefore, compute-and-forward with lattice codes enables the relay to manage interference and perform error correction concurrently. It is shown that compute-and-forward exhibits substantial improvement in the achievable rate compared with other state-of-the-art schemes for medium to high signal-to-noise ratio regime.

Despite several results that show the excellent performance of compute-and-forward, there are still important challenges to overcome before we can utilize compute-and-forward in practice. Some important challenges include the assumptions of "perfect timing synchronization "and "quasi-static fading", since these assumptions rarely hold in realistic wireless channels. So far, there are no conclusive answers to whether compute-and-forward can still provide substantial gains even when these assumptions

are removed. When lattice codewords are misaligned and mixed up, decoding integer combination of messages is not straightforward since the linearity of lattice codes is generally not invariant to time shift. When channel exhibits time selectivity, it brings challenges to compute-and-forward since the linearity of lattice codes does not suit the time varying nature of the channel. Another challenge comes from the emerging technologies for future 5G communication, e.g., autonomous driving and virtual reality, where low-latency communication with high reliability is necessary. In this regard, powerful short channel codes with reasonable encoding/decoding complexity are indispensable. Although there are fruitful results on designing short channel codes for point-to-point communication, studies on short code design specifically for compute-and-forward are rarely found.

The objective of this dissertation is threefold. First, we study compute-and-forward with timing-asynchronous users. Second, we consider the problem of compute-and-forward over block-fading channels. Finally, the problem of compute-and-forward for low-latency communication is studied. Throughout the dissertation, the research methods and proposed remedies will center around the design of lattice codes in order to facilitate the use of compute-and-forward in the presence of these challenges.

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest appreciation to my advisor, Dr. Krishna Narayanan. You are an excellent mentor who not only understand how to guide me to go through the Ph.D. journey but also willing to let me working on problems that I am interested in. Whenever my research work gets stuck, You are able to provide the necessary guidance with your profound insights. Without your advising, I would not become who I am today. Thank you. On the other hand, I would like to thank my committee memebers: Dr. Joseph Boutros, Dr. Alex Sprintson, and Dr. Axiao Jiang. The valuable comments and suggestions from you indeed elevate the quality of my dissertation.

To my former colleague: Dr. Yu-Chih Huang. We had countless discussions on multiple projects. You taught me a lot during my early stage of Ph.D. study. I have no doubt what a talented young researcher you are and I believe your academic career at NTPU will be very bright. I would also like to thank many of my colleagues in TAMU ECE: Avinash Vem, Ying Wang, Nagaraj Janakiraman, Arman Hasanzadeh, Chung-Chi Tsai, Ping-Chun Hsieh, and Meng-Jie Hsiao. All of you participated in a certain period of time during the last six years, from helping me out on issues related to course works to having some great discussions on research works. I can still remember those great moments with you.

Finally, I want to thank my wife, Yu-Fang Cheng, who supports me relentlessly. I couldn't reach my goal without your support. Thank you!

CONTRIBUTORS AND FUNDING SOURCES

This work was supported by a dissertation committee consisting of Dr. Krishna Narayanan (Advisor), Dr. Joseph Boutros, and Dr. Alex Sprintson of the Department of Electrical & Computer Engineering; and Dr. Anxiao Jiang of the Department of Computer Science & Engineering.

Part of the results reported in Chapter 3 was published in an article, P.-C. Wang, Y.-C. Huang, and K. Narayanan, Asynchronous physical layer network coding with quasi-cyclic codes, IEEE J. Sel. Areas Commun., vol. 33, no. 2, pp. 309V322, Feb. 2015. Part of the results reported in Chapter 4 was published in a conference paper, P.-C. Wang, Y.-C. Huang, K. Narayanan, and J. J. Boutros, "Physical layer network-coding over block fading channels with Root-LDA lattice codes," IEEE Intern. Conf. on Comm. (ICC), Kuala Lumpur, May 2016. The works in Chapter 4 are under preparation for journal submission. The results reported in Chapter 5 are under preparation for conference submission.

This doctoral study was supported by the National Science Foundation under Grant CIF 1302616.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

This section serves as a tutorial on major advancements in the compute-and-forward problem in the last decade.

Network coding is an effective approach to improve the throughput of the network by means of allowing intermediate nodes in the network to forward combination of received packets [1]. It has been shown that the upper bound of the single source multicast problem can be achieved by *linear* network coding. A similar situation arises at the physical layer in wireless relay networks. Here multiple source nodes may transmit messages in the same time slot which leads to the reception of a weighted sum of electromagnetic (EM) waves at the relay node. Conventionally, the relay decodes individual message one at a time while treating rest of the messages as interference. In contrast to the conventional approach, it is possible to embrace the interference by letting relay directly decodes combination of messages from the received signal. Hence, the situation has turned around where the interference is viewed as a constructive phenomenon instead of a destructive phenomenon. This novel concept is named as physical-layer network-coding (PLNC) since it exploits the connection between network coding and the weighted sum of EM waves at physical layer [2] [3].

One of the most popular channel models for studying PLNC is the two-way relay network (TWRN). In this model, two source nodes exchange information via an intermediate relay. The message exchange process consists of two phases. In the first phase, two source nodes send their own messages simultaneously to the relay. Upon receiving the superimposed signal, the relay applies certain signal processing techniques to the received signal which transforms it to combination of messages.

In the second phase, the relay then broadcasts the combination of messages to both sources. Since both sources have full knowledge of their own messages, they can recover the intended messages from the broadcasting signal. Note that PLNC can increase the exchange data rate of TWRN up to 100% when compared to the conventional approach in which all the nodes are scheduled to transmit in different time slots.

With the general concept of PLNC in mind, we now present the literature review of various relaying strategies for two-way relay network but not limited to it. In [2] [3], the concept of PLNC appears for the first time. The authors study uncoded TWRN with Gaussian channels. In their seminal work, the relay attempts to decode modulo-sum of the messages, i.e., real sum x_1+x_2 is transformed to modulo sum $x_1 \oplus x_2$ in $\mathbb{F}_2$. The same concept was independently proposed by Popovski *et al.* where they study the multiple-hop relay networks [3]. Decode-and-forward (DF) is another relaying strategy where the relay decodes all the messages from the received signal. Relay then forwards combination of the decoded messages in the second phase. The first phase can be considered as a multiple access problem while relay performs joint decoding, therefore DF is only optimal in asymptotically low signal-to-noise ratio (SNR) regime [4]. In [5] [6], the analog version of PLNC so-called amplify-and-forward (AF) is considered. The strategy is very simple where the relay directly forwards the received signal up to a scaling factor (in order to satisfy power constraint). Other related works on AF can be found in [7] [8]. The complexity to implement AF at a relay is very low since the only signal processing technique performed at the relay is scaling. However, AF has the drawback of noise accumulation, since the noise is also being scaled when the relay scales the received signal. This is a crucial problem especially in the low SNR regime where noise dominates. Denoise-and-forward (DNF) [9] is another relaying strategy proposed by Koike-Akino *et al.* where the relay aims to

remove the noise from received signal by mapping the received signal point to one of the signal points in an optimized constellation. The constellation is optimized in a way such that its minimum distance is maximized. DNF has the potential to achieve higher rate than DF and AF; However, the constellation and corresponding mapping rule (for broadcasting) will be changed according to instantaneous channel realization, therefore the memory storage requirement for relay to store all possible constellations and mapping rules are prohibitively high. This may not be favorable to practical communication systems which usually have limited memory storage. We have seen some of the advantages of employing DF, AF and DNF at the relay, but clearly, these relaying strategies have fundamental drawbacks as well. One of the problems for these relaying strategies is that the capability of error correction is not fully exploited. For DF, joint decoding is not optimal except in asymptotic low SNR regime. For AF and DNF, coded transmissions are not considered. On the other hand, the approach of computing modulo-sum directly from the received signal at the relay is a perfect match for error correction. In what follows, we will review the research works on this topic. From now on, when we say PLNC, we specifically imply the approach of computing modulo-sum at the relay.

In [2] [3], channel coding is not considered at the relay, thus erroneous message can only be corrected at final destination. This property is obviously undesirable to most of the communication systems. In [10], Narayanan *et al.* study the problem of joint design of PLNC and error correction at the relay for the first time. They propose a novel concept which employs identical nested lattice codes to all source nodes. By exploiting the algebraic structure of the lattice codes, one can perform error correction to the combination of messages at the relay. It is shown that the achievable computation rate is $\log(\frac{1}{2}+\text{snr})$ in two-way relay Gaussian channel, which is only 1/2 bit away from the upper bound. The result implies that the lattice coded

scheme is nearly optimal in the high SNR regime. In the sequel, Nam *et al.* study the capacity region of TWRN in the unequal channel gain scenario [11]. Later, the lattice coded scheme is generalized to wireless relay network with multiple source and relay nodes over quasi-static fading channels [12]. For the first time, the authors name PLNC with lattice codes as compute-and-forward problem. The concept of CF is quantizing the channel gains to integers such that the weighted-sum of signals can be decoded to an integer combination of messages at relay. After the pioneer works, CF has been studied by numerous researchers. Feng *et al.* study the general construction of algebraic codes for CF in the non-asymptotic regime [13]. Subsequently, several groups work on practical code constructions for CF [14] [15] [16] [17]. Multistage decoding for CF is proposed in [14] [18]. The former shows that lattice codes constructed from product construction can achieve the same computation rate as reported in [12] under multi-stage decoding. The latter derives universally achievable rate for the multilevel decoding scheme and provides the numerical results for the schemes that practical modulations are employed. CF over Eisenstein integer is considered by [19] [20]. The reason for exploiting alternative ring of integer is to have better quantization of the channel coefficients which leads to the improvement in outage and error-correcting performance.

Despite the exciting results on the achievable computation rate of CF, the skepticism on whether CF can operate reliably at the promised rate in practical wireless communication systems has never ceased. The main reason is that CF heavily relies on two ideal assumptions, otherwise the nice algebraic structure of lattice codes may be ruined and may no longer be easily exploited. These two assumptions are the "perfect timing synchronization"assumption and the "quasi-static channel"assumption. The former assumes that signals coming from all source nodes arrive at relay simultaneously; without which it will cause misalignment of lattice codewords and hence

the linearity is ruined. The latter assumes that channel coefficients are fixed within one codeword duration. Again, without this assumption, the linearity is no longer preserved and compute-and-forward is undermined. In many practical wireless communication systems, these assumptions are often too good to be true and solutions are called for in order to again make use of the structural gains offered by the channel.

[**Chapter 3: Timing-asynchronous users**] Recently, many attempts have been made on solving the problem of CF with timing-asynchronous source nodes [21] [22] [23] [24] [25]. Here we assume that receivers have perfect knowledge of the timing delays of all source nodes. In [21], the authors investigate asynchronous CF when delays are multiples of the symbol duration. In [22], the scenario where the delay can be any real value within one symbol duration is studied. Later, Wu *et al.* propose a novel scheme which can deal with any real-valued delay with the assistance of cyclic LDPC codes. The problem for this scheme is that codes in the family of cyclic LDPC codes are very limited, and their decoding performance under message-passing decoding is inferior to other well-designed LDPC codes. We notice that this timing asynchronism problem is either solved partially or no optimality is guaranteed. In Section 3, we will propose a new family of lattice codes constructed from quasi-cyclic (QC) LDPC codes which can deal with the challenges arising from timing asynchronism.

[**Chapter 4: Block-fading channels**] Compute-and-forward over channels that are not quasi static is far less discussed than the synchronization issue. In [26], Bakoury *et al.* investigate the impact of block fading for integer-forcing receiver, an application of CF to the multiple-input multiple output (MIMO) channel. Achievable rates based on CF with nested lattice codes of [27] and two decoders specifically designed for this scenario are derived. However, no optimality is shown. Moreover, from what we have learned in wireless communication [28], in the finite-length regime,

diversity is an important metric when evaluating the performance of a communication system in block-fading channel. The diversity determines the decreasing rate of the probability of block error as SNR increases. therefore we focus on designing finite-length lattice codes that can exploit full diversity offered by the channel while computing functions in Section 4.

[**Chapter 5: Low-latency communications**] Most of the preliminary works in compute-and-forward focus on the regime of long block length. Recently, with the advent of emerging technology in 5G, e.g., autonomous driving and virtual reality, low-latency communication with high reliability is necessary for these applications. Hence the design of powerful channel codes in short block length regime (block length $\leq$ 256) has drawn great attention. In the point-to-point channel model, the combination of polar code [29] and cyclic redundant check (CRC) [30] [31] [32] under list decoding can approach the finite length bound [33] in terms of codeword error rate. Meanwhile, research attempts on other codes are also visible, such as LDPC codes [34], turbo codes [35], and BCH codes [36], etc. Also, the performance of maximum likelihood decoding of various short codes is studied in [36] [37]. A survey on short channel codes can be found in [38]. In the context of multi-terminal communication, the study on the random error exponent of function computation in CF is recently presented by [39]. To the best of the author's knowledge, no other study is found. Thereby, we are interested in constructing powerful short lattice codes and investigating the decoding performance of function computations at relays using such codes.

1.1 Organization

The rest of the dissertation is organized as follows. In Section 2, The basics of LDA lattice codes are introduced and the problem of compute-and-forward is

formulated. Chapter 3 focuses on asynchronous compute-and-forward. Chapter 4 studies compute-and-forward when block-fading channel is considered. In the last Chapter, we study the performance of short lattice codes in the compute-and-forward paradigm.

1.2 Notations

Throughout the dissertation, $\mathbb{R}$, $\mathbb{C}$, and $\mathbb{Z}$ represent the set of real numbers, complex numbers, and integers, respectively. $\mathbb{F}_p$ denotes finite field of order p. Vectors and matrices are written in boldface lowercase and boldface uppercase, for example $\mathbf{x}$ and $\mathbf{X}$, respectively. For a vector $\mathbf{x}$, we use $\mathbf{x}^{(t)}$ to denote the right circularly shifted version of $\mathbf{x}$ by t. e.g., for $\mathbf{x} = [1, 2, 3, 4]$, $\mathbf{x}^{(1)} = [4, 1, 2, 3]$. Moreover, $\oplus$ and $\odot$ are addition and multiplication, respectively, over a finite field whose size is understood from the context.

2. BACKGROUND

The purpose of this section is twofold. In the first part, we introduce the basics of lattice codes with emphasis on LDA lattice codes. Second, we will formulate the problem of compute-and-forward rigorously such that the subsequent sections can be discussed based on the general framework.

2.1 LDA lattice codes

An LDA lattice code is a subset (finite points) of an LDA lattice, where an LDA lattice is an N-dimensional integer lattice built from a non-binary LDPC code via construction A [40]. It has been shown that there exists a Polytrev-capacity-achieving family of LDA lattices for the point-to-point AWGN channel [41]. Experimental results also show that the error performance of LDA lattices under message-passing decoding is close to capacity [40]. Based on these facts, and together with the goodness of lattice codes for CF problem, we are interested in constructing various types of LDA lattice codes for CF problems in three scenarios described in Chapter 1. We now introduce the construction of LDA lattices.

Definition 1. *LDA lattice*

Let $\mathcal{C}$ be an N dimensional non-binary (NB) LDPC code from some finite field $\mathbb{F}_p$, also let $\mathcal{M} : \mathbb{F}_p \to \mathbb{R}$ be the natural mapping given by

$$\mathcal{M}(u) \triangleq \begin{cases} u, & 0 \leq u \leq \frac{p-1}{2}, \\ u-p, & \frac{p-1}{2} < u < p, \end{cases} \tag{2.1}$$

when $p \geq 3$, and $\mathcal{M}(u) \triangleq u - \frac{1}{2}$ for $p = 2$ (i.e., BPSK). An LDA lattice is obtained by tiling $\mathcal{M}(\mathcal{C})$ to the entire $\mathbb{R}^N$ as

$$\Lambda \triangleq \mathcal{M}(\mathcal{C}) + p\mathbb{Z}^N, \tag{2.2}$$

where the sum in the equation is the Minkowski sum.

So far we have discussed the construction of the LDA lattices, yet a lattice consists of infinite points in $\mathbb{R}^N$, and the power at transmitters are limited in practical communication systems. One popular approach of using lattices for power-constrained problems is to carve a finite subset from the lattice via shaping techniques. Conventionally, spherical shaping is adopted where the codebook is the intersection of a sphere and the lattice [42]. Spherical shaping is power-efficient and leads to codebooks mimicking the capacity-achieving input distribution for the AWGN channel. However, the nice correspondence between lattice points and underlying linear codewords may no longer be preserved. Another popular approach for shaping is to use nested lattice shaping [27]. Let (Λ_f, Λ_c) be a pair of nested lattices such that $\Lambda_c \subseteq \Lambda_f$, where Λ_f is the fine lattice and Λ_c is the coarse lattice. The lattice code obtained by nested lattice shaping, henceforth referred to as nested lattice code, is $\mathcal{C} = \Lambda_f \cup \mathcal{V}(\Lambda_c)$ where $\mathcal{V}(\Lambda_c)$ denotes the fundamental Voronoi region of coarse lattice Λ_c. Under some goodness conditions, the resulting nested lattice code can achieve the capacity of AWGN channel with lattice decoding. Since nested lattice codes preserve most of the structures which facilitate function computation, they are adopted in most of the CF literature. Therefore in this work, we employ nested lattice shaping, by choosing Λ_c such that $p\mathbb{Z}^N \subseteq \Lambda_c$. It is worth emphasizing that this choice includes hypercubic shaping $\Lambda_c = p\mathbb{Z}^N$ which is simple to implement.

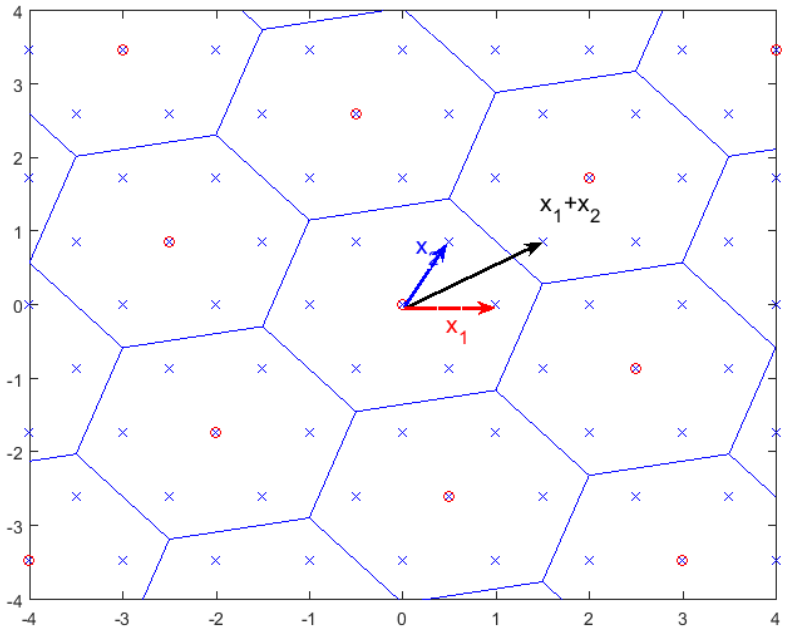


Figure 2.1: The sum of two lattice points is another lattice point

2.2 Compute-and-forward

In this section, we introduce the compute-and-forward relay network introduced by Nazer and Gastpar [12]. The compute-and-forward protocol suggested in [12] implements an *identical* nested lattice code [27] at each user and directly decodes the received signal to a modulo version of linear combination of the codewords with integer coefficients at the relay. This scheme is shown to provide a substantially higher computation rate in the medium signal-to-noise ratio (SNR) regime than existing schemes. For practical purposes, in [20], this nested lattice code is replaced by linear code over $\mathbb{F}_p$ together with a signal mapping possessing the property described earlier in Eq. (5.15), in order to exploit the structural gain. The relay then decodes the received signal to a linear combination of the codewords over $\mathbb{F}_p$. Indeed, the lattice code is carved out from construction A lattice with hyper-cubic shaping. To have

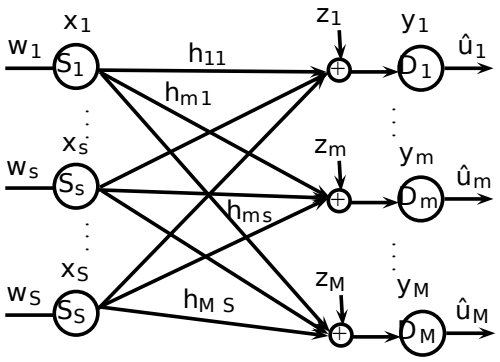


Figure 2.2: The compute-and-forward relay network.

better understanding of CF, we can take a look at an example with 2-dimensional lattice code shown in Fig. 2.1. The lattice codewords in the lattice codebook are the blue points in the figure. Due to the closure property of the lattice, the sum of two lattice codewords x_1 and x_2 is still a lattice codeword. Therefore error correction can be employed when computing the sum of two codewords. Indeed, any integer combination of the lattice codewords is a valid codeword. We will utilize this property to develop the subsequent compute-and-forward framework.

We now introduce the general system model of a compute-and-forward relay network. As shown in Fig. 2.2, in a compute-and-forward relay network, there are total S source nodes, denoted as $S_s, s = 1 \dots S$, and $M \geq S$ relay nodes, denoted as $D_m, m = 1 \dots M$. Each source node $s \in \{1, \dots, S\}$ encodes its message $\mathbf{w}_s \in \mathbb{F}_p^K$ to a codeword $\mathbf{c}_s \in \mathbb{F}_p^N$. This codeword is then modulated to the transmitted signal $\mathbf{x}_s \in \mathcal{A}^N$ via a mapping $\mathcal{M}$, the mapping is given by,

$$\mathcal{M}(u) \triangleq \begin{cases} u, & 0 \leq u \leq \frac{p-1}{2}, \\ u - p, & \frac{p-1}{2} < u < p, \end{cases} \tag{2.3}$$

for $p \geq 3$, and $\mathcal{M}(u) \triangleq u - \frac{1}{2}$ for $p = 2$ (i.e., BPSK)*. This mapping from $\mathbb{F}_p$ to $\mathbb{Z}/p\mathbb{Z}$ which is known to be a ring isomorphism, i.e., there exists one-to-one mapping between the operations in $\mathbb{F}_p$ and $\mathbb{Z}/p\mathbb{Z}$. In fact this is an important property for compute-and-forward (interesting readers can refer to [12] for details). Note that the above operation is precisely the standard procedure for constructing a lattice from linear codes via Construction A [43] [44]. The codeword is subject to a power constraint given by

$$\frac{1}{N}\|\mathbf{x}_s\|^2 = \frac{1}{N}\sum_{n=1}^{N}|x_s[n]|^2 \leq P. \tag{2.4}$$

At the m-th relay, the n-th received symbol is given by

$$y_m[n] = \sum_{s=1}^{S} h_{ms}x_s[n] + z_m[n], \quad n = 1, \ldots, N, \tag{2.5}$$

where $h_{ms} \in \mathbb{R}$ (or $\mathbb{C}$ depending on whether the signal constellation is real or complex) is the channel coefficient between the source node S_s and relay D_m, and $z_m[n] \sim \mathcal{CN}(0, 1)$. The relay node m is only interested in computing and forwarding a function of the messages, i.e., the relay node first chooses $\{a_{m1}, \ldots, a_{mS}\}(a_{ms} \in \mathbb{Z})$ such that $\sum_{s=1}^{S} a_{ms}\mathbf{x}_s$ can approximate $\sum_{s=1}^{S} h_{ms}\mathbf{x}_s$ carefully, then relay obtains $\{b_{m1}, \ldots, b_{mS}\}$ ($b_{ms} = a_{ms} \mod p$) and computes $\mathbf{u}_m = \oplus_{s=1}^{L} b_{ms}\mathbf{w}_s$. The optimal $\{a_{ms}\}$ can be determined in terms of computation rate, which is given by

$$\mathcal{R}(\mathbf{h}_m, \mathbf{a}_m) = \frac{1}{2}\log^+\left(\left(\|\mathbf{a}_m\|^2 - \frac{P|\mathbf{h}_m^H\mathbf{a}_m|^2}{1 + P\|\mathbf{h}_m\|^2}\right)^{-1}\right), \tag{2.6}$$

where $\mathbf{a}_m = [a_{m1}, \ldots, a_{mS}]^t$, and $\mathbf{h}_m = [h_{m1}, \ldots, h_{mS}]^t$. Note that the rate is achievable per real dimension. The computed function together with $\{b_{m1}, \ldots, b_{mS}\}$

*Note that the construction of lattice code we considered here indicates that the modulation (mapping) and coding can be considered separately

are then forwarded to a central destination which desires all the messages. After the central destination collect functions and coefficients $\{b_{ms}\}$ from all the relays, it is clear that as long as the coefficients $\{b_{ms}\}$ form a full-rank matrix in $\mathbb{F}_p$, the central destination would be able to invert the matrix and obtain all the messages.

3. COMPUTE-AND-FORWARD WITH TIMING ASYNCHRONOUS USERS*

In this section*, we study the asynchronous version of compute-and-forward relay network introduced by Nazer and Gastpar [12]. First, we will formulate the problem by describing the system model of asynchronous compute-and-forward relay network. In the second part, we will introduce the proposed lattice code for this problem, then the main results for asynchronous CF are presented. Finally, possible future directions are pointed out.

3.1 Introduction

Physical-layer network coding (or compute-and-forward) [12] [2] [45] has been shown to be a way to effectively harness interference in wireless networks and to provide significantly higher throughput than conventional strategies for many wireless networking problems. However, most of the results in the literature consider the case when the time delays from the multiple transmitters to a receiver are all identical (we refer to this as the synchronous case). One of the important open problems in this area is to determine whether the information rates achieved with compute-and-forward in the synchronous case can be obtained when the time delays from the multiple transmitters are different also (we refer to this as the asynchronous case). So far, this question has not been conclusively answered and our understanding of asynchronous physical-layer network coding is not as thorough as that of synchronous one. Recently, there have been some efforts in the literature trying to address such problems for some specific models such as inter symbol interference (ISI) [46] and asynchronous physical-layer network coding (and compute-and-forward as well) [47]

*Reprinted with permission from P. C. Wang, Y. C. Huang and K. R. Narayanan, "Asynchronous Physical-Layer Network Coding With Quasi-Cyclic Codes," in IEEE Journal on Selected Areas in Communications, vol. 33, no. 2, pp. 309-322, Feb. 2015.

[22] [23] [24]. In both cases, cyclic codes have been suggested for combating the time delays [46] [23].

While cyclic codes are quite useful for these problems, there has been no proof that rates achievable in the synchronous case are achievable in the asynchronous case also. One important reason for this is the fact that there is no proof showing the existence of ensembles of cyclic codes that can achieve capacity[†].

In this section, we show that there is no fundamental loss in the achievable information rates due to asynchronism, when the time delays introduced by the channel are integer multiples of a symbol duration. Interestingly, we also show that in some scenarios, time delays introduced by the channel can be exploited to achieve higher information rates than those achievable in the synchronous case. These results are based on two insights and novel ideas proposed in this section. The first is the insight that cyclic codes are *not necessary* to deal with asynchronism and that quasi-cyclic (QC) codes suffice. The second main idea is to use an interleve/deinterleave transform (IDT) which equips QC codes with the capability to combat time delays. With a slight rate reduction, this transform will convert any linear shift of an integer multiple of the symbol duration introduced by the channel into a circular shift of another integer value depending on the parameter we choose. Therefore, one can then utilize the IDT for *designing* the equivalent time delays seen at the transform output and implement a QC code accordingly. We then show the existence of an ensemble of QC codes that can achieve capacity for channels whose capacity can be achieved by linear codes and leverage this result to prove the aforementioned

[†]Recently, we notice that it has been shown that there exists ensemble of cyclic codes which can achieve the capacity of an erasure channel under maximum likelihood decoding [48]. However, the complexity of maximum likelihood decoder is extremely high, which cannot be realized in practical communication systems. Here, we are interested in designing a coding scheme which not only can achieve the desired information rate in theory but also exhibit excellent performance when efficient decoding algorithm is employed.

information-theoretic result for asynchronous physical-layer network coding.

To give concrete examples, we implement the proposed IDT together with QC codes for two instances of asynchronous physical-layer network coding, namely integer-forcing equalization for ISI channels and asynchronous compute-and-forward. For the integer-forcing equalization proposed in [46], we show that our IDT-QC codes achieve the upper bound on information rates presented in [46]. For asynchronous compute-and-forward, when the delays are integer multiples of the symbol duration, we first show that the rates achievable in the synchronous case can also be achieved in the asynchronous case. In addition to this, we also show that the proposed IDT-QC codes are capable of exploiting another dimension, namely the delay dimension which leads to rates exceeding those achieved in synchronous compute-and-forward [12]. Finally, we consider the case of non-integer valued delays and when rectangular pulses are used, we show that the proposed schemes achieves higher rates than the scheme in [24]. It is worth noting that the proposed IDT-QC codes are not limited to these two specific examples and can potentially be implemented for many networks with delays which cannot be easily compensated.

In addition to being of theoretical importance, the use of quasi-cyclic (QC) codes is of substantial practical importance as well. QC codes, QC low-density parity check (LDPC) codes in particular, are quite popular in modern coding theory due to their following desirable properties. They can be encoded using linear feedback shift registers [49] and a message passing decoder can be implemented efficiently in hardware in a partially parallel architecture [50]. Further, the QC property makes it efficient to route wires when implementing the message passing decoder [51]. Moreover, the family of QC codes is much larger than and subsumes as a special case of the family of cyclic codes. Due to these properties, QC LDPC codes have been included in many real world applications such as IEEE 802.11n [52], IEEE 802.16e [53], DVB-S2

[54], etc. In this section, we show that in addition to these desirable properties, when used with the IDT transform, QC codes can be a perfect candidate for combating time delays.

3.1.1 Organization

The section is organized as follows. In Section 3.2, we provide definitions of cyclic codes and QC codes and also review a well-known construction of QC codes based on protographs. We also review the modulation scheme typically used in the compute-and-forward literature. The modulation scheme has been shown to preserve the structures induced by the channel and hence are crucial for compute-and-forward. This review is of practical importance as our proposed scheme heavily relies on QC codes and the modulation scheme and the family of QC LDPC codes is one of the most popular classes of QC codes in practice. In Section 3.3, we elucidate the proposed IDT-QC codes and show some properties of the proposed codes which include the capacity-achieving property. Section 3.4 and Section 3.5 provide two interesting applications of the proposed IDT-QC codes, namely point-to-point communication over ISI channels and asynchronous compute-and-forward. In Section 3.6, we introduce a new joint detection and decoding scheme for our proposed IDT-QC codes. This section lifts the information-theoretic framework proposed in Section 3.3-3.5 towards practical implementation by explicitly introducing a practically implementable decoding scheme. Finally, Section 3.7 concludes the section.

3.2 Preliminaries

We first give definitions of cyclic codes and QC codes and then discuss a well-known construction of QC LDPC codes.

Definition 2. *Cyclic codes*

A linear code $\mathcal{C}$ is a cyclic code of length N if any circular shift of a codeword is a

codeword in $\mathcal{C}$, i.e., for every $\mathbf{c} \in \mathcal{C}$, $\mathbf{c}^{(i)} \in \mathcal{C}$, for all $i = 0, \ldots, N-1$.

Definition 3. *Quasi-cyclic codes - Representation I*

A linear code $\mathcal{C}$ is a QC code with shifting constraint b if any circular shift of a codeword by a multiple of b is a codeword in $\mathcal{C}$, i.e., for every $\mathbf{c} \in \mathcal{C}$, $\mathbf{c}^{(bi)} \in \mathcal{C}$, for all $i = 0, \ldots, \lfloor \frac{N}{b} \rfloor - 1$.

Definition 4. *Quasi-cyclic codes - Representation II*

A linear code $\mathcal{C}$ is a QC code with shifting constraint b if every codeword $\mathbf{c} \in \mathcal{C}$ consists of b sub-blocks and for each codeword, circularly shifting every sub-block by the same amount results in a codeword.

Note that the above two representations of QC codes are equivalent and such codes are referred to as b-QC codes. One can be converted to the other via an interleaver. Throughout the dissertation, unless mentioned otherwise, the first representation of QC codes is adopted (Definition 3). On the other hand, many constructions in the literature (e.g. [55]) adopt the second representation.

LDPC codes have been very popular in modern coding theory and in practice due to its ability to achieve near-capacity performance with low decoding complexity and outstanding performance in the finite-length regime. The family of QC LDPC codes is a special class of LDPC codes possessing the QC property that efficient encoding and decoding algorithms exist. In what follows, we briefly review the construction of QC LDPC codes. Most of the works in the literature consider using the protograph-based construction of [56] to generate QC LDPC codes, see for example [55] and the reference therein.

To construct a protograph of a length N b-QC LDPC code, one begins with a $c \times b$ base matrix and then replaces each entry in the base matrix by an $L \times L$ matrix where $L \triangleq N/b$. The replacement follows the rule that if the entry is 1, it is replaced

by a random $L \times L$ permutation matrix while if the entry is 0, it is replaced by an all-zero matrix. This would result in a $cL \times N$ LDPC matrix and can be used to generate an LDPC code. Now, if we further restrict those permutation matrices to be *circulant* matrices, then the output would be the parity-check matrix for a b-QC LDPC code. Unlike standard linear codes, the generator matrix of every QC code cannot be written in a systematic form without violating the QC property. However, many existing QC LDPC ensembles including the ones used for simulation and for the proofs satisfy those constraints.

In order to use the above QC codes for transmission, we modulate the codeword symbols onto elements in a constellation $\mathcal{A}$ to form the transmitted signals. Throughout this section, we consider using QC codes over a prime field $\mathbb{F}_p$ and restrict the constellation $\mathcal{A}$ to be pulse amplitude modulation (PAM) with p elements, i.e., $\mathcal{A} = \{-\frac{p-1}{2}, \ldots, 0, \ldots, \frac{p-1}{2}\}$. The mapping $\mathcal{M} : \mathbb{F}_p \to \mathcal{A}$ is given by

$$\mathcal{M}(u) \triangleq \begin{cases} u, & 0 \leq u \leq \frac{p-1}{2}, \\ u - p, & \frac{p-1}{2} < u < p, \end{cases} \tag{3.1}$$

for $p \geq 3$, and $\mathcal{M}(u) \triangleq u - \frac{1}{2}$ for $p = 2$ (i.e., BPSK). This mapping has the important property that $\mathcal{M}(u \oplus v) = \mathcal{M}(u) + \mathcal{M}(v) \mod p$ and $\mathcal{M}(u \otimes v) = \mathcal{M}(u) \cdot \mathcal{M}(v) \mod p$ for $p \geq 3$ and $\mathcal{M}(u \oplus v) + \frac{1}{2} = \mathcal{M}(u) + \frac{1}{2} + \mathcal{M}(v) + \frac{1}{2} \mod 2$ and $\mathcal{M}(u \otimes v) + \frac{1}{2} = (\mathcal{M}(u) + \frac{1}{2}) \cdot (\mathcal{M}(v) + \frac{1}{2}) \mod 2$ for $p = 2$. Note that the above operation is precisely the standard procedure for constructing a lattice from linear codes via Construction A [44]. In fact, one can easily show that the above construction results in lattices having QC property, i.e., QC lattices. Moreover, from a result by Forney *et al.* [57], we know that applying a capacity-achieving linear code to Construction A would result in a sphere-bound-achieving (or Poltyrev good) lattice. Therefore, existing

good QC codes such as AR4JA codes [55] can be adopted to generate good QC lattice codes via Construction A.

3.3 Proposed interleave/ieinterleave transformed quasi-cyclic code

Even though our ultimate application is in networks, in this section, we start with the point-to-point communication to facilitate the illustration of the proposed IDT transform. Consider point-to-point communication with additive white Gaussian noise (AWGN) and with time delays τ that is upper bounded by the maximal possible delay D_{max}. We assume that the transmitter only has access to D_{max} but the receiver knows both τ and D_{max}. For the point-to-point case, one can easily achieve the capacity by using a capacity-achieving code since τ is known and can be easily compensated. However, in a network where there are multiple source nodes, the signals may arrive at different time and are all mixed together so that this simple approach may no longer work. In order to obtain insight into this problem, we begin with the point-to-point case. Motivated by this issue, we propose a general framework called IDT which utilizes a pair of interleaver/deinterleaver to transform the received signal into the desired form. When combined with QC codes, the IDT allows us to combat time delays that may be introduced in many practical scenarios such as asynchronous channels and/or ISI channels. We nickname this combination as interleave/deinterleave transformed quasi-cyclic (IDT-QC) codes.

3.3.1 System model: point-to-point communication

Consider a point-to-point communication with AWGN and delay $\tau \in \{0, \ldots, D_{max}\}$. The transmitter wishes to send a message $\mathbf{w} \in \mathbb{F}_p^K$ to the receiver. It first feeds the message into an encoder $\mathcal{E}^N : \mathbb{F}_p^K \to \mathbb{F}_p^N$ to form the codeword $\mathbf{c} = \mathcal{E}^N(\mathbf{w}) \in \mathbb{F}_p^N$. The transmitter adopts the modulation scheme $\mathcal{M} : \mathbb{F}_p^N \to \mathcal{A}^N$ to form the transmitted signal $\mathbf{x} = \mathcal{M}(\mathbf{c}) \in \mathcal{A}^N$ where $\mathcal{A}$ is the signal constellation. The transmitted signal

$\mathbf{x}$ is subject to an input power constraint P.

$$\frac{1}{N}\|\mathbf{x}\|^2 = \frac{1}{N}\sum_{n=1}^{N}|x[n]|^2 \leq P. \tag{3.2}$$

The received signal is then given by

$$y[n] = x[n-\tau] + z[n], \quad n \in \{1, 2, \ldots, N+\tau\}, \tag{3.3}$$

where $\tau \in \{0, \ldots, D_{max}\}$ represents an integer delay and $z[n] \sim \mathcal{N}(0,1)$. Upon receiving $\mathbf{y}$, the receiver then forms an estimate of the message $\hat{\mathbf{w}} \in \mathbb{F}_p^K$ via $\mathcal{G}^N : \mathbb{R}^N \to \mathbb{F}_p^K$. Throughout the section, we assume that τ is unknown to the transmitter and is known to the receiver and D_{max} is known to both ends. For the ISI channel, this assumption implies that although the transmitter may not know how many taps we would have, it knows the maximal delay spread D_{max}. For asynchronous communication, this assumption models the scenario where there is only a very loose synchronization mechanism that would control the time delays to some degree D_{max}.

In what follows, we give the definition of codes, achievable rates, and capacity.

Definition 5. *Codes*

An (N, K) code consists of a pair of encoding/decoding functions $(\mathcal{E}^N, \mathcal{G}^N)$ described above and an error probability given by

$$P_e^{(N)} \triangleq \mathbb{P}\left(\hat{\mathbf{w}} \neq \mathbf{w}\right). \tag{3.4}$$

Definition 6. *Achievable rate and capacity*

For a given set of parameters P and D_{max}, a rate $R(P, D_{max})$ is achievable if for

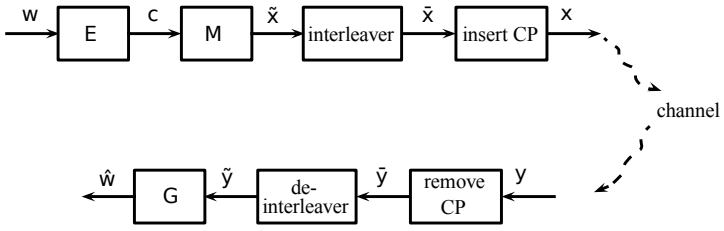


Figure 3.1: Block diagram of the proposed IDT.

any $\varepsilon > 0$ *there is an* (N, K) *code over* $\mathbb{F}_p$ *such that*

$$K \geq NR(P, D_{max})/\log(p) \text{ and } P_e^{(N)} \leq \varepsilon. \tag{3.5}$$

The capacity is defined as the supremum of all achievable rates given by

$$C(P, D_{max}) \triangleq \sup R(P, D_{max}). \tag{3.6}$$

3.3.2 IDT-QC codes

Let $\mathcal{C}$ be a (N', K) b-QC linear/lattice code with the design rate $R_d = r_d \cdot \log(p)$ where $r_d \triangleq K/N'$ and $r_d b \in \mathbb{Z}$. Also, we enforce the generator matrix of this code to be systematic. In the proposed IDT-QC codes shown in Fig. 3.1, the transmitter maps the message to a codeword $\mathbf{c} \in \mathcal{C}$ via the encoder $\mathcal{E}$. This codeword is modulated by $\mathcal{M}$ to form the signal $\tilde{\mathbf{x}}$. The signal is then fed into a $(b, N'/b)$ write column-wise transmit row-wise interleaver [58] to get a interleaved signal $\bar{\mathbf{x}}$ where the input-output relationship is given by

$$\bar{x}[n] = \tilde{x}\left[1 + (\lfloor n/L \rfloor) + b \cdot (n \mod L - 1)\right], \tag{3.7}$$

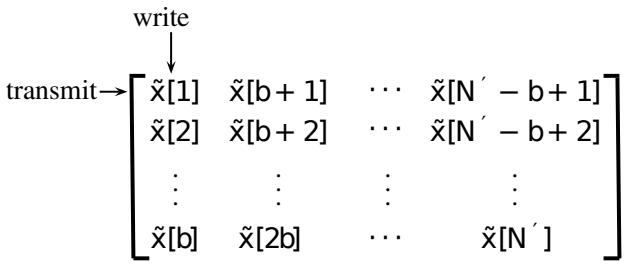


Figure 3.2: The write column-wise transmit row-wise interleaver.

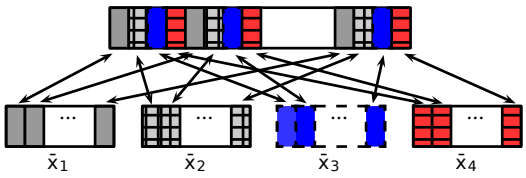


Figure 3.3: An example of the IDT-QC codes with $b = 4$.

where $L \triangleq N'/b$ is always an integer provided by the QC constraint. An illustration of interleaving can be found in Fig. 3.2 and one example with $b = 4$ is given in Fig. 3.3.

Note that one can write the interleaved codeword as the collection of b sub-blocks as

$$\bar{\mathbf{x}} = [\bar{\mathbf{x}}[1]\bar{\mathbf{x}}[2]\dots\bar{\mathbf{x}}[b]], \tag{3.8}$$

where each sub-block $\bar{\mathbf{x}}[s]$ for $s \in \{1, \dots, b\}$, is of length L. For each of the first $r_d b$ sub-blocks, we freeze the D_{max} last positions to be zero. This is possible since the encoder is systematic and the first $r_d b$ blocks correspond to the message part. We then insert a cyclic prefix (CP) of length D_{max} for each of the last $(1-r_d)b$ sub-blocks

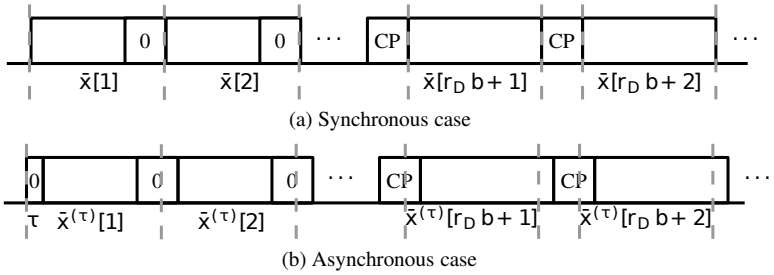


Figure 3.4: (a) Overall signal structure. (b) Asynchronous case.

by appending the last D_{max} symbols to the front. The overall transmitted signal is given by

$$\mathbf{x} = [\mathbf{x}[1], \mathbf{x}[2], \ldots, \mathbf{x}[b]], \tag{3.9}$$

where for $s \in \{1, \ldots, r_d b\}$, $\mathbf{x}[s] = \bar{\mathbf{x}}[s]$ whose last D_{max} symbols are 0, and for $s \in \{r_d b + 1, \ldots, r_d b\}$

$$\mathbf{x}[s] \triangleq [\underbrace{\bar{x}^{L-D_{max}+1}[s], \ldots, \bar{x}^{L}[s]}_{\text{CP with length } D_{max}}, \underbrace{\bar{x}^{1}[s], \ldots, \bar{x}^{L}[s]}_{=\bar{\mathbf{x}}[s]}]. \tag{3.10}$$

The total length of this signal is $N = N' + (1 - r_d) b D_{max}$. An illustration of the overall signal structure is given in Fig. 3.4.(a). In fact, for the purpose of IDT transform, one does not have to distinguish the parts using CPs and freezing symbols; it suffices to append CPs for all the sub-blocks. The reason that we choose to freeze symbols instead of inserting CPs for the first $r_d b$ sub-blocks will become apparent in Sections 3.4 and 3.5. At the receiver end, since the receiver knows the time delays τ and $\tau \leq D_{max}$, it first discards the CP for each sub-block to form $\bar{\mathbf{y}}$. As shown in Fig. 3.5, this signal $\bar{\mathbf{y}}$ is then fed to a $(b, N'/b)$ read row-wise output column-wise

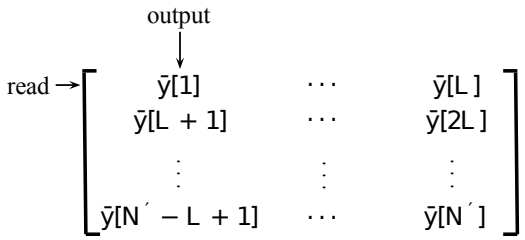


Figure 3.5: The read row-wise output column-wise deinterleaver.

deinterleaver to get output $\tilde{\mathbf{y}}$ where the input-output relationship is given by

$$\tilde{y}[n] = \bar{y}\left[1 + (\lfloor n/b \rfloor) + L \cdot (n \mod b - 1)\right], \tag{3.11}$$

which is then fed into the decoder of the QC code to form an estimate of the message. The actual rate of this IDT-QC code is given by

$$\begin{aligned} R_a &= \frac{K - r_d b D_{max}}{N' + (1 - r_d) b D_{max}} \log(p) \\ &= \left(1 - \frac{(2 - r_d) b D_{max}}{N' + (1 - r_d) b D_{max}}\right) R_d \end{aligned} \tag{3.12}$$

which tends to R_d asymptotically but introduces a rate loss for any finite N'.

In the following, we provide some properties of the IDT-QC codes.

Lemma 1. *If the receiver opts not to compensate the delay τ, i.e., the receiver observes a noisy version of the transmitted signal delayed by τ, $[\underbrace{0, \ldots, 0}_{\tau}, \mathbf{x}[1], \mathbf{x}[2], \ldots, \mathbf{x}[b]]$, then the proposed IDT transforms the received signal into a noisy version of the codeword circularly shifted by $b \cdot \tau$. Moreover, if a b-QC code is employed in conjunction with IDT, one can directly decode $\mathbf{c}^{(b\tau)}$.*

Proof. Let us first assume that there is no channel noise. For $\tau \leq D_{max}$, due to the

frozen bits for the first $r_d b$ sub-blocks and the insetion/removal of the CPs for the last $(1 - r_d)b$ sub-blocks, the linear shift by τ introduced by the channel has been transformed into circular shift of each sub-block by τ. This is written with a slight abuse of notation as

$$\bar{\mathbf{x}}^{(\tau)} \triangleq [\bar{\mathbf{x}}^{(\tau)}[1], \bar{\mathbf{x}}^{(\tau)}[2], \ldots, \bar{\mathbf{x}}^{(\tau)}[b]], \tag{3.13}$$

where for $s \in \{1, \ldots, b\}$,

$$\bar{\mathbf{x}}^{(\tau)}[s] = [\bar{x}^{L-\tau+1}[s], \ldots, \bar{x}^{L}[s], \bar{x}^{1}[s], \ldots, \bar{x}^{L-\tau}[s]] \tag{3.14}$$

is the circularly shifted version of $\bar{\mathbf{x}}[s]$ by τ positions. One can then verify that the output of the deinterleaver with this input would be $\tilde{\mathbf{x}}^{(b\tau)}$ which is corresponding to the codeword $\mathbf{c}^{(b\tau)}$ if a b-QC code is employed. Therefore, in the presence of channel noise, the received signal would be a noisy signal corresponds to $\mathbf{c}^{(b\tau)}$ and hence we can directly decode $\mathbf{c}^{(b\tau)}$ instead of $\mathbf{c}$. □

Theorem 1. *There exists a sequence of IDT-QC linear/lattice codes that achieve the capacity of the asynchronous point-to-point AWGN channel.*

Proof. Let $L > D_{max}$ and let $\mathcal{C}^1, \mathcal{C}^2, \ldots, \mathcal{C}^L$ be identical (b, k) linear/lattice codes that can approach the capacity [27] when $b \to \infty$ for a fixed k/b. i.e., for a $\varepsilon > 0$, there is a large enough b such that $k/b \log(p) > C(P, 0) - \varepsilon$ and $P_e^{(b)} < \varepsilon/L$. Moreover, in order to make these codes fit into the aforementioned form, the generator matrices of these codes should be systematic. We would like to construct a capacity-achieving IDT-QC codes $\mathcal{C}$ for the asynchronous AWGN channel from $\mathcal{C}^1, \mathcal{C}^2, \ldots, \mathcal{C}^L$. For every $\mathbf{c}^l \in \mathcal{C}^l$ for $l \in \{1, 2, \ldots, L\}$, we construct the codeword

$$\mathbf{c} = [\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^L]. \tag{3.15}$$

Using the fact that $\mathcal{C}^1, \mathcal{C}^2, \ldots, \mathcal{C}^L$ are identical linear/lattice codes, one can see that the collection of such codewords forms a (bL, kL) b-QC code with design rate $R_d = r_d \log(p)$ where $r_d = K/N' = k/b$. The codeword is then modulated to $\tilde{\mathbf{x}}$, fed into the interleaver to form $\bar{\mathbf{x}}$, bits-frozen and CP-appended to get $\mathbf{x}$.

The receiver observes a noisy version of the transmitted codeword delayed by τ which can be easily compensated as τ is known by the receiver. It then removes the CP and feeds the signal to the deinterleaver to get a noisy version of the original signal $\mathbf{x}$. The error probability of this IDT-QC code can be bounded using the union bound as

$$P_e^{(Lb)} < L \cdot \frac{\varepsilon}{L} = \varepsilon, \tag{3.16}$$

and from (3.12), one has the actual rate given by

$$R_a(b, L) = \left(1 - \frac{(2b-k)D_{max}}{bL + (b-k)D_{max}}\right) R_d. \tag{3.17}$$

Now, letting b go to infinity results in $R_d \to C$ and

$$R_a(L) = \lim_{b\to\infty} R_a(b, L) = \left(1 - \frac{(2 - r_d)D_{max}}{L + (1 - r_d)D_{max}}\right) C, \tag{3.18}$$

which in turn results in $\lim_{L\to\infty} R_a(L) = C$ which completes the proof. □

Remark 1. *Note that the ensemble of codes that we construct in Theorem 1 is not necessarily a good ensemble in the sense that for a particular target error rate, it requires a very long code length to achieve that error rate. In practice, QC codes are usually not constructed this way and QC codes with not very long block length comparing to the one constructed here can perform very well. Since this is the case, we only use this ensemble for the proof and construct QC codes for simulation by existing constructions such as AR4JA.*

3.3.3 Advantages and disadvantages of IDT-QC codes

Here, we provide a discussion of some advantages and disadvantages of the proposed IDT-QC codes.

Advantages:

- The proposed IDT-QC codes substantially generalize the idea of [47]. Unlike [47] which only considers allowing the use of convolutional codes and only works for the two-way relay channel, the proposed framework is more general in that it can take any QC codes and would work for a larger class of networks as will be shown later on. The use of QC codes provides significant improvement in practice as there are families of QC codes (e.g. AR4JA codes [59]) that can work very close to the Shannon limit with reasonable decoding complexity (iterative decoding). On the other hand, for convolutional codes, one has to use a very large constraint length (usually with formidably high decoding complexity) in order to approach the Shannon limit.

- Compared to the use of cyclic codes as in [46] [23], our approach enjoys a better error-correcting capability provided by QC codes. This can be easily seen by the fact that QC codes contain the family of cyclic codes as a special case. Our simulation in the following sections show that even when we consider the rate loss introduced by the IDT, IDT-QC codes significantly outperform cyclic codes. Moreover, unlike cyclic codes, the proposed IDT-QC codes can be easily shown to achieve the Shannon limit.

- In practice, QC codes (QC LDPC codes in particular) have been very popular and have been adopted in several communication standards (e.g., DVB-S2, WiMax, and 5G/NR) due to the existence of efficient encoding and decoding

algorithms . The proposed IDT-QC codes naturally inherit those practical benefits from QC codes and hence are practically attractive.

- As will be unveiled in the following sections, the proposed IDT-QC codes can not only harness interference in the presence of asynchronism, but also exploit asynchronism in some cases.

Disadvantages:

- Due to the use of interleaver and deinterleaver in our proposed IDT-QC codes, the transmitter has to wait until the entire codeword is generated before transmission. This results in an increased encoding latency.

- While the rate loss is negligible as $L \to \infty$ when we prove the capacity result in Theorem 1, it must be taken into account in the finite length regime. However, in the following sections, we will show that the proposed IDT-QC codes outperform cyclic codes under message-passing decoding even when rate loss is included.

3.4 Application 1: Integer-Forcing Equalization for ISI Channels

In this section, we consider point-to-point communication with ISI shown in Fig. 3.6. It has been known that the capacity of the ISI channel can be achieved by multi-carrier systems. However, the high peak-to-average power ratio makes this approach less attractive for applications requiring extremely low complexity such as wireless sensor networks. Another way to approach capacity is to code over time domain and uses a decision feedback equalizer (DFE) at the receiver. For this to work, a very long interleaver/deinterleaver (between multiple codewords) is required to avoid error propagation. Recently, Ordentlich and Erez in [46] proposed a new linear equalization technique called integer-forcing equalization. This new

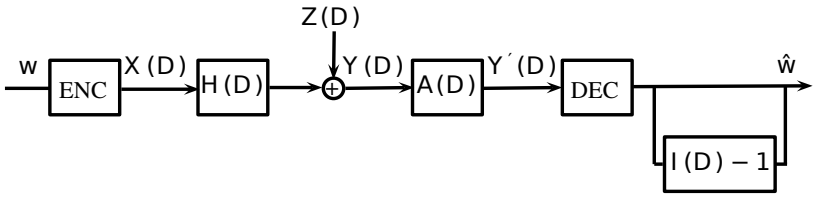


Figure 3.6: The integer-forcing equalization system.

equalization technique does not require inteleaver/deinterleaver between codewords and avoids the error propagation as no DFE is implemented. However, one of the drawbacks is that here the channel coding adopted is required to be cyclic. Although it has been shown that cyclic code can achieve the capacity of an erasure channel [48], but only under ML decoding. To the best of author's knowledge, the discovery of efficient ML decoding algorithms for cyclic codes remains to be an open problem. In what follows, we will replace the cyclic codes by the proposed IDT-QC codes which are not only able to achieve the upper bound on information rates presented in [46], but also exhibits excellent decoding performance when efficient message-passing decoder is adopted. It should be noted that unlike the DFE-based scheme, the interleaver/deinterleaver for the proposed framework is within a QC codeword and hence is substantially shorter than that in DFE-based schemes.

3.4.1 Problem statement

The transmitter encodes its message $\mathbf{w} \in \mathbb{F}_p^K$ to a codeword $\mathbf{c} \in \mathbb{F}_p^N$ which is then mapped to a signal $\mathbf{x} \in \mathcal{A}^N$ via $\mathcal{M}$ where $\mathcal{A}$ is the signal constellation (e.g., M-PAM) and $\mathcal{M}$ is the natural mapping. This signal is subject to a power constraint P and is sent over an AWGN channel with ISI $\mathbf{h} = [h_1, \ldots, h_{d_M}]$ where d_M depends on the

maximal delay spread and the sampling frequency. The received signal is given by

$$\mathbf{y} = \mathbf{h} * \mathbf{x} + \mathbf{z}. \tag{3.19}$$

i.e., the received signal would be a noisy version of a linear combination of the codeword linearly shifted by integers. We consider a recently proposed linear equalizer called integer-forcing equalizer proposed in [46]. This technique first passes $\mathbf{y}$ to a linear equalizer chosen in such a way that the equalized channel impulse responses are forced to be an integer vectors $\mathbf{i} \triangleq [i_0, i_1, \ldots, i_{D_{max}}]$. Also, one can easily transform linear convolution into circular convolution. The equalized signal is then given by

$$\mathbf{y} = \sum_{d=0}^{D_{max}} i_d \mathbf{x}^{(d)} + \mathbf{z}', \tag{3.20}$$

where $\mathbf{z}'$ is the filtered noise.

The authors in [46] then proposed using cyclic codes over $\mathbb{F}_p$ at the transmitter so that $\varphi(\sum_{d=0}^{D_{max}} i_d \mathbf{x}^{(d)})$ with $\varphi \triangleq \mathcal{M}^{-1} \circ \mod p$ is a codeword of the same cyclic code. Therefore, one can directly decode $\varphi(\sum_{d=0}^{D_{max}} i_d \mathbf{x}^{(d)})$ from $\mathbf{y} \mod p$. This decoded signal is then used to recover $\mathbf{x}$ and hence $\mathbf{w}$. In what follows, we propose using the IDT-QC codes to replace the cyclic codes. Since the problem of designing and analyzing integer-forcing equalizers has been well addressed in [46], we assume that the ISI channel has already been integral. i.e., $\mathbf{h} = \mathbf{i}$.

3.4.2 Using IDT-QC for point-to-point communication with ISI

After the integer-forcing equalizer, the signal becomes linear combination of linearly-shifted versions of the transmitted signal with integer coefficients $i_d \in \mathbb{Z}$ for $d \in \{0, \ldots, D_{max}\}$. As shown in Fig. 3.7 where an example with $D_{max} = 2$ is given, the receiver then removes the received signal at the positions where the CP of

the first tap's signal should be. Then the signal can be expressed as a noisy version of the linear combination of the codeword circularly shifted by integers given by

$$\bar{\mathbf{y}} = \sum_{d=0}^{D_{max}} i_d \bar{\mathbf{x}}^{(d)} + \bar{\mathbf{z}}, \tag{3.21}$$

where $\bar{\mathbf{x}}^{(d)}$ is as in (3.13) and elements in $\bar{\mathbf{z}}$ and those in $\mathbf{z}$ have the same distribution. As shown in Lemma 1, any linear shift by an integer $d \leq D_{max}$ introduced by the channel will be transformed by the IDT into $b \cdot d$ circular shift for the codeword. i.e., the channel would be transformed into

$$\tilde{\mathbf{y}} = \sum_{d=0}^{D_{max}} i_d \tilde{\mathbf{x}}^{(bd)} + \tilde{\mathbf{z}}, \tag{3.22}$$

where elements in $\tilde{\mathbf{z}}$ and those in $\mathbf{z}$ have the same distribution. Moreover, since IDT-QC codes adopt b-QC codes for channel coding, every $\tilde{\mathbf{x}}^{(bd)}$ in (3.22) corresponds to a valid codeword in the underlying QC code. This in turn allows us to directly decode $\tilde{\mathbf{y}}$ mod p to a valid codeword $\varphi\left(\sum_{d=0}^{D_{max}} i_d \tilde{\mathbf{x}}^{(bd)}\right)$ (or $\varphi(I(D^b)X(D))$ in the D-domain) in the same QC code. After this codeword is decoded, one can use the knowledge of frozen bits to strip out all the message bits. This is perhaps easier seen from the second representation and is shown in Fig. 3.7. It can be seen that within each sub-block corresponding to the message part, one can initiate the deconvolution since the last D_{max} bits are frozen.

It has been shown in Section 3.3 that there exists a sequence of the proposed IDT-QC codes that can achieve capacity. Thus, theoretically, using the proposed framework allows one to achieve the upper bound on information rates presented in [46] which may not be achievable for the cyclic coding scheme proposed therein. Therefore, the proposed framework bridges the gap-to-capacity for such integer-

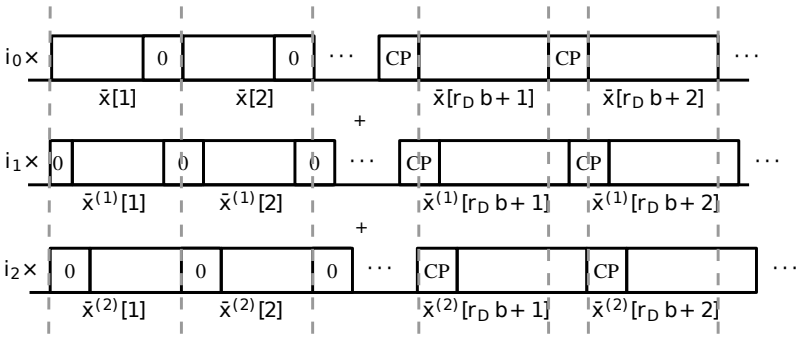


Figure 3.7: The idea of using IDT-QC for point-to-point communication with ISI.

forcing equalization schemes. In what follows, we provide some simulation results to demonstrate that the proposed IDT-QC codes outperform cyclic codes even though for the finite-length regime, the proposed IDT-QC codes suffer from a rate loss. It is worth mentioning that in addition to being of independent interest, the integer-forcing equalization for ISI channel will play an important role in using IDT-QC for asynchronous compute-and-forward.

3.4.3 Simulation results

We now provide some simulation results to compare the proposed IDT-QC framework and the cyclic coded scheme proposed in [46]. We consider the dicode channel whose impulse response is $I(D) = 1 + D$ (i.e., $i_0 = i_1 = 1$); therefore, $D_{max} = 1$. We construct an binary IDT-QC LDPC code from the AR4JA ensemble [55] with $N' = 4096$, $b = 32$, $K = 3072$, and the design rate $R_d = 0.75$. The actual rate of this code is $R_a = 0.742$. For comparison with the scheme in [46], we also construct a cyclic LDPC code from the ensemble proposed in [60] with $N' = N = 4095$, $K = 2703$, and the design rate $R_d = 0.66$. Note that for the cyclic coded scheme in [46], one has to freeze D_{max} bits for initializing the deconvolution. This results

in the actual rate $R_a \approx R_d = 0.66$. For the both codes, the decoding algorithm is a message-passing algorithm with at most 200 iterations. Since $I(D) = 1 + D$, the receiver attempts to decode $\mathbf{c} \oplus \mathbf{c}^{(b)}$. Simulation results presented in Fig. 3.8 show that in spite of having a higher rate, the proposed IDT-QC LDPC code provides roughly 1.1 dB gain when BER is at 10^{-5}. This is mainly because the proposed IDT transform enables the use of QC codes where very powerful ensembles such as AR4JA can be easily constructed. Moreover, the conventional scheme in [46] relies solely on the family of cyclic codes which is much smaller than that of QC codes.

We also provide the information rate corresponding to the independent uniformly distributed input distribution for the dicode channel estimated by the method in [61] (which is equivalent to the forward recursion of the BCJR algorithm). One observes that there is a roughly 4.4dB gap between the proposed scheme and the corresponding information rate at $P_e \approx 10^{-5}$. This gap comes from the following sources. First and foremost, a receive filter has not been used to harness all the energy in all the taps of the ISI channel. In this example, this contributes a 3 dB loss. A second source is the power loss inherited from the integer-forcing equalization approach which transforms the channel into a $\mod p$ channel (here $\mod 2$). The final source of this gap simply comes from the fact that the block length we consider here (4096) is rather small. Similar but larger gap can be observed for the cyclic coded integer-forcing equalization scheme.

It should be noted that for a single user ISI channel, using conventional equalization techniques such as a decision feedback equalizer (DFE) will provide better results than integer-forcing equalization. However, when a compute-and-forward problem is considered with multiple users and ISI, conventional equalization techniques will not be sufficient to efficiently compute functions of transmitted signals since the interference from multiple users and ISI cannot be simultaneously removed easily. In

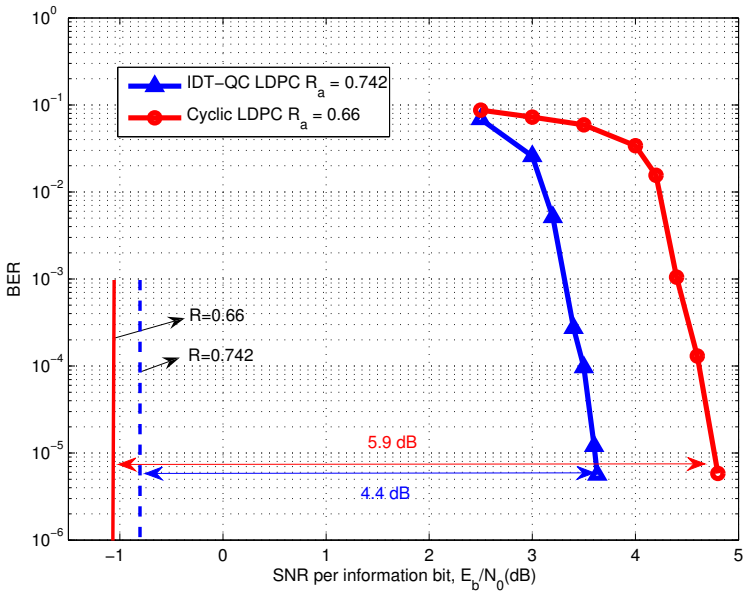


Figure 3.8: BER comparison of the IDT-QC LDPC code with $R_a = 0.742$ and the cyclic LDPC code with $R_a = 0.66$ over the dicode channel $(1 + D)$. The dashed and solid vertical lines are the required SNRs corresponding to the information rates $R = 0.742$ and $R = 0.66$, respectively.

these cases, integer-forcing equalization can substantially outperform conventional equalization techniques.

3.5 Application 2: Asynchronous Compute-and-Forward

In this section, we study the compute-and-forward relay network introduced by Nazer and Gastpar [12]. In particular, we consider the asynchronous version of this network where signals sent from different source nodes may arrive at a destination node at different times. In the synchronous case, the compute-and-forward strategy suggested in [12] implements an *identical* nested lattice code [27] at each user and directly decodes the received signal to a modulo version of linear combination of the codewords with integer coefficients at the destination. This scheme is shown to provide a substantially higher computation rate in the medium signal-to-noise ratio

(SNR) regime than existing schemes. For practical purposes, in [62], this nested lattice code is replaced by linear code over $\mathbb{F}_p$ together with a signal mapping possessing the property described in Section 3.2 in order to exploit the structural gain. The destination then decodes the received signal to a linear combination of the codewords over $\mathbb{F}_p$.

There have been a few attempts at using physical-layer network coding to this asynchronous setting. A convolutional coded scheme has been proposed in [47] to deal with synchronization errors; however, only integer-valued delays (i.e., frame-level asynchronism) are allowed. In [22], an over-sampling method was proposed and a graph-based decoding algorithm has been proposed specifically for this over-sampling model. This over-sampling method can take real-valued delays (i.e., symbol-level asynchronism) but only within one symbol time; thus, results in a stringent timing synchronization. In [24], frame-level and symbol-level asynchronous compute-and-forward are considered where the destinations are only able to compute synchronous functions. A very recent work in [23] has successfully applied cyclic codes to this problem so that asynchronous functions are computable and showed through simulation that cyclic codes are able to combat with real-valued delays within one *packet* time.

In this section, we replace the cyclic codes by the proposed IDT-QC codes and show that this replacement allows us to prove capacity results for both the frame-level and symbol-level cases. Moreover, the simulation results given in Section 3.6 show that this replacement substantially improves the performance. It should be noted that since the proposed scheme relies on the quasi-cyclic property instead of the cyclic property to deal with synchronization errors, the delay constraint is more stringent than the scheme in [23]. Nonetheless, it only requires the delays to be controlled within a certain range D_{max} (say few symbols time), which is practically

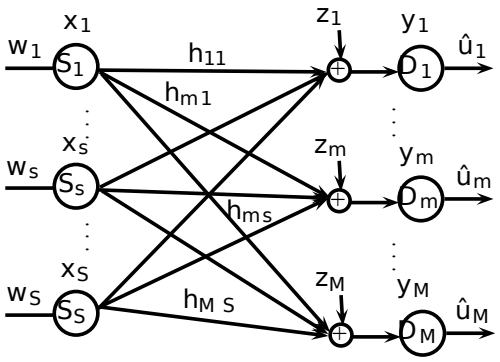


Figure 3.9: The compute-and-forward relay network.

reasonable.

3.5.1 Problem statement

As shown in Fig. 3.9, in a compute-and-forward network, there are total S source nodes and $M \geq S$ destination nodes. Each source node $s \in \{1, \ldots, S\}$ encodes its message $\mathbf{w}_s \in \mathbb{F}_p^K$ to a codeword $\mathbf{c}_s \in \mathbb{F}_p^N$. This codeword is then modulated to the transmitted signal $\mathbf{x}_s \in \mathcal{A}^N$ via a mapping $\mathcal{M}$ as described in Section 3.2. The codeword is subject to a power constraint given by

$$\frac{1}{N}\|\mathbf{x}_s\|^2 = \frac{1}{N}\sum_{n=1}^{N}|x_s[n]|^2 \leq P. \tag{3.23}$$

Let τ_{ms} be the delay experienced by the signal from source s to destination m. We will separately consider two cases, namely the frame-level asynchronous compute-and-forward where $\tau_{ms} \in \{0, \ldots, D_{max}\}$ and the symbol-level asynchronous compute-and-forward where $\tau_{ms} \in [0, T)$ with T being symbol duration. For the frame-level

asynchronous one, the received signal is given by

$$y_m[n] = \sum_{s=1}^{S} h_{ms} x_s[n - \tau_{ms}] + z_m[n], \tag{3.24}$$

where $h_{ms} \in \mathbb{R}$ (or $\mathbb{C}$ depending on whether the signal constellation is real or complex) is the channel coefficient between the source node s and destination m, and $z_m[n] \sim \mathcal{CN}(0, 1)$. For the symbol-level asynchronous compute-and-forward, one has to work with the continuous-time model given by

$$y_m(t) = \sum_{s=1}^{S} \sum_{n=1}^{N} h_{ms} x_s[n] p(t - nT - \tau_{ms}) + z(t), \tag{3.25}$$

where $p(t)$ is the pulse shaping function and $z(t)$ is a Gaussian process with zero mean and variance 1.

The destination node m is only interested in computing and forwarding a function of the messages. In particular, the compute-and-forward scheme in [12] confines itself to synchronous functions of the messages which mimics the behavior of linear network coding. i.e., the destination node m chooses $\{b_{ms}\}$ and computes $\mathbf{u}_m = \oplus_{s=1}^{L} b_{ms} \mathbf{w}_s$ such that the computation rate at node m is maximized. The computed functions together with $\{b_{ms}\}$ are then forwarded to a central destination which desires all the messages. It is clear that as long as the coefficients $\{b_{ms}\}$ form a full-rank matrix, the central destination would be able to invert the matrix and obtain all the messages. In the sequel, we will show that the use of IDT-QC codes allows one to compute asynchronous functions which may lead one to an increased computation rate. For ease of exposition, we will separately discuss the frame-level and symbol-level models and restrict ourselves to $S = M = 2$, but the proposed scheme works for general scenarios.

3.5.2 Frame-level asynchronous compute-and-forward

We illustrate the idea of using the proposed IDT-QC codes for frame-level asynchronous compute-and-forward. Each source node adopts a same b-QC code over $\mathbb{F}_p$ for encoding its message to the codeword $\mathbf{c}_s$ which is then modulated to the signal $\tilde{\mathbf{x}}_s = \mathcal{M}(\mathbf{c}_s)$. It will then be interleaved to form $\bar{\mathbf{x}}_s$ and further added frozen bits and appended CPs to form the transmitted signal $\mathbf{x}_s$. The length of the CPs is again set to be D_{max}. One difference here is that for a compute-and-forward network with S source nodes, one has to freeze SD_{max} positions instead of D_{max} positions for each sub-block corresponding to message part. As shown in Fig. 3.10, the receiver removes the signal at the positions where the first source node's CP should be and the received signal becomes

$$\bar{\mathbf{y}}_m = a_{m1}\bar{\mathbf{x}}_1^{(\tau_{m1})} + a_{m2}\bar{\mathbf{x}}_2^{(\tau_{m}2)} + \bar{\mathbf{z}}_{eq,m}, \tag{3.26}$$

where $\bar{\mathbf{x}}_s^{(\tau_{ms})}$ is as in (3.13) and $\bar{\mathbf{z}}_{eq,m}$ is the effective noise which consists of the noise and the self-interference [12]. The destination node then feeds this signal into the deinterleaver. As discussed in Lemma 1, the proposed IDT-QC codes transform any integer delay τ introduced by the channel into $b \cdot \tau_{ms}$ circular shifts for the codeword. Thus, the deinterleaver output is given by

$$\tilde{\mathbf{y}}_m = a_{m1}\tilde{\mathbf{x}}_1^{(b\tau_{m1})} + a_{m2}\tilde{\mathbf{x}}_2^{(b\tau_{m2})} + \tilde{\mathbf{z}}_{eq,m}, \tag{3.27}$$

where elements in $\tilde{\mathbf{z}}_{eq,m}$ and that in $\bar{\mathbf{z}}_{eq,m}$ have the same distribution. The receiver m then attempts to compute the lattice point $a_{m1}\tilde{\mathbf{x}}_1^{(b\tau_{m1})} + a_{m2}\tilde{\mathbf{x}}_2^{(b\tau_{m2})}$ and uses the

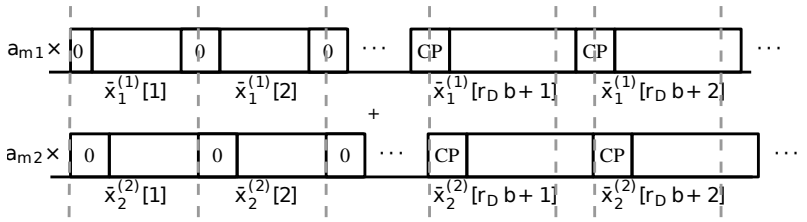


Figure 3.10: An example of using IDT-QC for asynchronous compute-and-forward where $\tau_{m1} = 1$, $\tau_{m2} = 2$, and $D_{max} = 2$.

ring homomorphism $\varphi \triangleq \mathcal{M}^{-1} \circ \mod p$ to map this lattice point to

$$\varphi(a_{m1}\tilde{\mathbf{x}}_1^{(b\tau_{m1})} + a_{m2}\tilde{\mathbf{x}}_2^{(b\tau_{m2})}) = b_{m1} \odot \mathbf{c}_1^{(b\tau_{m1})} \oplus b_{m2} \odot \mathbf{c}_2^{(b\tau_{m2})}, \tag{3.28}$$

where $b_{ms} \triangleq \varphi(a_{ms})$. It should be noted that since the underlying code we adopt is a b-QC code, $\mathbf{c}_s^{(b\tau_{ms})}$ is a codeword and so is $\mathbf{f}_m \triangleq b_{m1} \odot \mathbf{c}_1^{(b\tau_{m1})} \oplus b_{m2} \odot \mathbf{c}_2^{(b\tau_{m2})}$.

The computed functions and those coefficients are then forwarded to the central destination and are then further processed to recover all the messages. We now show that the compute-and-forward problem with full rank coefficients can be equivalently represented as ISI channel problems in Section 3.4 and can be solved by deconvolution if sufficient initial conditions are provided. For the sake of simplicity, we look at the interleaved version of $\mathbf{f}_m$ given by

$$\bar{\mathbf{f}}_m \triangleq b_{m1} \odot \bar{\mathbf{c}}_1^{(\tau_{m1})} \oplus b_{m2} \odot \bar{\mathbf{c}}_2^{(\tau_{m2})}. \tag{3.29}$$

In the D-domain, one has that

$$\begin{aligned}\bar{\mathbf{F}} &= \begin{pmatrix} \bar{F}_1(D) \\ \bar{F}_2(D) \end{pmatrix} \\ &= \begin{pmatrix} b_{11}D^{\tau_{11}} & b_{12}D^{\tau_{12}} \\ b_{21}D^{\tau_{21}} & b_{22}D^{\tau_{22}} \end{pmatrix} \odot \begin{pmatrix} \bar{C}_1(D) \\ \bar{C}_2(D) \end{pmatrix} \\ &\triangleq \bar{\mathbf{B}} \odot \bar{\mathbf{C}}. \end{aligned} \tag{3.30}$$

Note that mathematically, one can now left-multiply by the inverse of the matrix $\bar{\mathbf{B}}$ to get $\bar{\mathbf{C}}$. In order to endow this inverse an operational meaning, we note that for every full rank matrix $\bar{\mathbf{B}}$, one has

$$\bar{\mathbf{B}}^{-1} = \frac{\text{adj}(\bar{\mathbf{B}})}{\det(\bar{\mathbf{B}})}, \tag{3.31}$$

where det(.) is the determinant and adj(.) is the adjugate. One can then left multiply $\mathbf{F}$ with adj($\bar{\mathbf{B}}$) to form

$$\text{adj}(\bar{\mathbf{B}}) \odot \bar{\mathbf{F}} = \det(\bar{\mathbf{B}}) \odot \bar{\mathbf{C}}. \tag{3.32}$$

One observes that the problem has been converted into two separate ISI channel problems whose impulse responses are integer vectors. Moreover, since each element in $\bar{\mathbf{B}}$ has the range $\{0, \ldots, D_{max}\}$, each element in $\det(\bar{\mathbf{B}})$ has range $\{0, \ldots, 2D_{max}\}$ or in general $\{0, \ldots, SD_{max}\}$. Therefore, this problem can be solved by deconvolution provided that the transmitter freeze SD_{max} positions for each sub-block belonging

to the message part. The actual rate then becomes

$$\begin{aligned} R_a &= \frac{K - r_d bSD_{max}}{N' + (1-r_d)bD_{max}} \log(p) \\ &= \left(1 - \frac{(S+1-r_d)D_{max}}{L + (1-r_d)D_{max}}\right) R_d, \end{aligned} \tag{3.33}$$

which does not affect the asymptotic results. One example is given in the following.

Example 1. *Consider a 2-by-2 example over* $\mathbb{F}_2$*. Suppose that relay 1 receives*

$$y_1[n] = x_1[n-1] + x_2[n] + z_1[n], \tag{3.34}$$

and relay 2 receives

$$y_2[n] = x_1[n] + x_2[n-1] + z_2[n]. \tag{3.35}$$

Then

$$\bar{\mathbf{B}} = \begin{pmatrix} D & 1 \\ 1 & D \end{pmatrix}. \tag{3.36}$$

We have $\det(\bar{\mathbf{B}}) = 1 + D^2$*, and* $\mathrm{adj}(\bar{\mathbf{B}}) = \bar{\mathbf{B}}$*. Thus, by left-multiplying* $\mathrm{adj}(\bar{\mathbf{B}})$*, one has*

$$\begin{pmatrix} D\bar{F}_1(D) + \bar{F}_2(D) \\ \bar{F}_1(D) + D\bar{F}_2(D) \end{pmatrix} = \begin{pmatrix} (1+D^2)\bar{C}_1(D) \\ (1+D^2)\bar{C}_2(D) \end{pmatrix}, \tag{3.37}$$

which are two separate ISI channel problems. What (3.37) implies is that the receiver separately decodes $\bar{\mathbf{c}}_1 \oplus \bar{\mathbf{c}}_1^{(2b)}$ *and* $\bar{\mathbf{c}}_2 \oplus \bar{\mathbf{c}}_2^{(2b)}$ *where* b *is the shifting constraint. Then using the frozen bits, deconvolutions are performed to obtain* $\bar{\mathbf{c}}_1$ *and* $\bar{\mathbf{c}}_2$

We now present the main theorem of this section.

Theorem 2. *Consider the frame asynchronous case where* $\tau_{ms} \in \{0, \ldots, D_{max}\}$*. At*

relay m, given $\mathbf{h}_m$ and $\mathbf{a}_m$, a computation rate of

$$\mathcal{R}(\mathbf{h}_m, \mathbf{a}_m) = \frac{1}{2}\log^+\left(\left(\|\mathbf{a}_m\|^2 - \frac{P|\mathbf{h}_m^H\mathbf{a}_m|^2}{1+P\|\mathbf{h}_m\|^2}\right)^{-1}\right), \tag{3.38}$$

where $a_{ms} \in \mathbb{Z}$, is achievable per real dimension.

Proof. We start by reviewing the nested lattice code of Erez and Zamir [27] adopted by [12] for achieving the computation rate in the absence of synchronization errors. Let Λ_f and Λ_c be two b-dimensional lattices with the relationship $\Lambda_c \subseteq \Lambda_f$. A nested lattice code is a code using the minimum-energy coset representatives of Λ_f/Λ_c as codewords. i.e., $\mathcal{C} \triangleq \Lambda_f \cap \mathcal{V}_{\Lambda_c}$ where $\mathcal{V}_\Lambda$ is the fundamental Voronoi region of a lattice Λ. The rate of a nested lattice codes is given by

$$R = \frac{1}{b}\log\left(\frac{\mathrm{Vol}(\mathcal{V}_{\Lambda_c})}{\mathrm{Vol}(\mathcal{V}_{\Lambda_f})}\right). \tag{3.39}$$

Moreover, lattices in [27] are constructed by Construction A with (b,k) linear codes over $\mathbb{F}_p$; hence, one can further rewrite the rate as $R = k/b\log(p)$. We denote such nested lattice codes as (b,k) nested lattice codes.

Now, let $\mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \ldots, \mathcal{C}^{(L)}$ be identical (b,k) nested lattice code $\Lambda_f \cap \mathcal{V}_{\Lambda_c}$ that can achieve $\mathcal{R}(\mathbf{h}_m, \mathbf{a}_m)$ given in (4.24) in the absence of synchronization errors. i.e., for a $\varepsilon > 0$, there exist sufficiently large b and p such that $k/b\log(p) > \mathcal{R}(\mathbf{h}_m, \mathbf{a}_m) - \varepsilon$ and $P_e^{(b)} < \varepsilon/L$. Let us again concatenate L codes to form a super-code being the collection of $\mathbf{c} = [\mathbf{c}^1, \mathbf{c}^2, \ldots, \mathbf{c}^L]$. We then feed codewords of the super-code into the IDT transform to freeze bits and add CP. Note that every Construction A lattice can be easily put into a systematic form and hence freezing bits is feasible.

From (3.27), one can see that the proposed IDT transform would make the re-

ceived signal a noisy version of

$$[a_{m1}\mathbf{x}^{L-\tau_{m1}+1},\ldots,a_{m1}\mathbf{x}^{L-\tau_{m1}}]+$$
$$[a_{m2}\mathbf{x}^{L-\tau_{m2}+1},\ldots,a_{m2}\mathbf{x}^{L-\tau_{m2}}]. \tag{3.40}$$

Each b sub-block is now a perfectly synchronized compute-and-forward problem. Therefore, we can compute $a_{m1}\mathbf{x}^l + a_{m2}\mathbf{x}^{l-\tau_{m2}+\tau_{m1} \bmod L}$ for each $l \in \{1,\ldots,L\}$ separately and use the mapping φ to obtain linear combinations in $\mathbb{F}_p$. The error probability can be union bounded by $P_e^{Lb} < \varepsilon$ and the actual rate of this strategy is given by (3.33). Now, letting $b, p \to \infty$ results in vanishing ε and the rate of each nested lattice sub-code would approach $\mathcal{R}(\mathbf{h}_m, \mathbf{a}_m)$. Moreover, letting $L \to \infty$ would make the actual rate converge to the design rate $\mathcal{R}(\mathbf{h}_m, \mathbf{a}_m)$. This completes the proof. □

3.5.3 Achieving higher rates than in the synchronous case

One important observation here is that the proposed QC-IDT scheme allows one to exploit another dimension, namely the delay dimension. This is due to the fact that the QC nature of the proposed scheme enables the computation of asynchronous functions in addition to synchronous ones. Sometimes, this allows one to achieve rates *surpassing* that achieved by tightly synchronous compute-and-forward with the same channel coefficients. For example, the matrix $\bar{\mathbf{B}}_{sync} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$ is not invertible; however, the matrix in (3.36) is invertible. It must be noted that the delays are completely determined by the channel so that one does not have control over those parameters. But instead of being limited by those delays, the proposed scheme is capable of exploiting them. One example of how the delay may improve the system performance is given in the following.

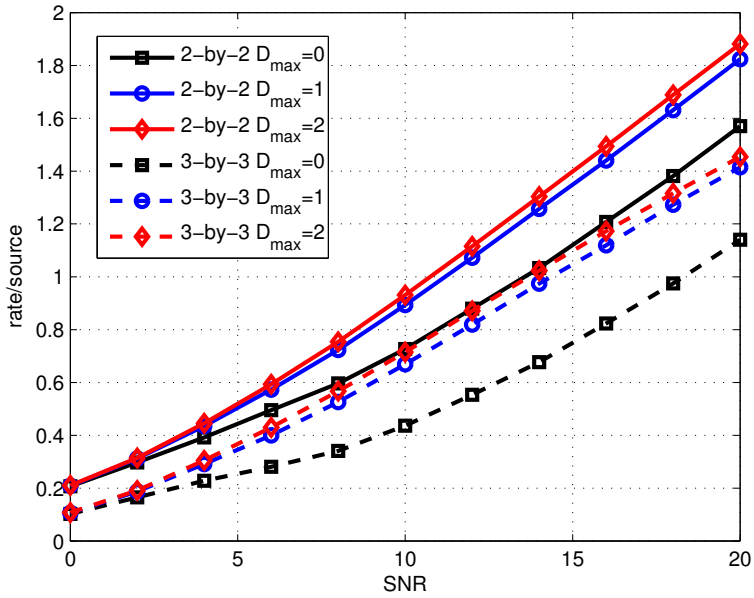


Figure 3.11: Achievable rates of asynchronous compute-and-forward (average over 10000 realizations).

Example 2. *In Fig. 3.11, we plot the achievable computation rates of asynchronous compute-and-forward, for the cases where $S = M = 2$ and $S = M = 3$ respectively. The channel coefficients h_{ms} are drawn from i.i.d. Rayleigh distribution. One can see from this figure that for both cases, increasing D_{max} substantially increases achievable computation rates. This effect is most pronounced when D_{max} is increased from 0 to 1. This example demonstrates that when the channel introduces delays, using the proposed scheme which allows the decoding of asynchronous functions results in higher achievable computation rates.*

Remark 2. *Frame asynchronous compute-and-forward has been considered in [24]. The scheme therein does not possess the QC or cyclic properties so that the relays are forced to compute synchronous functions only. As a consequence, they have to use multiple antenna at the relays to rotate the received signal in order to recover syn-*

chronous functions in the presence of frame-synchronization errors. This introduces a huge loss not just in rates but also in degrees of freedom because multiple antennas are used just for computing one function at each relay. On the other hand, thanks to the QC nature of the proposed scheme, the computation rates given in Theorem 2 have the exactly same form with that in [12], i.e., as there was no frame asynchronism at all. This is a direct consequence of enabling computing asynchronous functions which are undecodable in [24]. However, this gain does not come for free; this gain comes with an increased burden in the next phase since in addition to a_{ms} (or b_{ms} equivalently), the relays also have to forward the delay profile to the central destination. But this is usually not an issue as the bottleneck is usually the first phase.

3.5.4 Using IDT-QC for symbol-level asynchronous compute-and-forward

We now focus on the symbol-level asynchronous compute-and-forward. i.e., $\tau_{ms} \in [0, T)$ and the continuous-time model in (3.25) is considered. Similar to [17] [22] [25], we further assume that the pulse shaping function adopted is the ideal (rectangular) pulse. Let π_m be the permutation operation at the relay m defined by

$$\pi_m(1, \ldots, S) = (j_1, \ldots, j_S), \tag{3.41}$$

such that $\tau_{mj_1} \leq \ldots \leq \tau_{mj_S}$. In order to extract out all the energy, in general, one can perform S different matched filter $p_{mi}(t)$ for $i \in \{1, \ldots, S\}$ as

$$p_{mi}(t) = \begin{cases} 0, & \tau_{mj_{i-1}} + (n-1)T \leq t < \tau_{mj_i} + (n-1)T \\ \sqrt{P}, & \tau_{mj_i} + (n-1)T \leq t < \tau_{mj_{i+1}} + (n-1)T \\ 0, & \tau_{mj_{i+1}} + (n-1)T \leq t < nT, \end{cases} \tag{3.42}$$

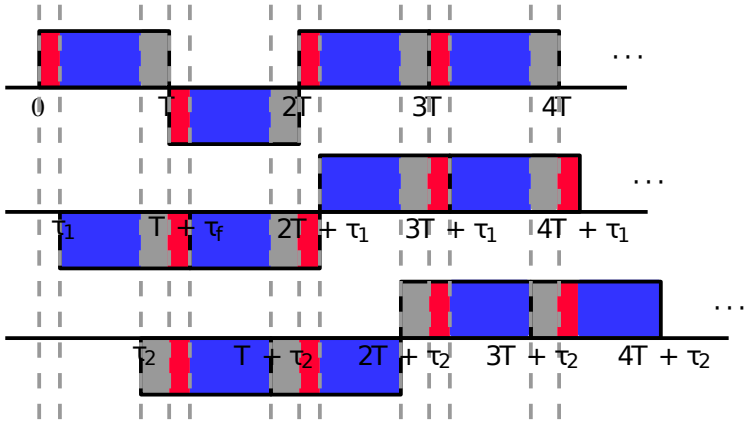


Figure 3.12: An example of SNR loss resulting from symbol-level asynchronous model where $\tau_{j_1} = 0$, $\tau_{j_2} = \tau_1$, and $\tau_{j_2} = \tau_2$.

where $\tau_{mj_0} = 0$ and $\tau_{mj_{S+1}} = T$ for each m. Note that the sampled output of different matched filters would correspond to different functions. For example, as shown in Fig. 3.12, for the $S = 3$ case, three different matched filters would correspond to three different functions, namely, $\mathbf{c}_{j_1} \oplus \mathbf{c}_{j_2}^{(b)} \oplus \mathbf{c}_{j_3}^{(b)}$, $\mathbf{c}_{j_1} \oplus \mathbf{c}_{j_2} \oplus \mathbf{c}_{j_3}^{(b)}$, and $\mathbf{c}_{j_1} \oplus \mathbf{c}_{j_2} \oplus \mathbf{c}_{j_3}$. The corresponding SNR are then given by

$$P_{mi} = PT(\tau_{mj_i} - \tau_{mj_{i-1}}). \tag{3.43}$$

We then pick the one with the highest SNR for compute-and-forward and discard the others. This would result in the following achievable computation rates.

Theorem 3. *Consider the symbol asynchronous case where $\tau_{ms} \in [0, T)$. At relay m, given $\mathbf{h}_m$ and $\mathbf{a}_m$, a computation rate of*

$$\mathcal{R}(\mathbf{h}_m, \mathbf{a}_m) = \frac{1}{2} \log^+ \left(\left(\|\mathbf{a}_m\|^2 - \frac{P_m |\mathbf{h}_m^H \mathbf{a}_m|^2}{1 + P_m \|\mathbf{h}_m\|^2} \right)^{-1} \right), \tag{3.44}$$

where $a_{ms} \in \mathbb{Z}$ *and* $P_m = \max_{i\in\{1,\dots,S\}} P_{mi}$*, is achievable per real dimension.*

Proof. Similar to the proof of Theorem 2 but replacing P by P_m. □

Remark 3. *In [24], the same problem has been studied and achievable computation rates similar to* (3.44) *but with* P_m *replaced by* P_{mjs} *has been achieved as only synchronous functions are computable. Our proposed scheme provides increased computation rates through allowing the computation of asynchronous functions.*

Remark 4. *The above scheme only works with the signals corresponding to the function with the highest SNR and completely ignores those corresponding to other functions. However, as will be discussed in the next section, in terms of error probability, one can take advantage of those information by jointly considering detection and decoding.*

3.6 Practical Detection and Decoding for Asynchronous Compute-and-Forward

In this section, we introduce a joint detection and decoding scheme for the proposed IDT-QC codes to alleviate the SNR loss in the presence of symbol-level asynchronism. This decoder is then used for generating simulation results which demonstrate that the proposed framework substantially outperforms the cyclic coding scheme [23]. We again begin with the continuous-time model in (3.25). We further restrict our attention to a specific relay and drop the subscript m for the sake of simplicity. This allows us to assume $\tau_1 = 0$ and $\tau_2 = \tau$ without loss of generality. Let $\tau \in [0, D_{max}]$ where $\tau = \tau_f + \tau_s$ with $\tau_f \in \{0, 1, ..., D_{max} - 1\}$ being the frame-level asynchronism and $\tau_s \in [0, 1)$ being the symbol-level asynchronism.

We use a set of matched filters similar to (3.42) to over-sample the received signal [22]. This will result in the following sampled outputs

$$r[2n-1] = h_1 x_1[n] + h_2 x_2[n-1] + z[2n-1], \tag{3.45}$$

with $x_2[0] = 0$ and

$$r[2n] = h_1 x_1[n] + h_2 x_2[n] + z[2n], \tag{3.46}$$

and $r[2(N-\tau_f)+1] = h_2 x_2[N-\tau_f] + z[2(N-\tau_f)+1]$ where $z[2n-1] \sim \mathcal{N}(0, 1/\tau_s)$, $z[2(N-\tau_f)+1] \sim \mathcal{N}(0, 1/\tau_s)$, and $z[2n] \sim \mathcal{N}(0, 1/(1-\tau_s))$. In what follows, we describe how to perform the detection and decoding based on this over-sampling model.

3.6.1 Joint MAP detection and JCF decoding

We now propose a joint detection and decoding scheme which can be deemed as the decoding scheme in [22] tailored specifically for the IDT-QC codes. The decoding algorithm is based on the Tanner graph given in Fig. 3.13. The top part of the Tanner graph with zigzag fashion is associated with the MAP detection which accommodates the correlation between two consecutive over-sampling symbols. The bottom part of the Tanner graph is precisely that of the underlying QC code but over $\mathbb{F}_{p^2}$, i.e., the ACNC decoder in [17] which we refer to as the joint compute-and-forward (JCF) decoder. Unlike [22], there is a pair of interleaver/deinterleaver between the MAP detection and the JCF decoding parts. Moreover, thanks to the ability described in Lemma 1, depending on the corresponding SNRs, the receiver can opt to decode either $\mathbf{c}_1 \oplus \mathbf{c}_2^{(b\tau_f)}$ or $\mathbf{c}_1 \oplus \mathbf{c}_2^{(b(\tau_f+1))}$. This is represented as solid and dash edges, respectively.

3.6.2 Simulation results

We now provide some simulation results. For the sake of simplicity, we only consider coding over $\mathbb{F}_2$ with BPSK. The channel parameters are set to be $h_1 = h_2 = 1$ and $D_{max} = 5$. We again construct a IDT-QC LDPC code from the AR4JA ensemble with $N = 4096$, $b = 32$, $K = 3072$, and the design rate $R_d = 0.75$. The actual rate

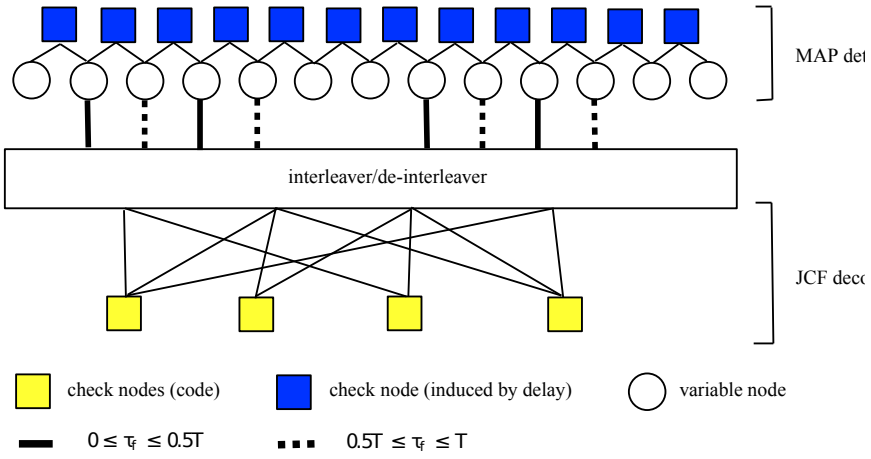


Figure 3.13: Graph representation of iterative receiver

is then $R_a = 0.685$. We would like to recall that for a same set of D_{max} and b, the longer the code the smaller the rate loss. Also, we construct the same cyclic LDPC code as in Section 3.4 with design rate $R_d = 0.66$. Note that for using cyclic codes for asynchronous compute-and-forward, one has to freeze SD_{max} bits. Thus, the actual rate of the cyclic LDPC code is $R_a = 0.658$. The decoding algorithm for the both codes is the joint MAP detection and JCF decoding described above with 40 outer iterations and 5 inner iterations.

In Fig. 3.14, BER versus SNR curve is plotted. One can observe that despite of having a higher rate, the proposed IDT-QC LDPC code outperforms the cyclic LDPC code by roughly 1.1 dB when $\tau = 0$, i.e., under perfect synchronization. This is the coding gain offered by the AR4JA code over the cyclic code adopted. When $\tau = 0.5$, the proposed IDT-QC LDPC code provides roughly 1.5 dB gain over the cyclic-LDPC considered. This enlarged gap may be explained by the observation that the joint graph of the zigzag detection and the parity check of a cyclic code is more likely to create short cycles compared to QC LDPC codes.

In Fig. 3.15, BER versus delay curve is plotted for SNR= 3.5 dB. One observes

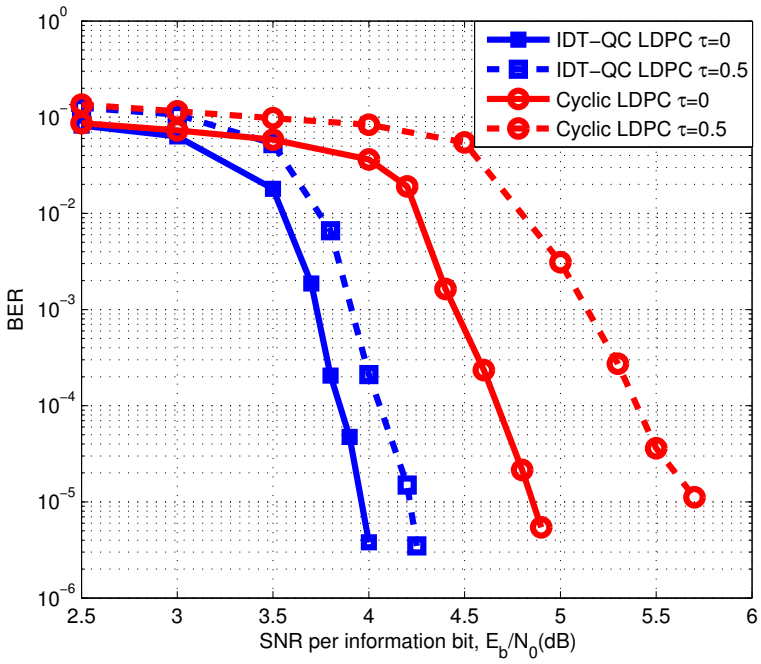


Figure 3.14: BER performance of the proposed IDT-QC LDPC code with $R_a = 0.685$ and the cyclic LDPC code with $R = 0.658$ for compute-and-forward.

that in the region $\tau \in [0, T]$, the proposed IDT-QC LDPC code always performs better than the cyclic LDPC code in terms of BER. Moreover, we observe a symmetric behavior of BER about $\tau = 0.5$. This is a consequence of allowing decoding to asynchronous functions so that the performance would only depend on how close the delay τ is to an integer. According to this observation, one expects a periodic behavior for other $[kT, (k+1)T]$ within $[0, D_{max}]$. Another interesting observation is that there is a local minimum at around $\tau = 0.5$. This may be explained by the observation that at around $\tau = 0.5$, two codewords are well separated and hence the zigzag detection and JCF would perform like a decode-and-forward decoder.

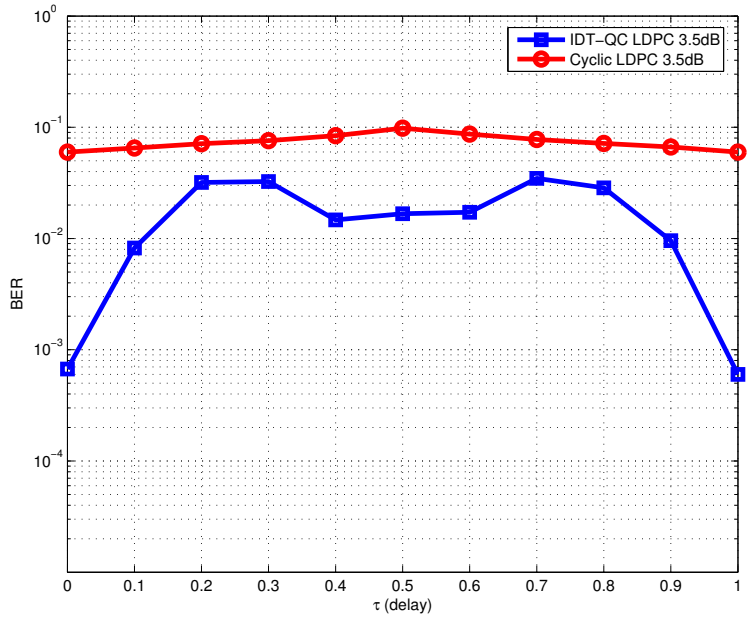


Figure 3.15: BER versus τ for the proposed IDT-QC LDPC code with $R = 0.685$ and the cyclic LDPC code with $R = 0.658$ for compute-and-forward.

3.7 Concluding Remarks and Future Work

The problem of communication in the presence of time delays that cannot be easily compensated, such as what we encounter in asynchronous physical layer network coding, has been studied. There are three main results in the section. Theorem 2 establishes that fundamentally there is no loss in the information rates achievable in the presence of timing delays which are integer multiples of symbol duration in comparison to the synchronous case. This is the first result to show that integer-valued asynchronism does not cause any reduction in the achievable rates. This result is obtained through the use of a novel framework called the interleave-deinterleave (IDT) transform in conjunction with quasi-cyclic codes. Secondly, in Section 3.5.3, we have shown that delays from the channel can be exploited to decode an increased set of

functions at the relays, thereby obtaining higher rates than in the synchronous case in some scenarios. Finally, an achievable rate in the presence of non-integer valued delays is given in Theorem 13. The rates achievable are higher than those reported earlier in the literature in [24].

As applications, the IDT-QC codes have been implemented as coding schemes for two channel models, namely the integer-forcing equalized ISI channel and the asynchronous compute-and-forward relay network. For the former, the proposed IDT-QC framework was able to achieve the upper bound of the information rate. For the latter, it has been shown that the proposed IDT-QC scheme not only provides significantly higher rates than the state of the art but also allows the exploitation of the delay dimension which may lead one to rates beyond those achieved by tightly synchronous compute-and-forward. Moreover, practical implementation of the proposed scheme has also been considered where a joint detection and decoding scheme was considered. Simulation results for the two transmitters and one receiver case have further confirmed the theoretical analysis and observations.

Interesting future work includes the following. Theoretically, it is of interest to see how one can further benefit from the delay dimension introduced by the channel. For example, for the symbol-level asynchronous compute-and-forward, it could be the case that some relays are unable to compute any function with a rate above the threshold but some relays are able to compute more than one functions (possibly have the same coefficients but different delays) with rates above the threshold. If the second phase bandwidth is not an issue, the central destination can pick S functions with the highest computation rates, regardless of where the functions are computed. This may lead to higher computation rates than that provided by tightly synchronous compute-and-forward. Practically, it is interesting to design spatially-coupled QC LDPC codes which can further bridge the gap to those theoretical results.

Moreover, it is of interest to investigate the impact of using a practical pulse shaping function.

4. COMPUTE-AND-FORWARD IN BLOCK-FADING CHANNELS*

In this section*, we study the problem of compute-and-forward in block-fading channels. Over the quasi-static channel, the compute-and-forward paradigm, which exploits the linearity of lattice codes and performs function computation according to the channel coefficients at relays, has shown promising gains. However, when channel exhibits time selectivity, it brings challenges to compute-and-forward since the linearity of lattice codes does not suit the time varying nature of the channel. In this section, we propose a new family of lattice codes called root-LDA lattice codes, to effectively exploit the diversity provided by the block-fading channels. It is shown that the proposed scheme under message-passing decoding can provide full-diversity gains over block-fading channels while incurring almost no loss to the coding gains as compared to existing codes over quasi-static channels. To further illustrate the benefit of the proposed scheme, the proposed compute-and-forward with root-LDA lattice codes and amplify-and-forward with root-LDA lattice codes are compared in multi-hop line networks with block fading. Simulation results show that amplify-and-forward suffers from not only the noise accumulation but also higher probability of uncorrectable deep-fading event (deep fadings occur at multiple sub-blocks) at the end node. On the other hand, the proposed strategy is free from the both and hence provides a much better error performance.

4.1 Introduction

Ever since the concept of physical-layer network coding (PLNC) was introduced [2], it has drawn great attention as one of the relaying strategies to boost the through-

*Reprinted with permission from P. C. Wang, Y. C. Huang, K. R. Narayanan and J. J. Boutros, "Physical-layer network-coding over block fading channels with root-LDA lattice codes," 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, 2016, pp. 1-6.

put in wireless relay networks. The idea is to employ structured codes at source nodes and then allow function computation in the physical layer at the relay. The functions are chosen according to the channel's and codes' structures. For example, in [2] and [63], the codes adopted at the source nodes are identical linear/lattice codes and the channel coefficients are set to 1; hence, the computed function is the modulo-sum of the codewords. Later in [12] where the name compute-and-forward (CF) first appeared, Nazer and Gastpar study this problem for a more general relay network in quasi-static fading channels in which the codes are still identical lattice codes but the channel coefficients are random. In this case, the computed functions become modulo version of linear integer combinations of codewords which correspond to linear combinations of messages. They then allow the relays to adaptively choose functions according to the channel coefficients for maximizing the performance.

The compute-and-forward paradigm described above heavily relies on some ideal assumptions, otherwise the nice algebraic structure of lattice codes may be ruined and may no longer be easily exploited. Among many of these assumptions, two of the most crucial ones are the perfect timing synchronization and the quasi-static channel assumption. The former basically assumes that signals coming from all the source nodes arrive at a relay simultaneously; without which it will cause misalignment of lattice codewords and hence the linearity is ruined. The latter assumes that channel coefficients are fixed within one codeword duration. Again, without this assumption, the linearity no longer preserves and compute-and-forward is undermined. In many practical communication systems, these assumptions are often too good to be true and solutions are called for in order to again making use the structural gains offered by the channel. For the problem of CF with timing-asynchronous users, interesting readers can refer to Chapter 3 and the reference therein.

In this section, we tackle the other problem, namely the quasi-static channel

assumption. This problem is far less discussed than the synchronization issue. In [26], Bakoury *et al.* investigate the impact of block fading for integer-forcing receiver, an application of CF to the multiple-input multiple output (MIMO) channel. Achievable rates based on CF with nested lattice codes of [27] and two decoders specifically designed for this scenario are derived. However, no optimality is shown. Moreover, from what we have learned in wireless communication [28], in the finite length regime, diversity is more important than achievable rates in block-fading channels. Therefore, in this section, we turn our focus to design finite-length codes that can exploit full diversity offered by the channel while computing functions.

For the point-to-point communication counterpart, Boutros *et al.* propose the family of root-LDPC codes which can attain full diversity under iterative decoding [64]. The idea is to recover the information bits under deep fading with the help of the bits that are not in deep fading, and this can be done by introducing rootchecks in LDPC codes. On the other hand, one can use Construction A [44] together with LDPC codes to construct LDA lattices [40] [41] [65]. This ensemble of lattices is shown to achieve the capacity of AWGN channel under lattice decoding [41]. Combining the above two ideas, we first construct root-LDA lattices via Construction A with root-LDPC codes and show that nested lattice codes from root-LDA lattices can achieve full diversity gain for the point-to-point communication in block-fading channels.

We then adopt the proposed root-LDA lattice codes to CF in block-fading channels. It is previously unclear whether fixing the integer coefficients throughout the entire codeword duration still makes sense for time-varying channels. We answer the question in affirmative by fixing one function and viewing the event corresponds to large self-interference (approximation errors incurred by the difference between a version of channel coefficients and integer coefficients) as a deep-fading event. Then,

it is clear that the proposed root-LDA lattice codes can take care of the cases where not all the blocks are in deep fade and thus can provide full diversity gains. Furthermore, the performance comparison between CF and amplify-and-forward (AF) [5] [66] in block-fading channels is studied. Despite its low complexity, AF suffers from noise accumulation because noise is progressively added up and this will eventually become a serious issue as the number of hops increases. Interestingly, in this work, we show by simulation that CF outperforms AF in terms of codeword error rate in the line network not only because noise accumulation but also deep-fading events accumulation, which basically reflects that the impact of channel coefficients in different hops are all coupled together.

The rest of the section is organized as follows. Section 4.2 describes the general system model. Section 4.3 introduces the proposed root-LDA lattice codes, and the error rate performance of these codes in point-to-point channel is investigated. In section 4.4, we study compute-and-forward with root-LDA lattice codes in two-way relay networks. Section 4.5 extends the study to the line network. Finally, section 4.6 concludes the section.

4.2 Problem Statement

Consider a line network in Fig. 4.1 with L nodes, $S_1, S_2, \ldots, S_L$, $L \geq 2$. Source nodes S_1 and S_L wish to exchange information via a series of relay nodes $S_2, S_3, \ldots, S_{L-1}$. Each node is equipped with single antenna and operates in half-duplex mode. Each node can only communicate with its neighbors, i.e., S_1 can only communicate with S_2, S_L can only communicate with S_{L-1}, and S_i can only communicate with S_{i-1} and S_{i+1}, $i = 2, 3, \ldots, L-1$. We arrange these nodes into two sets, $\mathcal{A}_o = \{S_i | i \text{ is odd}\}$, and $\mathcal{A}_e = \{S_i | i \text{ is even}\}$. By taking advantage of the nature of wireless channel, the transmission policy is as follows.

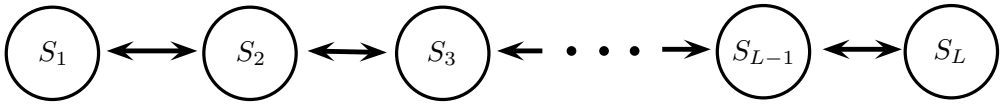


Figure 4.1: A line network with L nodes

- Nodes in $\mathcal{A}_o$ are allowed to transmit simultaneously in phase t, t is odd.
- Nodes in $\mathcal{A}_e$ are allowed to transmit simultaneously in phase t, t is even.

For ease of exposition, we consider the case where S_1 and S_L both have only one information sequence needs to be transmitted. Also we assume number of nodes L is odd, however it can be easily generalized to any L. Initially, source nodes S_1 and S_L encode information sequences $\mathbf{u}_1 \in \mathbb{F}_p^K$ and $\mathbf{u}_L \in \mathbb{F}_p^K$ to codewords $\mathbf{x}_1 \in \mathbb{R}^N$ and $\mathbf{x}_L \in \mathbb{R}^N$, respectively. The transmitted codeword $\mathbf{x}_i$ is subject to power constraint given by

$$\frac{1}{N}\|\mathbf{x}_i\|^2 = \frac{1}{N}\sum_{n=1}^{N}|x_i[n]|^2 \leq P,\ i = 1, \ldots, L. \tag{4.1}$$

S_1 and S_L transmit codewords $\mathbf{x}_{1,1}$ and $\mathbf{x}_{L,1}$ to S_2 and S_{L-1} in the first phase, respectively. Here $\mathbf{x}_{i,t}$ denotes the codeword transmitted by S_i in phase t. Then relay S_2 and S_{L-1} receive noisy codewords from S_1 and S_L in the first phase, respectively. Suppose they decode the received codewords correctly, S_2 and S_{L-1} then transmit codewords $\mathbf{x}_{2,2} = \mathbf{x}_{1,1}$ and $\mathbf{x}_{L-1,2} = \mathbf{x}_{L,1}$ to S_3 and S_{L-2} in the second phase, respectively. We keep doing this and eventually, two codewords will arrive at the relay S_M ($M = \frac{L+1}{2}$) in phase $t_M = \frac{L-1}{2}$. The relay node S_M computes a function $f(\mathbf{x}_{M-1,t_M}, \mathbf{x}_{M+1,t_M})$. In the subsequent phase, S_M broadcasts $\mathbf{x}_{M,t_M+1} = f(\mathbf{x}_{M-1,t_M}, \mathbf{x}_{M+1,t_M})$ to S_{M-1} and S_{M+1}. This function is then propagated in opposite directions, one towards S_1, and another towards S_L. Here we assume source nodes S_1 and S_L have perfect knowledge of the functions f. S_L (S_1)

then tries to recover the codeword $\mathbf{x}_{M-1,t_M} = \mathbf{x}_1$ ($\mathbf{x}_{M+1,t_M} = \mathbf{x}_L$) based on the knowledge of f, and $\mathbf{x}_{M+1,t_M} = \mathbf{x}_L$ ($\mathbf{x}_{M-1,t_M} = \mathbf{x}_1$), which is originated from itself. The total phases needed for information exchange between S_1 and S_L is $L-1$.

In general, both source nodes typically have multiple information sequences instead of merely one information sequence needed to be transmitted. Therefore one should expect to see multiple codewords from the same source node appear in the network which leads to multiple times of codeword collisions. As a result, more function computations and self-interference cancellations are involved. This model will be discussed in Section 4.5. The main point here is to illustrate that in such the network, function computation is common and inevitable.

In this work, unlike most of the work in the literature, the channel model for each hop is the block-fading channel, i.e., the channel coherence time (ℓ symbols) is smaller than the codeword duration (number of channel uses) N. We assume that channel gains are fixed in ℓ symbol duration (one sub-block), N is divisible by ℓ, and $b = N/\ell$ is number of sub-blocks experienced by each codeword. Let $h_{i,i+1}^{(j)} \in \mathbb{R}$ ($h_{i+1,i}^{(j)} \in \mathbb{R}$) be the channel gain experienced by the signal from node S_i (S_{i+1}) to S_{i+1} (S_i) in the j-th sub-block. All the channel gains are i.i.d., in addition, channel reciprocality is assumed throughout this work, i.e., $h_{i,i+1}^{(j)} = h_{i+1,i}^{(j)}$. The collection of channel gains from node S_i to S_{i+1} is given by $\mathbf{h}_{i,i+1} = [h_{i,i+1}^{(1)}, \ldots, h_{i,i+1}^{(b)}]$. We denote by $\mathbf{h}_i^{(j)} = [h_{i-1,i}^{(j)} h_{i+1,i}^{(j)}]$ as the collection of channel gains seen by relay node S_i in the j-th sub-block.

The received signal at S_i in phase t is then given by

$$\begin{aligned} y_{i,t}[n] &= h_{i-1,i}^{(j)} x_{i-1,t}[n] + h_{i+1,i}^{(j)} x_{i+1,t}[n] + z_{i,t}[n] \\ n &= 1, \ldots, N, \quad j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor, \end{aligned} \tag{4.2}$$

where $z_{i,t}[n] \sim \mathcal{N}(0,1)$ is white Gaussian noise seen at node S_i in phase t. For some cases a node only receives signal from one neighbor, we can set signal $\mathbf{x}_{i-1}$ or $\mathbf{x}_{i+1}$ to all-zero sequence. The node S_i is assumed to have the channel state information (CSI) of $\mathbf{h}_i^{(1)}, \ldots, \mathbf{h}_i^{(b)}$. The transmitters do not have CSI but have the knowledge of channel coherence time, i.e., it knows how long a sub-block lasts. The end-to-end codeword error rate (or frame error rate) is defined as

$$P_e \triangleq \mathbb{P}\left\{\hat{\mathbf{u}}_1 \neq \mathbf{u}_1 \cup \hat{\mathbf{u}}_L \neq \mathbf{u}_L\right\}, \tag{4.3}$$

where $\hat{\mathbf{u}}_1$ is an estimate of $\mathbf{u}_1$ made at node S_L and $\hat{\mathbf{u}}_L$ is an estimate of $\mathbf{u}_L$ made at node S_1. In our simulations, for simplicity, we only consider frame error rate at S_L, which is given as

$$\tilde{P}_e \triangleq \mathbb{P}\left\{\hat{\mathbf{u}}_1 \neq \mathbf{u}_1\right\}. \tag{4.4}$$

Nevertheless, eq. (4.3)(4.4) will exhibit similar behavior.

For the rest of the sections, we will use the line networks introduced here with different numbers of nodes to demonstrate the proposed scheme and its properties. Specifically, in Section 4.3, the network with $L = 2$ nodes is studied. This reduces to the point-to-point communication problem since no relay node is deployed. By studying point-to point problem, it helps us constructing lattice codes for block-fading channels. In Section 4.4, we study the network consists of $L = 3$ nodes, which is a fundamental building block of large networks and is also known as the two-way relay network (TWRN). The reason for studying the TWRN is two-fold: (1) it is the simplest case which involves relaying. (2) it helps us understanding the unique structure of the line network in Fig. 4.1; any scheme for TWRN can be easily extended to the general line network. In section 4.5, we study the line network with

$L = 5$ nodes which serves as an example with multiple hops and allows us to see the benefits of the proposed scheme against AF.

4.3 Proposed Root-LDA Lattice Codes

In this section, we introduced the proposed root-LDA lattice ensemble and examine error performance of these codes in point-to-point channel. Point-to-point channel can be considered as a special case of the line network in Fig. 4.1 with $L = 2$ nodes. To avoid heavy notation, we drop as many subscripts as possible. The problem becomes that S_1 encodes information sequence $\mathbf{u}_1$ into $\mathbf{x}_1$, which is sent to S_2. The received signal at S_2 is given by

$$\begin{aligned} y[n] &= h^{(j)}x[n] + z[n], \\ n &= 1, \ldots, N, \quad j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor . \end{aligned} \tag{4.5}$$

In what follows, we generalize the binary root-LDPC codes proposed by Boutros *et. al* [64] to non-binary (NB) case, which lead to NB root-LDPC codes. The NB root-LDPC codes are then used in conjunction with Construction A to obtain the proposed root-LDA lattices. As we focus on power-constrained communication systems, shaping is needed to carve out a finite subset from a lattice (a lattice is an infinite constellation). Simulation results show that as long as the shaping process is done carefully, the proposed root-LDA lattice codes exhibit full diversity order in block-fading point-to-point channel.

4.3.1 Non-binary root-LDPC codes

Mathematically, a code is said to provide b diversity order if its frame error rate (FER) decreases as $1/\text{SNR}^b$ when SNR is sufficiently large. Binary root-LDPC codes with rate $R = \frac{K}{N} \leq \frac{1}{b}$ have been reported to achieve full-diversity order in point-to-

point block-fading channel (with b sub-blocks) under message-passing decoding [64]. For the sake of simplicity, consider point-to-point block-fading channel with number of sub-blocks $b = 2$. The root-LDPC codes introduce the notion of rootchecks, which is a check node with restricted connectivity. The rule is that every information bit $m[n]$ participates in at least one rootcheck such that all the other bits participate in this rootcheck experience different fade than $m[n]$. Let us use a binary (3,6) root-LDPC code for example to illustrate this idea. The Tanner graph associated with the parity check matrix of this code is shown in Fig. 4.2. The check nodes in the Tanner graph should be designed following these two criterions:

1. There are 2 types of rootchecks denoted as CHK_1 and CHK_2, the number of rootchecks for each type is $N/4$.

2. Type 1(2) rootcheck is connected with 5 variable nodes experiencing fading $h^{(2)}$ ($h^{(1)}$) and 1 information variable node experiencing fading $h^{(1)}$ ($h^{(2)}$).

For CHK_1 in Fig.4.2, if $h^{(1)}$ is deep fade while $h^{(2)}$ is not, then white node can be recovered by passing information from all black node through CHK_1. Similarly, another case is that $h^{(2)}$ is deep fade while $h^{(1)}$ is not, then white node can recover the information by itself. The same reasoning holds for black node which is participated in CHK_2. Once the structure of Tanner graph is established, immediately the parity-check matrix is known which can be written as

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{H}_{2i} & \mathbf{H}_{2p} \\ \mathbf{H}_{1i} & \mathbf{H}_{1p} & \mathbf{I} & \mathbf{0} \end{bmatrix}, \tag{4.6}$$

where all sub-arrays have dimension $N/4 \times N/4$. $\mathbf{I}$ is identity matrix and $\mathbf{0}$ is all-zero matrix. The rest of sub-arrays are random matrices subject to row-column weight constraint of $\mathbf{H}$. Since we aim to construct (3,6) code, the column weights of $\mathbf{H}_{1i}$ and

$\mathbf{H}_{2i}$ are 2. The column weights of $\mathbf{H}_{1p}$ and $\mathbf{H}_{2p}$ are 3. The row weights of $[\mathbf{H}_{1i} \quad \mathbf{H}_{1p}]$ and $[\mathbf{H}_{2i} \quad \mathbf{H}_{2p}]$ are both 5. These sub-matrices are optimized by progressive edge-growth (PEG) algorithm [67] to ensure no small cycles in the bipartite graph. Column 1 through $N/4$ and $(N/2)+1$ through $(3/4)N$ are corresponding to information bits. The first $N/2$ rows are corresponding to CHK_1 and the rest of the rows can be regarded as CHK_2. We refer codes constructed in this fashion as root-LDPC (4π) codes [68], where 4π denotes that there are four permutation sub-matrices besides $\mathbf{I}$ and $\mathbf{0}$ in $\mathbf{H}$.

It's worth mentioning that the root-LDPC parity-check matrix can be designed carefully with code doping [68]. The idea is to let a fraction θ of parity bits enjoy full-diversity after two or more decoding iterations. This technique has been shown to provide higher coding gains than that provided by the original root-LDPC codes. To design the H with code doping, the sub-matrices $\mathbf{H}_{1p}$ and $\mathbf{H}_{2p}$ are given by

$$\mathbf{H}_{jp} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{H}_{jp,1} & \mathbf{H}_{jp,2} \end{bmatrix}, j = 1, 2, \tag{4.7}$$

where $\mathbf{I}$ is $N/8 \times N/8$ identity matrix, in addition $\mathbf{H}_{jp,1}$ and $\mathbf{H}_{jp,2}$ are $N/8 \times N/8$ matrices with appropriate weight distributions such that the weight distribution of $\mathbf{H}$ is satisfied. The columns of $\mathbf{H}$ (eq. (4.6)) that first $N/8$ columns of $\mathbf{H}_{jp}$ reside in correspond to the additional parity bits that enjoy full-diversity. We refer to codes constructed in this fashion as root-LDPC (2π) codes. The name 2π comes from the fact $\mathbf{H}_{jp}, j = 1, 2$ are not permutation sub-matrices anymore which only two permutation sub-matrices remain in $\mathbf{H}$. In order to construct NB root-LDPC codes, the corresponding parity-check matrices can be constructed by replacing the ones in $\mathbf{H}$ with non-zero elements in $\mathbb{F}_p$

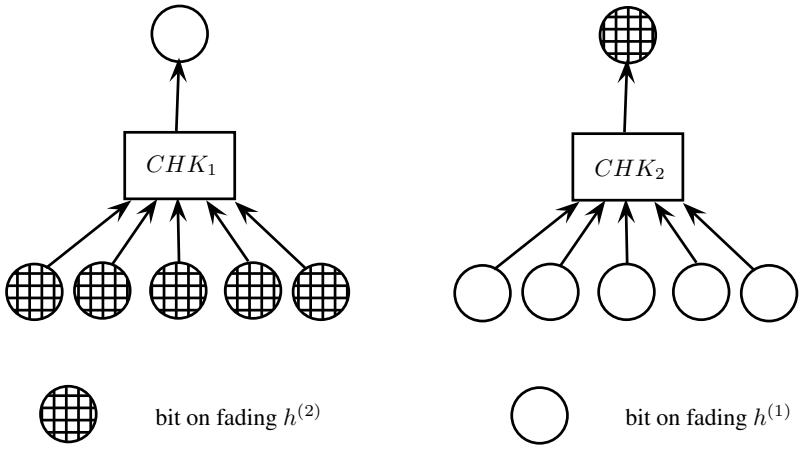


Figure 4.2: Rootchecks of (3,6) root-LDPC code

4.3.2 Root-LDA lattice codes

An LDA lattice is constructed via construction A [44] with underlying code being an NB LDPC code [40]. We now summarize the construction of LDA lattices. Let $\mathcal{C}$ be an N dimensional NB LDPC code from some finite field $\mathbb{F}_p$, also let $\mathcal{M} : \mathbb{F}_p \to \mathbb{R}$ be the natural mapping given by

$$\mathcal{M}(u) \triangleq \begin{cases} u, & 0 \leq u \leq \frac{p-1}{2}, \\ u-p, & \frac{p-1}{2} < u < p, \end{cases} \tag{4.8}$$

when $p \geq 3$, and $\mathcal{M}(u) \triangleq u - \frac{1}{2}$ for $p = 2$ (i.e., BPSK). An LDA lattice is obtained by tiling $\mathcal{M}(\mathcal{C})$ to the entire $\mathbb{R}^N$ as

$$\Lambda \triangleq \mathcal{M}(\mathcal{C}) + p\mathbb{Z}^N, \tag{4.9}$$

where the sum in the equation is the Minkowski sum. The fact that Λ is indeed a lattice can be easily shown by noticing that $\mathcal{M}$ is an isomorphism and $\mathcal{C}$ is a linear

code. We are interested in a subclass of LDA lattices, called root-LDA lattices, where the underlying codes are restricted to NB root-LDPC codes. According to different types of NB root-LDPC codes, we further categorize the root-LDA lattices discussed in this work into two subclasses as follows. The first one is the root-LDA (4π) lattice which adopts the NB root-LDPC (4π) code and the second one is root-LDA (2π) lattice which uses the NB root-LDPC (2π) code.

So far we have discussed the construction of the proposed root-LDA lattices. A lattice consists of infinite points in $\mathbb{R}^N$. One popular approach of using lattices for power-constrained problems is to carve a finite subset from the lattices via shaping techniques. Conventionally, spherical shaping is adopted where the codebook is the intersection of a sphere and the lattice [42]. Spherical shaping is power-efficient and leads to codebooks mimicking the capacity-achieving input distribution for the AWGN channel. However, the nice correspondence between lattice points and underlying linear codewords may no longer be preserved. Another popular approach for shaping is to the nested lattice shaping [27]. Let (Λ_f, Λ_c) be a pair of nested lattices such that $\Lambda_c \subseteq \Lambda_f$, where Λ_f is the fine lattice and Λ_c is the coarse lattice. The lattice code obtained by nested lattice shaping, henceforth referred to as nested lattice code, is $\mathcal{C} = \Lambda_f \cap \mathcal{V}(\Lambda_c)$ where $\mathcal{V}(\Lambda_c)$ denotes the fundamental Voronoi region of coarse lattice Λ_c. Under some goodness conditions, the resulting nested lattice code can achieve the capacity of AWGN channel with lattice decoding. Since nested lattice codes preserve most of the structures which facilitate function computation, they are adopted in most of the CF literature.

Since our goal is to enable CF for networks with block-fading channels, we employ nested lattice shaping. Here, we reveal another crucial difference between spherical shaping and nested lattice shaping for using lattice codes in block-fading channels. That is, it may not be an easy task for lattice codes with spherical shaping to enjoy

full diversity order even though the underlying linear codes can do so. This is simply because the part from $p\mathbb{Z}^N$ is not protected from any code; thus, any lattice code containing more than one elements in the same coset $\mathbf{c}+p\mathbb{Z}^N$ for some $\mathbf{c}\in\mathcal{C}$ cannot enjoy full diversity order. And it is not easy for spherical shaping to prevent this from happening. On the contrary, for nested lattice shaping, one can easily avoid the above problem by choosing Λ_c such that $p\mathbb{Z}^N\subseteq\Lambda_c$. It is worth emphasizing that this choice includes the famous and low-complex hypercube shaping $\Lambda_c=p\mathbb{Z}^N$.

4.3.3 Lattice decoder for root-LDA lattice codes

In this section, we discuss the implementation of lattice decoder based on message-passing algorithm [40], which can be used to efficiently decode the proposed root-LDA lattice codes. The idea is to rely on the low-complexity belief-propagation decoder for the underlying non-binary code to decode high-dimensional lattice points, otherwise directly decode the lattice points is computationally prohibitive. Although we are discussing lattice decoding, it can be easily seen that lattice code decoding is a special case of lattice decoding where the decoding space is finite.

Let

$$\mathbf{x}=\mathcal{M}(\mathbf{c})+p\mathbf{k} \tag{4.10}$$

be the transmitted lattice point where $\mathbf{x}\in\Lambda$ and the corresponding received signal is (4.5). The lattice decoding based on message-passing algorithm is as follows:

Initialization: The first step is to compute the *a posteriori* probabilities (APPs) of $\mathbf{c}$ element-wise,

$$\mathbb{P}(c[n]=v|y[n])=\sum_{\chi\in\Gamma_v}\mathbb{P}(h^{(j)}\chi|y[n]),\quad v=1,\ldots,p\quad n=1,\ldots,N \tag{4.11}$$

where $\Gamma_v = \{\chi | \chi = \mathcal{M}(v) + p\zeta, \zeta \in \mathbb{Z}\}$ and

$$\mathbb{P}(h^{(j)}\chi | y[n]) \propto \exp(-\frac{(y[n] - h^{(j)}\chi)^2}{2\sigma^2}), \tag{4.12}$$

Note that there are infinite number of elements in Γ_v, however one can truncate Γ_v to a finite set for approximation. In practice, picking the 3 closest elements to $y[n]$ is sufficient.

Message passing: The probability vectors are written as

$$\tilde{\mathbf{p}}_n = [\mathbb{P}(c[n] = 0 | y[n]), \ldots, \mathbb{P}(c[n] = p | y[n])] \quad n = 1, \ldots, N. \tag{4.13}$$

$\tilde{\mathbf{p}}_n$ is fed into the n-th variable node of underlying NB LDPC code as initial value. These values are then passed to the check nodes. After check node update, the refined probability vectors are passed back to variable nodes, which completes one iteration. Each variable node jointly processes the incoming probabilities and initial channel value and then passes them to check nodes in the second iteration. The iterative process terminates either when the probabilities converge or the number of iterations reaches a predefined value.

Hard decision: The final decision rule on $\hat{\mathbf{c}} = [\hat{c}[1], \ldots, \hat{c}[N]]$ is

$$\hat{c}[n] = \underset{v}{\operatorname{argmax}}\ \mathbb{P}^{(m)}(c[n] = v | \mathcal{C}, \mathbf{y} \backslash y[n])) \mathbb{P}(c[n] = v | y[n]). \tag{4.14}$$

After the hard decision on $\hat{\mathbf{c}}$, the decoder then looks into the coset $\mathcal{M}(\hat{\mathbf{c}}) + p\mathbb{Z}^N$ and the received signal $\mathbf{y}$ is quantized to the nearest point inside this coset to form the estimate $\hat{\mathbf{x}}$.

4.3.4 Simulation results

We construct three LDA lattice codes and investigate their FER performance in point-to-point channels. The first one is a root-LDA (4π) lattice code where the underlying code is a root-LDPC (4π) code. The second one is a root-LDA (2π) lattice code where the underlying code is root-LDPC (2π) code. The last one is LDA lattice code where the underlying code is LDPC code optimized by PEG algorithm [67]. All the LDPC codes are (3,6) regular non-binary codes constructed over $\mathbb{F}_5$. The block length is set to $N = 1200$ and information length is $K = 600$. Also, hyper-cubic shaping is considered for all lattice codes. The fading coefficients are i.i.d., drawn from Rayleigh distribution. We set the maximum decoding iteration to 100 for the message-passing decoder. To have a sense about how good these schemes are, we also provide the information outage probability [64] defined as

$$P_{out}(\gamma, R) = Pr\{\mathcal{I}(\gamma, \mathbf{h}) < R\}, \tag{4.15}$$

where γ is signal-to-noise ratio and $\mathbf{h} = [h^{(1)} \dots h^{(b)}]$, $\mathcal{I}(\gamma, \mathbf{h})$ is the instantaneous input-output mutual information between the input and output of the channel given

$$\mathcal{I}(\gamma, \mathbf{h}) = \frac{1}{b} \sum_{i=1}^{b} I_{\text{AWGN},p}(\gamma {h^{(j)}}^2), \tag{4.16}$$

$I_{\text{AWGN},p}(s)$ is the input-output mutual information of an AWGN channel with SNR s and input as $\mathbb{Z}$ hypercubically shaped by $p\mathbb{Z}^N$ [69].

We first investigate the FER performance in quasi-static fading ($b = 1$) channel which is shown in Fig. 4.3. It is observed that both root-LDA lattice codes exhibit similar performance comparing with LDA lattice code; thus, we argue that there is no fundamental loss in coding gain by designing lattice codes specifically for block-fading

channels. In Fig. 4.4, we then compare the FER performance of these codes in block-fading channel ($b = 2$). We observe that the LDA-lattice code is unable to achieve full-diversity since it is not optimized for block-fading channel. In contrast, both the root-LDA (4π) and the root-LDA (2π) lattice codes achieve full-diversity. They outperforms LDA-lattice code by 3dB and 8dB at FER=10^{-3} and 10^{-4}, respectively. In addition, root-LDA (2π) lattice code has additional coding gain as compared to root-LDA (4π) lattice codes [†], where the extra coding gain comes from the additional full-diversity parity symbols. In summary, the proposed root-LDA lattice codes enjoy full-diversity order while do not incur fundamental loss in coding gains for the AWGN channel. An interesting application of this scheme is a wireless network where the channel could be either quasi-static or block-fading, depending on the velocity of the user. We can apply the proposed scheme to this scenario and argue that the proposed coding scheme is robust against dynamic channel conditions.

4.4 Root-LDA Lattice Codes for Two-Way Relay Networks

In this section we study the network consists of $L = 3$ nodes, i.e., TWRN. Again the channel model we are interested in is block-fading channel. Information exchange in TWRN consists of 2 phases: In the first phase, S_1 transmits $\mathbf{x}_1$ and S_3 transmits $\mathbf{x}_3$ simultaneously. The relay S_2 receives $\mathbf{y}_2 = [y_2[1], \ldots, y_2[N]]$ where

$$\begin{aligned} y_2[n] &= h_{1,2}^{(j)} x_1[n] + h_{3,2}^{(j)} x_3[n] + z_2[n] \\ n &= 1, \ldots, N, \quad j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor . \end{aligned} \tag{4.17}$$

[†]In [70], which is the conference version of this work, only the root-LDA (4π) lattice codes were investigated. Here, we include the root-LDA (2π) lattice codes and will later show that lattice codes from this ensemble can provide better performance than that provided by those from the root-LDA (4π).

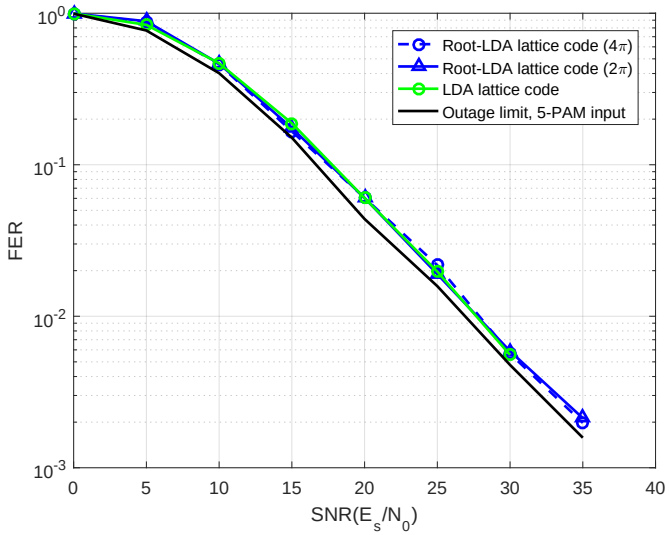


Figure 4.3: FER vs. SNR curves for root-LDA lattice codes in point-to-point quasi-static (b=1) fading channel

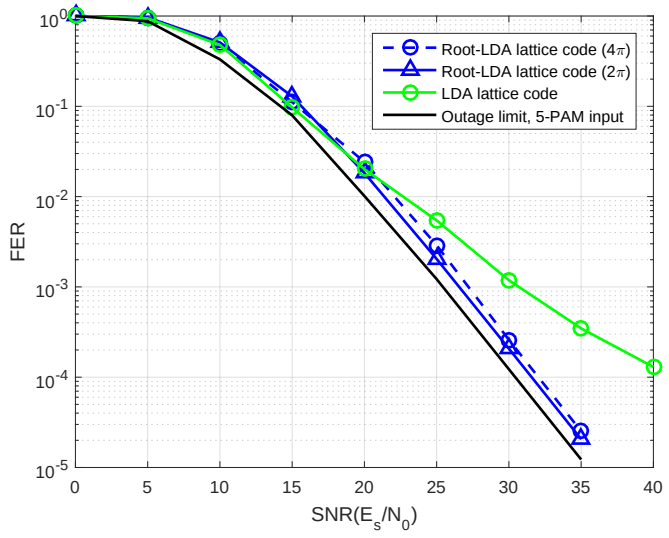


Figure 4.4: FER vs. SNR curves for root-LDA lattice codes in point-to-point block-fading channel (b=2)

Subsequently the relay performs necessary signal processing techniques on the received sequence $\mathbf{y}_2$ to form an estimate of $\mathbf{x}_2 = f(\mathbf{x}_1, \mathbf{x}_3)$, for which $\mathbf{x}_2$ is a function of $\mathbf{x}_1$ and $\mathbf{x}_3$. For CF, the function $f(\cdot)$ has to satisfy the following property: One can reconstruct $\mathbf{x}_3$ ($\mathbf{x}_1$) given the knowledge of $f(\cdot)$ and $\mathbf{x}_1$ ($\mathbf{x}_3$). In the second phase, relay S_2 broadcasts the estimate of $\mathbf{x}_2 = f(\mathbf{x}_1, \mathbf{x}_3)$ to S_1 and S_3. Since S_1 and S_3 hold the information that was sent by themselves, and together with the knowledge of $f(\cdot)$, they can extract the information sent from another source node. In this section, we focus on function computation at the relay while the second phase is just broadcast channel, therefore the subscript for the phase is dropped. In what follows, we discuss CF, separation-based CF, and AF for this model.

4.4.1 Compute-and-forward

Following [26], we enforce S_1 and S_3 employ an identical nested lattice code $\mathcal{C} = \Lambda_f \cap \mathcal{V}_{\Lambda_c}$. S_1 and S_3 transmit $\mathbf{x}_1 = (\mathbf{t}_1 - \mathbf{d_1}) \mod \Lambda_c$ and $\mathbf{x}_3 = (\mathbf{t}_3 - \mathbf{d}_3) \mod \Lambda_c$ respectively, where $\mathbf{t}_1, \mathbf{t}_3 \in \mathcal{C}$, $\mathbf{d_1}$ and $\mathbf{d_3}$ are dither vectors. After relay S_2 receives signals given by eq. (4.17), a geometric mean (GM) receiver chooses one integer vector $\mathbf{a}_2 = [a_{1,2} \quad a_{3,2}] \in \mathbb{Z}^2$, according to the channel vectors $\{\mathbf{h}_2^{(1)}, \mathbf{h}_2^{(2)}, \ldots, \mathbf{h}_2^{(b)}\}$. GM receiver then scales the signals in each sub-block individually, where j-th sub-block of $\mathbf{y}_2$ is scaled by $\alpha_2^{(j)}$, the scaled version of $\mathbf{y}_2$ is given by,

$$\tilde{y}_2[n] = \alpha_2^{(j)} y_2[n], n = 1, \ldots, N, j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor, \tag{4.18}$$

then adds dithers back to $\tilde{\mathbf{y}}_2$ results in what follows,

$$\begin{aligned} \bar{\mathbf{y}}_2 &= (\tilde{\mathbf{y}}_2 + a_{1,2}\mathbf{d}_1 + a_{3,2}\mathbf{d}_3) \mod \Lambda_c \\ &= (\mathbf{t}_{eq,2} + \mathbf{z}_{eq,2}) \mod \Lambda_c, \end{aligned} \tag{4.19}$$

where

$$\mathbf{t}_{eq,2} = (a_{1,2}\mathbf{t}_1 + a_{3,2}\mathbf{t}_3) \mod \Lambda_c, \tag{4.20}$$

and

$$\mathbf{z}_{eq,2} = \tilde{\mathbf{z}}_2 + \mathbf{w}_1 + \mathbf{w}_3, \tag{4.21}$$

which each element in $\tilde{\mathbf{z}}_2$ is given by

$$\tilde{z}_2[n] = \alpha_2^{(j)} z_2[n], \quad n = 1, \ldots, N, \quad j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor, \tag{4.22}$$

and each element in $\mathbf{w}_i$ is given by

$$w_i[n] = (\alpha_2^{(j)} h_2^{(j)} - a_{i,2}) x_i[n], \quad n = 1, \ldots, N, \quad j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor. \tag{4.23}$$

Since the linear combination of two lattice codewords is still a lattice codeword, relay S_2 can reliably decode function $\mathbf{t}_{eq,2} = f(\mathbf{t}_1, \mathbf{t}_2)$, with rate arbitrary close to the achievable computation rate [26]

$$\begin{aligned} \mathcal{R}(\mathbf{h}_2^{(1)}, \ldots, \mathbf{h}_2^{(b)}, \mathbf{a}_2) &= \frac{1}{b} \sum_{j=1}^{b} \frac{1}{2} \log^+ \left(\left(\|\mathbf{a}_2\|^2 - \frac{P |\mathbf{h}_2^{(j)^H} \mathbf{a}_2|^2}{1 + P\|\mathbf{h}_2^{(j)}\|^2} \right)^{-1} \right) \\ &= \frac{1}{b} \sum_{j=1}^{b} \frac{1}{2} \log^+ \left(\frac{P}{\alpha_2^{(j)} + P\|\alpha_2^{(j)} \mathbf{h}_2^{(j)} - \mathbf{a}_2\|^2} \right), \end{aligned} \tag{4.24}$$

where $\log^+(x) \triangleq \max(\log(x), 0)$. When the channel coefficients and integer vector are fixed, the corresponding achievable computation rate varies with the scaling factors $\alpha_2^{(j)}$. The purpose of scaling factor is to scale the received signals of j-th sub-block such that the channel coefficients are close to integer vector $\mathbf{a}_2$, which suppresses the second term of the denominator in (4.24). It is shown in [12] that the optimal choice

is the MMSE scaling factor given by

$$\alpha_{2,\mathrm{MMSE}}^{(j)} = \frac{P\mathbf{h}_2^{(j)^H}\mathbf{a}_2}{1+P\|\mathbf{h}_2^{(j)}\|^2}, j = 1, \ldots, b. \tag{4.25}$$

Now, we still need to decide on $\mathbf{a}_2$ such that the achievable computation rate is maximized given the channel coefficients. However it remains open to find polynomial-time algorithms which can find near-optimal integer vectors in block-fading channels. For the simulations in this work, we exhaustively search the integer vectors. Luckily, according to Lemma 1 in [12], we only need to search those integer vectors within $\{\mathbf{a}_2 \mid \exists j, \|\mathbf{a}_2\|^2 < 1 + \|\mathbf{h}_2^{(j)}\|^2 P\}$ because everything outside will result in zero achievable rate.

4.4.2 Separation-based compute-and-forward

In spite of the achievability result of compute-and-forward discussed in last subsection, we employ separation-based compute-and-forward [71] in this work. It is because CF adopts the sub-optimal lattice decoding, which greatly simplifies the analysis and therefore leads to a closed-form achievable rates. On the other hand, separation-based CF views the coding and modulation separately and adopts optimal maximum *a posteriori* probability (MAP) decoding. Moreover, if the underlying linear codes are sparse, one can use the low-complex message-passing algorithm to approximate the MAP decoding.

In a nutshell, separation-based CF consists of two parts:

1. Find MAP estimate of the coset (with message-passing algorithms).

2. Find the nearest point to the received signal in the coset.

We now describe the details of this decoder. Suppose S_1 transmits $\mathbf{x}_1 = \mathcal{M}(\mathbf{c}_1) + p\mathbf{k}_1$ and S_3 transmits $\mathbf{x}_3 = \mathcal{M}(\mathbf{c}_3) + p\mathbf{k}_3$, where $\mathbf{c}_1, \mathbf{c}_3 \in \mathcal{C}$ and $\mathcal{C}$ is a linear code over $\mathbb{F}_p$.

Here we ignore dithers for the sake of simplicity. At the relay S_2, the integer vector $\mathbf{a}_2 = [a_{1,2} \quad a_{3,2}]$ is chosen to maximize (4.24). Let $b_{1,2} = a_{1,2} \mod p$, $b_{1,2} \in \mathbb{F}_p$ and $b_{3,2} = a_{3,2} \mod p$, $b_{3,2} \in \mathbb{F}_p$. We aim to decode $\mathcal{M}(\mathbf{c}_f) + p\mathbb{Z}^N$ where $\mathbf{c}_f = b_{1,2}\mathbf{c}_1 \bigoplus b_{3,2}\mathbf{c}_3 \in \mathbb{F}_p^N$. The decoding algorithm mostly follows the algorithm in Section 4.3.3 but with some modifications described as follows. The relay first computes the joint APP of $c_1[n]$ and $c_3[n]$ given $y_2[n]$,

$$P_{APP}(c_1[n] = v_1, c_3[n] = v_2) = \sum_{\chi_1 \in \Gamma_{v_1}} \sum_{\chi_3 \in \Gamma_{v_3}} \mathbb{P}(h_{1,2}^{(j)}\chi_1 + h_{3,2}^{(j)}\chi_3 | y_2[n]), \tag{4.26}$$

where $\Gamma_v = \{\chi | \chi = \mathcal{M}(v) + p\zeta, \zeta \in \mathbb{Z}\}$, and

$$\mathbb{P}(h_{1,2}^{(j)}\chi_1 + h_{3,2}^{(j)}\chi_3 | y_2[n]) \propto \exp(-\frac{(y_2[n] - h_{1,2}^{(j)}\chi_1 - h_{3,2}^{(j)}\chi_3)^2}{2\sigma^2}),$$

the APP for $\mathbf{c}_f[n]$ is then given by

$$\mathbb{P}(c_f[n] = v | y_2[n]) = \sum_{b_{1,2}v_1 \oplus b_{3,2}v_3 = v} P_{APP}(v_1, v_3). \tag{4.27}$$

Subsequently, the decoder performs standard message-passing decoding as described in section 4.3.3. The output of the decoder is $\hat{\mathbf{c}}_f$ which determines the coset $\hat{\mathbf{x}}_f = \mathcal{M}(\hat{\mathbf{c}}_f) + p\mathbb{Z}^N$. The final step is to quantize $\mathbf{y}_2$ to the nearest point in this coset. Note that if $p\mathbb{Z}^N \subseteq \Lambda_c$, then there is no need to perform the last step since $\mathcal{M}(\mathbf{c}_f)$ is the only lattice codeword in the coset.

We also study the achievable computation rate of separation-based CF. The primary purpose is to understand what is the fundamental rate loss of using separation-based CF (with hyper-cubic shaping) as compared to CF. The idea is to convert the network problem into an equivalent point-to-point channel problem and then we can

focus on the mutual information of this equivalent point-to-point channel. Conceptually, (4.17) can also be interpreted as the output of a point-to-point AWGN channel with input being

$$x_f[n] = h_{1,2}^{(j)} x_1[n] + h_{3,2}^{(j)} x_3[n], \quad n = 1, \ldots, N, \quad j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor, \tag{4.28}$$

where $\mathbf{x}_f$ is obtained from $\mathbf{c}_f = b_{1,2}\mathbf{c}_1 \oplus b_{3,2}\mathbf{c}_3$ via mapping functions $\mathcal{M}_{\mathbf{h}_2^{(1)},\mathbf{a}_2}(\cdot), \ldots, \mathcal{M}_{\mathbf{h}_2^{(b)},\mathbf{a}_2}(\cdot)$, which $\mathcal{M}_{\mathbf{h}_2^{(j)},\mathbf{a}_2}$ is the mapping function for symbols in j-th sub-block given by,

$$\begin{aligned} \mathcal{M}_{\mathbf{h}_2^{(j)},\mathbf{a}_2}(c_f[n] = b_{1,2}c_1[n] \oplus b_{3,2}c_3[n]) = h_{1,2}^{(j)}\mathcal{M}(c_1[n]) + h_{3,2}^{(j)}\mathcal{M}(c_3[n]). \\ n = (j-1)b + 1, \ldots, (j-1)b + \ell. \end{aligned} \tag{4.29}$$

An example is provided as follows.

Example 3. *In Fig. 4.5-(a), S_1 and S_3 modulate symbols $c_1[n]$ and $c_3[n]$ (numbers in blue) in GF(3) onto 3-PAM constellation ($\{$-1,0,1$\}$) with natural mapping $\mathcal{M}$. In Fig. 4.5-(b), the transmitted signals of S_1 and S_3 are distorted by quasi-static fading channels with $h_{12} = 1$ and $h_{32} = 0.5$, respectively. The integer coefficients are chosen to be $a_{12} = 2$ and $a_{32} = 1$ which maximize* (4.24)*. The corresponding integer coefficients in GF(3) are $b_{12} = 2$ and $b_{32} = 1$. Then the mapping $\mathcal{M}_{\mathbf{h}_2^{(1)},\mathbf{a}_2}$ is illustrated in Fig. 4.5-(c), where the mapping is mapped from joint symbol in $\mathbb{F}_3^2$ to $\mathbb{R}$. Here only one mapping function is needed since we consider quasi-static fading channel. Take a received modulated symbol* 0.5 *as example, it could be mapped from 12 or 01, where 12 indicates $c_1[n] = 1$ and $c_3[n] = 2$, equivalently $c_f[n] = b_{1,2}c_1[n] \oplus b_{3,2}c_3[n] = 1$, and 01 indicates $c_1[n] = 0$ and $c_3[n] = 1$, equivalently $c_f[n] = b_{1,2}c_1[n] \oplus b_{3,2}c_3[n] = 1$. One can verify the other input-output mappings in* (4.29) *also holds given the mapping in Fig. 4.5-(c).*

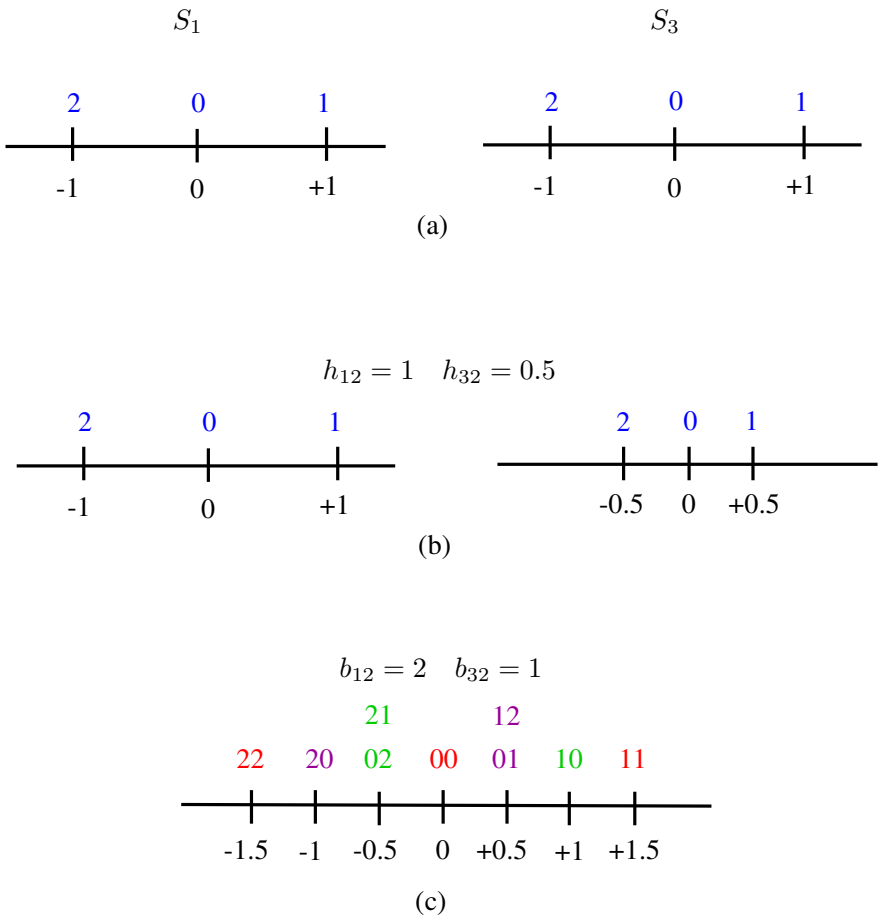


Figure 4.5: Example: separation-based compute-and-forward with 3-PAM

From the above example, the mapping falls into the class of PAM modulation. In block-fading channel of b sub-blocks, each sub-block is associated with an equivalent point-to-point AWGN channel and mapping function. Therefore the achievable computation rate of separation-based CF is the average mutual information rate of b equivalent channels given by

$$\mathcal{R}_s(\mathbf{h}_2^{(1)}, \ldots, \mathbf{h}_2^{(b)}, \mathbf{a}_2) = \frac{1}{b} \sum_{i=1}^{b} I_{\text{AWGN},eq}(\mathbf{h}_2^{(i)}, \mathbf{a}_2), \tag{4.30}$$

where $I_{\text{AWGN},eq}(\mathbf{h}^{(i)}, \mathbf{a}_2)$ is the mutual information rate of the equivalent point-to-point AWGN channel (with PAM input $\mathcal{M}_{\mathbf{h}_2^{(j)}, \mathbf{a}_2}(\cdot)$) for i-th sub-block.

4.4.3 Amplify-and-forward

In AF, decoding is only performed at the end nodes S_1 and S_3 and the relay node S_2 only scales the received signal to satisfy the power constraint before forwarding. Based on the received signal in (4.17), the relay scales the received signal as

$$x_2[n] = \beta_2^{(j)} y_2[n], \quad n = 1, \ldots, N, \quad j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor, \tag{4.31}$$

where

$$\beta_2^{(j)} = \sqrt{\frac{P}{|h_{1,2}^{(j)}|^2 P + |h_{3,2}^{(j)}|^2 P + 1}}. \tag{4.32}$$

It's worth mentioning that the scaling factor $\beta_2^{(j)}$ is for the interest of power control, whereas the scaling factor in compute-and-forward scheme serves as the role of approximating channel gains to integer coefficients. After scaling the received signal, the relay S_2 then broadcasts $\mathbf{x}_2$ to S_1 and S_3. We look into the receiver processing at S_3, nevertheless the argument holds similarly for S_1. Assuming S_3 has prior knowledge of its own information and perfect CSI of all links, therefore after canceling the

signals corresponding to its own message, the received signals in the j-th sub-block at S_3 can be viewed as the outputs of the point-to-point AWGN channel with the equivalent SNR being

$$\frac{|h_{1,2}^{(j)}\beta_2^{(j)}h_{2,3}^{(j)}|^2 P_1}{|\beta_2^{(j)}h_{2,3}^{(j)}|^2+1}. \tag{4.33}$$

S_3 then simply employs a regular decoder for point-to-point channel, say the one described in Section 4.3 for example.

4.4.4 Simulation results

Before presenting the simulation results, we first derive the corresponding outage probability for this model. This outage probability will again give us a sense about how good these schemes are. Depending on either CF or separation-based CF is considered, we can derive the corresponding outage probability based on the achievable computation rate of CF in (4.24) or (4.30) as follows,

$$P_{out,CF}(R) = Pr\{\mathcal{R}(\mathbf{h}_2^{(1)},\ldots,\mathbf{h}_2^{(b)},\mathbf{a}_2) < R\}, \tag{4.34}$$

or

$$P_{out,SCF}(R) = Pr\{\mathcal{R}_s(\mathbf{h}_2^{(1)},\ldots,\mathbf{h}_2^{(b)},\mathbf{a}_2) < R\}. \tag{4.35}$$

We now investigate the FER performance at the relay node of the TWRN. We first look at the case of quasi-static fading channel ($b = 1$) in Fig. 4.6 where one observes that the proposed root-LDA lattice (2π) code is only 1.5 dB away from the outage limit of the separation-based CF. In addition, there is no loss in coding gain comparing against the LDA lattice code even though the root-LDA lattice code is designed for block-fading channel. One also observes that there is 1 dB gap between the outage limit of the CF and that of the separation-based CF. This gap corresponds

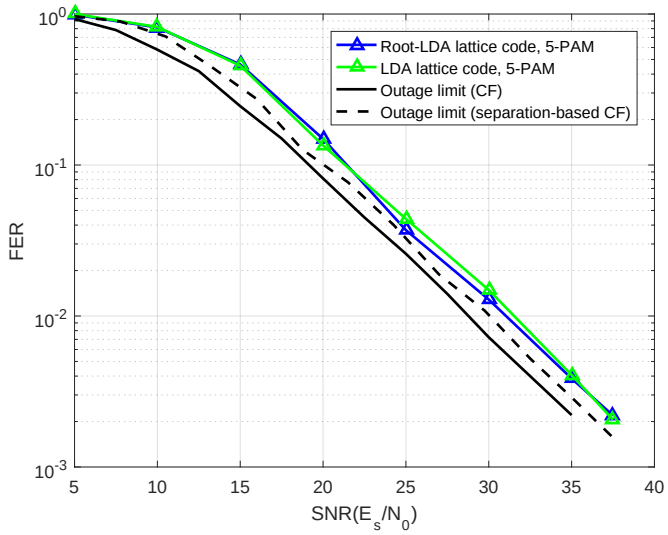


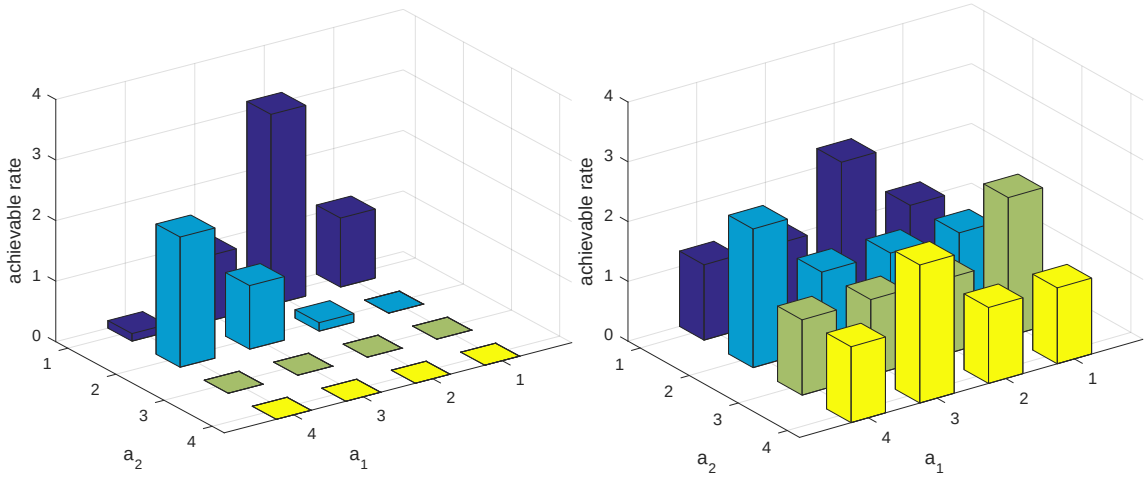
Figure 4.6: FER vs. SNR curves for root-LDA lattice codes in two-way relay network with quasi-static fading channel (uplink)

to the shaping gain as the nested lattice shaping employed by CF attains the optimal Gaussian shaping while the separation-based CF uses hypercube shaping.

For block-fading channel with $b = 2$, as shown in Fig. 4.8, the proposed root-LDA lattice code achieves full-diversity and the slope of FER is the same as that of the outage limit of separation-based CF. One interesting observation is that the outage limit of the separation-based CF is better than that of the CF in this scenario. This phenomenon can be explained with the example shown in Fig. 4.7. Suppose we consider quasi-static fading channel in two-way relay network. The channel gains are $h_1 = 1$, $h_2 = 0.5$, and SNR=25dB. We compare the achieve computation rates of two schemes with different choices of integer coefficients a_1 and a_2. CF achieves the highest rate 3.15 when $a_1 = 2$ and $a_2 = 1$, meanwhile the highest rate separation-based CF achieves is 2.32, also when $a_1 = 2$ and $a_2 = 1$. In this case, CF achieves higher computation rate than separation-based CF. This is reasonable since the channel gains are well approximated by the integer vector and thereby, the self-interference

is small. Hence, the decoding problem can be equivalently thought of as function computation in the high SNR regime where lattice decoding is approximately optimal [63]. Then the extra shaping gain makes CF outperform separation-based CF. On the other hand, with other choices of integer vectors, CF performs poorly while separation-based CF performs better. The phenomenon can be explained as follows. When the channel gains are not well approximated by the integer vector, the self-interference is large and the decoding problem can be thought of as function computation in the low SNR regime. In this regime, lattice decoding performs poorly [63] while the separation-based CF enabling ML decoding outperforms CF. Back to the block-fading channel case, the integer coefficients should be fixed during codeword duration, therefore CF typically can only approximate the channel gains of one sub-block well but not for other sub-blocks. However the level of sensitivity to different integer coefficients is lower for separation-based CF which leads to resiliency to channel variations throughout the codeword. For the rest of the FER simulations, CF means separation-based CF unless otherwise specified.

We then compare the end-to-end FER performance of CF and AF in TWRN in Fig. 4.9. We adopt the same root-LDA (2π) lattice code for both CF and AF. We also assume that the uplink and downlink channels are reciprocal. The outage limit (separation-based CF) is derived based on eq. (4.35) and eq. (4.15) which correspond to phase 1 and phase 2, respectively. The outage limit (AF) is derived based on eq. (4.15), since AF can be regarded as point-to-point communication after interference cancellation at receiver. In Fig. 4.9. We can observe that the two protocols demonstrate comparable FER performance and the slopes of FER are consistent with the outage limits.



(a) Compute-and-forward (b) separation-based compute-and-forward

Figure 4.7: Achievable rates of compute-and-forward and separation-based compute-and-forward associate with different integer coefficients when $h_1 = 1$ and $h_2 = 0.5$ and SNR=25dB.

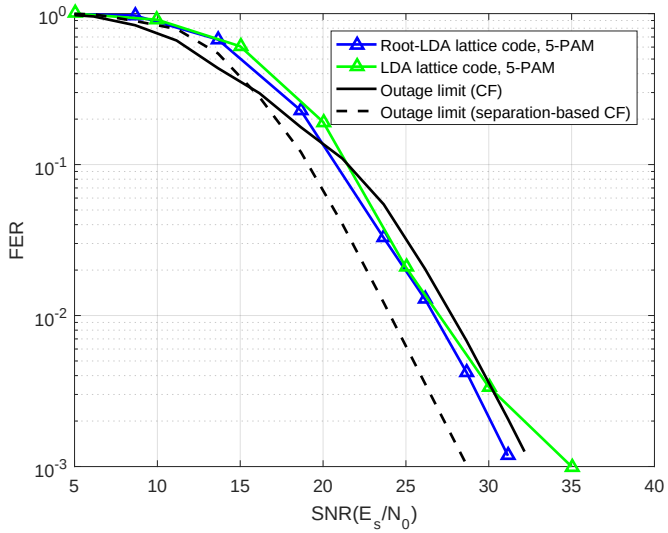


Figure 4.8: FER vs. SNR curves for root-LDA lattice codes in two-way relay network with block-fading (b=2) channel (uplink)

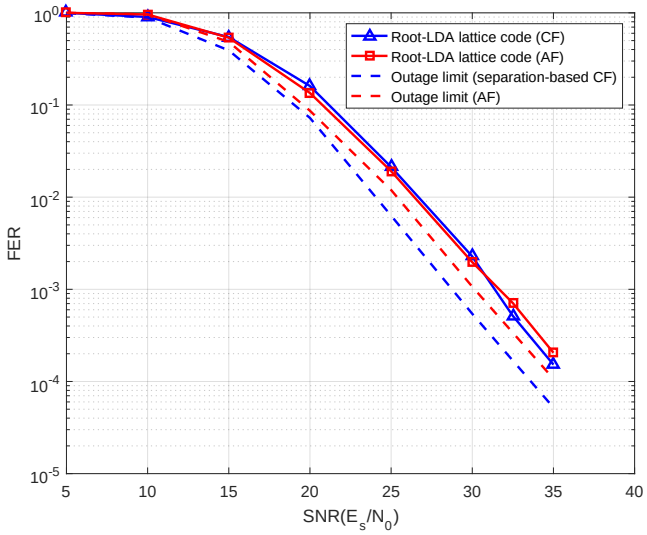


Figure 4.9: End-to-end FER curves for CF and AF relaying strategies in two-way relay network with block-fading (b=2) channel

4.5 Root-LDA Lattice Codes for Multi-Hop Line Networks

In this section, we consider a line network which consists of 5 nodes. S_1 and S_5 are sources nodes who want to exchange information. S_2, S_3, and S_4 are relay nodes. The line network can be decomposed into multiple point-to-point communication and TWRN for which the transmission schemes mentioned in Section 4.3 and Section 4.4 can be applied.

4.5.1 Compute-and-forward in line network

We focus on how to deliver a codeword $\mathbf{x}_{1,t}$ from S_1 to S_5. Initially, at phase t, S_1 and S_3 transmit codeword $\mathbf{x}_{1,t}$ and $\mathbf{x}_{3,t}$ to relay S_2 simultaneously. $\mathbf{x}_{1,t}$ carries the information originated from S_1. $\mathbf{x}_{3,t} = f(\mathbf{x}_{2,t-1}, \mathbf{x}_{4,t-1})$ (function of codewords from previous phase $t-1$). The decoding algorithm mentioned in section 4.4.2 is adopted at relays S_2, S_3, and S_4, respectively. At S_2, it first computes the joint APP of $c_1[n]$

and $c_3[n]$ given $y_2[n]$,

$$P_{APP}(c_1[n] = v_1, c_3[n] = v_3) = \sum_{\chi_1 \in \Gamma_{v_1}} \sum_{\chi_3 \in \Gamma_{v_3}} \mathbb{P}(h_{1,2}^{(j)} \chi_1 + h_{3,2}^{(j)} \chi_3 | y_2[n]), \tag{4.36}$$

where $\Gamma_v = \{\chi | \chi = \mathcal{M}(v) + p\zeta, \zeta \in \mathbb{Z}\}$, and

$$\mathbb{P}(h_{1,2}^{(j)} \chi_1 + h_{3,2}^{(j)} \chi_3 | y_2[n]) \propto \exp(-\frac{(y_2[n] - h_{1,2}^{(j)} \chi_1 - h_{3,2}^{(j)} \chi_3)^2}{2\sigma^2}),$$

the APP for $\mathbf{c}_f[n]$ is then given by

$$\mathbb{P}(c_f[n] = v | y_2[n]) = \sum_{b_{1,2} v_1 \oplus b_{3,2} v_3 = v} P_{APP}(v_1, v_3). \tag{4.37}$$

The APPs are fed into the message-passing decoder, then the decoder outputs the estimated codeword $\hat{\mathbf{c}}_f$. The final step is quantizing $\mathbf{y}_2$ to the nearest point which denotes $\hat{f}_{2,t}(\mathbf{x}_{1,t}, \mathbf{x}_{3,t})$, where $\hat{f}_{2,t}$ denotes the function estimated by S_2 in phase t. In phase $t+1$, S_2 transmits $\mathbf{x}_{2,t+1} = \hat{f}_{2,t}(\mathbf{x}_{1,t}, \mathbf{x}_{3,t})$ to relay S_3, in the meantime, S_4 also transmits $\mathbf{x}_{4,t+1} = \hat{f}_{4,t}(\mathbf{x}_{3,t}, \mathbf{x}_{5,t})$ to S_3. The same decoding procedure is applied again, but this time it's performed at S_3 and the decoding output is $\hat{f}_{3,t+1}(\mathbf{x}_{2,t+1}, \mathbf{x}_{4,t+1})$. In phase $t+2$, S_3 transmits $\mathbf{x}_{3,t+2} = \hat{f}_{3,t+1}(\mathbf{x}_{2,t+1}, \mathbf{x}_{4,t+1})$ to relay S_4, in the mean time, S_5 also transmits $\mathbf{x}_{5,t+2}$ to S_4. The decoding output at relay S_4 is $\hat{f}_{4,t+2}(\mathbf{x}_{3,t+2}, \mathbf{x}_{5,t+2})$, then it is transmitted to S_5 in phase $t+3$, and the last hop is just point-to-point transmission. Suppose all functions are decoded correctly, the decoded codeword at S_5 is a function of codewords from S_1 and S_5. Based on the knowledge of its own codewords and previously decoded codewords (from S_1), S_5 is able to reconstruct the most recent codewords sent from S_1.

Here, we only focus on transmitting single codeword from S_1 to S_5, in general

there can be multiple codewords being transmitted in the network from both S_1 and S_5 simultaneously. Below we demonstrate an example of 5-node line network with CF.

- Nodes in $\mathcal{A}_o = \{S_1, S_3, S_5\}$ allow to transmit simultaneously (S_1 and S_5 always transmit new codewords) in phase t, t is odd.
- Nodes in $\mathcal{A}_e = \{S_2, S_4\}$ allow to transmit simultaneously in phase t, t is even.

We further describe the transmission protocol explicitly in Fig. 4.10. To distinguish different codewords sent by the same source node, we say $\mathbf{x}_{i,j}$ is the j-th codeword sent by node S_i. For the ease of explanation, we first assume the channels are AWGN channels. In phase 1, S_1 transmits $\mathbf{x}_{1,1}$ to S_2 and S_5 transmits $\mathbf{x}_{5,1}$ to S_4. S_3 is idle since it has not received anything. In phase 2, S_2 and S_4 forward $\mathbf{x}_{1,1}$ and $\mathbf{x}_{5,1}$, respectively, and S_3 computes the codeword corresponding to the linear combination $\mathbf{x}_{1,1} + \mathbf{x}_{5,1}$. After phase 2, we only focus on information flow from S_1 and S_5 where the other way works in a similar fashion. In phase 3, S_3 broadcasts the previously computed codeword to S_2 and S_4. Meanwhile, S_1 and S_5 transmit new codewords $\mathbf{x}_{1,2}$ and $\mathbf{x}_{5,2}$ to S_2 and S_4, respectively. S_2 then computes the codeword corresponding to linear combination $\mathbf{x}_{1,2} + \mathbf{x}_{1,1} + \mathbf{x}_{5,1}$ and S_4 computes the codeword corresponding to the linear combination $\mathbf{x}_{5,2} + \mathbf{x}_{1,1} + \mathbf{x}_{5,1}$. In phase 4, S_2 broadcasts its computed function to S_1 and $\mathbf{x}_{5,1}$ can be extracted out from this function with the message originated at S_1. S_4 also broadcasts its function to S_5 and $\mathbf{x}_{1,1}$ can be extracted out from this function with the message originated at S_5. It's worth mentioning that the extracted codewords will become side information for future decoding. Phase 5 and 6 follows the same process and $\mathbf{x}_{5,2}$ can be decoded by S_1 and $\mathbf{x}_{1,2}$ can be decoded by S_5, respectively, at the end of phase 6.

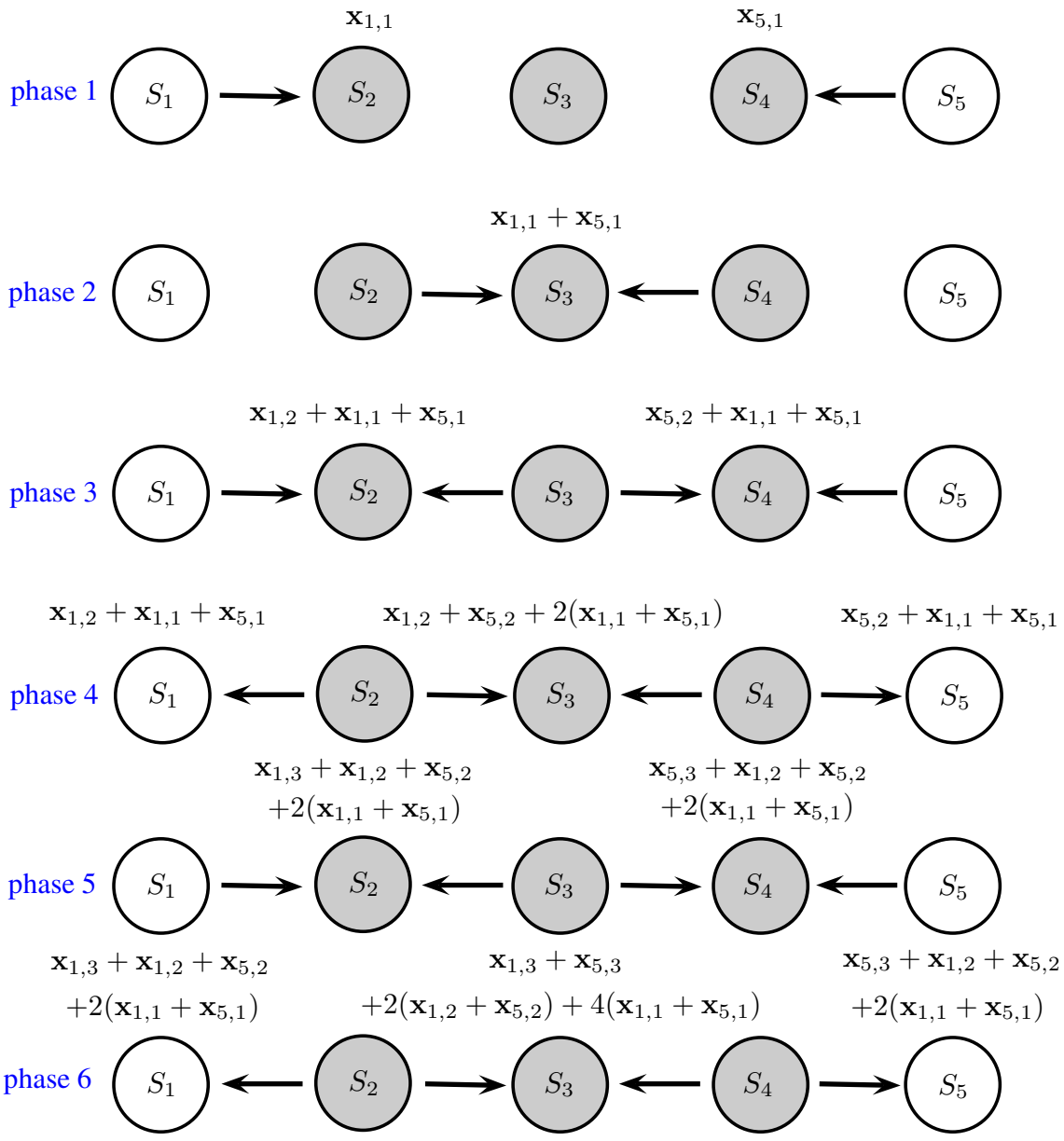


Figure 4.10: Transmission protocol for 5-node line network

4.5.2 Amplify-and-forward in line network

Based on the signal model in (4.2), the forwarding signals at S_2, S_3, S_4 with amplify-and-forward protocol are given by

$$x_i[n] = \beta_i^{(j)} y_i[n], \quad i = 2, 3, 4, \quad n = 1, \ldots, N, \quad j = 1 + \left\lfloor \frac{(n-1)}{\ell} \right\rfloor, \tag{4.38}$$

where

$$\beta_i^{(j)} = \sqrt{\frac{P_i}{|h_{i-1,i}^{(j)}|^2 P_{i-1} + |h_{i+1,i}^{(j)}|^2 P_{i+1} + 1}}. \tag{4.39}$$

Again, let us focus on the signal flows from S_1 to S_5. The received signal at S_5 consists of three parts: it's own codewords, previously decoded codewords from S_1, and $\mathbf{x}_1$ where the former two components can be canceled out. Therefore after cancellation, the received signals in the j-th sub-block at S_5 can be regarded as the outputs of point-to-point AWGN channel with equivalent SNR being

$$\frac{|h_{1,2}^{(j)} \beta_2^{(j)} h_{2,3}^{(j)} \beta_3^{(j)} h_{3,4}^{(j)} \beta_4^{(j)} h_{4,5}^{(j)}|^2 P_1}{|\beta_2^{(j)} h_{2,3}^{(j)} \beta_3^{(j)} h_{3,4}^{(j)} \beta_4^{(j)} h_{4,5}^{(j)}|^2 + |\beta_3^{(j)} h_{3,4}^{(j)} \beta_4^{(j)} h_{4,5}^{(j)}|^2 + |\beta_4^{(j)} h_{4,5}^{(j)}|^2 + 1}, \tag{4.40}$$

where the i-th term in the denominator corresponds to the variance of additive noise at S_{i+1} seen at S_5. The AF strategy then performs decoding based on this equivalent point-to-point signal.

4.5.3 Simulation results

In Fig. 4.11, we investigate the end-to-end FER performance of CF and AF in block-fading channels with $b = 2$. The same root-LDA lattice code (2π) is used for both the strategies. The outage limit (separation-based CF) is derived based on eq. (4.35) and eq. (4.15). The outage limit (AF) is derived based on eq. (4.15). In Fig. 4.9. The FER performance shows that CF with proposed coding scheme still

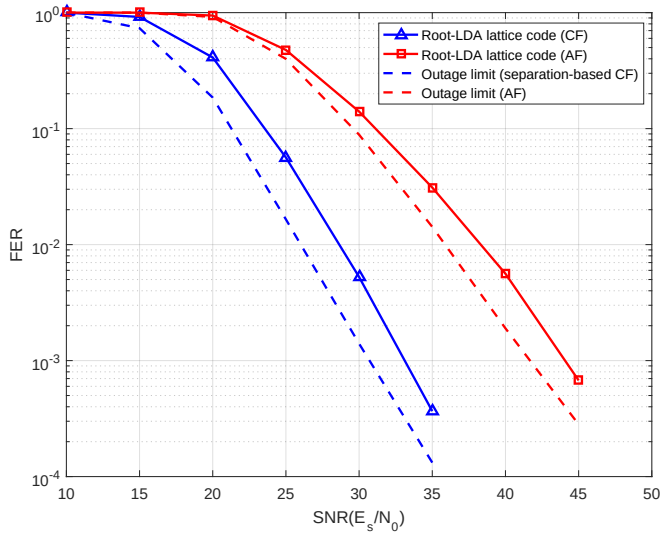


Figure 4.11: FER comparison between CF and AF in 5-node line network with block-fading channel ($b = 2$).

achieves full diversity in multiple-hop network. Meanwhile, the FER of AF is 11 dB away from CF at FER=10^{-3}. On the other hand, both simulation and outage limit of AF cannot achieve full diversity. There are two factors that contribute to this loss. The first one is noise accumulation at relays which lead to the entire FER curve shifting to the right. Moreover, the slope of the FER curve for AF is not as steep as that of the FER curve for CF. The reason that this phenomenon occurs is due to accumulation of deep-fading events which can be easily explained by looking into (4.40) that any one of the four channels is in deep fade results in a deep fade for this sub-block. Therefore, there is an obvious reason to choose CF over AF in block-fading channels as CF tries to clean up errors at every hop which prevents both noise accumulation and deep-fading events accumulation.

4.6 Conclusion

In this section, we have proposed root-LDA lattice codes for CF in relay networks over block-fading channel. The proposed root-LDA lattice codes have allowed relays

to compute integer linear combination of codewords in block-fading channels. Simulation results have shown that the proposed solution exhibits full diversity in both TWRN and the multiple-hop line network with 5 nodes while the conventional LDA lattice codes not designed specifically for block-fading channels cannot do so. In addition, simulation results have also suggested that there is no fundamental penalty in coding gain when we optimize the code specifically to deal with block-fading channels. For the line network with 5 nodes, we have shown that CF substantially outperforms AF as the CF with the proposed root-LDA lattice codes cleans up errors at every hop and thereby prevents both noise accumulation and deep-fading event accumulation from happening.

5. COMPUTE-AND-FORWARD FOR ULTRA-RELIABLE LOW-LATENCY COMMUNICATION

5.1 Introduction

In this section, we study the problem of compute-and-forward with short lattice codes. As new applications for future 5G communication are emerging, e.g., autonomous driving and virtual reality, low-latency communication with high reliability is necessary, otherwise the requirements of these applications cannot be met. In this regard, powerful short channel codes with reasonable encoding/decoding complexity are indispensable. Although we have seen fruitful results on code design for point-to-point communication, studies on code design specifically for compute-and-forward are rarely found. We aim to tackle this problem in a two-step approach. First, we propose the family of binary generalized LDPC (GLDPC) codes. Compared to conventional binary LDPC codes, the check nodes of the proposed GLDPC codes can be jointly characterized by a convolutional code, rather than multiple single parity-check codes. We compare the codeword error rate performance of the proposed GLDPC codes with state-of-the-art codes in the point-to-point AWGN channel. From what we learned in the point-to-point channel, we then construct good short lattice codes for compute-and-forward problem. The proposed lattice codes are constructed from non-binary GLDPC codes via construction A and proper shaping. We investigate the codeword error rate performance of the proposed lattice codes in a compute-and-forward network. Comparison with other lattice codes is provided as well.

5.2 Point-to-Point Communication

5.2.1 Problem statement

Consider a point-to-point communication problem that a source node S wishes to send information to a destination node D. The source node first generates the message $\mathbf{u} \in \mathbb{F}_p^K$. The message $\mathbf{u}$ is then encoded to codeword $\mathbf{c} \in \mathbb{F}_p^N$. $\mathbf{c}$ is then mapped into $\mathbf{x} \in \mathcal{A}^N$ via a mapping $\mathcal{M}$, where the mapping is given by,

$$\mathcal{M}(u) \triangleq \begin{cases} u, & 0 \leq u \leq \frac{p-1}{2}, \\ u-p, & \frac{p-1}{2} < u < p, \end{cases} \tag{5.1}$$

for $p \geq 3$, and $\mathcal{M}(u) \triangleq u - \frac{1}{2}$ for $p = 2$ (i.e., BPSK). The codeword is subject to a power constraint where

$$\frac{1}{N}\|\mathbf{x}\|^2 = \frac{1}{N}\sum_{n=1}^{N} |x[n]|^2 \leq P. \tag{5.2}$$

At the destination node, the n-th received symbol is given by

$$y[n] = hx[n] + z[n], \quad n = 1, \ldots, N, \tag{5.3}$$

where $h \in \mathbb{R}$ (or $\mathbb{C}$ depending on whether the signal constellation is real or complex) is the channel coefficient between the source node S and destination node D, and $z[n] \sim \mathcal{CN}(0,1)$ is the white Gaussian noise seen at destination node.

5.2.2 Conventional LDPC codes

In this section, we first review the basics of binary LDPC code. Subsequently, we focus on discussing the proposed generalized LDPC code.

A binary LDPC code can be fully characterized by the Tanner graph [72], where

a Tanner graph is a bipartite graph which is the graph representation of the LDPC code's parity check matrix. As shown in Fig. 5.1, a Tanner graph consists of variable nodes (circles), check nodes (squares), and edges. Each variable node is corresponding to a code symbol and each check node is corresponding to a parity check equation. An edge is established between i-th check node and j-th variable node if the entry at i-th row and j-th column of the parity check matrix is 1. The number of edges connected to each node is specified by a given degree distribution. A straightforward way to define the degree distribution is specifying the fraction of variable nodes with degree i as L_i, and specifying the fraction of check nodes with degree j as R_j. Now the so-called node-perspective degree distribution is given by

$$L(x) = \sum L_i x^i, \tag{5.4}$$

and

$$R(x) = \sum R_j x^j, \tag{5.5}$$

where $\sum L_i = 1$ and $\sum R_j = 1$. Similarly, and more commonly, the degree distribution can be defined from edge perspective. The so-called edge-perspective degree distribution is given by

$$\lambda(x) = \sum \lambda_i x^{i-1}, \tag{5.6}$$

and

$$\rho(x) = \sum \rho_j x^{j-1}, \tag{5.7}$$

here λ_i denotes the fraction of edges that are incident with degree i variable node and ρ_j denotes the fraction of edges that are incident with degree j check node. What follows is the relationship between node-perspective degree distribution and

edge-perspective degree distribution,

$$L_i = \frac{\lambda_i}{i \int_0^1 \lambda(x)dx}, \tag{5.8}$$

and

$$R_j = \frac{\rho_j}{j \int_0^1 \rho(x)dx}. \tag{5.9}$$

Therefore, given the degree polynomials $\lambda(x)$ and $\rho(x)$, one can then name the set of LDPC codes that satisfy the degree distribution as (λ, ρ) LDPC ensemble (λ, ρ). The LDPC ensemble is said to be regular if all the variable nodes have same degree L_i and all check nodes have same degree R_j, otherwise it is irregular. For instance ,we say that a code drawn from the (3,6) LDPC ensemble will have variable nodes of degree 3 and check nodes of degree 6.

It has been proved that there exists optimal degree distribution such that the associate LDPC ensemble is capacity achieving in the binary erasure channel with message-passing decoding, under the assumption of no cycle in the corresponding Tanner graph [73]. This assumption is true in the asymptotic regime where block length N goes to infinity. In the asymptotic regime, the number of edges comparing to the number of variable/check nodes is relatively small, therefore the graph is sparse which ensures that the probability of cycle occurs tends to zero. On the other hand, for finite-length LDPC codes (say few thousand bits), their corresponding Tanner graphs consist of small cycles which may deteriorate the decoding performance of message-passing decoder [74] [75]. Fortunately, for codes with medium length or above ($N > 512$), there are many preliminary works which focus on designing graphs with high girth [76] [77] [78], where girth is defined as the length of the shortest cycle in the graph. One of the most famous approaches is employing progressive edge

growth (PEG) algorithm to design the Tanner graph [76], which PEG algorithm is a greedy algorithm that establishes edges between variable nodes and check nodes one at a time, and the process terminates when all nodes satisfy the degree distribution. Despite its sub-optimality, often times the resulting codes have excellent decoding performance under message-passing decoding.

We realize that the problem of designing graph with high girth is even more challenging in short-block length regime, e.g., $N < 256$. One straightforward approach is reducing the number of edges in the graph, then the possibility of seeing small cycles is decreased. Yet, reducing edges results in a lower average degree of variable nodes. It has been reported that the minimum distance is upper-bounded by $(J + 1)!$ for regular LDPC codes, where J is the degree of variable nodes [79]. In [37], extensive simulation results for various short LDPC codes are reported. The authors observe that there has always existed a gap between the codeword error rate performance of the ML decoder and message-passing decoder. The main reason is that the Tanner graphs have low girth which prevents the message-passing decoder from converging. With these existing issues for conventional short LDPC codes in mind, we aim to design novel LDPC codes which have good minimum distance and good for message-passing decoding simultaneously.

5.2.3 Generalized LDPC codes

In order to construct codes that have good minimum distance, while in the same time good for message-passing decoding, we aim to construct LDPC codes having stronger check nodes where all the check nodes can be jointly characterized by the convolutional constraint. We name the proposed codes as generalized LDPC codes. The Tanner graph of GLDPC code is presented in Fig. 5.2. Here the variable nodes are arranged in a slightly different fashion, where the variable nodes on top of the

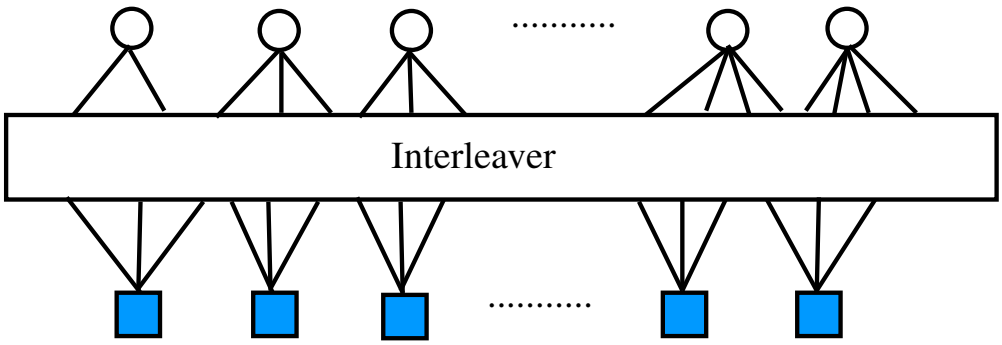


Figure 5.1: Tanner graph of LDPC code

graph are corresponding to information bits, and the variable nodes in the bottom are corresponding to parity bits.

We then use the set of auxiliary variable nodes and check nodes to describe a convolutional code, where the i-th auxiliary variable node (from left side) is defined as the state of convolutional encoder at time $i-1$, and the i-th check node represents the trellis step from state at time $i-1$ to state at time i. Thereby the value of i-th parity symbol is determined by the information symbols connected to the $i-$th check node and the auxiliary variable node on the left, (current encoder state). The rate of GLDPC code is give by

$$R = \frac{K}{qK(n-k/n)} = \frac{n}{q(n-k)}, \tag{5.10}$$

where q is the degree of the variable nodes corresponding to information bits, K is the number of information bits, k is the number of input bits of the convolutional code at each time instant, and n is the number of output bits of the convolutional code at each time instant. In Fig. 5.2, we let the degree of the variable nodes associated with information bits to be 2 and the degree of the variable nodes associated with parity

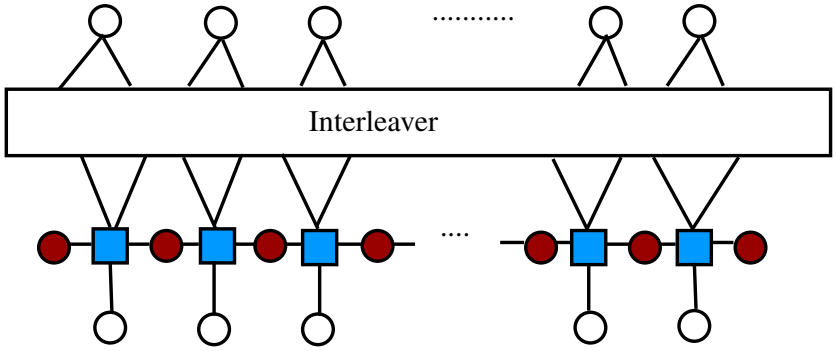


Figure 5.2: Tanner graph of generalized LDPC code

bits to be 1. The code rate of the convolutional code is 2/3, and the repetition factor for each information bit is $q = 2$, therefore the resulting GLDPC code has code rate $1/2$ *.

Based on the described Tanner graph, the GLDPC code consists of a repetition code, interleaver, and convolutional code. Each of the components is optimized in what follows. For the repetition code, the repetition factor is 2, therefore all the variable nodes corresponding to the information bits have degree 2. Apart from conventional LDPC codes which the associated Tanner graph have the higher variable degree (usually ≥ 3), here we set the variable degree to be either 2 or 1. We then can argue that the GLDPC code is more sparse than a conventional LDPC code form the perspective of variable node. We expect the GLDPC codes exhibit better convergence performance than conventional LDPC codes (which typically have higher degree of variable nodes) under message-passing decoding. Subsequently, we enforce the interleaving pattern to be what follows: For the input bits of the interleaver,

*The proposed generalized LDPC codes coincident with self-concatenated codes proposed in [80, 81]. Nevertheless, to the best of author's knowledge, these seminal works focus on code designs in the regime of large block length. The scope of designing short length GLDPC/self-concatenated code has not been studied in the literature.

if they are induced from the same information bit, then they should be s-position away from each other after interleaving. The interleaving pattern can eliminate the occurrence of cycles of length 4 in the Tanner graph. Finally, convolutional codes with high free distance are selected, which can increase the minimum distance of the GLDPC code.

5.2.4 Encoding of GLDPC codes

The encoder is illustrated in Fig 5.3, which consists of repeater, interleaver and systematic convolutional encoder. The information sequence is first fed into the repeater serially, where the number of repetition q for each information bit is corresponding to the degree of the information variable nodes. The output sequence of the repeater is then randomly interleaved. Subsequently, we feed the interleaved sequence into the systematic convolutional encoder serially. Here the convolutional encoder generates the parity bits of the convolutional codeword only. We then concatenate the original information sequence $\mathbf{c}_I = \mathbf{u}$ and the parity bits of the convolutional codeword, which denoted as $\mathbf{c}_R$. Finally, the GLDPC codeword is given by

$$\mathbf{c} = [\mathbf{c_I}\ \mathbf{c_R}]. \tag{5.11}$$

Therefore, the GLDPC codeword is in systematic form. Since a convolutional encoder is incorporated in the GLDPC encoder, the ending state of the convolutional encoder should be terminated properly. One approach is forcing the ending state to all-zero state [82], and this can be done by enforcing the last m (constraint length of the convolution code) input bits of the convolutional encoder to certain values. However, this leads to significant rate loss since the codeword length is small. To overcome the problem of rate loss, we employ tail-biting (TB) convolutional code [83]. An important characteristic of tail-biting convolutional codes is that the encoder always

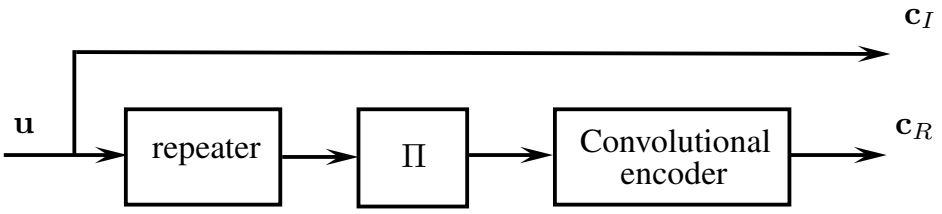


Figure 5.3: Encoder for the GLDPC code

forces the starting state and the ending state to be the same [84]. The starting state is determined in what follows. If it is a feedforward TB convolutional encoder, the starting state can be determined by the last m information bits (suppose the constraint length of the convolutional code is m). We first start at the all-zero state. We then feed the last m information bits to the encoder. Now the ending state of the encoder will be the actual starting state of the encoding process. By doing this, we can show that the starting state and ending state of the encoder will be the same since the state of the feedforward encoder only depends on the last m input bits. For feedback encoder, the initialization of the encoder is not as straightforward as the feedforward case. This is reasonable since the state of the encoder depends on all previous input bits. For interesting readers, the detail can be found in [84]. The advantage of TB convolutional code is that no termination sequence required for the encoder, therefore we can overcome the problem of rate loss. Note that since the starting state is dependent on the input bits, thus the receiver does not have the knowledge of the starting state

5.2.5 Decoding of GLDPC codes

In this section, the decoding algorithm for the GLDPC code is discussed. After receiving the signals from the channel, we first compute the log likelihood ratio (LLR)

of the n-th codeword bit which is given by

$$L_c[n] = \log \frac{p([n] = 1|y[n])}{p(c[n] = 0|y[n]}. \tag{5.12}$$

As shown in Fig. 5.2, we jointly consider the sequence of bits that are connected to the check nodes as a convolutional codeword. We then consider the set of all check nodes as a super check node that is shown in Fig. 5.4. Within the super check node is the symbol-wise maximum a posteriori probability (MAP) decoder which can be realized by the BCJR algorithm [85]. The BCJR algorithm computes the extrinsic LLRs then pass them back to the variable nodes which one turbo iteration is completed. In the next turbo iteration, the variable nodes apply standard LLR calculations then send the updated LLRs to the BCJR decoder again. The decoding process continues until a valid codeword is found[†] or the maximum turbo iteration is reached. Note that the complexity of decoding high-rate convolutional codes is substantially higher than the complexity of decoding low-rate convolutional codes. The additional complexity comes from the fact that the number of paths in the trellis is increased. On the other hand, it is equivalent to decode the dual code (low-rate code) which is computationally more efficient. We, therefore, employ the BCJR algorithm proposed in [86] which can compute the desired extrinsic information via decoding the dual convolutional code. Note that since tail-biting convolutional code is exploited and the exact starting (ending) state is unknown to the decoder, thus a circular-typed BCJR decoding is performed on the extended trellis [87] with extension depth t.

[†]Hard decisions are performed at the variable nodes in blue. The valid codeword should satisfy both the convolutional constraint and the constraints imposed by repetition codes

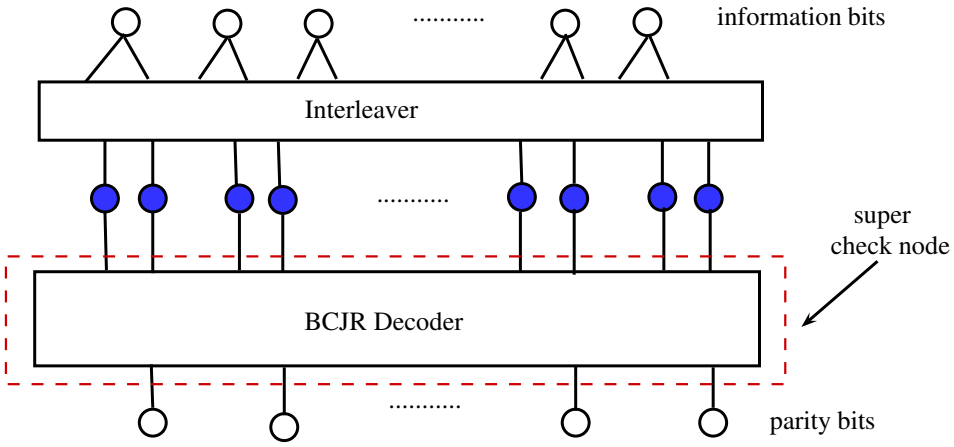


Figure 5.4: Decoder for the GLDPC code

5.2.6 Simulation results

In this section, we present simulation results of FER for various codes. To have a fair comparison, the decoding iteration of the proposed GLDPC code is set to 20, which the decoding complexity is roughly the same as polar code + CRC with list size of L=32 (Fig. 5.5). Note that number of basic operations (including additions, subtractions, multiplications, divisions and binary additions) is adopted as the metric for complexity analysis, which is also considered in [88]. Note that the constraint length of the convolutional code within the G-LDPC is set to $m = 4$. The performance of all the codes exclude G-LDPC codes are from [88]. We compare the FER performance at the target FER of 10^{-4}.

We first study the FER performance (Fig. 5.6). For length $N = 128, K = 64$ codes, Tail-biting convolutional (TB CC) code with constraint length (m=10) has the best FER performance, which is only 1dB away from the Polyanskiv finite length bound [33]. Polar codes have similar FER performance as TB-CC in waterfall region, but error floor is observable at FER=10^{-4}. One possible reason that causes error

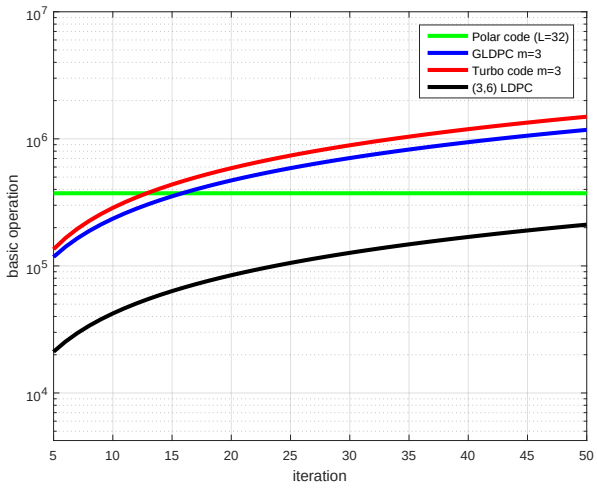


Figure 5.5: Decoder complexity: number of basic operations

floor from happening may be the short CRC (8 bits), which is not powerful enough for error detection. The proposed GLDPC code is only 0.6 dB away from Polar + CRC (L=32) and 0.2dB away from non-binary LDPC ($\mathbb{F}_{256}$). Moreover, we didn't observe error floor for G-LDPC. Also, GLDPC code has much lower decoding complexity than non-binary LDPC ($\mathbb{F}_{256}$) [88]. Without considering TB-CC, GLDPC code is a promising candidate for very short block length regime.

For length $N = 256, K = 128$ codes, first of all, TB CC is not showed in Fig, 5.7, due to its inferior FER in such length. The inferior FER performance is caused by low-weight codewords. Among the codes we present in Fig. 5.7, Polar code + CRC with list size (L=256) and non-binary LDPC ($\mathbb{F}_{256}$) are 0.5 dB and 0.8 dB away from Polyanskiy bound, respectively. However, the excellent performances are attained at the cost of highly complex decoders [88]. For the interests of codes with reasonable decoding complexity, we then compare polar code + CRC with list size (L=32) and the proposed GLDPC codes. The FER performances of both codes are comparable, which are 1 dB away from finite length bound. Finally, LTE turbo code

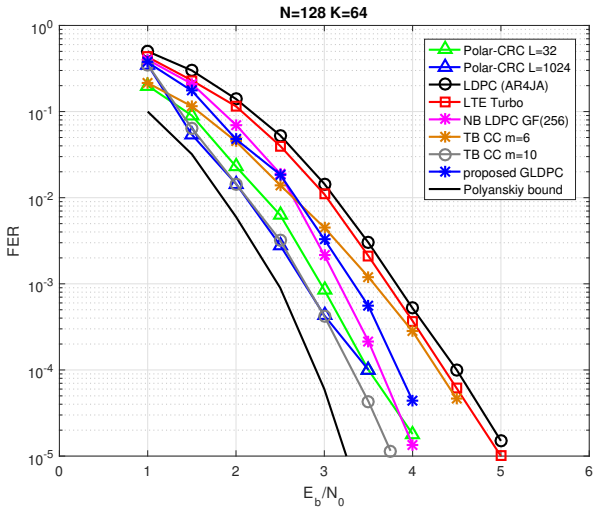


Figure 5.6: BER performance comparison of various codes (n=128, k=64) in point-to-point AWGN channel

and binary LDPC code chosen from AR4JA family are plotted as well, which the proposed GLDPC code is approximately 0.5 dB better than them.

5.3 Compute-and-Forward

5.3.1 Problem statement

In this section, we consider a compute-and-forward relay network with two source nodes S_1 and S_2, and one destination node D. S_1 and S_2 have messages $\mathbf{u}_1 \in \mathbb{F}_p^K$ and $\mathbf{u}_2 \in \mathbb{F}_p^K$, respectively. Each source node encodes its message $\mathbf{u}_s$ to a codeword $\mathbf{c}_s \in \mathbb{F}_p^N$. The codeword is then modulated to the transmitted signal $\mathbf{x}_s \in \mathcal{A}^N$ As shown in Fig. 5.8, S_1 and S_2 transmit signals $\mathbf{x}_1$ and $\mathbf{x}_2$ simultaneously to the destination node D. The received signal at the destination node is given by

$$y[n] = h_1 x_1[n] + h_2 x_2[n] + z[n], \quad n = 1, \ldots, N, \tag{5.13}$$

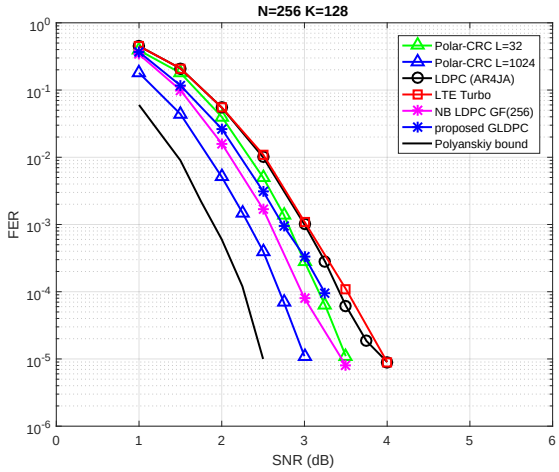


Figure 5.7: BER performance comparison of various codes (n=256, k=128) in point-to-point AWGN channel

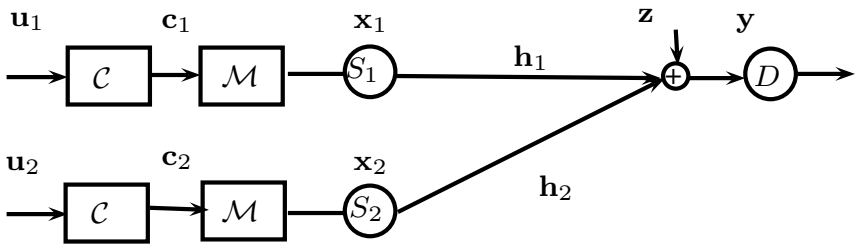


Figure 5.8: The compute-and-forward relay network with two source nodes and one destination(relay) node.

where h_s is the channel coefficient for the channel between source node S_s and destination node D and $z[n] \sim \mathcal{N}(0,1)$ is white Gaussian noise seen at node D. The node D is assumed to have the channel state information (CSI) of h_1 and h_2. The source nodes have knowledge of the distribution of the channel coefficients but have no knowledge of the channel realizations.

The relay node is interested in computing and forwarding a function of the messages, i.e., the relay node first chooses $\{a_1, a_2\}(a_i \in \mathbb{Z})$ such that $a_1\mathbf{x}_1 + a_2\mathbf{x}_2$ can approximate the received signal $h_1\mathbf{x}_1 + h_2\mathbf{x}_2$ carefully (suppose the channel is noiseless), then relay obtains $\{b_1, b_2\}$ ($b_s = a_s \mod p$) and computes $\mathbf{f} = b_1\mathbf{u}_1 \oplus b_2\mathbf{u}_2$. The optimal $\{a_s\}$ can be determined in terms of computation rate, which is given by

$$\mathcal{R}(\mathbf{h}, \mathbf{a}) = \frac{1}{2}\log^+\left(\left(\|\mathbf{a}\|^2 - \frac{P|\mathbf{h}^H\mathbf{a}|^2}{1+P\|\mathbf{h}\|^2}\right)^{-1}\right), \tag{5.14}$$

where $\mathbf{a} = [a_1\ a_2]^t$, and $\mathbf{h} = [h_1\ h_2]^t$. Note that the rate is achievable per real dimension. The computed function together with $\{b_1, b_2\}$ are then forwarded to a central destination. Ideally, the central destination will receive many of such functions from multiple relays. The central destination can then obtain the desired information from source nodes by solving all the received equations. For brevity, here we only analyze the minimal non-trivial case where there are two source nodes and one relay node, and we are interested in analyzing the error probability of the computed function at the relay.

5.3.2 Short lattice codes

In this section, we construct short lattice codes for the compute-and-forward problem. We employ construction A to construct a lattice then carve out a portion of the lattice to be the lattice code. We now introduce the construction of lattices.

Let $\mathcal{C}$ be an N dimensional non-binary (NB) code from some finite field $\mathbb{F}_p$, also let $\mathcal{M} : \mathbb{F}_p \to \mathbb{R}$ be the natural mapping given by

$$\mathcal{M}(u) \triangleq \begin{cases} u, & 0 \leq u \leq \frac{p-1}{2}, \\ u-p, & \frac{p-1}{2} < u < p, \end{cases} \tag{5.15}$$

when $p \geq 3$, and $\mathcal{M}(u) \triangleq u - \frac{1}{2}$ for $p = 2$ (i.e., BPSK). A lattice is obtained by tiling $\mathcal{M}(\mathcal{C})$ to the entire $\mathbb{R}^N$ as

$$\Lambda \triangleq \mathcal{M}(\mathcal{C}) + p\mathbb{Z}^N, \tag{5.16}$$

where the sum in the equation is the Minkowski sum. The fact that Λ is indeed a lattice can be easily shown by noticing that $\mathcal{M}$ is an isomorphism and $\mathcal{C}$ is a linear code. To construct the GLDPC lattice, we enforce the linear code $\mathcal{C}$ to be the non-binary GLDPC code. The non-binary GLDPC codes is constructed from the binary GLDPC code proposed in the previous section, which the ones in the binary parity check matrix are replaced by non-zero elements randomly chosen from the non-binary field. Similarly, the polar lattice can be constructed from the non-binary polar code.

So far we have discussed the construction of the lattice, yet a lattice consists of infinite points in $\mathbb{R}^N$, and the power at transmitters are limited in practical communication systems. One popular approach of using lattices for power-constrained problems is to carve a finite subset from the lattice via shaping techniques. Conventionally, spherical shaping is adopted where the codebook is the intersection of a sphere and the lattice [42]. Spherical shaping is power-efficient and leads to codebooks mimicking the capacity-achieving input distribution for the AWGN channel. However, the nice correspondence between lattice points and underlying linear codewords may no longer be preserved. Another popular approach for shaping is to

the nested lattice shaping [27]. Let (Λ_f, Λ_c) be a pair of nested lattices such that $\Lambda_c \subseteq \Lambda_f$, where Λ_f is the fine lattice and Λ_c is the coarse lattice. The lattice code obtained by nested lattice shaping, henceforth referred to as nested lattice code, is $\mathcal{C} = \Lambda_f \cup \mathcal{V}(\Lambda_c)$ where $\mathcal{V}(\Lambda_c)$ denotes the fundamental Voronoi region of coarse lattice Λ_c. Under some goodness conditions, the resulting nested lattice code can achieve the capacity of AWGN channel with lattice decoding. Since nested lattice codes preserve most of the structures which facilitate function computation, they are adopted in most of the CF literature. Therefore in this section, we employ nested lattice shaping, by choosing Λ_c such that $p\mathbb{Z}^N \subseteq \Lambda_c$. It is worth emphasizing that this choice includes the famous and low-complex hypercubic shaping $\Lambda_c = p\mathbb{Z}^N$.

5.3.3 Simulation results

In this section, we compare codeword error rate performance of function computation at the relay. We study the performance of GLDPC lattice code and CRC-aided polar lattice code. All the lattices are constructed from non-binary codes over $\mathbb{F}_3$ via construction A, then hypercubic shaping is applied to obtain lattice codes. GLDPC lattice codes with length $N = 128$ and $N = 256$ are constructed. Also, polar lattice codes with length $N = 128$ and $N = 256$ are constructed. All lattice codes are rate 1/2 codes. In the simulation, we let all channel gains to be 1. The function that maximizes the achievable computation rate (Eq. 5.14) at the relay will then be $x_1 \oplus x_2$, where the addition in $\mathbb{F}_3$.

For decoding algorithms, we employ message-passing algorithm for the GLDPC lattice code. Specifically, non-binary BCJR algorithm [89] is employed to decode the dual convolutional code at the check nodes. The polar lattice code is CRC-aided, thus non-binary successive cancellation list decoding is used. To have a fair comparison, the decoding iteration of the proposed GLDPC lattice code is set to 20, which the

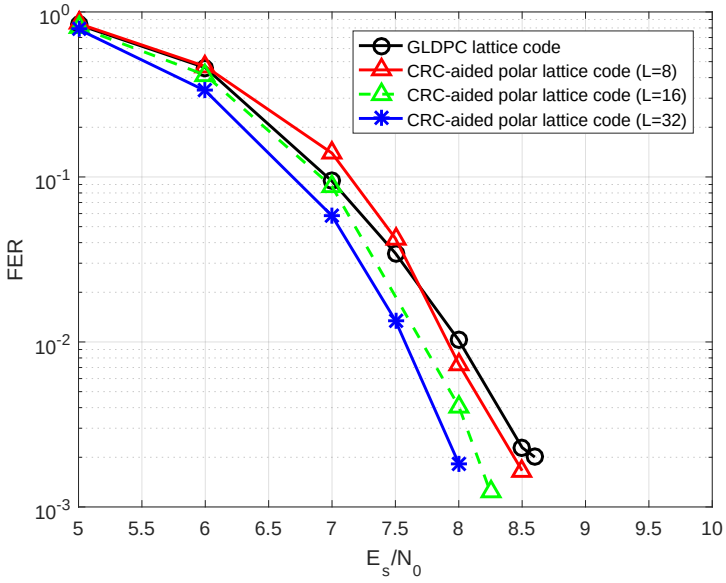


Figure 5.9: FER performance at relay with 2 source nodes and 1 relay (N=128)

decoding complexity is roughly the same as polar lattice code + CRC with list size of L=32.

For length $N = 128$ lattice codes Fig. 5.9, CRC aided polar lattice code with list size 32 outperforms the proposed GLDPC lattice code at FER= 10^{-3}. GLDPC lattice code exhibits similar FER performance with CRC-aided polar lattice code with list size=8 (L=8). We then investigate length $N = 256$ lattice codes Fig. 5.10, the proposed GLDPC lattice code and CRC-aided polar lattice code with list size =32 (L=32) exhibit comparable performance at FER= 10^{-3}. However, GLDPC lattice code exhibits better FER performance in the waterfall region.

5.4 Conclusion

In this section, we study the design of short lattice codes for compute-and-forward networks. We first investigate code designs in the scenario of point-to-point com-

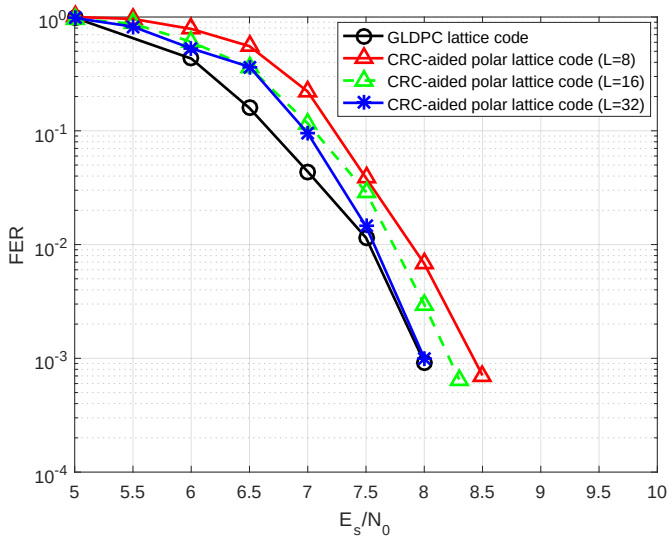


Figure 5.10: FER performance at relay with 2 source nodes and 1 relay (N=256)

munication. The binary generalized LDPC code is proposed. Apart from conventional LDPC codes, check equations can be jointly described by the convolutional constraint. The simulation shows that the proposed binary GLDPC code under message-passing decoding exhibits comparable codeword error rate performance to the CRC-aided binary polar code under successive cancellation list decoding of size $L = 32$. We then generalize the result to compute-and-forward networks. We construct GLDPC lattice codes and CRC-aided polar lattice codes which can facilitate function decoding at relays. Simulation result also shows that the proposed GLDPC lattice code also exhibits comparable codeword error rate performance to the CRC-aided polar lattice code.

6. CONCLUSION

In this dissertation, we first tackle two of the most challenging problems that compute-and-forward has encountered in practical wireless channels. The first problem is compute-and-forward with timing asynchronous users and the second problem is compute-and-forward in block-fading channels.

In Chapter 3, we study the first problem. We show that fundamentally there is no loss in the information rates achievable in the presence of timing delays which are integer multiples of symbol duration in comparison to the synchronous case. This is the first result to show that integer-valued asynchronism does not cause any reduction in the achievable rates. Moreover, practical implementation of the proposed scheme has also been considered where a joint detection and decoding scheme was considered. Simulation results for the two transmitters and one receiver case have further confirmed the theoretical analysis and observations. In Chapter 4, the second problem is studied. We have proposed root-LDA lattice codes for CF in relay networks over block-fading channel. The proposed root-LDA lattice codes have allowed relays to compute integer linear combination of codewords in block-fading channels. Simulation results have shown that the proposed solution exhibits full diversity in both TWRN and the multiple-hop line network with 5 nodes. In addition, simulation results have also suggested that there is no fundamental penalty in coding gain when we optimize the code specifically to deal with block-fading channels. When jointly considered these two problems, it can be easily shown that an universal solution exists, where a quasi-cyclic root-LDA lattice code can be constructed. The universal solution has its importance especially for practical communication systems since only one code needs to be implemented at the transceivers.

Finally, in Chapter 5, we study compute-and-forward with short lattice codes. We first investigate code designs in the scenario of point-to-point communication. The binary generalized LDPC code is proposed. Apart from conventional LDPC codes, check equations can be jointly described by the convolutional constraint. The simulation shows that the proposed binary GLDPC code under message-passing decoding exhibits comparable codeword error rate performance to the CRC-aided binary polar code under successive cancellation list decoding of size $L = 32$. We then generalize the result to compute-and-forward networks. We construct GLDPC lattice codes and CRC-aided polar lattice codes which can facilitate function decoding at relays. Simulation result also shows that the proposed GLDPC lattice code also exhibits comparable codeword error rate performance to the CRC-aided polar lattice code.

REFERENCES

[1] S. Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, pp. 371–381, Feb 2003.

[2] S. Zhang, S. C. Liew, and P. P. Lam, "Hot topic: Physical-layer network coding," pp. 358–365, Sept. 2006.

[3] P. Popovski and H. Yomo, "The anti-packets can increase the achievable throughput of a wireless multi-hop network," in *2006 IEEE International Conference on Communications*, vol. 9, pp. 3885–3890, June 2006.

[4] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[5] S. Katti, S. Gollakota, and D. Katabi, "Embracing wireless interference: Analog network coding," pp. 397–408, Oct. 2007.

[6] P. Popovski and H. Yomo, "Bi-directional amplification of throughput in a wireless multi-hop network," in *2006 IEEE 63rd Vehicular Technology Conference*, vol. 2, pp. 588–593, May 2006.

[7] B. Schein and R. Gallager, "The gaussian parallel relay network," in *2000 IEEE International Symposium on Information Theory (Cat. No.00CH37060)*, pp. 22– , 2000.

[8] S. Borade, L. Zheng, and R. Gallager, "Amplify-and-forward in wireless relay networks: Rate, diversity, and network size," *IEEE Transactions on Information Theory*, vol. 53, pp. 3302–3318, Oct 2007.

[9] T. Koike-Akino, P. Popovski, and V. Tarokh, "Optimized constellations for two-way wireless relaying with physical network coding," *IEEE Journal on Selected Areas in Communications*, vol. 27, pp. 773–787, June 2009.

[10] K. Narayanan, M. P. Wilson, and A. Sprintson, "Joint physical layer coding and network coding for bi-directional relaying," in *Forty-Fifth Annual Allerton Conference*, Sep 2007.

[11] W. Nam, S. Y. Chung, and Y. H. Lee, "Capacity bounds for two-way relay channels," in *2008 IEEE International Zurich Seminar on Communications*, pp. 144–147, March 2008.

[12] B. Nazer and M. Gastpar, "Compute-and-forward: Harnessing interference through structured codes," *IEEE Transactions on Information Theory*, vol. 57, pp. 6463–6486, Oct 2011.

[13] C. Feng, D. Silva, and F. R. Kschischang, "An algebraic approach to physical-layer network coding," *IEEE Transactions on Information Theory*, vol. 59, pp. 7576–7596, Nov 2013.

[14] B. Hern and K. Narayanan, "Multilevel coding schemes for compute-and-forward," in *2011 IEEE International Symposium on Information Theory Proceedings*, pp. 1713–1717, July 2011.

[15] O. Ordentlich, J. Zhan, U. Erez, M. Gastpar, and B. Nazer, "Practical code design for compute-and-forward," in *2011 IEEE International Symposium on Information Theory Proceedings*, pp. 1876–1880, July 2011.

[16] J. C. Belfiore, "Lattice codes for the compute-and-forward protocol: The flatness factor," in *2011 IEEE Information Theory Workshop*, pp. 1–4, Oct 2011.

[17] S. Zhang and S. C. Liew, "Channel coding and decoding in a relay system operated with physical-layer network coding," *IEEE Journal on Selected Areas in Communications*, vol. 27, pp. 788–796, June 2009.

[18] Y.-C. Huang, K. Narayanan, and N. E. Tunali, "Multistage compute-and-forward with multilevel lattice codes based on product constructions," *arXiv:1401.2228 [cs.IT]*, Jan. 2014.

[19] Q. T. Sun, J. Yuan, T. Huang, and K. W. Shum, "Lattice network codes based on eisenstein integers," *IEEE Transactions on Communications*, vol. 61, pp. 2713–2725, July 2013.

[20] N. E. Tunali, Y. C. Huang, J. J. Boutros, and K. R. Narayanan, "Lattices over eisenstein integers for compute-and-forward," *IEEE Transactions on Information Theory*, vol. 61, pp. 5306–5321, Oct 2015.

[21] D. Wang, S. Fu, and K. Lu, "Channel coding design to support asynchronous physical layer network coding," in *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, pp. 1–6, Nov 2009.

[22] L. Lu and S. C. Liew, "Asynchronous physical-layer network coding," *IEEE Transactions on Wir eless Communications*, vol. 11, pp. 819–831, February 2012.

[23] X. Wu, C. Zhao, and X. You, "Joint ldpc and physical-layer network coding for asynchronous bi-directional relaying," *IEEE Journal on Selected Areas in Communications*, vol. 31, pp. 1446–1454, August 2013.

[24] H. Najafi, M. O. Damen, and A. Hjorungnes, "Asynchronous compute-and-forward," *IEEE Transactions on Communications*, vol. 61, pp. 2704–2712, July 2013.

[25] Q. Yang and S. C. Liew, "Asynchronous convolutional-coded physical-layer network coding," *IEEE Transactions on Wireless Communications*, vol. 14, pp. 1380–1395, March 2015.

[26] I. E. Bakoury and B. Nazer, "The impact of channel variation on integer-forcing receivers," in *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 576–580, June 2015.

[27] U. Erez and R. Zamir, "Achieving 1/2 log (1+snr) on the awgn channel with lattice encoding and decoding," *IEEE Transactions on Information Theory*, vol. 50, pp. 2293–2314, Oct 2004.

[28] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[29] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, pp. 3051–3073, July 2009.

[30] I. Tal and A. Vardy, "List decoding of polar codes," in *2011 IEEE International Symposium on Information Theory Proceedings*, pp. 1–5, July 2011.

[31] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Transactions on Information Theory*, vol. 61, pp. 2213–2226, May 2015.

[32] K. Niu and K. Chen, "Crc-aided decoding of polar codes," *IEEE Communications Letters*, vol. 16, pp. 1668–1671, October 2012.

[33] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Transactions on Information Theory*, vol. 56, pp. 2307–2359, May 2010.

[34] L. Wei, "Several properties of short ldpc codes," *IEEE Transactions on Communications*, vol. 52, pp. 721–727, May 2004.

[35] T. Jerkovits and B. Matuz, "Turbo code design for short blocks," in *2016 8th Advanced Satellite Multimedia Systems Conference and the 14th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, pp. 1–6, Sept 2016.

[36] J. V. Wonterghem, A. Alloumf, J. J. Boutros, and M. Moeneclaey, "Performance comparison of short-length error-correcting codes," in *2016 Symposium on Communications and Vehicular Technologies (SCVT)*, pp. 1–6, Nov 2016.

[37] M. Helmling, S. Scholl, F. Gensheimer, T. Dietz, K. Kraft, S. Ruzika, and N. Wehn, "Database of Channel Codes and ML Simulation Results." www.uni-kl.de/channel-codes, 2017.

[38] G. Liva, L. Gaudio, T. Ninacs, and T. Jerkovits, "Code design for short blocks: A survey," *CoRR*, vol. abs/1610.00873, 2016.

[39] S. S. Ullah, G. Liva, and S. C. Liew, "Physical-layer network coding: A random coding error exponent perspective," in *arXiv:1702.01311v1.*

[40] N. di Pietro, J. J. Boutros, G. Zmor, and L. Brunel, "Integer low-density lattices based on construction a," in *2012 IEEE Information Theory Workshop*, pp. 422–426, Sept 2012.

[41] J. J. B. N. di Pietro, G. Zemor, "LDA lattices without dithering achieve capacity on the gaussian channel," *arXiv:1603.02863 [cs.IT]*, Mar. 2016.

[42] H. A. Loeliger, "New turbo-like codes," in *Proceedings of IEEE International Symposium on Information Theory*, pp. 109–, Jun 1997.

[43] J. Leech and N. J. A. Sloane, "Sphere packing and error-correcting codes," Canad. J. Math., vol. 23, no. 4, pp. 718–745, 1971.

[44] J. Conway and N. Sloane, *Sphere Packings, Lattices, and Groups*. Springer Verlag, 1999.

[45] B. Nazer and M. Gastpar, "Reliable physical-layer network coding," *Proceedings of the IEEE*, vol. 99, pp. 438–460, Mar. 2011.

[46] O. Ordentlich and U. Erez, "Cyclic-coded integer-forcing equalization," *IEEE Transactions on Information Theory*, vol. 58, pp. 5804–5815, Sept 2012.

[47] D. Wang, S. Fu, and K. Lu, "Channel coding design to support asynchronous physical layer network coding," pp. 1–6, Nov. 2009.

[48] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Şaşoğlu, and R. L. Urbanke, "Reed muller codes achieve capacity on erasure channels," *IEEE Transactions on Information Theory*, vol. 63, pp. 4298–4316, July 2017.

[49] Z. Li, L. Chen, L. Zeng, S. Lin, and W. Fong, "Efficient encoding of quasi-cyclic low-density parity-check codes," vol. 54, no. 1, pp. 71–81, 2006.

[50] Y. Chen and K. K. Parhi, "Overlapped message passing for quasi-cyclic low-density parity check codes," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, pp. 1106–1113, June 2004.

[51] Z. Wang and Z. Cui, "Low-complexity high-speed decoder design for quasi-cyclic ldpc codes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, pp. 104–114, Jan 2007.

[52] "Ieee standard for information technology– local and metropolitan area networks– specific requirements– part 11: Wireless lan medium access control (mac)and physical layer (phy) specifications amendment 5: Enhancements for higher throughput," *IEEE Std 802.11n-2009 (Amendment to IEEE Std 802.11-*

2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, and IEEE Std 802.11w-2009), pp. 1–565, Oct 2009.

[53] "Ieee standard for local and metropolitan area networks part 16: Air interface for broadband wireless access systems," *IEEE Std 802.16-2009 (Revision of IEEE Std 802.16-2004)*, pp. 1–2080, May 2009.

[54] "Frame structure channel coding and modulation for a second generation digital terrestrial television broadcasting system (dvb-t2)," *Digital Video Broadcasting (DVB)*, July 2015.

[55] "Low density parity check codes for use in near-earth and deep space applications (131.1-o-2 orange book)," *Consultative Committee for Space Data Systems*, Sep 2007.

[56] J. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," in *JPL, IPN Progress Report*, pp. 42–154, Aug. 2003.

[57] G. D. Forney, M. D. Trott, and S.-Y. Chung, "Sphere-bound-achieving coset codes and multilevel coset codes," vol. 46, pp. 820–850, May 2000.

[58] T. Guess and M. Varanasi, "A new successively decodable coding technique for intersymbol-interference channels," p. 102, June 2000.

[59] A. Abbasfar, D. Divsalar, and K. Yao, "Accumulate-repeat-accumulate codes," vol. 55, pp. 692–702, Apr. 2007.

[60] Q. Huang, Q. Diao, S. Lin, and K. Abdel-Ghaffar, "Cyclic and quasi-cyclic LDPC codes on constrained parity-check matrices and their trapping sets," vol. 58, pp. 2648–2671, May 2012.

[61] H. D. Pfister, J. B. Soriaga, and P. H. Siegel, "On the achievable information rates of finite state isi channels," in *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, vol. 5, pp. 2992–2996 vol.5, 2001.

[62] N. E. Tunali, K. Narayanan, J. Boutros, and Y.-C. Huang, "Lattices over eisenstein integers for compute-and-forward," Oct. 2012.

[63] M. P. Wilson, K. Narayanan, H. D. Pfister, and A. Sprintson, "Joint physical layer coding and network coding for bidirectional relaying," *IEEE Transactions on Information Theory*, vol. 56, pp. 5641–5654, Nov 2010.

[64] J. Boutros, A. Guillén i Fàbregas, E. Biglieri, and G. Zémor, "Low-density parity-check codes for nonergodic block-fading channels," vol. 56, pp. 4286–4300, Sept. 2010.

[65] N. di Pietro, J. J. Boutros, G. Zémor, and L. Brunel, "New results on low-density integer lattices," pp. 1–6, Feb. 2013.

[66] I. Maric, A. Goldsmith, and M. Medard, "Multihop analog network coding via amplify-and-forward: The high snr regime," *IEEE Transactions on Information Theory*, vol. 58, pp. 793–803, Feb 2012.

[67] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Progressive edge-growth tanner graphs," in *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, vol. 2, pp. 995–1001 vol.2, 2001.

[68] J. J. Boutros, "Diversity and coding gain evolution in graph codes," in *2009 Information Theory and Applications Workshop*, pp. 34–43, Feb 2009.

[69] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. 28, pp. 55–67, Jan 1982.

[70] P. C. Wang, Y. C. Huang, K. R. Narayanan, and J. J. Boutros, "Physical-layer network-coding over block fading channels with root-lda lattice codes," in *2016 IEEE International Conference on Communications (ICC)*, pp. 1–6, May 2016.

[71] N. E. Tunali, Y. C. Huang, J. J. Boutros, and K. R. Narayanan, "Lattices over eisenstein integers for compute-and-forward," *IEEE Transactions on Information Theory*, vol. 61, pp. 5306–5321, Oct 2015.

[72] R. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, pp. 533–547, Sep 1981.

[73] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 619–637, Feb 2001.

[74] T. J. Richardson, "Error floors of ldpc codes," in *Proc. 41st Annual Allerton Conf. on Communications, Control and Computing*, 2003.

[75] S. K. Chilappagari, S. Sankaranarayanan, and B. Vasic, "Error floors of ldpc codes on the binary symmetric channel," in *2006 IEEE International Conference on Communications*, vol. 3, pp. 1089–1094, June 2006.

[76] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, pp. 386–398, Jan 2005.

[77] M. P. C. Fossorier, "Quasicyclic low-density parity-check codes from circulant permutation matrices," *IEEE Transactions on Information Theory*, vol. 50, pp. 1788–1793, Aug 2004.

[78] Y. Wang, J. S. Yedidia, and S. C. Draper, "Construction of high-girth qc-ldpc codes," in *2008 5th International Symposium on Turbo Codes and Related Top-*

ics, pp. 180–185, Sept 2008.

[79] D. J. C. Mackay and M. C. Davey, "Evaluation of gallager codes for short block length and high rate applications," *Code, Syatems, and Graphical Models, B. Marcus and J. Rosenthal, eds. New York: Springer-Verlag*, pp. 113–130, 2001.

[80] D. Divsalar and F.Pollara, "Serial and hybrid concatenated codes with applications," in *International Symposium and Turbo Codes and Related Topics*, Sep 1997.

[81] M. F. U. Butt, S. X. Ng, and L. Hanzo, "Self-concatenated code design and its application in power-efficient cooperative communications," *IEEE Communications Surveys Tutorials*, vol. 14, pp. 858–883, Third 2012.

[82] S. Lin and D. J. Costello, *Error Control Coding, Second Edition*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2004.

[83] H. Ma and J. Wolf, "On tail biting convolutional codes," *IEEE Transactions on Communications*, vol. 34, pp. 104–111, Feb 1986.

[84] C. Weiss, C. Bettstetter, S. Riedel, and D. J. Costello, "Turbo decoding with tail-biting trellises," in *1998 URSI International Symposium on Signals, Systems, and Electronics. Conference Proceedings (Cat. No.98EX167)*, pp. 343–348, Oct 1998.

[85] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (corresp.)," *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, Mar 1974.

[86] S. Riedel, "Map decoding of convolutional codes using reciprocal dual codes," *IEEE Transactions on Information Theory*, vol. 44, pp. 1176–1187, May 1998.

[87] W. Sung, "Minimum decoding trellis lengths for tail-biting convolutional codes," *Electronics Letters*, vol. 36, pp. 643–645, Mar 2000.

[88] O. Iscan, D. Lentner, and W. Xu, "A comparison of channel coding schemes for 5g short message transmission," in *2016 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, Dec 2016.

[89] J. Berkmann, "On turbo decoding of nonbinary codes," *IEEE Communications Letters*, vol. 2, pp. 94–96, April 1998.