DIPLOMARBEIT

SOFT-INFORMATION-BASED NETWORK CODING FOR WIRELESS RELAY NETWORKS

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Diplomingenieurs

unter der Leitung von
Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Gerald Matz
Univ.Ass. Dipl.-Ing. Clemens Novak
Projektass. Dipl.-Ing. Stefan Schwandter
Institut für Nachrichtentechnik und Hochfrequenztechnik

eingereicht an der Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik

von
Andreas Winkelbauer
Landstraßer Gürtel 21/14
1030 Wien

Wien, im Juni 2010 ______________________

—— To my parents ——

Abstract

This thesis deals with network coding for wireless relay networks. Network coding is a technique which, in contrast to routing, allows intermediate network nodes to combine the data received from multiple sources for subsequent transmission. It has been shown that network coding is necessary to achieve capacity in networks with multicast transmission. Moreover, it is known that the separation of network coding and channel coding is suboptimal in general. It is therefore advantageous to perform joint network-channel coding at the physical layer. This applies in particular to wireless networks, where the individual network nodes are connected to each other by error-prone links.

We first give a brief introduction to network coding and discuss the state of the art in channel coding and relay-based transmission. Then, we present a physical layer network coding scheme for the multiple-access relay channel (multiple sources assisted by a single relay). Contrary to existing schemes, our method builds on forwarding of soft information about the network-coded bits. This has the advantage that error-free decoding at the relay is not required. Subsequently, the processing at the relay and at the destination, which employs an iterative joint network-channel decoder, is described in full detail. Furthermore, the issue of log-likelihood ratio (LLR) quantization is discussed. We show that proper LLR quantization is vital for the performance of the proposed system. Finally, numerical simulation results for different transmission settings are presented. A comparison with reference systems shows the gains achieved with our physical layer network coding scheme.

Kurzfassung

Diese Diplomarbeit beschäftigt sich mit Netzkodierung für drahtlose Relaisnetze. Netzkodierung ist eine Technik die es den Netzknoten, im Gegensatz zu Routing, erlaubt, Daten von mehreren Quellen für die darauffolgende Übertragung zu kombinieren. Es wurde gezeigt, dass Netzkodierung notwendig ist um die Kapazität in Netzen mit Multicast-Übertragung zu erreichen. Darüber hinaus ist bekannt, dass die Trennung von Netzkodierung und Kanalkodierung im Allgemeinen suboptimal ist. Es ist daher vorteilhaft, Netz- und Kanalkodierung gemeinsam auf der physikalischen Schicht durchzuführen. Das trifft insbesondere auf drahtlose Netze zu, in denen die einzelnen Netzknoten durch fehleranfällige Übertragungsstrecken miteinander verbunden sind.

Zuerst geben wir eine kurze Einführung in die Netzkodierung und behandeln den Stand der Technik bezüglich Kanalkodierung und relaisbasierter Übertragung. Danach präsentieren wir ein Übertragungsschema mit Netzkodierung auf der physikalischen Schicht für den Mehrfachzugriffsrelaiskanal. Im Gegensatz zu bestehenden Übertragungsverfahren basiert unsere Methode auf der Weiterleitung von „soft“ Information über die netzkodierten Daten. Dies hat den Vorteil, dass eine fehlerfreie Dekodierung am Relaisknoten nicht erforderlich ist. Anschließend wird die Signalverarbeitung am Relais- und am Empfangsknoten, der einen iterativen Netz- und Kanaldekoder einsetzt, ausführlich beschrieben. Zudem wird die Quantisierung von „log-likelihood ratios“ (LLRs) besprochen. Wir zeigen, dass eine korrekte LLR Quantisierung entscheidend für die Leistungsfähigkeit des vorgeschlagenen Systems ist. Zum Abschluss präsentieren wir numerische Simulationsergebnisse für unterschiedliche Übertragungsparameter. Ein Vergleich mit Referenzsystemen zeigt die Gewinne, die durch unser Übertragungsverfahren mit Netzkodierung auf der physikalischen Schicht erzielt werden.

Acknowledgements

I would like to express my gratitude to a number of people who supported me in a variety of ways. Without their help my work on this thesis would not have been possible.

First, I want to thank Gerald Matz for giving me the opportunity to work on this thesis and for being an excellent supervisor. His constant advice and interest in my thesis are sincerely appreciated.

I am indebted to my supervisors Clemens Novak and Stefan Schwandter for countless enlightening discussions, suggestions and comments. Their careful proofreading has improved this thesis significantly.

I want to thank Prof. Norbert Görtz for sharing his profound expertise in source coding with me.

Very special thanks go to my love, Veronika. Her support and encouragement has been essential for the progress of my work. With her cheerful disposition she can make me feel happy and comfortable in the wink of an eye. Only she knows how much I am indebted to her.

Above all, I want to express my deep gratitude to my parents, Barbara and Helmut. Thank you for your patience, for giving me confidence and especially for giving me the possibility to pursue my studies in my own way.

Contents

1 Introduction 1

1.1 Outline . . . 5

1.2 State of the Art . . . 6

1.2.1 Channel Coding . . . 6

1.2.2 Wireless Relay Networks . . . 12

2 Physical Layer Network Coding for the MARC 16

2.1 Introduction . . . 16

2.2 System Description . . . 17

2.2.1 System Parameters and Assumptions . . . 20

2.3 Processing at the Sources . . . 22

2.4 Processing at the Relay . . . 23

2.5 Processing at the Destination . . . 30

2.5.1 Iterative Joint Network-Channel Decoder . . . 31

2.6 Reference Systems . . . 34

3 Quantization of Soft Information 36

3.1 Introduction . . . 36

3.2 Quantizer Design for Soft Information Quantization . . . 37

3.3 The Lloyd Algorithm . . . 40

3.4 Equiprobable Output Quantizer . . . 41

3.5 The Information Bottleneck Method . . . 43

3.6 Quantizer Comparison . . . 47

4 Simulation Results 53

4.1 Introduction 53

4.2 The Symmetric Case 54

4.3 The General Case 63

5 Summary and Outlook 69

5.1 Summary 69

5.2 Outlook 70

Bibliography 72

Notation 77

List of Abbreviations 79

1

Introduction

Since the publication of the seminal paper on network information flow by Ahlswede et. al., network coding [1] has attracted increasing attention in the information theory and communications research communities. The core notion of network coding is to allow intermediate network nodes to combine the data received from multiple links for subsequent transmission.

In the above mentioned paper, the authors show that, contrary to one's intuition, it is in general not optimal to consider the information to be multicast in a network as a "fluid" which can simply be routed or replicated at the intermediate nodes. Rather, network coding has to be employed in order to achieve optimality.

Before we continue our introduction to network coding, let us first define the terms *unicast*, *multicast* and *broadcast*.

Definition 1.1. *A* unicast *denotes the transmission of information from one source to one sink. The term* multicast *is used for transmissions from a single source to multiple sinks. Finally, a transmission from one source to all available sinks is referred to as* broadcast*. In the multicast and broadcast case, all receiving sinks want to obtain the* same *information.*

It is well known that the maximum information flow in networks with one source and a single sink is limited by the weakest set of links which completely cut the source from the sink. This cut can be seen as a bottleneck for the information flowing from the source to the sink. The max-flow min-cut theorem [2] provides a way to calculate the maximum information flow in any (single source, single sink) network. In this case the maximum flow can be achieved by routing. Note that routing is in fact a special case of network coding.

However, in the case of a network with multiple sinks, i.e., for a multicast transmission, it is not obvious how to determine the maximum information flow. In [1] it has been shown that in the multicast case the max-flow min-cut interpretation is still valid. Moreover, network coding is mandatory in order to achieve the maximum information flow. This major result of [1] is summarized in the following theorem.

Theorem 1.1. *Consider a network with K sources, each transmitting at a certain rate, and N sinks, where each sink wants to decode all sources. If the source rates are such that, without network coding, the network can support each sink in isolation (i.e. each sink can decode all sources in case it is the only sink in the network), then there exists a (linear) network code such that the network can support all sinks* simultaneously*.*

In the following, we will describe the basic principle of network coding in terms of a simple example.

Example 1.1. Let us consider a network with one source and two sinks as it is depicted in Figure 1.1. This network is sometimes referred to as "butterfly network" [1]. Here, we wish to multicast the bits u_1 and u_2 from the source node A to the sink nodes F and G. Each point-to-point link in the network is assumed to be error-free with a capacity of 1 bit per channel use.

A routing solution to this problem is shown in Figure 1.2(a). Note that the link from node D to node E has to be used twice in this case.

The aforementioned bottleneck can be circumvented if we allow network coding, see Figure 1.2(b). Here, node D combines the incoming bits and transmits the bit $u_3 = u_1 \oplus u_2$. Each sink receives u_3 and one of the bits u_1, u_2. The remaining bit can be recovered by reversing the encoding operation performed at node D, i.e., node F obtains $u_2 = u_3 \oplus u_1$ and, similarly, node G decodes $u_1 = u_3 \oplus u_2$. In this example, one bit less has to be transmitted and we can therefore save one channel use.

In addition, we can also observe that in this example, network coding will increase the throughput. Considering the transmission of an (infinite) stream of bits u_1 and u_2, one can see that with network coding both sinks can simultaneously receive 2 bits per

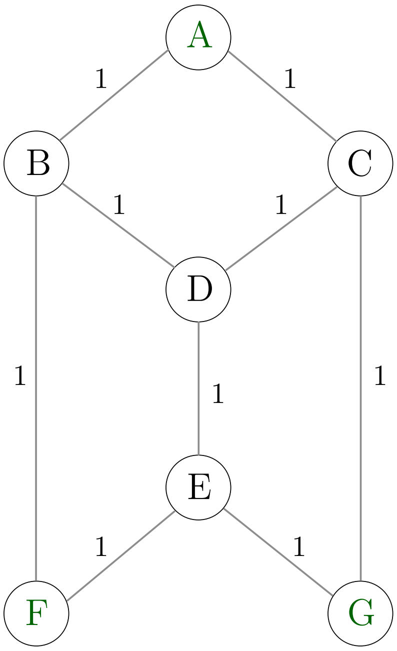
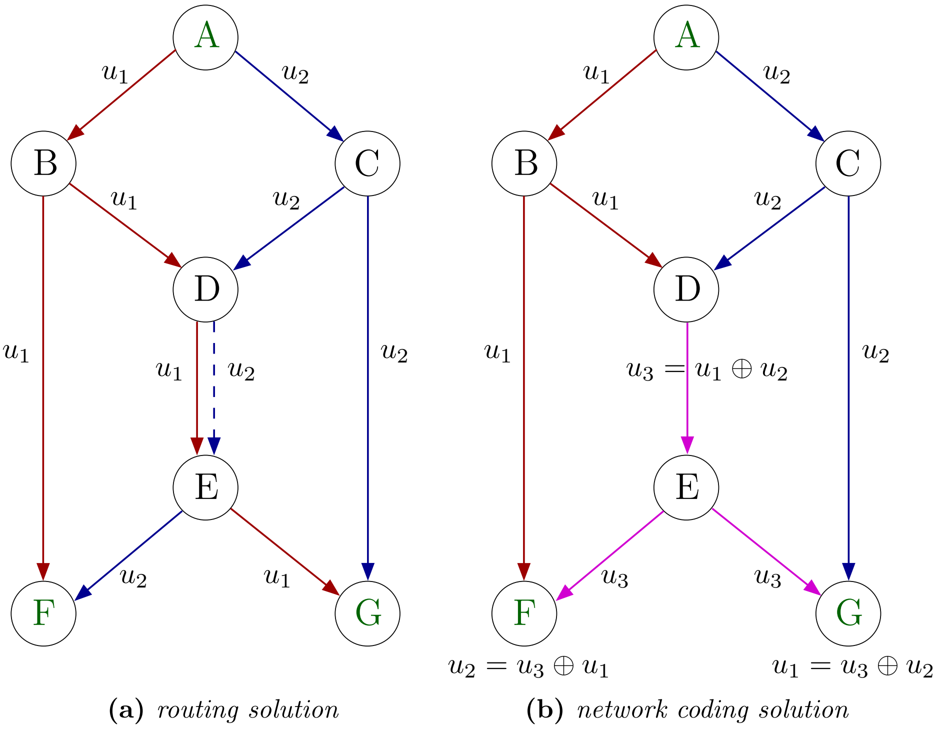


Figure 1.1: *Error-free network with one source node and two sink nodes. Each point-to-point link has a capacity of 1 bit per channel use.*



(a) *routing solution* **(b)** *network coding solution*

Figure 1.2: *Transmission of two bits, u_1 and u_2, from node A to the nodes F and G. Note that in (a) we cannot transmit u_1 and u_2 simultaneously from node D to E.*

time instant. On the other hand, if we do not allow coding at the network nodes, the sink nodes will, on average, receive only 1.5 bits per time instant.

This example shows that even a very simple network code can provide significant gains.

- A *throughput of 2 bits is achievable.* This is a gain of 33.3 % compared to the 1.5 bits obtained without network coding.
- We *save 10 % in channel uses* compared to the routing solution, because we need to transmit one bit less.

□

Although Theorem 1.1 is stated in terms of multicast flows, network coding can be beneficial also for other traffic types such as unicast flows transmitted by multiple sources. Furthermore it is important to note that the authors of [1] considered error-free networks only. In an error-free network, each link can transport information at a given, deterministic rate without introducing any residual decoding errors.

An implementation of network coding could, for instance, combine messages at the packet level, i.e., on the network layer. Today, a number of applications use random network coding [3] in such settings, e.g., for large-scale file distribution systems [4], [5]. In random network coding, intermediate nodes combine incoming messages at random in a completely independent and decentralized manner. This strategy has the advantage that network nodes do not need to have any knowledge about the network topology. Moreover, the randomized approach is robust against node departures and link failures. In order to allow decoding of randomly network-coded messages, the transmitted packets have to contain an encoding vector in addition to the payload data. The encoding vector specifies the coefficients which were used to combine the messages included in the respective packet. A sink node can decode N transmitted messages as soon as it has received sufficiently many packets containing (at least) N linearly independent encoding vectors.

From a physical layer point of view, real transmission links are never completely error-free due to the presence of impairments like interference, signal distortion and noise. Therefore the assumption of error-free links is reasonable only if network coding is applied on top of a sophisticated physical layer, employing forward error correction (FEC) and automatic repeat request (ARQ) techniques. Due to the physical nature of wireless channels, it is, in contrast to wireline networks, practically impossible to ensure completely error-free links in the wireless domain. Hence, a joint network-channel coding approach is more promising for wireless systems. In addition it has been shown that joint network-channel coding is superior to separate network and channel coding [6].

Recently, the application of the network coding principle to wireless cooperative networks has received tremendous attention from the research community. Using network coding we can alleviate the loss in spectral efficiency, which is due to the half-duplex constraint of practical wireless systems. Furthermore, in a wireless scenario, the broadcast nature of radio waves can be exploited to increase power efficiency.

The term “physical layer network coding” (PLNC) has been coined for a set of techniques combining channel coding and network coding in various relay-based communication scenarios, such as two-way communication, multiple access, multicasting, and broadcasting. In this thesis we use the terms “physical layer network coding” and “joint network-channel coding” (JNCC) interchangeably. PLNC is also referred to as “algebraic code superposition” [7] by some authors.

Relay-based cooperative networks are now also being considered by standardization organizations. The 3rd Generation Partnership Project (3GPP), for example, considers the introduction of relays for users located at the edge of a cell in order to increase throughput and coverage area [8].

The main goals of this thesis are to study PLNC at relays and to analyze the gains PLNC can provide in wireless relay networks.

1.1 Outline

This thesis investigates PLNC in wireless relay networks and provides an extensive analysis of a PLNC scheme for the multiple-access relay channel (MARC). Moreover, we discuss quantization of soft information which turns out to be an important issue for certain relay-based transmission schemes.

The text at hand is structured as follows.

- The remainder of this chapter describes the state of the art in channel coding and relay-based transmission schemes.

- The starting point of **Chapter 2** is the physical layer network coding scheme presented in [9]. We improve and extend this scheme in several directions. A detailed description of the processing at each node, with a focus on the iterative decoder at the destination, is given. Furthermore, reference systems, which will be used for performance comparison, are introduced.

- In **Chapter 3** we discuss the important issue of soft information quantization. A comparison between different quantizer design methods is given. From this comparison it can be seen that proper log-likelihood ratio (LLR) quantization is vital for the system performance.
- In **Chapter 4** numerical simulation results of our PLNC scheme are presented. We show the BER performance of our PLNC scheme for a wide range of scenarios and system parameters. Furthermore, a comparison with reference systems shows the gains that are obtained using our PLNC scheme.
- Finally, **Chapter 5** concludes this thesis with a summary and proposes several directions for future research.

1.2 State of the Art

1.2.1 Channel Coding

Channel Capacity: The promise of channel coding. In Shannon's groundbreaking 1948 paper [10], it has been shown that reliable communication over a noisy channel is possible, with an *arbitrarily small* error probability, as long as the rate R does not exceed the channel capacity C. Furthermore, error-free communication cannot be achieved if $R > C$. In this case the probability of a decoding error is bounded away from zero.

Channel coding introduces redundancy in a controlled manner, in order to protect against transmission errors. Random codes with large blocklength are used in the proof of the channel capacity theorem. Unfortunately, such codes are not practical because they do not provide any structure to allow encoding and decoding with feasible computational complexity. Therefore, all practically relevant channel codes allow for efficient encoding and decoding due to their inherent structure.

Before we describe some important types of channel codes, we will introduce the notion of *soft information*.

Soft Information and log-likelihood ratios. Let us consider the transmission of data over a noisy channel with the input-output relation $\mathbf{y} = \mathbf{x} + \mathbf{w}$, where $\mathbf{x}$ is the transmitted signal and $\mathbf{w}$ is additive noise, statistically independent of $\mathbf{x}$ and distributed according to some known pdf. Over this channel we want to transmit a block of information bits $\mathbf{u}$, which we encode using a linear channel code with generator matrix $\mathbf{G}$. Let $\varphi(\cdot)$ denote the modulator mapping, i.e., the mapping from the code bits to the transmitted signal. Then, we obtain $\mathbf{y} = \varphi(\mathbf{Gu}) + \mathbf{w} = \varphi(\mathbf{c}) + \mathbf{w}$ for the received signal, where $\mathbf{c}$ is a vector of code bits.

At the receiver, the signal $\mathbf{y}$ is first processed by a demodulator, whose output is then fed into the channel decoder. It is well known that, in a coded system, the receiver should use soft decisions instead of hard decisions in order to decrease the probability of residual decoding errors [11].

In contrast to hard decisions, soft decisions also reveal the *reliability* associated to the respective observation. The a posteriori probability (APP) $\mathrm{P}\{c_i = 0 \,|\, \mathbf{y}\}$ is a proper measure of the reliability information about the code bit c_i. There is, of course, $\mathrm{P}\{c_i = 1 \,|\, \mathbf{y}\} = 1 - \mathrm{P}\{c_i = 0 \,|\, \mathbf{y}\}$. Often, it is more convenient to work with log-likelihood ratios (LLRs) instead of bit probabilities. The a posteriori LLR of c_i, given the observation $\mathbf{y}$, is defined as[1]

$$\Lambda(c_i \,|\, \mathbf{y}) = \log \frac{\mathrm{P}\{c_i = 1 \,|\, \mathbf{y}\}}{\mathrm{P}\{c_i = 0 \,|\, \mathbf{y}\}}. \tag{1.1}$$

The relation between a LLR $\Lambda(c_i \,|\, \mathbf{y})$ and the hard decision $\hat{c}_i$ is[2]

$$\hat{c}_i = \begin{cases} 0, & \Lambda(c_i \,|\, \mathbf{y}) < 0 \\ 1, & \Lambda(c_i \,|\, \mathbf{y}) \geq 0 \end{cases} , \tag{1.2}$$

i.e., the sign of $\Lambda(c_i \,|\, \mathbf{y})$ contains the information about the value of the bit. The magnitude of the LLR expresses the reliability of the associated hard decision. A large (small) value $|\Lambda(c_i)|$ corresponds to more (less) certainty about the value of the bit c_i.

The representation of soft information by LLRs is fully equivalent to using a posteriori bit probabilities. Given some LLR value $\Lambda(c_i)$, we obtain

$$p(b) = \frac{e^{b\Lambda(c_i)}}{1 + e^{\Lambda(c_i)}}, \tag{1.3}$$

where $p(b) = \mathrm{P}\{c_i = b \,|\, \mathbf{y}\}$ and $b \in \{0, 1\}$.

For two statistically independent bits u_1 and u_2, one can readily see that

$$\Lambda(u_1 \oplus u_2) = \log \frac{e^{\Lambda(u_1)} + e^{\Lambda(u_2)}}{1 + e^{\Lambda(u_1) + \Lambda(u_2)}} = -2 \operatorname{atanh}\left(\tanh\left(\frac{\Lambda(u_1)}{2}\right) \tanh\left(\frac{\Lambda(u_2)}{2}\right) \right). \tag{1.4}$$

[1] With a slight abuse of notation, the conditioning will be suppressed in case it is clear from the context that Λ denotes an a posteriori LLR. Also note that some authors define LLRs such that the sign is flipped compared to our definition.

[2] In case the pdf of $\Lambda(c_i \,|\, \mathbf{y})$, $f_\Lambda(\xi)$, is continuous at $\xi = 0$ (which is usually true) it does not matter if we attribute $\Lambda = 0$ to $\hat{c}_i = 0$ or to $\hat{c}_i = 1$.

To simplify notation, we will usually express the above calculation in terms of the "boxplus" operator "⊞" as

$$\Lambda(u_1) \boxplus \Lambda(u_2) \triangleq \Lambda(u_1 \oplus u_2), \tag{1.5}$$

i.e., the boxplus in the LLR domain is analogous to the modulo-2 addition in the bit domain. Note that the pair $(\mathbb{R}, \boxplus)$ forms an abelian monoid. A further introduction to LLR algebra is given in [12].

Convolutional Codes. In contrast to block codes, convolutional codes are not restricted to finite-length data blocks that are encoded independently of each other. Rather, a stream of information bits u_k is mapped to a stream of code bits c_n.

Encoders for convolutional codes have an internal frame structure. The sequence u_k is split up into data frames of length K each. Then, the sequence of data frames is mapped to a sequence of code frames of length $N \geq K$ each. The encoding procedure is not memoryless. Each code frame depends on the current data frame and the past m data frames. Here, m denotes the *frame memory order* defined as

$$m = \max_{j=0,\ldots,K-1} m_j, \tag{1.6}$$

where m_j is the length of the jth shift register. Therefore, the frame memory order is equal to the encoder's maximum shift register length. For brevity we will henceforth denote the frame memory order m as *encoder memory*.

In most communication systems, data is usually organized in blocks of a certain length. In order to convert a convolutional code into a linear block code, two standard modifications, known as termination and truncation, can be applied. We then obtain a code rate R of (roughly) K/N, assuming the blocklength is large compared to the encoder memory m.

Whereas for block codes, a large blocklength is required for good performance, the performance of convolutional codes is mainly determined by the encoder memory m. Practical convolutional codes show reasonable performance, also for relatively short blocklength (after termination or truncation). However, they are far away from attaining the Shannon limit, i.e., channel capacity.

Linear shift register circuits provide an efficient way of implementing encoders for convolutional codes. Encoders for non-systematic convolutional codes (NSCCs) are usually implemented by feedforward shift register circuits, whereas shift register circuits with feedback are commonly used to implement encoders for recursive systematic

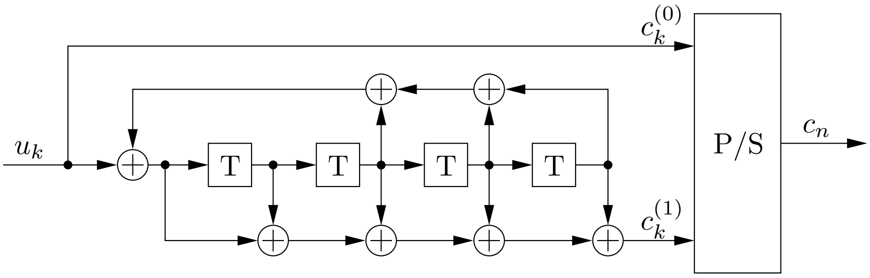


Figure 1.3: *Shift register implementation of a rate 1/2 recursive systematic convolutional code with generator polynomials $(23, 37)_8$.*

convolutional codes (RSCCs).

Figure 1.3 shows an encoder for an RSCC with $K = 1$, $N = 2$ and generator polynomials $(23, 37)_8$, where the first generator polynomial, 23_8, corresponds to the feedback part of the encoder. The generator polynomials are given in octal notation and they specify which outputs of the shift register taps are used to form the respective code sequence $c_k^{(i)}$. The block labeled “P/S” denotes a parallel to serial converter, which interleaves the sequences $c_k^{(0)}$ and $c_k^{(1)}$ to produce a single output sequence c_n.

There are numerous algorithms for decoding convolutional codes. The Viterbi algorithm [13] is based on the trellis representation of a convolutional code and performs (approximate) maximum likelihood sequence decoding. There is also a soft-output variant of the Viterbi algorithm (SOVA) [14]. The BCJR algorithm [15] also belongs to the class of graph-searching decoding techniques. However, in contrast to the Viterbi algorithm, the BCJR algorithm performs (soft-output) maximum a posteriori *symbol* decoding and thus minimizes the symbol error probability. Unfortunately, this comes at the price of a higher computational complexity, compared to the Viterbi algorithm. The computational complexity of the BCJR algorithm grows exponentially with the encoder memory m and linearly with the blocklength.

Other decoding techniques, known as sequential decoding, are well suited for decoding convolutional codes with large encoder memory m (corresponding to a large number of trellis states). Examples for sequential decoding techniques are the Fano algorithm and the stack algorithm [16].

Parallel Concatenated Convolutional Codes (PCCCs). PCCCs, better known as *turbo codes* [17], have revolutionized channel coding since their introduction in 1993. Turbo codes were the first practical channel codes that enabled reliable transmission over the AWGN channel at bit rates close to the channel capacity.

The encoder of a PCCC, depicted in Figure 1.4, consists of two constituent RSCCs

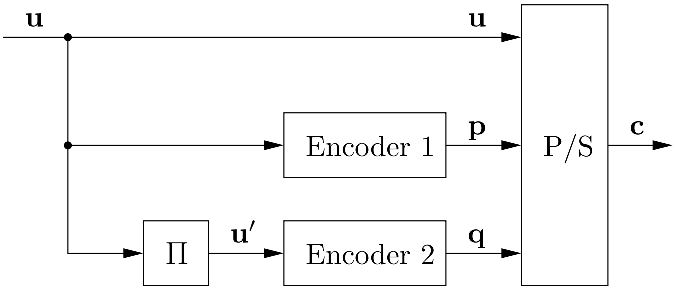


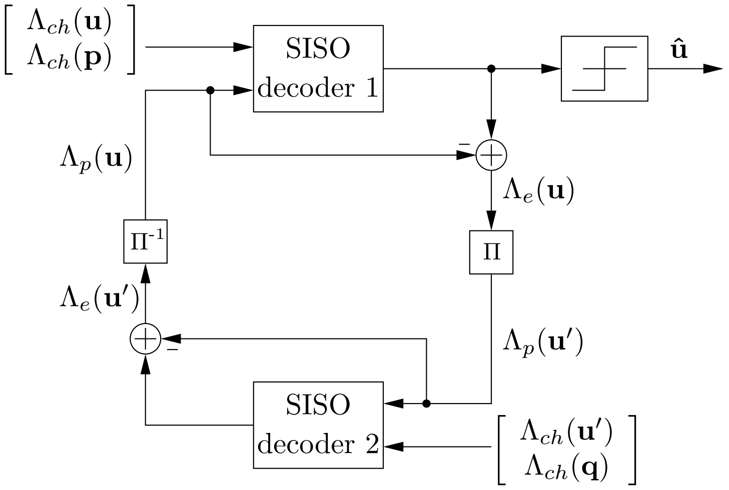
Figure 1.4: *Encoder for a PCCC with two constituent codes.*

and an interleaver Π. In many cases the two constituent codes are identical, but this is not a requirement. The second constituent encoder processes the interleaved information bits, $\mathbf{u}' = \Pi(\mathbf{u})$. Both constituent encoders output only the parity bits, i.e., the systematic bits are punctured from the output of the constituent encoders. The interleaver is particularly important for the performance of the code. Its aim is to rearrange the input bits in a pseudo-random manner. This is achieved in principle by choosing a large interleaver depth, but by doing so the encoding delay is increased, which might be undesirable in some applications. Interleaver design for turbo codes is discussed in [18].

PCCCs were termed turbo codes in [17], because the proposed iterative decoding method feeds back soft information analogous to the reuse of exhaust gas in turbo engines. Iterative decoding of turbo codes is attractive, because it attains ML decoding performance with reasonable computational complexity.

An iterative turbo decoder is shown in Figure 1.5. Two SISO decoders are used to decode the constituent codes. Each SISO decoder receives soft information about its code bits from the demodulator, denoted by the vectors $[\Lambda_{ch}(\mathbf{u})\,\Lambda_{ch}(\mathbf{p})]^T$ and $[\Lambda_{ch}(\mathbf{u}')\,\Lambda_{ch}(\mathbf{q})]^T$ respectively. Furthermore both decoders exchange a priori information about the information bits. The extrinsic LLRs $\Lambda_e(\mathbf{u})$ are computed by subtracting the prior LLRs $\Lambda_p(\mathbf{u})$ from the posterior LLRs at the output of the first constituent decoder. This ensures that only information which was newly gained in the previous decoding step is forwarded to the second constituent decoder. The prior LLRs for SISO decoder 2, $\Lambda_p(\mathbf{u}')$, are obtained by interleaving the extrinsic LLRs of SISO decoder 1, $\Lambda_e(\mathbf{u})$. In this manner the SISO decoders exchange information for a certain number of iterations, until eventually a hard decision is performed on the posterior LLRs at the output of one of the constituent decoders.

A detailed description of the concept of turbo coding can be found in [19].



$\begin{bmatrix} \Lambda_{ch}(\mathbf{u}) \\ \Lambda_{ch}(\mathbf{p}) \end{bmatrix}$
SISO decoder 1
$\hat{\mathbf{u}}$
$\Lambda_p(\mathbf{u})$
$-$
$\Lambda_e(\mathbf{u})$
Π^{-1}
Π
$\Lambda_e(\mathbf{u}')$
$-$
$\Lambda_p(\mathbf{u}')$
SISO decoder 2
$\begin{bmatrix} \Lambda_{ch}(\mathbf{u}') \\ \Lambda_{ch}(\mathbf{q}) \end{bmatrix}$

Figure 1.5: *An iterative turbo decoder. Information is passed in the form of log-likelihood ratios for single bits being grouped into vectors.*

Low Density Parity Check (LDPC) Codes. After the invention of PCCCs in 1992 other code constructions have been found which, again combined with iterative decoding techniques, also allow reliable transmission close to the Shannon limit. One of these code constructions are LDPC codes, which were introduced by Gallager as early as in 1962 [20]. Soon after their invention, they were largely forgotten, and "reinvented" in the mid-1990's [21], [22], [23]. In the following, we will consider only binary LDPC codes for the sake of simplicity. However, LDPC codes can be generalized to nonbinary alphabets.

An LDPC code is a linear block code given by the nullspace of an $m \times n$ parity-check matrix $\mathbf{H}$ with a low density of 1's. This sparsity property makes LDPC codes amenable to various iterative decoding algorithms with near-optimal performance. An LDPC code is called *regular* if $\mathbf{H}$ has a constant column weight g and a row weight r, where $r = g(n/m)$ and $g \ll m$. The code rate of a regular LDPC code is bounded as $R \geq 1 - m/n = 1 - g/r$, with equality when $\mathbf{H}$ is full rank. If $\mathbf{H}$ is low density, but the number of 1's in each column or row is not constant, then the LDPC code is called *irregular*. The term "low density" is rather vague and cannot be precisely quantified, although a density of 1 % or lower, i.e., 1 % or fewer entries of $\mathbf{H}$ are 1's, can be called low density [24].

In addition to the matrix representation, LDPC codes can equivalently be represented by factor graphs. A factor graph is a bipartite graph with two different kinds of nodes. The two types of nodes are called *variable nodes* and *check nodes*. A variable node corresponds to a code bit and a check node corresponds to a check equation. The factor graph of an LDPC code will usually contain cycles. By running loopy belief propagation on the factor graph [25] one can obtain approximate posterior LLRs of the code bits, given a noisy observation.

Since short cycles can significantly degrade the performance of the iterative decoding algorithm, LDPC codes should be designed such that there are as few short cycles as possible. Moreover, cycles may contribute to the error floor of LDPC codes. For these reasons code design for LDPC codes is an important research topic.

A comprehensive treatment of LDPC codes can be found in [24].

1.2.2 Wireless Relay Networks

The increasing demand for high data rates in wireless networks calls for new transmission strategies. As already mentioned previously, relay-based transmission schemes are considered as a viable option for future wireless systems.

In the following we state some advantages of wireless relaying, compared to point-to-point communication.

- Two signal observations, one coming from the source and one from the relay, are available at the destination. Both, small-scale fading due to multipath propagation and large-scale fading due to shadowing can be assumed independent for the source-destination and the relay-destination link. Thus, relaying allows us to gain spatial diversity, increasing the robustness of the overall communication system.

- Assuming the relay is positioned in a clever way, it is very likely that the link from the source to the relay is of better quality than the direct link between source and destination. In this case, relaying promises higher data rates than simple point-to-point communication. Moreover, the power efficiency of the transmission is increased, i.e., the source can save transmission energy which is especially beneficial for mobile devices. In cellular systems one can also expect increased coverage area due to the use of relays.

- The probability of a line-of-sight connection is increased by the use of a relay. This is especially true for the link between a fixed relay and a fixed destination. This allows the use of spectrum, on the relay-destination link, which is currently not used due to its vulnerability against non-line-of-sight conditions.

An obstacle which is common to all relay-based transmission schemes is the fact that, currently, practical RF implementations of wireless relays impose a half-duplex transmission constraint. This reduces the spectral efficiency of the overall system. Furthermore, depending on the network topology, the half-duplex constraint may also pre-

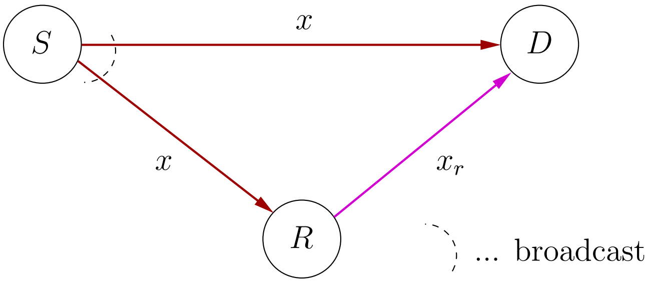


Figure 1.6: *The relay channel.*

vent simultaneous multiple-access and therefore restrict transmissions to time division multiple-access (TDMA) or frequency division multiple-access (FDMA) schemes.

Transmission schemes for the relay channel, shown in Figure 1.6, provide spatial diversity by introducing cooperation between the source S and the relay R. Furthermore, it is natural to exploit the broadcast nature of wireless transmissions, i.e., the relay will overhear the transmission from the source S to the destination D.

Depending on the processing that is performed at the relay, we distinguish between different forwarding strategies. In the following, three commonly considered forwarding strategies are briefly described.

- **Amplify-and-Forward (AF).** In the AF case the relay recovers the equivalent complex baseband samples from the received signal, amplifies them and subsequently retransmits the new, amplified signal. This method is simple, but it has the drawback of noise enhancement on the relay-destination link.

- **Decode-and-Forward (DF).** In the DF case the relay demodulates and decodes the received signal. If the relay was able to decode without error, indicated, e.g., by a cyclic redundancy check (CRC), it forwards a new codeword to the destination. In the simplest case the forwarded codeword is equal to the received codeword or a punctured version of it. Thereby the relay provides additional redundancy to the decoder at the destination. Thus, the code rate at the source can be increased, because successful decoding using solely the codeword transmitted by the source is not necessary. However, the relay could also forward a different codeword, e.g., by interleaving and re-encoding the received information bits. This concept is referred to as *distributed turbo coding* [26]. In any case the channel code is *distributed* between the source and the relay.

  If the CRC at the relay indicates non-successful decoding, the relay has the following options.

  - The relay ignores the CRC indication and re-encodes the wrongly decoded information bits anyway. The drawback of this approach is *error propagation.*
  - In order to prevent error propagation, the relay remains silent and does not transmit at all. In doing so, any information available at the relay about the signal transmitted by the source is discarded.
  - The DF scheme could be extended to *soft* decode-and-forward. In this case, the relay is not required to decode the message from the source correctly. Here, the relay would either forward soft information obtained by a SISO decoder or it would use an interleaver and a SISO *en*coder to perform distributed turbo coding with soft information relaying [27], [28].
  - If there is a feedback channel from the relay to the source, the relay could indicate to the source that decoding was not successful. Then, the source would transmit additional code bits in order to increase the probability of successful decoding at the destination.

  Note that the DF scheme works well when the probability of a decoding error at the relay is small, i.e., when the relay is close to the source.

- **Compress-and-Forward (CF).** Instead of decoding the information transmitted by the source, the relay can also help the destination by forwarding a compressed version of its received signal. Compared to DF, compress-and-forward features a lower computational complexity, because the relay does not need to decode the received signal. Furthermore, CF is known to outperform DF when the relay is close to the destination [29].

It is important to note that even for the relay channel with one source, one relay and one destination, as it is depicted in Figure 1.6, the capacity is still unknown, except for special cases discussed, e.g., in [30].

In conjunction with network coding, we consider wireless relay networks with more than one source and/or sink. In Chapter 2 a PLNC scheme for the multiple-access relay channel (MARC) will be presented. The MARC, depicted in Figure 1.7, consists of two source nodes, S_1 and S_2, one relay R and a destination node D. Here, the relay supports both sources in their transmission to the destination.

A general model of the MARC is given in [31]. Capacity bounds for the general MARC, where the sources and the relay are allowed to transmit simultaneously, can be found in [29].

Another wireless relay network which has recently been considered for physical layer

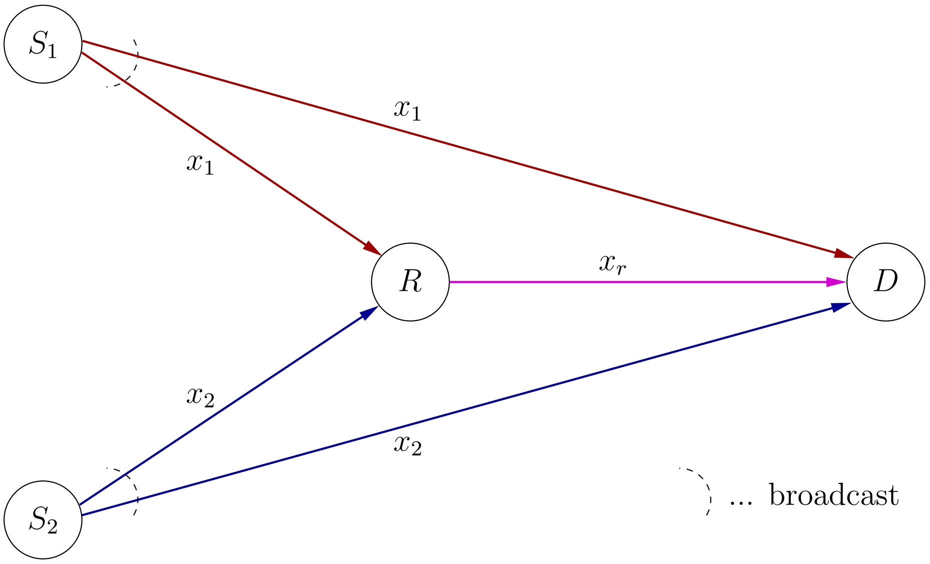


Figure 1.7: *The multiple-access relay channel.*

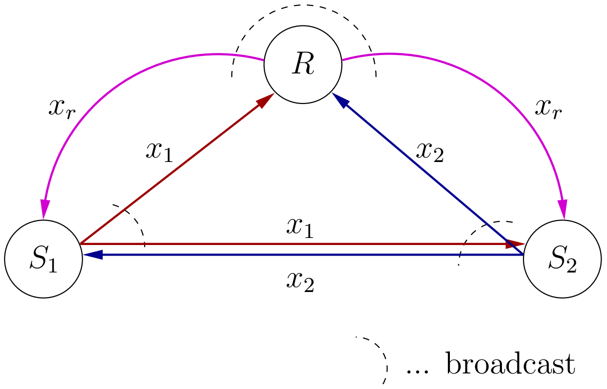


Figure 1.8: *The two-way relay channel.*

network coding is the two-way relay channel (TWRC), shown in Figure 1.8. Here, two sources, supported by one relay, want to exchange data in both directions.

The achievable rates for the TWRC simultaneous multiple-access and broadcast transmissions of all nodes are given in [32]. A PLNC scheme for the TWRC can be found in [33].

2

Physical Layer Network Coding for the Multiple-Access Relay Channel

2.1 Introduction

In this chapter we present a physical layer network coding (PLNC) scheme for the multiple-access relay channel (MARC). The MARC, shown in Figure 1.7, is a wireless network with two sources, S_1 and S_2, which transmit data, with the help of a single relay R, to one common destination node D. Note that the MARC does not consider direct cooperation between the two sources, instead the sources cooperate with the relay.

Compared to the general MARC model [31], we assume orthogonal channels for the transmission of x_1, x_2 and x_r (cf. Figure 1.7). This assumption is suboptimal, but of great practical relevance, since it allows the nodes to operate in a half-duplex mode without the need for synchronization on a symbol or even carrier phase level.

In our PLNC scheme we treat network coding and channel coding jointly. Separate network and channel coding is possible, but it leads to worse performance when compared to joint network-channel coding. A comparison between separate network and channel coding and joint network-channel coding for the MARC is given in [34]. The joint network-channel coding approach we follow allows us to exploit the redundancy in the network code to support the channel code for improved performance. This is similar to joint source-channel coding where the remaining redundancy after the source encoder aids the channel code to combat signal corruption. In joint network-channel coding, a channel code is distributed in the network to include several links. Therefore, at the destination node, the network and channel codes are decoded jointly by passing soft information between the network decoder and the channel decoders. This is a major difference in comparison to separate network and channel coding, where channel coding is performed locally and separately for each link.

In this chapter we describe the proposed system and explain the processing performed at the nodes. Furthermore, we also introduce reference schemes, which allow to assess the performance of our PLNC scheme. Numerical simulation results, providing a comparison between these schemes for different system parameters, are then presented in Chapter 4.

2.2 System Description

The system under consideration in this chapter is based on the decode-and-forward (DF) strategy and uses two channel decoders at the relay in order to decode the messages transmitted by the two sources. This is similar to previously proposed network coding schemes for the MARC, which would then interleave, network encode and channel encode the decoded bits at the relay and subsequently transmit the re-encoded bits to the destination.

A fundamental problem of this strategy is that in order to avoid error propagation, the relay is required to decode both source messages correctly. The detection of residual decoding errors is usually implemented by appending a frame check sequence (FCS), generated by a cyclic redundancy check (CRC) code, to the transmitted information bits[1]. With reasonably designed CRC codes the probability of an undetected error is usually negligible. Specifically, for an (N, K) CRC-p code with $p = N - K$ bits of redundancy, the probability of an undetected error asymptotically approaches 2^{-p} (from below) as K increases [35]. Optimization of CRC codes with 24 and 32 parity bits is discussed in [36].

However, if, in such a scheme, the relay decodes one or both messages in error, it

[1] Since most current wireless systems already employ CRC codes, this does not represent additional overhead required by schemes with physical layer network coding.

has only two clearly suboptimal choices. First, the relay could remain silent in order to avoid the deteriorating effect of error propagation, thereby discarding all information obtained by the channel decoders and wasting available transmission time and energy. Second, the relay could, at the risk of severe error propagation, ignore the decoding error(s) and process the wrongly decoded data for subsequent transmission.

This dilemma is common to all relaying schemes that require error-free decoding at the relay. In such schemes, the relay would usually remain silent in case of residual decoding errors. Note that in case only one source message has been decoded in error, the system could fall back to a relay channel (cf. Figure 1.6) and one point-to-point link. This strategy would, however, require additional signaling overhead from the relay to the destination to indicate that one source message could not be decoded correctly.

In order to circumvent this unfortunate situation the authors of [9] have proposed a PLNC scheme in which the relay does not merely transmit re-encoded bits, but rather soft information about these bits. The advantage of this strategy is that all the information delivered by the *soft-output* channel decoders at the relay can be exploited, irrespective of whether the source messages are decoded correctly or not.

By forwarding LLRs about network-coded bits from the relay to the destination, the relay operation resembles message passing in belief propagation. In [9], however, the transmission of the LLRs Λ_r over the relay-destination link is tentatively modeled as an analog transmission according to

$$\widetilde{\Lambda}_r = \Lambda_r + w, \tag{2.1}$$

where w is additive white Gaussian noise with zero mean and variance σ_w^2. The SNR on the relay-destination channel is then given by

$$\rho_{rd} = \frac{\mathrm{E}\{\Lambda_r^2\}}{\sigma_w^2}. \tag{2.2}$$

Since analog LLR transmission is not suited for digital communication systems, coded transmission of the LLRs Λ_r should be considered instead of the transmission model given by (2.1).

The system we propose is based on the idea of [9], with several extensions and improvements. Figure 2.1 shows the system model of our PLNC scheme for the MARC. Before describing the processing performed at each node, we will discuss the system parameters and assumptions related to this system.

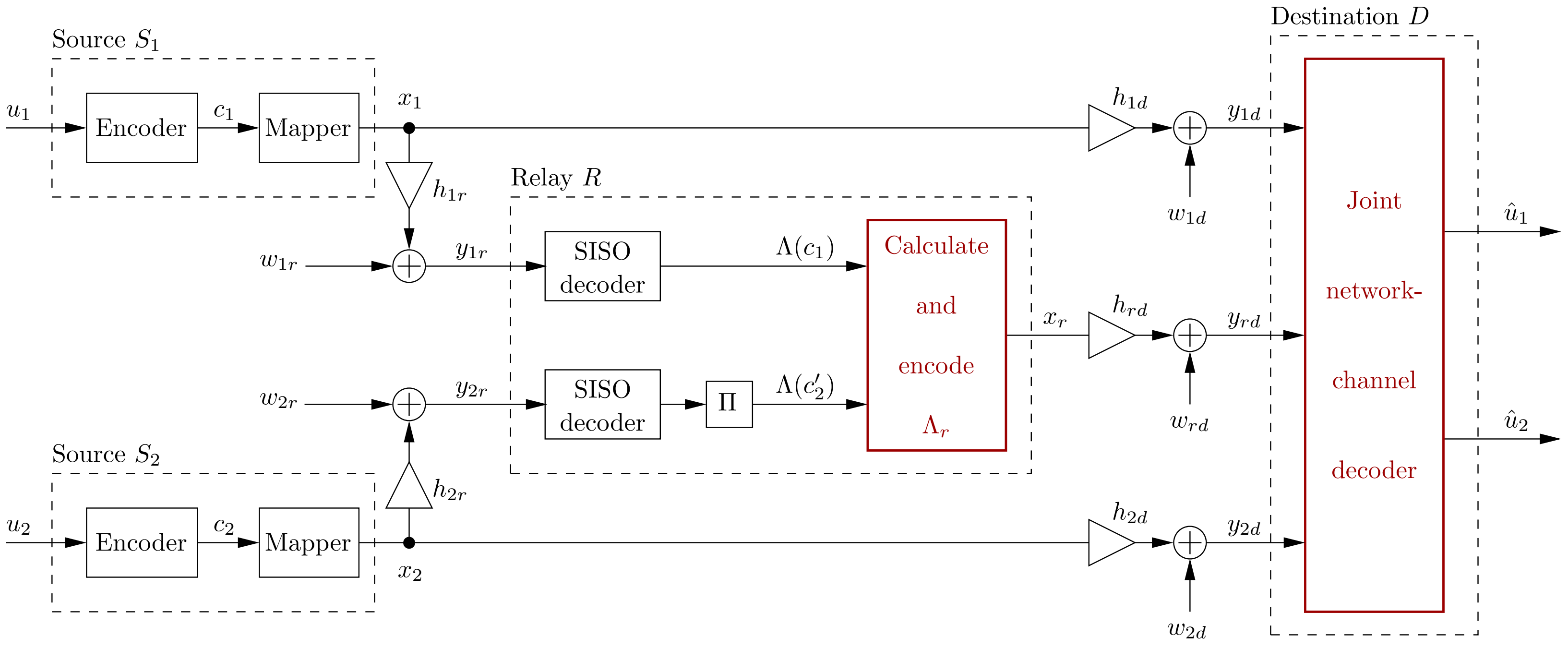


Figure 2.1: *System model of the PLNC scheme under consideration.*

2.2.1 System Parameters and Assumptions

The four nodes of the MARC are connected to each other by five non-interfering, i.e., orthogonal, transmission channels. On each channel, we assume ISI-free passband PAM transmission, modeled in the equivalent complex baseband domain on a symbol level. Moreover, we assume flat fading channels with additive white noise. Therefore the input-output relation of the individual channels is given by

$$y_{ij}[n] = h_{ij}[n]\, x_i[n] + w_{ij}[n]\,, \tag{2.3}$$

where i and j denote the transmitting and receiving node, respectively. The indices 1 and 2 are used for the sources S_1 and S_2, whereas the indices r and d are used for the relay R and the destination D.

In (2.3) the symbols transmitted by node i are denoted by $x_i[n]$, where each symbol x_i comes from the symbol alphabet $\mathcal{A}$, $x_i \in \mathcal{A}$. We assume that all $h_{ij}[n]$ are complex Gaussian random variables, $h_{ij} \sim \mathcal{CN}(0,1)$, which are independent over time, $\mathrm{E}\{h_{ij}[n]\, h^*_{ij}[m]\} = \delta[n-m]$. Therefore the magnitude of the channel coefficients $h_{ij}[n]$ is Rayleigh distributed with $\mathrm{E}\{|h_{ij}[n]|^2\} = 1$. Furthermore, all $h_{ij}[n]$ are independent and identically distributed (i.i.d.). The special case of an AWGN channel is given by $h_{ij}[n] \equiv 1$, yielding

$$y_{ij}[n] = x_i[n] + w_{ij}[n]\,. \tag{2.4}$$

A broken link corresponds to $h_{ij}[n] \equiv 0$.

The noise $w_{ij}[n]$ is i.i.d., complex Gaussian distributed, $w_{ij} \sim \mathcal{CN}(0, \sigma^2_{w_{ij}})$, and independent of h_{ij} and x_i. Thus, the instantaneous SNR γ_{ij} and the average SNR ρ_{ij} are given by

$$\gamma_{ij} = |h_{ij}|^2 \frac{P_A}{\sigma^2_{w_{ij}}} \qquad \text{and} \qquad \rho_{ij} = \mathrm{E}\{\gamma_{ij}\} = \frac{P_A}{\sigma^2_{w_{ij}}}, \tag{2.5}$$

where P_A denotes the mean power of the transmitted symbols, $P_A = \mathrm{E}\{|x_i|^2\}$. Unless noted otherwise, the term "SNR" will henceforth refer to the average SNR. We assume that each node transmits symbols $x_i[n]$ using the same Gray labeled QPSK mapping and therefore P_A is equal for all nodes, except the relay, as described later.

For reasons that will become clear in Section 2.4, we will often consider symmetric channels. In the symmetric case the average SNR on both source-relay channels and both source-destination channels is equal, meaning $\rho_{1r} = \rho_{2r}$ and $\rho_{1d} = \rho_{2d}$. To simplify notation we will then denote the source-relay SNR by ρ_{sr} and the source-destination

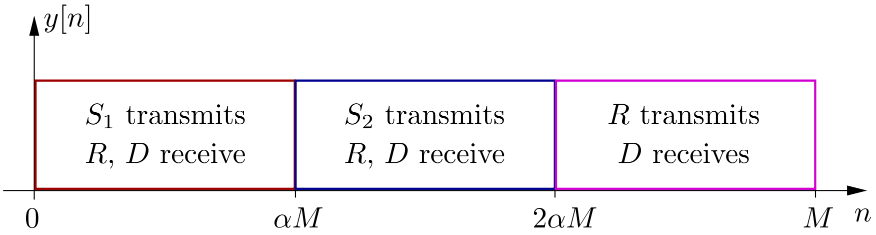


Figure 2.2: *Frame structure obtained by using TDMA with three timeslots.*

SNR will be denoted ρ_{sd}.

We assume perfect channel state information (CSI) at each receiving node about its incoming links. This means that the relay is aware of h_{1r}, h_{2r}, and the destination is aware of h_{1d}, h_{2d} and h_{rd}. It is important to stress that the relay has no CSI about the source-destination links. This side information would put the relay in a better position, but it would also require permanent transmission of CSI from the destination to the relay.

Time division multiple-access (TDMA) with three timeslots is used to achieve the aforementioned orthogonality between all transmissions in the network. In the first timeslot source 1 will broadcast its data to the relay and the destination. The second timeslot will be used in the same manner by source 2. Finally, the relay transmits to the destination in the third timeslot. Henceforth we will use the term *frame* in order to denote three consecutive timeslots. The total number of channel uses per frame is M. An equal amount of $M_s = \alpha M$ $(0 < \alpha < 1/2)$ channel uses is allocated to each source and the remaining $M_r = (1 - 2\alpha)M$ channel uses are assigned to the relay. The resulting frame structure is depicted in Figure 2.2. The value of α is closely related to the processing at the relay and to the signal constellations used by the sources and the relay. Possible values for α will be discussed in Section 2.4.

The transmit energy per frame at the relay is assumed to be fixed to E_r, independent of the actual number of channel uses M_r. In order to satisfy this energy constraint, the transmit power at the relay has to be scaled by the factor

$$\beta = \frac{E_r}{P_A M_r}, \tag{2.6}$$

which leads to an average transmit power of $\widetilde{P}_A = \beta P_A$ and to an average relay-destination SNR of $\rho_{rd} = \widetilde{P}_A / \sigma^2_{w_{rd}} = \beta P_A / \sigma^2_{w_{rd}}$.

The optimization of transmit time and/or power allocation is beyond the scope of this thesis. Furthermore we do not consider synchronization issues and assume perfect

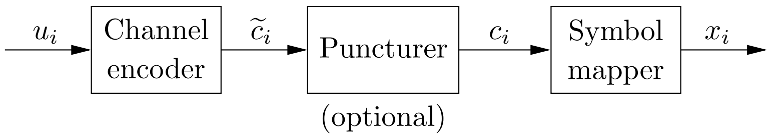


Figure 2.3: *Overview about the processing at the sources.*

synchronization, i.e., non-interfering transmissions.

2.3 Processing at the Sources

In this section the processing at the two sources is described. Figure 2.3 gives an overview about source processing.

The information bits u_i, transmitted by source i, are assumed to be statistically independent and equiprobable. Furthermore all information bits u_1 and u_2 are assumed to be mutually independent. Using a block of K_i information bits u_i as an input, the channel encoder produces a block of $\widetilde{N}_i$ code bits $\widetilde{c}_i$. The code rate $\widetilde{R}_{c,i}$ is thus given by $\widetilde{R}_{c,i} = K_i/\widetilde{N}_i$. The output of the channel encoder is then processed by a puncturer, which reduces the number of code bits to $N_i \leq \widetilde{N}_i$, yielding an overall code rate of $R_{c,i} = K_i/N_i$. In Figure 2.1 the channel encoder including the puncturer is combined in the block "Encoder".

Our PLNC scheme is very flexible with respect to the channel code type and code rate. The sources can basically use any channel code for which a soft-output decoder is available. We have used LDPC codes, turbo codes and convolutional codes with possibly different rates $R_{c,1}$ and $R_{c,2}$ as channel codes. We have also verified that the system performs well even if both sources use different types of channel codes. For simplicity we have restricted the code rates $R_{c,1}$ and $R_{c,2}$ such that both encoders (including puncturing) produce the same number of code bits, i.e., $N = N_1 = N_2$. While this assumption slightly simplifies the processing at the relay and at the destination, it does not impose a significant restriction. A way how to lift this restriction is described in Section 2.4.

It is important to note that channel codes which work well on point-to-point links are not necessarily well suited for the MARC with PLNC. Code design for the MARC is nevertheless beyond the scope of this thesis. For the MARC it can be shown that the largest code rate yielding a diversity order of 2, i.e., full diversity, is given by $R_{c,max} = 2/3$. A design strategy for LDPC codes which guarantees a diversity order of 2 at a code rate $R_c = 2/3$ is presented in [37].

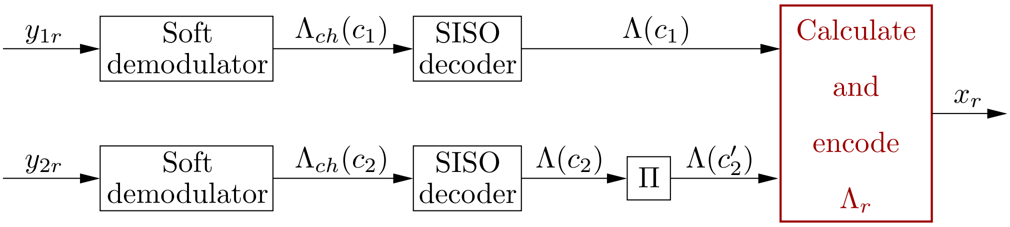


Figure 2.4: *Overview about the processing at the relay.*

Finally, the block of N code bits is mapped to $\lceil N/2 \rceil$ QPSK symbols. With the symbol alphabet $\mathcal{A} = \left\{a^{(1)}, a^{(2)}, a^{(3)}, a^{(4)}\right\}$, two bits $(b_0, b_1) \in \mathrm{GF}(2)\times\mathrm{GF}(2)$ are mapped to one QPSK symbol $a \in \mathcal{A}$ according to the following Gray labeled mapping.

$$\begin{array}{ccl} (b_0, b_1) & \longleftrightarrow & a \in \mathcal{A} \\ (0,0) & \longleftrightarrow & a^{(1)} = (-1-j)\sqrt{P_A/2} \\ (0,1) & \longleftrightarrow & a^{(2)} = (-1+j)\sqrt{P_A/2} \\ (1,0) & \longleftrightarrow & a^{(3)} = (1-j)\sqrt{P_A/2} \\ (1,1) & \longleftrightarrow & a^{(4)} = (1+j)\sqrt{P_A/2} \end{array} \tag{2.7}$$

More generally, assuming a source symbol alphabet with $M_{a,i} = |\mathcal{A}_i|$ different symbols, each symbol carries $l_i = \log_2(M_{a,i})$ bits. Thus, $M_i = N/l_i$ channel uses are needed per source and frame. Each source transmits at a rate of $l_i R_{c,i}$ in its timeslot. This yields an average rate of $R_i = \alpha l_i R_{c,i}$ for the ith source, $i \in \{1, 2\}$. With QPSK as symbol alphabet this leads to $M_i = N/2$ channel uses per frame and an average rate of $R_i = 2\alpha R_{c,i}$.

2.4 Processing at the Relay

In conventional relaying schemes with multiple sources, the relay can support only one source at a time. On the other hand, in a PLNC scheme the relay can support multiple sources simultaneously. In this section we describe the processing that is necessary at the relay in our PLNC scheme for the MARC. An overview about the relay processing is given in Figure 2.4. For reasons of space, the soft demodulators have been omitted in Figure 2.1.

The relay receives two signals, y_{1r} and y_{2r}, coming from the two sources, S_1 and S_2, respectively. Since the sources transmit in non-overlapping timeslots, the signals y_{1r} and y_{2r} do not interfere with each other and can therefore be processed independently. The

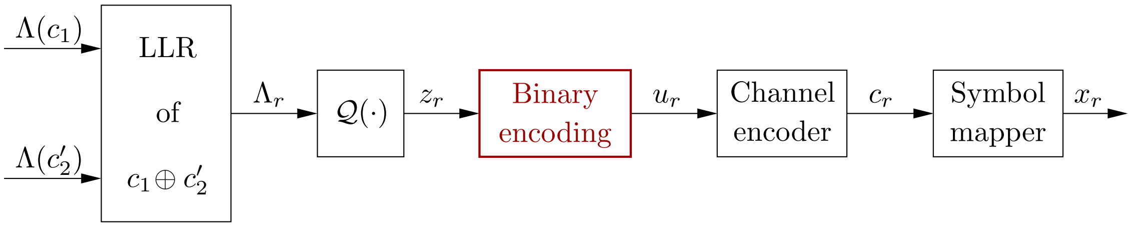


Figure 2.5: *Calculation, quantization and encoding of the soft network-coded LLR Λ_r.*

received signals are first processed by soft demodulators which compute LLRs about the code bits transmitted by the sources, based on the respective channel observation. These LLRs, $\Lambda_{ch}(c_i)$, do not take the redundancy introduced by the channel encoder into account. Rather, they serve as an input to the SISO decoders, which deliver posterior LLRs $\Lambda(c_i)$ for the code bits. We use an implementation of the BCJR algorithm to perform MAP symbol decoding of convolutional codes. Turbo codes are decoded by the iterative decoder shown in Figure 1.5, where each constituent code is again decoded by the BCJR algorithm. In order to decode LDPC codes we run the sum-product algorithm on the code's factor graph. Note that in any case no hard decisions are taken at the relay.

As a next step, the sequence of LLRs $\Lambda(c_2)$ are interleaved, yielding the LLR sequence $\Lambda(c_2') = \Pi\left(\Lambda(c_2)\right)$, where the code bits obtained by interleaving c_2 are denoted by c_2'. Here, interleaving is necessary in order to avoid short cycles in the iterative decoder at the destination, discussed in Section 2.5. The posterior LLR sequence $\Lambda(c_1)$ and the posterior interleaved LLR sequence $\Lambda(c_2')$ are then combined to form a new LLR sequence Λ_r, which is processed as shown in Figure 2.5.

In linear network coding over GF(2) the encoding and decoding operations are performed by computing the bit-wise modulo-2 sum of the correctly received bits (cf. Example 1.1). Taking transmission errors into account, the authors of [9] proposed to apply network coding to the soft values instead of the hard bits. This strategy allows us to apply network coding even if the source messages have not been decoded correctly at the relay node which performs the encoding operation. By applying Equation (1.4) we can express the soft network-coded LLR $\Lambda_r = \Lambda(c_1 \oplus c_2')$ in terms of the posterior LLRs $\Lambda(c_1)$ and $\Lambda(c_2')$, delivered by the channel decoders at the relay. This means that the output of the leftmost block in Figure 2.5 is given by

$$\Lambda_r = \Lambda(c_1) \boxplus \Lambda(c_2') = -2\,\mathrm{atanh}\left[\tanh\left(\frac{\Lambda(c_1)}{2}\right)\tanh\left(\frac{\Lambda(c_2')}{2}\right)\right]. \tag{2.8}$$

Due to the similarity of this expression to the XOR operation (modulo-2 sum), we will also call the coding rule in (2.8) *soft XOR* coding. In case the two blocks of code bits are not equally long, the shorter block of LLRs is simply padded with $\Lambda(c_i) = \infty$. This yields $\Lambda_r = \Lambda(c_j) \boxplus \Lambda(c_i) = \Lambda(c_j) \boxplus \infty = \Lambda(c_j)$.

It can be readily seen that the magnitude of the output of the boxplus operation is upper bounded by the minimum magnitude of its inputs, i.e.,

$$|\Lambda_r| \leq \min\left\{|\Lambda(c_1)|, |\Lambda(c_2')|\right\}. \tag{2.9}$$

Therefore, Λ_r will be unreliable, i.e., small in magnitude, as soon as one of the LLRs $\Lambda(c_1)$ and $\Lambda(c_2')$ is unreliable. Note that $\Lambda(c)\boxplus 0 = 0$ and also $\infty \boxplus 0 = 0$. It can be seen that the combination of LLRs according to (2.8) will work well, i.e., Λ_r will be large in magnitude, if $\Lambda(c_1)$ and $\Lambda(c_2')$ are large and approximately equal in magnitude. This is the case when the SNR on both source-relay channels is relatively high and equal, i.e., $\rho_{1r} = \rho_{2r} \gg 1$. In the case of unsymmetric source-relay channels, meaning $\rho_{1r} \neq \rho_{2r}$, the source with the worse source-relay channel could lower its code rate in order to increase the corresponding LLRs on average and compensate for the lower SNR on its source-relay link.

We employ the mutual information $I(C_1, C_2'; \Lambda_r)$ between the code bits c_1 and c_2' and the soft network-coded LLR Λ_r to gain insight into how helpful the relay can be, given certain source-relay SNRs ρ_{1r} and ρ_{2r}. However, note that this mutual information has no direct operational meaning in the information theoretic sense. Rather, it indicates how significantly the relay can support *both* sources. For $I(C_1, C_2'; \Lambda_r)$ we obtain

$$\begin{aligned} I(C_1, C_2'; \Lambda_r) &= \sum_{(c_1, c_2') \in [\mathrm{GF}(2)]^2} \int_{-\infty}^{\infty} p(\lambda_r, c_1, c_2') \log \frac{p(\lambda_r, c_1, c_2')}{p(\lambda_r) p(c_1, c_2')} d\lambda_r \\ &= \frac{1}{4} \sum_{(c_1, c_2') \in [\mathrm{GF}(2)]^2} \int_{-\infty}^{\infty} p(\lambda_r | c_1, c_2') \log \frac{4 p(\lambda_r | c_1, c_2')}{\sum\limits_{\tilde{c}_1, \tilde{c}_2'} p(\lambda_r | \tilde{c}_1, \tilde{c}_2')} d\lambda_r, \end{aligned} \tag{2.10}$$

where the second equality is obtained by applying Bayes rule and assuming that the code bits are equally likely and statistically independent (which is true, since the code bits belong to independent messages transmitted by two independent sources). Note that the conditional LLR distributions $p(\lambda_r | c_1, c_2')$ depend on the source-relay channel SNRs and the channel codes that are used on the source-relay links. Since we lack an analytical model of the posterior LLRs at the output of the SISO decoders, all four conditional pdfs $p(\lambda_r | c_1, c_2')$ have to be obtained using Monte Carlo simulations. However, note that due to the symmetry of the coding rule (2.8) we have $p(\lambda_r|0,0) = p(\lambda_r|1,1)$ and $p(\lambda_r|1,0) = p(\lambda_r|0,1)$. Therefore we can simplify (2.10) to

$$I(C_1, C_2'; \Lambda_r) = I(C_n; \Lambda_r) = \frac{1}{2} \sum_{c_n \in \mathrm{GF}(2)} \int_{-\infty}^{\infty} p(\lambda_r|c_n) \log \frac{2p(\lambda_r|c_n)}{\sum_{\tilde{c}_n} p(\lambda_r|\tilde{c}_n)} d\lambda_r, \tag{2.11}$$

where $c_n = c_1 \oplus c_2'$.

Figure 2.6 depicts a plot of the mutual information $I(C_1, C_2'; \Lambda_r)$ versus the average SNR ρ_{sr} (symmetric case) for two different channel codes. For the blue curve each source uses an RSCC with the channel encoder shown in Figure 1.3. The red curve is obtained when the sources use an irregular LDPC code[2] with code rate 1/2 and a blocklength of $N = 64\,000$. In any case the source-relay channels are modeled according to (2.3). From Figure 2.6 we can see that with increasing ρ_{sr} the confidence about the value of $c_1 \oplus c_2'$ is growing and eventually saturates at $I(C_1, C_2'; \Lambda_r) = 1$ for high SNR. On the other hand, if ρ_{sr} is too low, then also the mutual information is small. This means that the relay won't be able to significantly support the sources in their transmission.

The mutual information $I(C_1, C_2'; \Lambda_r)$ in case of non-symmetric source-relay channels is depicted in Figure 2.7. Here, both sources use the RSCC used in Figure 2.6 and all other system parameters are the same as above. It can be clearly seen that in case of non-symmetric source-relay channels the soft XOR network coding strategy is suboptimal. If one source has a very bad channel to the relay, it will render all LLRs Λ_r unreliable, irrespective of the SNR on the other source-relay channel. As pointed out previously, one strategy to compensate for non-symmetric source-relay channels is to adapt the code rates of the two sources. A more sophisticated strategy is to apply vector quantization directly to the two posterior LLRs from the channel decoders, instead of first combining them and performing a scalar quantization afterwards. A PLNC scheme for the MARC using LLR vector quantization is presented in [38].

The subsequent processing should be designed such that the information about $c_1 \oplus c_2'$ in Λ_r can be recovered as accurately as possible at the destination. In order to allow forwarding of the LLRs $\Lambda_r \in \mathbb{R}$ using digital transmission, we have to quantize the LLRs. To this end, we use a scalar quantizer $\mathcal{Q}(\cdot)$ with Q levels, i.e., we quantize using $\log_2(Q)$ bits per LLR. For each LLR the quantizer produces an integer $z_r \in \{0, 1, \ldots, Q-1\}$, the quantization index, at its output. Several quantizer design methods for soft information quantization are discussed and compared in Chapter 3.

The quantization indices z_r are subsequently converted to a binary representation,

[2] The LDPC code was designed using the EPFL web-tool at `http://ipgdemos.epfl.ch/ldpcopt/`.

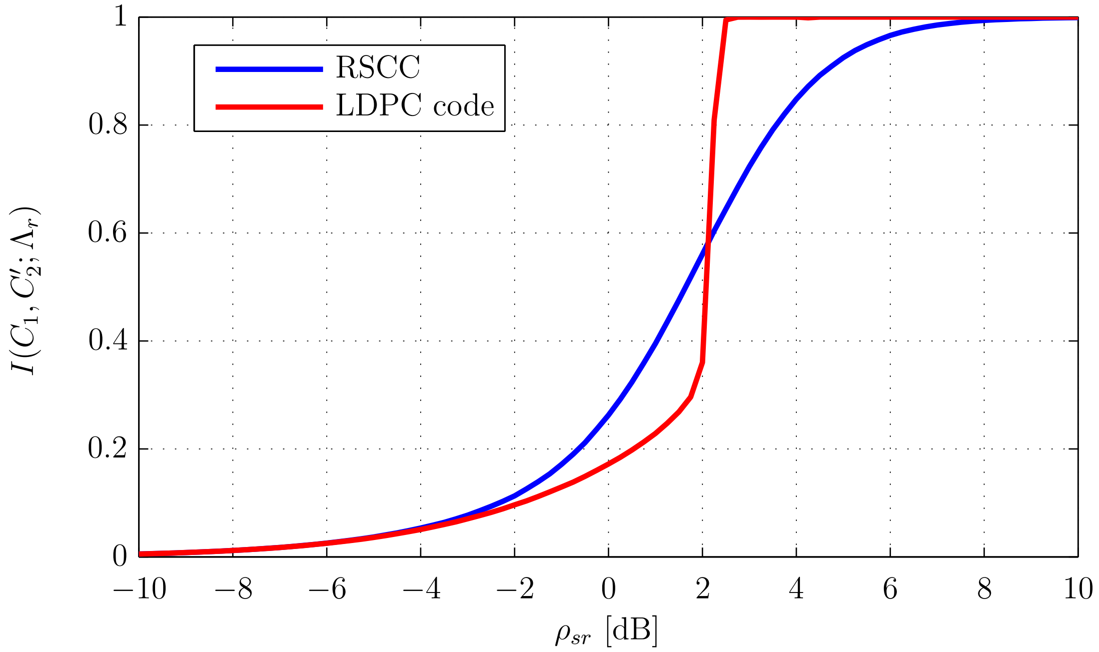


Figure 2.6: *Mutual information $I(C_1, C_2'; \Lambda_r)$ between the code bits transmitted by the sources and the soft network-coded LLR Λ_r for an RSCC and an LDPC code versus ρ_{sr}, the average SNR on the source-relay links (symmetric case). Note that the decoding threshold of the LDPC code at $\rho_{sr} \approx 2$ dB is clearly visible in $I(C_1, C_2'; \Lambda_r)$.*

denoted by u_r. One possibility for this conversion is depicted in Figure 2.8(a). Here, we take each integer z_r and map it to its natural binary representation using $\lceil \log_2(Q) \rceil$ bits. However, note that in general, for $Q > 2$, the quantizer levels will be used with unequal probabilities. Therefore it is possible to find a binary representation $\widetilde{u}_r$ of length $\widetilde{N}_r$ using fewer bits by applying a source code with variable rate, i.e., lossless data compression, to the quantization indices z_r as shown in Figure 2.8(b). We define the source coding rate, i.e., the data compression ratio, R_s as the number of uncoded bits over the expected number of source-coded bits,

$$1 \leq R_s = \frac{\lceil \log_2(Q) \rceil N}{\mathrm{E}\{\widetilde{N}_r\}} \leq \frac{\lceil \log_2(Q) \rceil}{H(Z_r)}, \tag{2.12}$$

where $H(Z_r)$ is the entropy of the quantizer output. In order to recover z_r from $\widetilde{u}_r$ using a source decoder, we have to ensure that $\widetilde{u}_r$ is available at the destination without errors. Therefore we append a frame check sequence, e.g., the parity bits of a CRC code, to $\widetilde{u}_r$ in order to allow error detection at the destination. In the following, the overhead due

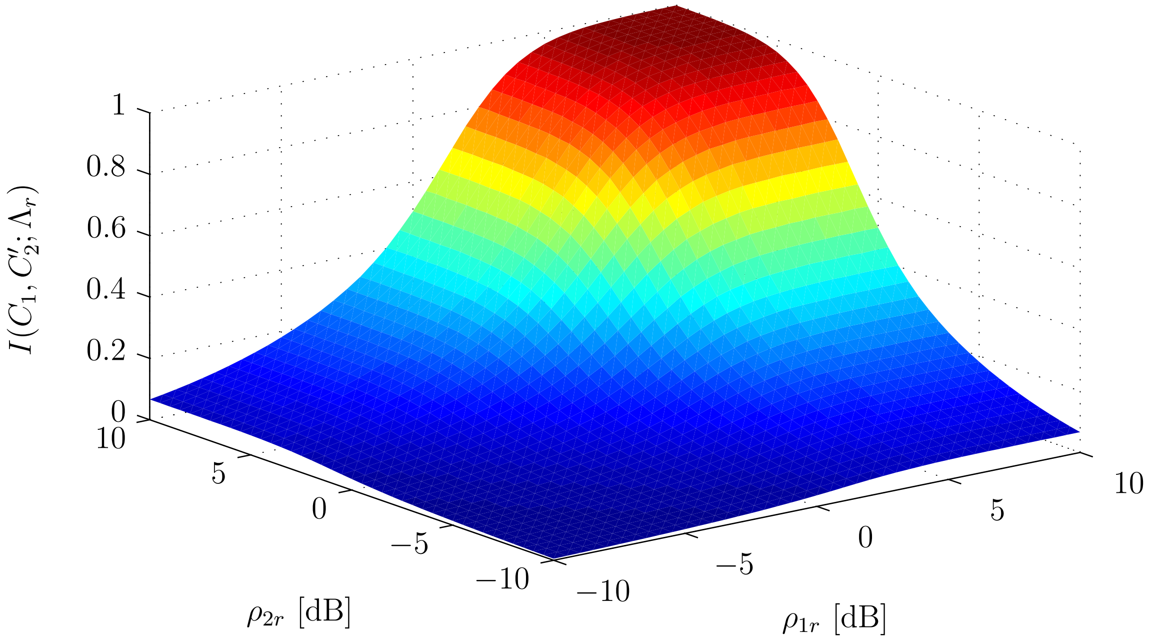
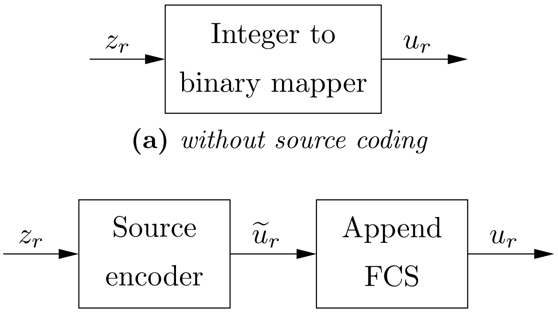


Figure 2.7: *Mutual information $I(C_1, C_2'; \Lambda_r)$ versus ρ_{1r} and ρ_{2r} (non-symmetric case). It can be seen that soft XOR network coding is suboptimal in the general case of non-symmetric source-relay channels. Here, both sources use an RSCC as channel code.*

to the FCS will be neglected.

If, with source coding, an error is detected at the destination, the whole block transmitted by the relay has to be discarded in order to avoid error propagation through the source decoder. The relay-destination channel, seen at the output of the source decoder, is thereby converted to a binary (block) erasure channel. For lossless compression of quantizer indices we use arithmetic coding, which is well suited for long sequences from sources having skewed symbol distributions and small symbol alphabets. An introduction to arithmetic coding can be found in [39].

The remaining processing steps, namely channel encoding and symbol mapping, are similar to the processing at the sources (cf. Section 2.3). More specifically, the relay uses a channel code with code rate (including puncturing) of $R_{c,r}$ and it transmits using the QPSK mapping defined in (2.7), i.e., each symbol carries $l_r = 2$ bits. The relay has to transmit $\lceil \log_2(Q) \rceil N/R_s$ bits per frame, where $N = R_{c_i} K_i, i \in \{1, 2\}$, and R_s is the source coding rate ($R_s = 1$ in case source coding is not employed). Including channel coding this amounts to $\lceil \log_2(Q) \rceil N/(R_{c,r} R_s)$ code bits per frame. Using a symbol alphabet with $M_{a,r} = |\mathcal{A}_r|$ different symbols, we have $l_r = \log_2(M_{a,r})$ bits per



(a) *without source coding*

(b) *with source coding*

Figure 2.8: *Binary encoding of the quantization index z_r.*

symbol. Therefore we obtain

$$M_r = \frac{\lceil \log_2(Q) \rceil N}{l_r R_{c,r} R_s} \tag{2.13}$$

transmitted symbols, i.e., relay channel uses, per frame. Assuming $l_1 = l_2$ and therefore $M_1 = M_2$, we get

$$M = 2M_i + M_r = N \left(\frac{2}{l_i} + \frac{\lceil \log_2(Q) \rceil}{l_r R_{c,r} R_s} \right) \tag{2.14}$$

for the total number of channel uses per frame. Since $M_i = \alpha M$ we obtain the following expression for the time sharing parameter α,

$$\alpha = \left(2 + \frac{\lceil \log_2(Q) \rceil}{R_{c,r} R_s} \frac{l_i}{l_r} \right)^{-1} . \tag{2.15}$$

Table 2.1 shows some possible values for α depending on the relay symbol alphabet size $M_{a,r} = 2^{l_r}$ and on the number of quantizer levels Q. For a TDMA system with two sources and one destination, but without the relay, the total transmission time would be divided equally between both sources, yielding $\alpha = 1/2$. From Table 2.1 it can be seen that the relay should use a large symbol alphabet size $M_{a,r}$ in order to keep the rate loss, compared to the case without relay, small. For system deployment this means that the relay should be positioned such that the relay-destination SNR ρ_{rd} allows the use of relatively large signal constellations.

Finally, we conjecture that it would be beneficial to design the coding and modulation at the relay such that the sign of the LLRs is better protected than the magnitude information. However, in this thesis we do not consider such unequal error protection techniques.

$M_{a,r}$	Q	α
4	2	1/4
4	4	1/6
4	8	1/8
16	2	1/3
16	4	1/4
16	8	1/5
64	2	3/8
64	4	3/10
64	8	1/4

Table 2.1: *Value of the time sharing parameter α for different relay symbol alphabet sizes $M_{a,r} = 2^{l_r}$ and for different numbers of quantizer levels Q. Here, $R_{c,r}$ is equal to 1/2, l_i is fixed to 2 and R_s is assumed to be 1, e.g., no source coding is used.*

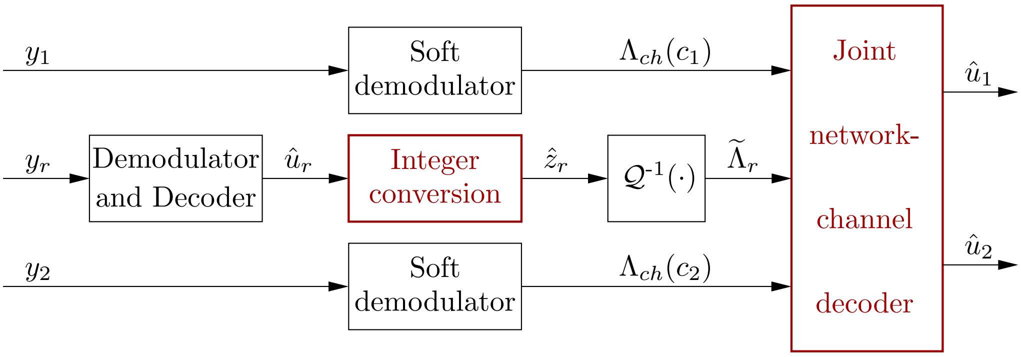
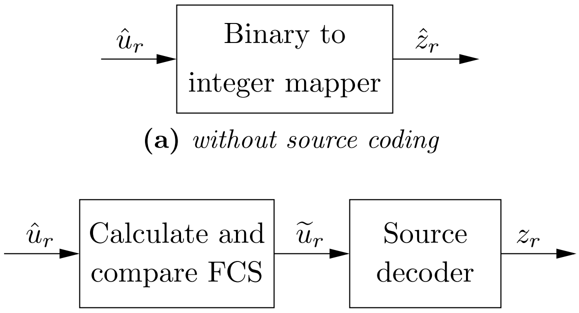


Figure 2.9: *Overview about the processing at the destination.*

2.5 Processing at the Destination

In this section we describe the processing at the destination with a focus on the iterative joint network-channel decoder. An overview about the destination processing is given in Figure 2.9.

The destination can avail of three observations, two coming directly from the sources and one coming from the relay. The received signals y_1 and y_2 are processed by soft demodulators in order to calculate the posterior LLRs $\Lambda_{ch}(c_i)$ about the code bits transmitted by the sources. From the signal y_r the destination seeks to recover the quantization indices z_r. To this end, y_r is demodulated and decoded to obtain an estimate $\hat{u}_r$ about the corresponding bits u_r. These bits then have to be converted to the respective quantization indices.



(a) *without source coding*

(b) *with source coding*

Figure 2.10: *Conversion of the decoded bits $\hat{u}_r$ to the quantization index $\hat{z}_r$.*

In case no source codec has been used for the transmission of the quantization indices, this amounts to taking blocks of $\lceil \log_2(Q) \rceil$ bits $\hat{u}_r$ and calculating the corresponding integer $\hat{z}_r$, as shown in Figure 2.10(a). Note that in this case $\hat{z}_r$ is not necessarily equal to z_r. On the other hand, if a source codec has been used, we have to ensure that the source-coded bits $\widetilde{u}_r$ have been decoded correctly at the destination. This is accomplished by calculating the FCS of the decoded bits and comparing it to the FCS that has been received, as depicted in Figure 2.10(b). In case no decoding error is detected, the bits $\widetilde{u}_r$ are fed into the source decoder which then produces the *true* quantization indices z_r at its output. If a decoding error has been detected, the source decoder and the block labeled "$\mathcal{Q}^{-1}(\cdot)$" (cf. Figure 2.9) are bypassed and all LLRs $\widetilde{\Lambda}_r$ are set to zero. In all cases except the latter, the decoded quantization indices $\hat{z}_r$ are used to obtain the respective quantizer reproducer value $\widetilde{\Lambda}_r$. This operation is performed by the block "$\mathcal{Q}^{-1}(\cdot)$", depicted in Figure 2.9. Note, we assume the destinations knows which quantizer has been used at the relay, i.e., the mapping $z_r \leftrightarrow \widetilde{\Lambda}_r$ is known at the destination. As we will see in Chapter 3 the choice of the quantizer mapping, for given channel codes, depends only on the source-relay channel SNRs. In this thesis we do not consider the impact of noisy transmission of the relay's quantizer choice to the destination. Furthermore, we neglect the overhead that is due to the transmission of the quantizer choice in a practical system.

The LLRs $\Lambda_{ch}(c_1), \Lambda_{ch}(c_2)$ and $\widetilde{\Lambda}_r$, corresponding to the three observations at the destination, form the input to the joint network channel decoder discussed in the following subsection.

2.5.1 Iterative Joint Network-Channel Decoder

The iterative joint network-channel decoder of our PLNC scheme, shown in Figure 2.11, essentially consists of two channel decoders, one for each source, and two identical

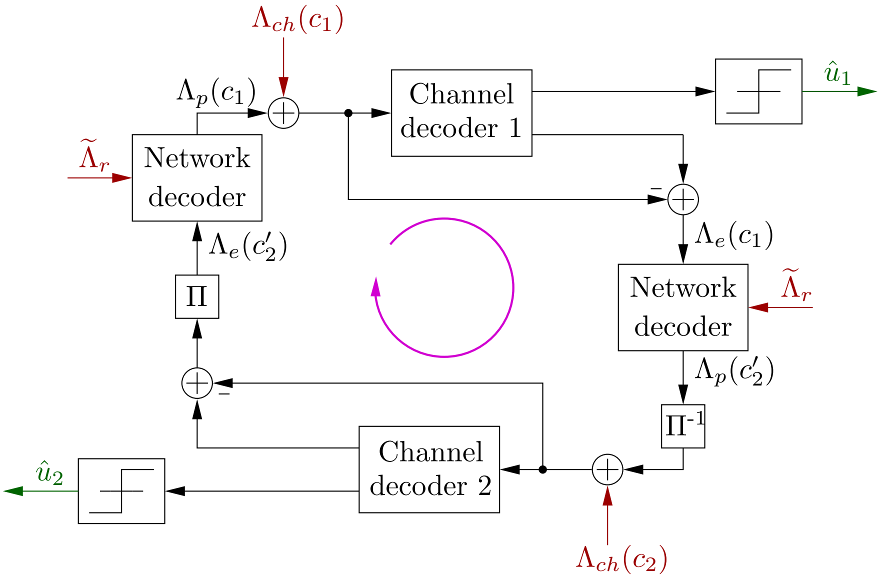


Figure 2.11: *Iterative joint network-channel decoder. The inputs to the iterative decoder are indicated in red and the outputs are indicated in green.*

network decoders (since only one node, namely the relay, performs network encoding).

Both channel decoders are coupled and exchange information via the network decoder in an iterative manner. The coupling is due to the fact that the destination receives soft information about the network-coded bits $c_n = c_1 \oplus c_2'$ from the relay. This information, together with information about either c_1 or c_2', can be used to obtain prior information about the respective other code bit and thus induces coupling between the channel decoders.

In the following we will explain the processing of the iterative decoder. Let us assume decoding starts with the invocation of channel decoder 1. In the first iteration there is no prior information available, i.e., $\Lambda_p(c_1) = 0$. Thus the LLRs $\Lambda_{ch}(c_1)$, corresponding to the direct observation from source S_1, form the input for channel decoder 1. At the output of the channel decoder, we obtain posterior LLRs about the information bits u_1 and about the code bits c_1. The extrinsic LLRs $\Lambda_e(c_1)$ are calculated by subtracting the LLRs at the input from the posterior LLRs at the output of the channel decoder. These extrinsic LLRs, together with $\widetilde{\Lambda}_r$, are fed into the network decoder, which produces prior information about the code bits transmitted by source S_2. The prior LLRs at the output of the network decoder, $\Lambda_p(c_2')$, are deinterleaved and added to the LLRs $\Lambda_{ch}(c_2)$. The resulting LLRs form the input to channel decoder 2. Similarly to channel decoder 1, the second channel decoder calculates posterior LLRs about the information bits and code bits transmitted by source S_2. The interleaved extrinsic LLRs $\Lambda_e(c_2')$ and $\widetilde{\Lambda}_r$ are then used by the network decoder to calculate prior information about the code bits c_1. This prior information is used by channel decoder 1 in the next iteration. The decoder will perform a certain number of iterations until a stopping criterion is met. Then, a hard decision on the posterior LLRs about the information bits u_1 and u_2 is performed in order to obtain the corresponding estimates $\hat{u}_1$ and $\hat{u}_2$.

The level of confidence the destination has about the value of $c_1 \oplus c_2'$ determines how tightly the channel decoders are coupled. From this point of view, the network decoder can be seen as a gate, controlled by $\widetilde{\Lambda}_r$, for the amount of information transfered between the channel decoders. In case there is no knowledge about the value of $c_1 \oplus c_2'$ at the destination, i.e., $\widetilde{\Lambda}_r = 0$, the two channel decoders are completely decoupled. On the other hand, as $\widetilde{\Lambda}_r \to \pm\infty$, all the extrinsic information of one decoder will be used as prior information for the other decoder. In this case the overall decoder works basically like the turbo decoder shown in Figure 1.5. As we will presently see, the tighter the coupling between the channel decoders is, the more gain will the iterations yield and the more turbo-like will the joint network-channel decoder perform.

Let us now discuss the processing that is performed by the network decoder. Figure 2.12 shows the factor graph corresponding to the network decoder. Here, the function node f is defined as

$$f(c_1, c_2', c_n) = \begin{cases} 1, & c_n = c_1 \oplus c_2' \\ 0, & \text{otherwise} \end{cases}. \tag{2.16}$$

The output of the network decoder, in this case $\Lambda_p(c_2')$, can be determined using belief propagation [25]. By applying the message update rule, we obtain

$$\begin{aligned} \Lambda_p(c_2') &= \sum_{\sim c_2'} f(c_1, c_2', c_n)\Lambda_e(c_1)\widetilde{\Lambda}_r(c_n) \\ &= -2 \operatorname{atanh}\left(\tanh\frac{\Lambda_e(c_1)}{2}\ \tanh\frac{\widetilde{\Lambda}_r(c_n)}{2}\right) \\ &= \Lambda_e(c_1) \boxplus \widetilde{\Lambda}_r(c_n), \end{aligned} \tag{2.17}$$

i.e., the output of the network decoder is the boxplus of its inputs. We observe that the network decoder performs the same calculation as the network encoder does. This is analogous to network decoding in the bit domain, where encoding and decoding, amount to a modulo-2 addition (cf. Example 1.1). However, note that $c_n \oplus c_j = (c_j \oplus c_i) \oplus c_j = c_i$, but $\Lambda_r \boxplus \Lambda(c_j) = \Lambda(c_j \oplus c_i) \boxplus \Lambda(c_j) \neq \Lambda(c_i)$. Thus, the boxplus operation is not an involution.

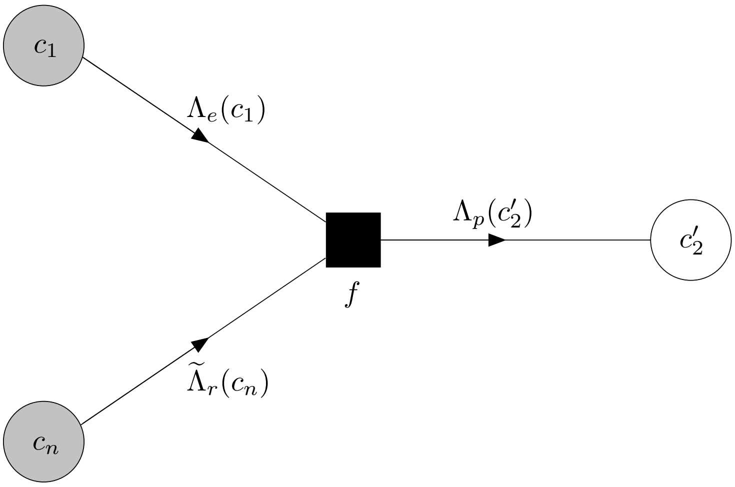


Figure 2.12: *Factor graph representation of the network decoder.*

Now it is also evident that the network decoder, controlled by $\widetilde{\Lambda}_r$, determines the information exchange between the channel decoders. Recalling the rules for the boxplus calculation, we see that $\Lambda \boxplus 0 = 0$ and $\Lambda \boxplus \pm\infty = \pm\Lambda$. For these two extreme cases we therefore obtain $\Lambda_p(c_i) = \Lambda_e(c_j) \boxplus 0 = 0$ (channel decoders are completely decoupled, iterations yield no gain) and $\Lambda_p(c_i) = \Lambda_e(c_j) \boxplus \pm\infty = \pm\Lambda_e(c_j)$ (channel decoders are perfectly coupled, yielding turbo-like performance).

2.6 Reference Systems

In order to assess the performance of our scheme, we will compare it to three different reference schemes. In the following, these schemes are briefly described.

The first reference system is equal to the scheme we propose, except for the LLR forwarding strategy. This reference system uses analog LLR forwarding from the relay to the destination (cf. [9]) modeled according to (2.1). Since the analog LLR transmission described in [9] can be considered as a gedankenexperiment, it is difficult to find a way for a fair comparison to schemes using coded LLR transmission. We therefore compare this scheme to our scheme using the same system parameters and an equal SNR on the relay-destination channel.

The second reference system is a decode-and-forward scheme, where the relay uses half of its timeslot for source S_1 and the other half for source S_2. The relay forwards the decoded code bits without re-encoding the received data. Thereby the relay equally supports both sources in each frame. The destination combines the LLRs obtained for the respective codewords and decodes both messages separately. In case the relay can only decode one of the two source messages, it spends its complete timeslot for that source. If the relay was unable to decode any message it will remain silent. In any case, the time sharing parameter α_{ref} is fixed to 1/3. Note, if the relay would use its timeslot to support only one source in each frame, the average BER for each source could at best be halved. This corresponds to a small SNR gain compared to the case where the relay does not help the sources at all. Therefore it is clearly advantageous to support both sources in each frame (if possible).

The third reference system considers multiple-access of two sources to a common destination, i.e., the network is obtained by removing the relay node from the MARC. Here, we again assume TDMA with equally long timeslots for each source. Therefore the time sharing parameter α_{ref} is equal to 1/2.

To allow for a fair comparison between the systems we assume that in each case both sources transmit the same number of bits using the same channel code and signal constellation. Therefore the sources need the same number of channel uses per frame in each system. However, the time sharing parameters α (for the proposed system) and α_{ref} may be different because the transmission time occupied by the relay (if present) may differ. This means, we have to compensate for the differences in the average rates when comparing two systems. We obtain

$$\frac{R_i}{R_{i,\text{ref}}} = \frac{\alpha}{\alpha_{\text{ref}}}, \tag{2.18}$$

or, equivalently,

$$\frac{(E_b/N_0)_i}{(E_b/N_0)_{i,\text{ref}}} = \frac{\alpha_{\text{ref}}}{\alpha}. \tag{2.19}$$

This leads to the following relation between $(E_b/N_0)_{i,\text{ref}}^{[\text{dB}]}$ and $(E_b/N_0)_i^{[\text{dB}]}$

$$(E_b/N_0)_{i,\text{ref}}^{[\text{dB}]} = (E_b/N_0)_i^{[\text{dB}]} - 10\log_{10}\left(\frac{\alpha_{\text{ref}}}{\alpha}\right)\text{dB}. \tag{2.20}$$

3 Quantization of Soft Information

3.1 Introduction

In this chapter we discuss the important issue of soft information quantization. As we have seen in Chapter 2, quantization of LLRs is necessary for the PLNC scheme we propose in order to forward soft information from the relay to the destination using digital transmission. In the following we want to further motivate soft information quantization.

In Chapter 1 we have introduced the notion of soft information. In contrast to hard decisions, soft decisions provide additional reliability information. Therefore, soft information processing is, in terms of performance, generally superior to the use of hard decisions. In many applications of digital communications concatenated modules (iteratively) exchange soft information to perform a certain task. Today, virtually all advanced receiver concepts build on the processing and exchange of soft information, e.g., the turbo decoder shown in Figure 1.5. Moreover, in wireless relay networks, as we have explained previously, it is beneficial to exchange soft information between the individual nodes.

However, compared to hard decisions, which can be represented by a single bit, soft decisions are real numbers and therefore cannot be represented exactly using a finite number of bits. Nevertheless, practical implementations usually impose constraints with respect to memory consumption and computational complexity. Thus, soft information has to be quantized, i.e., represented by a *finite* number of bits, whenever it is processed, stored or transmitted. In general one would like to spend as few bits as possible for each soft decision. The number of bits that is used to represent the quantized values is especially critical in case the information has to be exchanged between spatially distant nodes. On the other hand, we have to expect an increasing loss in performance the cruder we represent the available soft information. This fundamental tradeoff is common to all lossy source coding problems.

Clearly, the number of bits per quantized value, henceforth called the (quantizer) *rate*, is not the only thing that matters. If we are given a fixed rate then we can still choose the quantization regions and the associated reproducer values. These choices should of course be taken according to some optimality criterion. In the following section we will discuss the details of quantizer design for soft information quantization.

3.2 Quantizer Design for Soft Information Quantization

Soft information can be represented in various ways, in the following we will, however, focus on the quantization of log-likelihood ratios. We want to design a scalar quantizer $\mathcal{Q}$ with fixed rate R_Q. In the case of scalar quantization, R_Q is an integer and the number of quantizer levels is given by $Q = 2^{R_Q}$. Likewise there are Q quantization regions $\mathcal{R}^{(i)}$, given by the intervals $[g_{i-1}, g_i] \subseteq \mathbb{R}$, $i \in \{1, 2, \ldots, Q\}$. Here, g_i denotes the decision boundary between the quantization regions $\mathcal{R}^{(i)}$ and $\mathcal{R}^{(i+1)}$. Note that we set $g_0 = -\infty$ and $g_Q = \infty$. The quantizer $\mathcal{Q}$ maps an LLR $\Lambda \in \mathbb{R}$ to one reproducer value λ_i according to the following relation.

$$\hat{\Lambda} \triangleq \mathcal{Q}(\Lambda) = \lambda_i \qquad \text{if} \quad \Lambda \in \mathcal{R}^{(i)},\ i = 1, 2, \ldots, Q. \tag{3.1}$$

Quantizer design thus amounts to specifying the quantization regions $\mathcal{R}^{(i)}$ and the corresponding reproducer values λ_i. For the transmission of the quantized LLR $\hat{\Lambda}$ we will use the index $i \in \mathbb{N}$ of the respective quantization region, instead of the reproducer value $\lambda_i \in \mathbb{R}$ itself. This implies, however, that the reproducer values λ_i have to be known at the destination. Note that for $Q > 2$ the individual quantizer levels might be used with pronounced unequal probabilities. We can therefore perform data compression as described in Section 2.4 in order to represent a sequence of quantizer indices i by a smaller number of bits.

The LLRs we want to quantize are the observations $\Lambda_r = \Lambda(c_1) \boxplus \Lambda(c_2')$, obtained at the relay of our PLNC scheme for the MARC (cf. Figure 2.5). As in Chapter 2 the quantization index corresponding to Λ_r will be denoted by z_r. For the design of the quantizer $\mathcal{Q}$ we use the conditional LLR distributions $p(\lambda_r|c_1, c_2')$, which are obtained by Monte Carlo simulations. Thus the quantizer design happens “offline”. Performing on-the-fly quantizer design during data transmission could be beneficial, since we would not have to presuppose knowledge of the conditional LLR distributions. However, in our PLNC scheme for the MARC all information about the quantizer design would have to be signaled from the relay to the destination in case of an on-the-fly quantizer design approach.

It is important to note that the conditional LLR distributions $p(\lambda_r|c_1, c_2')$ do not depend on c_1 and c_2' independently. Rather, they depend only on $c_1 \oplus c_2'$, i.e., we have the symmetries $p(\lambda_r|0,0) = p(\lambda_r|1,1)$ and $p(\lambda_r|1,0) = p(\lambda_r|0,1)$. Moreover, the unconditional LLR distribution $p(\lambda_r)$ is an even function, yielding $p(-\lambda_r) = p(\lambda_r)$. This leads to the additional symmetry relation $p(-\lambda_r|0,0) = p(\lambda_r|0,1)$ between the two different conditional LLR distributions. We will therefore use a symmetric quantizer design in order to reflect the aforementioned symmetries. Specifically we get

$$g_{Q-i} = -g_i, \qquad i \in \left\{ \left\lceil \frac{Q}{2} \right\rceil, \left\lceil \frac{Q}{2} \right\rceil + 1, \ldots, Q \right\} \tag{3.2}$$

for the decision boundaries, and

$$\lambda_{Q-i+1} = -\lambda_i, \qquad i \in \left\{ \left\lceil \frac{Q+1}{2} \right\rceil, \left\lceil \frac{Q+1}{2} \right\rceil + 1, \ldots, Q \right\} \tag{3.3}$$

for the quantizer reproducer values. This implies that $g_{Q/2} = 0$ if Q is even and $\lambda_{\lceil Q/2 \rceil} = 0$ in case Q is odd.

Figure 3.1 shows the unconditional LLR distribution $p(\lambda_r)$ for two different source-relay SNRs. We can see that for a relatively low SNR of 2 dB the LLR distribution is unimodal, whereas for a higher SNR of 7 dB the distribution is multimodal. In addition to the SNR, the LLR distribution also depends on the channel model and the channel code employed by the sources. Here, we used the same RSCC as in the previous chapter and we assumed fast i.i.d. Rayleigh flat fading. Obviously, the quantizer we use has to be designed for a wide range of SNR values. The relay node in our PLNC scheme then has to select the quantizer that was designed for the SNR that is closest to the current channel SNR. Note that in our scheme we assume perfect CSI (about the source-relay links) at the relay and thus the relay will always choose the correct quantizer. We do not study the effect of imperfect CSI on the quantizer choice and its impact on the system performance.

The framework of rate-distortion theory is commonly used to analyze lossy source coding problems. Rate-distortion theory characterizes the tradeoff between the signal representation size, i.e., the rate, and the average distortion of the reconstructed signal. In particular, it tells us how large the average bit rate R_Q has to be in order to represent

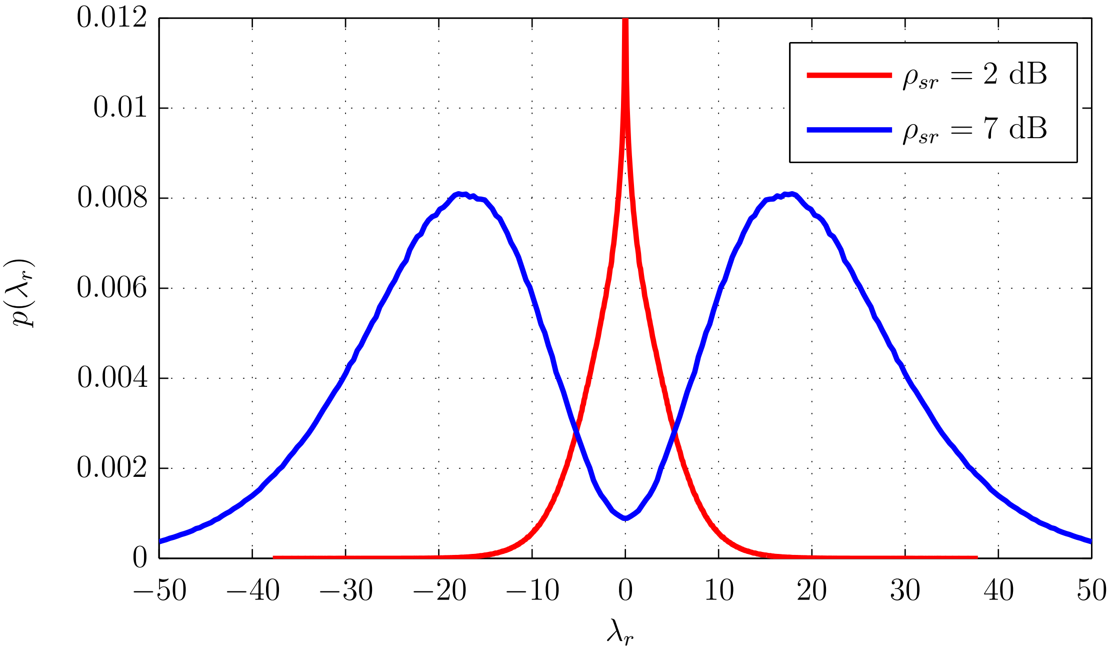


Figure 3.1: *Unconditional LLR distribution $p(\lambda_r)$ for two different values of the source-relay SNR ρ_{sr}. Note the symmetry of the LLR distributions.*

a source with some given distortion D_Q (rate-distortion function), or, equivalently, how small the mean distortion D_Q can be, when encoding a source at some given rate R_Q (distortion-rate function, the inverse of the rate-distortion function). However, the rate distortion function is generally not known analytically and moreover rate-distortion theory only allows us to find upper limits for the potential system performance, since it provides only asymptotic results.

For a source producing symbols $\lambda \in \mathcal{L}$, distributed according to $p(\lambda)$, rate-distortion theory tells us that the minimum rate required to represent the source by a random variable $Z \in \mathcal{Z}$, with a mean distortion of at most D_Q, is given by [40]

$$R_Q(D_Q) = \min_{p(z|\lambda)} I(\Lambda; Z) \qquad \text{subject to} \quad \mathrm{E}\{d(z, \lambda)\} \leq D_Q. \tag{3.4}$$

Here, $d : \mathcal{Z} \times \mathcal{L} \rightarrow \mathbb{R}_0^+$ is the distortion measure. At this point the problem is that the distortion measure d has to be fixed in advance in order to solve this rate-distortion problem. *The choice of the distortion measure is not part of rate-distortion theory.* Unfortunately, there is no general approach for finding a good or even the best distortion measure for a specific problem.

In the following sections we describe three different quantizer design approaches based on different optimality criteria.

3.3 The Lloyd Algorithm

The Lloyd algorithm [41], [42] is based on the squared-error distortion measure, i.e., $d(\hat{x}, x) = (\hat{x} - x)^2$. The main idea of the Lloyd algorithm is to minimize the mean square error (MSE) distortion by adapting the quantizer reproducer values to the pdf of the signal that shall be quantized. Therefore the Lloyd algorithm allows us to design optimal scalar quantizers (with respect to the MSE distortion measure). The Lloyd algorithm is frequently applied to quantizer design for waveform signals as well as in image processing. However, the Lloyd algorithm is applicable only if the source distribution is either known a priori or if it can be estimated sufficiently well.

For a fixed-rate quantizer with Q levels, the mean power of the quantizer distortion P_D is given by

$$P_D = \mathrm{E}\left\{d(\mathcal{Q}(\Lambda), \Lambda)\right\} = \mathrm{E}\{(\hat{\Lambda} - \Lambda)^2\} = \sum_{i=0}^{Q-1} \int\limits_{g_i}^{g_{i+1}} (\lambda_{i+1} - \lambda)^2 p(\lambda) d\lambda, \tag{3.5}$$

where λ_i denotes the ith quantizer reproducer value and $p(\lambda)$ is the pdf for which we want to optimize the quantizer. A direct minimization of P_D is generally not possible because all g_i and λ_i are unknown. However, by applying the Lloyd algorithm, this optimization problem can be solved iteratively. As a first step, let us assume we knew the reproducer values. Then we can optimize the decision boundaries g_i by computing

$$\frac{\partial P_D}{\partial g_j} = (\lambda_j - g_j)^2 p(g_j) - (\lambda_{j+1} - g_j) p(g_j) \overset{!}{=} 0. \tag{3.6}$$

This leads to

$$g_j = \frac{\lambda_{j+1} + \lambda_j}{2}, \qquad j = 1, 2, \ldots, Q-1, \tag{3.7}$$

and $g_0 = -\infty$, $g_Q = \infty$, i.e., the optimal decision boundaries are in the middle between two reproducer values. Now we can do the same thing for the reproducer values. Setting to zero the partial derivatives

$$\frac{\partial P_D}{\partial \lambda_{j+1}} = \frac{\partial}{\partial \lambda_{j+1}} \int\limits_{g_j}^{g_{j+1}} (\lambda_{j+1} - \lambda)^2 p(\lambda) d\lambda = 2 \int\limits_{g_j}^{g_{j+1}} (\lambda_{j+1} - \lambda) p(\lambda) d\lambda \tag{3.8}$$

yields

$$\lambda_{j+1} = \frac{\displaystyle\int_{g_j}^{g_{j+1}} \lambda p(\lambda) d\lambda}{\displaystyle\int_{g_j}^{g_{j+1}} p(\lambda) d\lambda}, \qquad j = 1, 2, \ldots, Q. \tag{3.9}$$

This means that the optimal reproducer values are given by the centroid of the corresponding decision region.

By evaluating (3.7) and (3.9) the optimal scalar quantizer can be determined. However, these equations cannot be solved independently since they are coupled. In order to obtain a solution one can start with a guess for the reproducer values and iteratively compute the new decision boundaries and the updated reproducer values. The iterations are performed until all λ_i do not (significantly) change anymore. The Lloyd algorithm is guaranteed to converge to a *local* minimum of P_D, as P_D decreases in each iteration (until convergence). Therefore the initial guess for the reproducer values will have an impact on the quality of the solution. In order to obtain a good quantizer, multiple instances of the Lloyd algorithm can be run on a sufficiently large number of initial guesses. The best quantizer, i.e., the one yielding the smallest average distortion, is then chosen as the result of the quantizer design procedure.

The Lloyd algorithm can also be used with other distortion measures than the squared-error distortion. However, using a different distortion measure will presumably not yield closed-form expressions for the decision boundaries and the reproducer values. Furthermore the optimization problem of minimizing P_D can be constrained by introducing Lagrange multipliers to obtain solutions adapted to the given constraints.

An extension of the Lloyd algorithm to vector quantization is given by the LBG (Linde, Buzo and Gray) algorithm [43].

3.4 Equiprobable Output Quantizer

In contrast to the Lloyd algorithm the equiprobable output quantizer [44] has been developed with communications applications in mind. In communication systems concatenated modules exchange information to perform certain tasks. Motivated by the data processing inequality, it seems natural to strive for minimum information loss in a processing chain.

Consider the generic transmission system shown in Figure 3.2. Here, the information bits u are encoded and the code bits c are transmitted over a channel. The channel

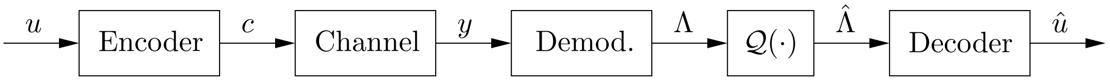


Figure 3.2: *A generic transmission system with LLR quantization in which the channel decoder processes quantized LLRs. Note that complexity and memory constraints may necessitate LLR quantization.*

output y is then demodulated, yielding posterior LLRs Λ about the code bits. These LLRs are subsequently quantized resulting in quantized LLRs $\hat{\Lambda}$ which are finally processed by a decoder to produce estimates $\hat{u}$ about the transmitted bits u. Note that LLR quantization might be necessary due to memory or complexity constraints.

Quantization of LLRs for maximum mutual information, or, equivalently, minimum mutual information loss, has been discussed in [45] for the special case of BPSK transmission over an AWGN channel. In that paper the LLR quantization was designed such that the mutual information $I(C;\hat{\Lambda})$ is maximized. However, an extension of this approach to more sophisticated modulation schemes and different channel models appears infeasible. Hence the authors of [44] proposed a quantization scheme which deviates from mutual information maximization and allows for simple implementation while leading only to a small decrease of the information rate.

In the following we will describe the basic idea of the equiprobable output quantizer. From Figure 3.2 it can be seen that $c - \Lambda - \hat{\Lambda}$ forms a Markov chain. Hence, the data processing inequality implies $I(C;\hat{\Lambda}) \leq I(C;\Lambda)$. In order for $I(C;\hat{\Lambda})$ to be as close as possible to $I(C;\Lambda)$ for a fixed number of quantizer levels Q, [44] proposes to maximize $I(\Lambda;\hat{\Lambda})$. This mutual information can be expressed as

$$I(\Lambda;\hat{\Lambda}) = H(\hat{\Lambda}) - H(\hat{\Lambda}|\Lambda) = H(\hat{\Lambda}), \tag{3.10}$$

where $H(\hat{\Lambda}|\Lambda)$ is zero since $\hat{\Lambda} = \mathcal{Q}(\Lambda)$ is a deterministic function of Λ. Thus the equiprobable output quantizer maximizes $H(\hat{\Lambda})$ which is achieved by a uniform distribution of $\hat{\Lambda}$. This implies that the quantizer decision boundaries g_i have to be such that $\mathrm{P}\{\hat{\Lambda} = \lambda_i\} = 1/Q$. Using the cumulative distribution function (cdf)

$$F_\Lambda(\lambda) = \mathrm{P}\{\Lambda \leq \lambda\} = \int_{-\infty}^{\lambda} p_\Lambda(\xi) d\xi \tag{3.11}$$

the optimal decision boundaries can be obtained by finding the arguments of $F_\Lambda(\lambda)$ for which $F_\Lambda(\lambda) = i/Q$, i.e.,

$$g_i = F_\Lambda^{-1}(i/Q), \qquad i = 1, 2, \ldots, Q-1. \tag{3.12}$$

As before we set $g_0 = -\infty$ and $g_Q = \infty$.

It is important to realize that the mutual information $I(C; \hat{\Lambda})$ only depends on the distribution $p(\hat{\Lambda})$, i.e., on the probabilities $\mathrm{P}\{\hat{\Lambda} = \lambda_i\}$. The actual values of the quantizer reproducers λ_i do not influence the mutual information, however, they are important in order to provide the decoder with proper reliability information. In view of the equivalent discrete channel with input c and output $\hat{\Lambda}$ the quantizer reproducer values λ_i are chosen as LLRs given by [46]

$$\lambda_i = \log \frac{\mathrm{P}\{c = 1 \,|\, \Lambda \in \mathcal{R}^{(i)}\}}{\mathrm{P}\{c = 0 \,|\, \Lambda \in \mathcal{R}^{(i)}\}} = \log \frac{P_{1,i}}{P_{0,i}}. \tag{3.13}$$

Since closed-form expressions for the distribution of the LLRs Λ are not available, except for a few special cases, Monte Carlo simulations are generally required to determine the probabilities $P_{1,i}$ and $P_{0,i}$. However, carefully note that in [44] the authors also proposed a method for designing the equiprobable output quantizer during data transmission. In [44] it has been shown that this on-the-fly quantizer design leads only to a negligible performance loss compared to offline quantizer design. Therefore knowledge of the (conditional) LLR distributions is not mandatory for the design of the equiprobable output quantizer.

The equiprobable output quantizer can be extended to N dimensional vector quantization by applying N independent equiprobable output quantizers, one for each dimension, if all vector components are statistically independent.

3.5 The Information Bottleneck Method

As we have mentioned before, an analysis of quantization in terms of the of rate-distortion theory requires that we fix a specific distortion measure in advance. However, the “right” distortion measure is rarely available or known. We therefore consider the approach presented in [47] by Tishby et al. who have introduced the notion of *relevance through another variable*. Note that information theory in its original formulation by Shannon does not provide a quantitative notion of “relevant” information.

Relevant information in a random variable X is defined as the information this random variable provides about another random variable Y. The variable Y is called *relevance variable.* Clearly, X and Y must not be statistically independent. The aim of the information bottleneck method (IBM) is to find a representation of X, using as few bits (or codewords) as possible, while preserving a certain amount of information about Y. The name "information bottleneck method" comes from the idea that we squeeze the information that X provides about Y through a bottleneck formed by the limited set of codewords. In contrast to rate-distortion theory the IBM does not constrain the average distortion. Rather, it requires that $\widetilde{X}$, the quantized version of X, provides a minimum amount of information about Y. Note that this is equivalent to maximizing the relevant information for a fixed compression of the original variable X. This constrained optimization problem can be seen as a generalization of rate-distortion theory in which the distortion measure *emerges* from the joint statistics of X and Y. For the IBM we have to assume that we have access to the joint distribution $p(x, y)$.

In terms of the setting described in Chapter 2 this means that we want to quantize the LLRs Λ_r, yielding a quantized representation Z_r, where we require that Z_r carries a minimum amount of information about the network-coded bits $C_n = C_1 \oplus C_2'$ (cf. Figure 2.5). More specifically, we want to solve the optimization problem

$$\min_{p(z_r|\lambda_r)} I(\Lambda_r; Z_r) \qquad \text{subject to} \quad I(C_n; Z_r) \geq I'. \tag{3.14}$$

As in the previous section we observe that $C_n - \Lambda_r - Z_r$ forms a Markov chain. Due to the data processing inequality we have $I(C_n; Z_r) \leq I(C_n; \Lambda_r)$, which implies that $0 \leq I' \leq I(C_n; \Lambda_r)$. Since the IBM is phrased in terms of discrete random variables we have to treat Λ_r as a discrete random variable in the following. Hence, we can think of the distribution $p(\lambda_r)$ as a discretized version of the true pdf of Λ_r, where the discretization is performed with a sufficiently high resolution.

Let us now take a more detailed look at the left hand side of the constraint in (3.14). We obtain

$$\begin{aligned}
I(C_n; Z_r) &\overset{(1)}{=} I(C_n; \Lambda_r) + \underbrace{I(C_n; Z_r|\Lambda_r)}_{=0} - I(C_n; \Lambda_r|Z_r) \\
&= I(C_n; \Lambda_r) - \sum_{c_n,\lambda_r,z_r} p(c_n, \lambda_r, z_r) \log \frac{p(c_n, \lambda_r|z_r)}{p(c_n|z_r)p(\lambda_r|z_r)} \\
&\overset{(2)}{=} I(C_n; \Lambda_r) - \sum_{\lambda_r,z_r} p(\lambda_r, z_r) \sum_{c_n} p(c_n|\lambda_r) \log \frac{p(c_n|\lambda_r)}{p(c_n|z_r)} \\
&= I(C_n; \Lambda_r) - \sum_{\lambda_r,z_r} p(\lambda_r, z_r) \mathrm{D}(p(c_n|\lambda_r) \,||\, p(c_n|z_r)) \\
&= I(C_n; \Lambda_r) - \mathrm{E}\left\{\mathrm{D}(p(c_n|\lambda_r) \,||\, p(c_n|z_r))\right\},
\end{aligned} \tag{3.15}$$

where $\mathrm{D}(\cdot\,||\,\cdot)$ denotes the Kullback-Leibler divergence. In (3.15) equality (1) is obtained by expanding $I(C_n;\Lambda_r,Z_r)$ as

$$I(C_n;\Lambda_r,Z_r)=I(C_n;Z_r)+I(C_n;\Lambda_r|Z_r)=I(C_n;\Lambda_r)+I(C_n;Z_r|\Lambda_r). \tag{3.16}$$

Furthermore $I(C_n;Z_r|\Lambda_r)$ is equal to zero since $Z_r=\mathcal{Q}(\Lambda_r)$ is a deterministic function of Λ_r. Equality (2) is obtained by applying the chain rule for probability distributions and by noting that $p(c_n|\lambda_r,z_r)=p(c_n|\lambda_r)$ due to the Markov chain property.

From (3.15) we can see that the Kullback-Leibler divergence has emerged as the right distortion measure for our problem. We can now rewrite (3.14) in the way that is usual for rate-distortion problems, i.e.,

$$\min_{p(z_r|\lambda_r)} I(\Lambda_r;Z_r) \qquad \text{subject to} \quad \mathrm{E}\left\{\mathrm{D}(p(c_n|\lambda_r)\,||\,p(c_n|z_r))\right\}\le I'', \tag{3.17}$$

where we have set $I''=I(C_n;\Lambda_r)-I'$. In [47] the authors have shown that the optimal probability assignment solving (3.17) is given by

$$p(z_r|\lambda_r)=\frac{p(z_r)}{\zeta(\lambda_r,\beta)}\exp\left[-\beta\,\mathrm{D}(p(c_n|\lambda_r)\,||\,p(c_n|z_r))\right], \tag{3.18}$$

with some $\beta>0$ and a normalization term (partition function) $\zeta(\lambda_r,\beta)$ which ensures that $p(z_r|\lambda_r)$ is a valid probability distribution. However, it must be emphasized that (3.18) is only a formal solution since both $p(z_r)$ and $p(c_n|z_r)$ are defined implicitly in terms of $p(z_r|\lambda_r)$. For $p(c_n|z_r)$ we obtain

$$p(c_n|z_r)=\frac{1}{p(z_r)}\sum_{\lambda_r}p(c_n,\lambda_r)p(z_r|\lambda_r), \tag{3.19}$$

where we have used Bayes rule and the Markov chain condition $C_n-\Lambda_r-Z_r$. Clearly, due to consistency, $p(z_r)$ can be expressed as

$$p(z_r)=\sum_{\lambda_r}p(z_r|\lambda_r)p(\lambda_r). \tag{3.20}$$

Note that in all expressions above we assume that the joint distribution $p(c_n,\lambda_r)$ is known. Thus, also for the IBM, we have to obtain the conditional LLR statistics by Monte Carlo simulations.

Although $p(z_r|\lambda_r)$ cannot be obtained in closed form, an iterative algorithm can be used to obtain the solution of (3.17) numerically. The iterative IBM algorithm is guaranteed to converge to a local optimum and can be seen as a generalization of the

Blahut-Arimoto algorithm [48], which is used for the numerical computation of rate-distortion functions and channel capacities. In Algorithm 1 we summarize the iterative IBM algorithm for the computation of $p(z_r|\lambda_r)$.

Algorithm 1 *Iterative IBM algorithm for the computation of* $p(z_r|\lambda_r)$.

```
Input: $p(c_n, \lambda_r)$, $\beta$, $Q = |\mathcal{Z}|$ (number of quantizer levels), $K$ (number of iterations)
Require: $\beta > 0, Q \geq 2, K \geq 1$ and $p(c_n, \lambda_r)$ is a valid probability distribution
Initialization: randomly choose $p_0(z_r|\lambda_r)$, $k \leftarrow 1$
  while $k \leq K$ do
    $p_k(z_r) \leftarrow \sum_{\lambda_r} p_{k-1}(z_r|\lambda_r)p(\lambda_r)$
    $p_k(c_n|z_r) \leftarrow \frac{1}{p_k(z_r)} \sum_{\lambda_r} p(c_n, \lambda_r)p_{k-1}(z_r|\lambda_r)$
    $p_{k+1}(z_r|\lambda_r) \leftarrow p_k(z_r) \exp\left[-\beta\, \mathrm{D}(p(c_n|\lambda_r) \,||\, p_k(c_n|z_r))\right]$
    $p_{k+1}(z_r|\lambda_r) \leftarrow p_{k+1}(z_r|\lambda_r) / \sum_{z_r} p_{k+1}(z_r|\lambda_r)$
    $k \leftarrow k + 1$
  end while
  $p(z_r|\lambda_r) \leftarrow p_k(z_r|\lambda_r)$
```

Clearly, our quantizer design using the iterative IBM algorithm should yield a deterministic mapping, i.e., each λ_r is mapped to exactly one z_r. This means that for each λ_r we have $p(z_r^\star|\lambda_r) = 1$ for one specific $z_r = z_r^\star$ and $p(z_r|\lambda_r) = 0$ for all other z_r. In order to obtain such a $0-1$ probability distribution for $p(z_r|\lambda_r)$ the value of β has be to chosen large enough. In our quantizer design we have chosen β such that $100 \leq \beta \leq 300$, with increasing values of β for higher source-relay SNRs. Note that values of β that are too large will likely lead to numerical problems in the IBM algorithm.

With $p(z_r|\lambda_r)$ the Q quantizer decision regions $\mathcal{R}^{(i)}$ are fixed. The corresponding quantizer reproducer values λ_i are not determined by the IBM since they are irrelevant for the mutual information. However, in order to provide the decoder in our PLNC scheme with proper reliability information, we again set the reproducer values according to (3.13).

We emphasize that the choice of $p_0(z_r|\lambda_r)$ is important, since the iterative IBM algorithm will in general only converge to a local optimum. It is vital for the quality of the result $p(z_r|\lambda_r)$ to choose $p_0(z_r|\lambda_r)$ at random in the initialization of the IBM algorithm. In order to obtain good results, multiple instances of the IBM algorithm can be run on a sufficiently large number of different initializations $p_0(z_r|\lambda_r)$. The best mapping $p(z_r|\lambda_r)$ in terms of mutual information loss is then chosen as the result of the quantizer design procedure.

The number of iterations needed until the result of the IBM algorithm converges sufficiently well generally depends on the source-relay SNR. We observed that for lower source-relay SNRs the required number of iterations is larger that for higher source-relay SNR. For high source-relay SNR, i.e., when $I(C_n; \Lambda_r)$ is close to 1 bit, roughly 20 iterations are sufficient for convergence, whereas for low source-relay SNR about 100 iterations may be necessary to reach convergence.

It is important to note that we applied the IBM in our quantization problem to maximize the mutual information $I(C_n; Z_r)$ for a fixed rate, i.e., for a fixed number of quantizer levels Q. However, due to the transmission of the quantizer indices z_r over a noisy channel from the relay to the destination, the decoded quantizer indices $\hat{z}_r$ at the destination might not be equal to the ones that were transmitted (cf. Figure 2.9). Therefore we should include the effect of the relay-destination channel in our quantizer design, i.e., we should maximize the mutual information $I(C_n; \hat{Z}_r)$ for a fixed number of quantizer levels (instead of $I(C_n; Z_r)$). But due to our assumptions regarding channel state information this is impossible, since the relay has no CSI about the relay-destination link.

3.6 Quantizer Comparison

In this section we will compare the three quantizer design approaches described in the previous sections. We apply the respective quantizers to our PLNC scheme for the MARC, where we use the RSCC depicted in Figure 1.3 as channel code and we model all links as fast i.i.d. Rayleigh flat fading channels.

Figure 3.3 shows the LLR statistic and the corresponding quantizer design results for quantizers with $Q = 4$ levels at a source-relay SNR of $\rho_{sr} = 3.75\,\text{dB}$. Note that due to symmetry we plot the results in Figure 3.3 only for $\lambda_r > 0$. In this figure, the quantizer decision boundaries are indicated by vertical lines and the reproducer values are plotted as squares on the abscissa. It is interesting to note that the decision boundaries and reproducer values of the IBM quantizer are the smallest, then come those of the equiprobable output quantizer and the Lloyd quantizer yields the largest values for the decision boundaries and quantizer reproducers. In addition it can be seen that the reproducer values of the Lloyd quantizer are significantly larger than the reproducer values of the other quantizers. This is due to the fact that the Lloyd quantizer chooses the reproducers such that the MSE is minimized (cf. Equation (3.9)), whereas for the other two quantizers the reproducer values are given by (3.13).

Since $\Lambda_r = \Lambda(c_1) \boxplus \Lambda(c_2')$, the decision boundaries g_i in Figure 3.3 are related to

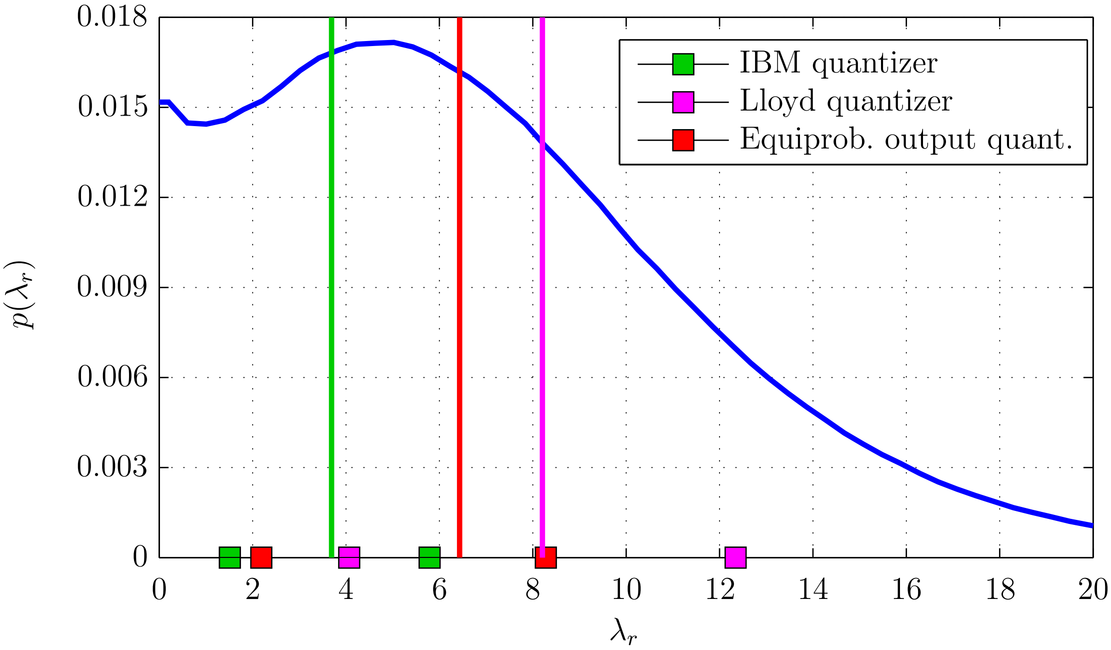


Figure 3.3: *LLR distribution, quantizer decision boundaries and quantizer reproducer values for $Q = 4$ levels at a source-relay SNR of 3.75 dB. Note that the LLR statistics and the quantizer designs are symmetric around $\lambda_r = 0$.*

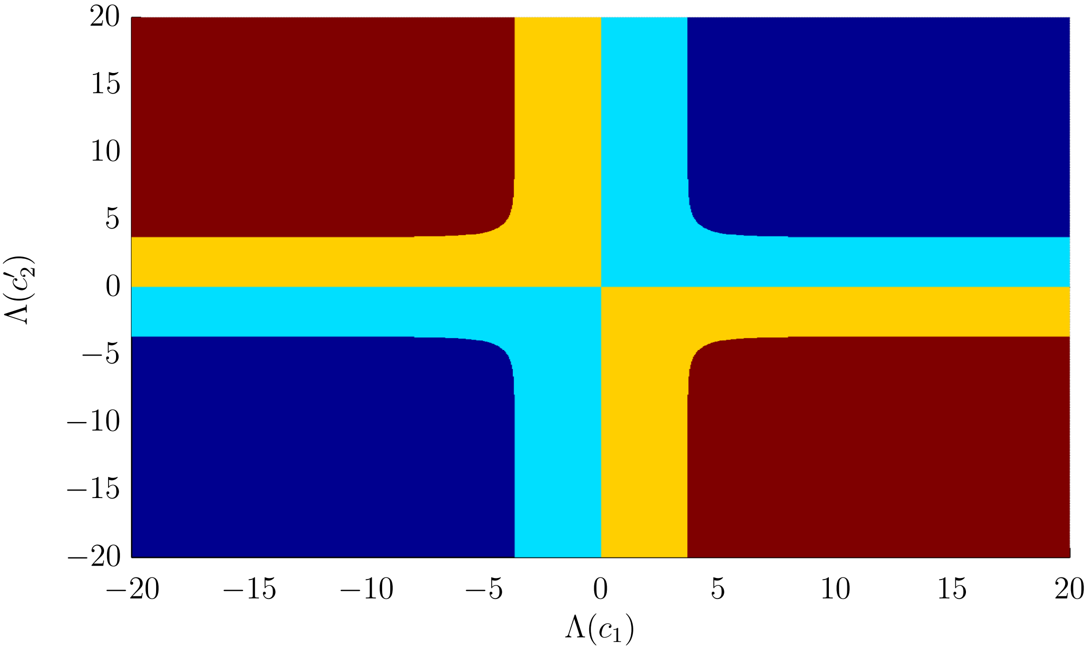


Figure 3.4: *Two dimensional representation of the IBM quantizer decision regions from Figure 3.3 in terms of $\Lambda(c_1)$ and $\Lambda(c_2')$. The boundaries of the two dimensional decision regions are given by $\Lambda(c_1) \boxplus \Lambda(c_2') = g_i$.*

the respective code bit LLRs by $\Lambda(c_1) \boxplus \Lambda(c_2') = g_i$. A two dimensional representation of the quantizer decision regions of the IBM quantizer from the example in Figure 3.3 is depicted in Figure 3.4. The shape of the borders of the decision regions directly reflects the soft XOR encoding rule. The boxplus operation together with the scalar IBM quantizer which lead to the result depicted in Figure 3.3 are equivalent to a two dimensional vector quantizer, with the decision regions shown in Figure 3.4, operating directly on $\Lambda(c_1)$ and $\Lambda(c_2')$. However, note that the two dimensional quantizer decision regions are disconnected. Here, each decision region consists of two disjoint components. This is in contrast to the Voronoi tessellation generated by usual vector quantization algorithms which lead to connected Voronoi cells.

Figure 3.5 shows the quantizer decision boundary g_3 versus the source-relay SNR ρ_{sr} for $Q = 4$ levels. Note that all other decision boundaries are determined by (3.2). Here we can see that there is a large difference between the value of the decision boundary of the IBM quantizer and the other quantizers. Furthermore the value of the decision boundary of the Lloyd quantizer is always the largest, whereas for the equiprobable output quantizer it is the smallest for low SNR and closely follows the value of the decision boundary of the Lloyd quantizer for high SNR. In addition we can observe that it should be possible to approximate the curve corresponding to the IBM quantizer by a (piecewise) linear function with small approximation error. This would allow for on-the-fly design of the quantizer during data transmission.

In Figure 3.6 we plot the quantizer reproducer values λ_3 and λ_4, again for $Q = 4$ levels, versus the the source-relay SNR ρ_{sr}. Here, the dashed lines correspond to the larger reproducer value λ_4, whereas the solid lines correspond to λ_3. Note that $\lambda_1 = -\lambda_4$ and $\lambda_2 = -\lambda_3$ due to symmetry. We note that the behavior of the reproducer values versus the SNR is similar to the behavior of the quantizer decision boundaries versus the SNR. However, we observe that λ_3 of the equiprobable output quantizer follows the result of the IBM quantizer much more closely than it does for λ_4. Again, we note that an approximation of the curves corresponding to the IBM quantizer by piecewise linear functions should lead to good results.

Figure 3.7 shows a comparison of the three quantizer design approaches we have considered in terms of the mutual information loss $\Delta I = I(C_1, C_2'; \Lambda_r) - I(C_1, C_2'; Z_r)$. Note that $I(C_1, C_2'; \cdot) = I(C_n; \cdot)$, with $C_n = C_1 \oplus C_2'$. The comparison in terms of mutual information loss is motivated by the data processing inequality, i.e., we want to lose as little information as possible due to the quantization of the LLRs Λ_r. We will show in Chapter 4 that the metric of mutual information loss is indeed relevant for the performance of our PLNC scheme. The mutual information on the abscissa of Figure 3.7 is shown in Figure 2.6 versus the source-relay SNR ρ_{sr}.

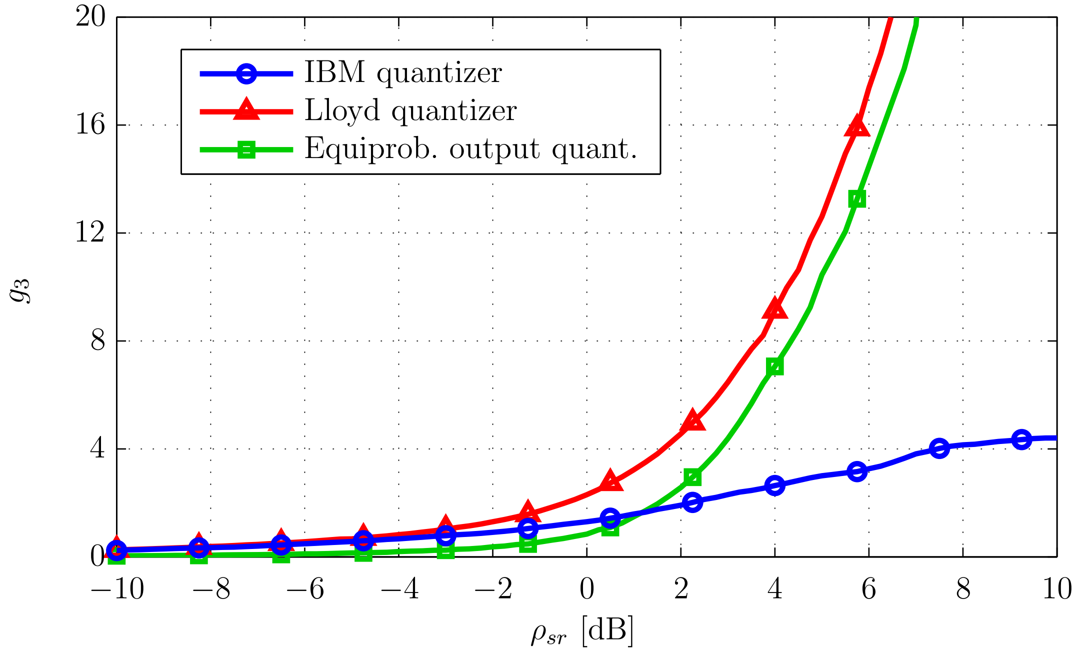


Figure 3.5: *Quantizer decision boundary g_3 versus the source-relay SNR ρ_{sr} for quantizers with $Q = 4$ levels. Note that the other quantizer boundaries are given by $g_0 = -\infty, g_1 = -g_3, g_2 = 0$ and $g_4 = \infty$ due to symmetry.*

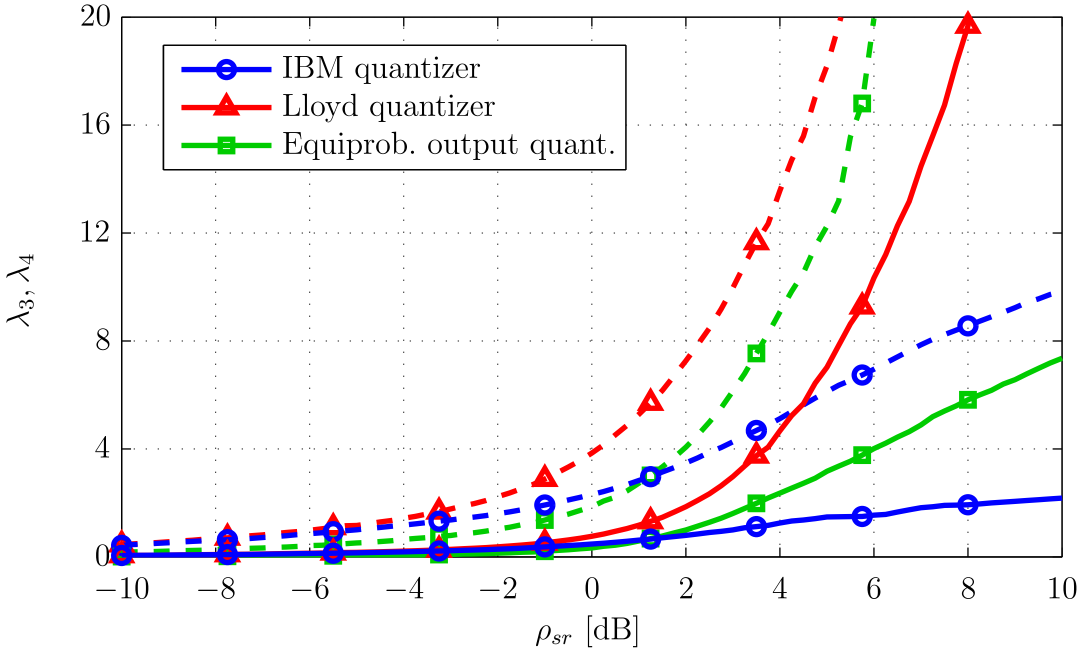


Figure 3.6: *Quantizer reproducer values λ_3 (solid lines) and λ_4 (dashed lines) versus the source-relay SNR ρ_{sr} for quantizers with $Q = 4$ levels. Note that the other reproducer values are given by $\lambda_1 = -\lambda_4$ and $\lambda_2 = -\lambda_3$ due to symmetry.*

We compare the different quantizers for $2, 4$ and 8 levels. For an odd number of levels both the equiprobable output quantizer and the Lloyd quantizer show rather poor performance, particularly for relatively high source-relay SNR or, equivalently, for $I(C_1, C_2'; \Lambda_r)$ close to 1. Note that when Q is odd we have a quantizer reproducer value at $\lambda = 0$. Therefore, the equiprobable output quantizer will, especially for high SNR, map a large part of the LLR distribution to $\lambda = 0$ which leads to a large mutual information loss.

From Figure 3.7 we can see that for $Q = 2$ levels all quantizers show equal performance. This is due to the fact that the only (finite) quantizer decision boundary is, due to symmetry, at $g_1 = 0$ for $Q = 2$. Since the choice of the quantizer reproducer values does not effect the mutual information, it is obvious that all quantizers with 2 levels are equal in terms of mutual information loss. Note that a quantization of the LLRs with $Q = 2$ levels, i.e., with 1 bit/LLR, corresponds to a hard decision on the sign of the LLRs. Furthermore, since we use a scalar quantizer, the rate cannot be any smaller than 1 bit/LLR.

Clearly, if the number of levels is increased, the mutual information loss will decrease. For 4 and 8 quantizer levels, we can see that the IBM quantizer is superior to the other quantizers in terms of mutual information loss. This is due to the fact that the IBM quantizer maximizes $I(C_1, C_2'; Z_r)$ and thus minimizes the mutual information loss ΔI. For those values of the mutual information $I(C_1, C_2'; \Lambda_r)$ which are relevant for the operation of our PLNC scheme, the equiprobable output quantizer outperforms the Lloyd quantizer. However, we can see that the maximization of $I(\Lambda_r; Z_r)$ performed by the equiprobable output quantizer does not minimize the mutual information loss. In this comparison the Lloyd quantizer performs worst, which means that quantizer design for minimum MSE distortion is far away from minimizing the mutual information loss that is due to the quantization. Finally it is interesting to note that the maximum of the mutual information loss seems to be shifted to smaller values of the mutual information with an increasing number of levels for the IBM quantizer, whereas for the other quantizers just the opposite is the case.

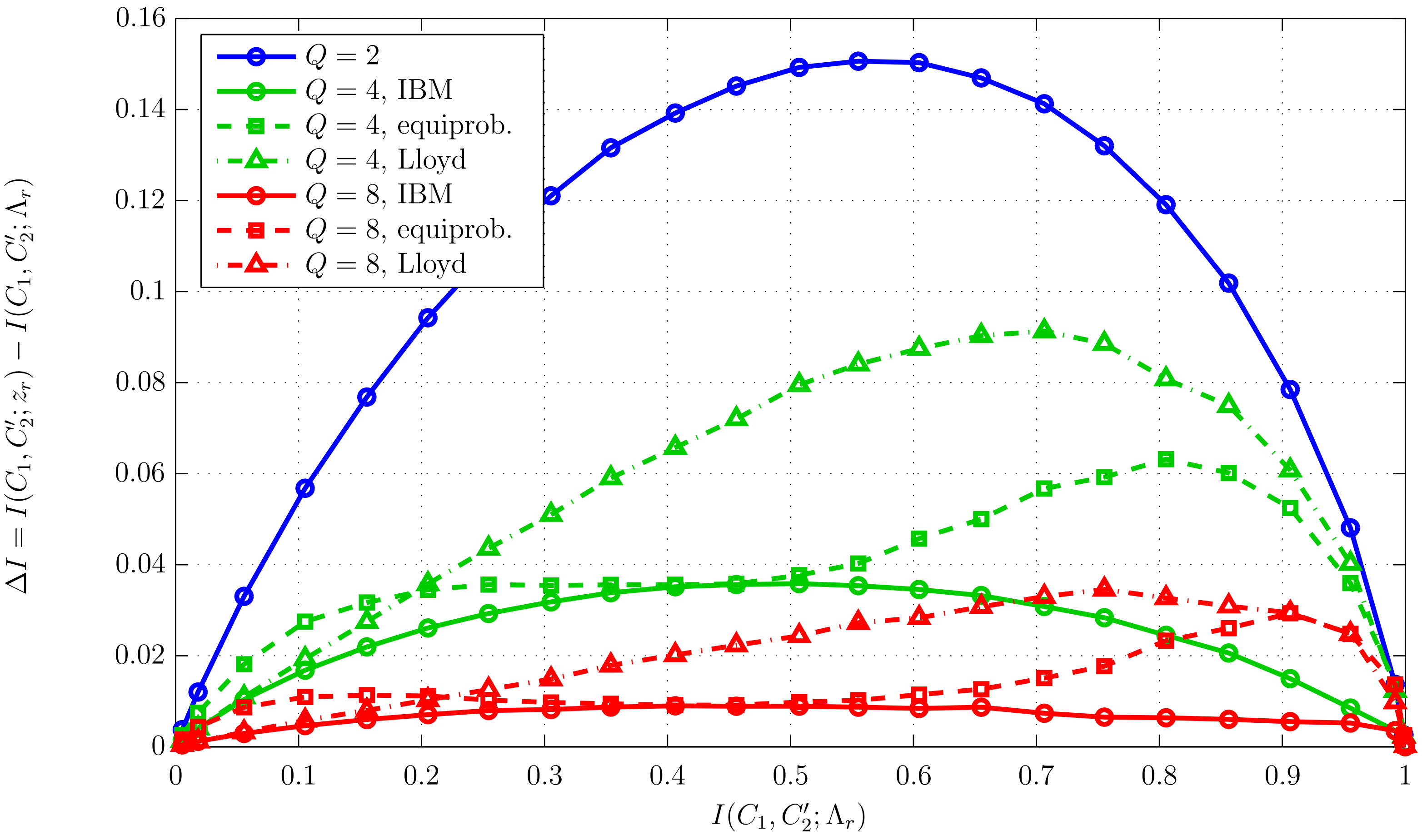


Figure 3.7: *Quantizer comparison in terms of mutual information loss ΔI. Note that for $Q = 2$ levels all quantizer designs are equal since the only decision boundary $g_1 = 0$ for all quantizers. The IBM quantizer is designed to minimize the mutual information loss and thus it is superior to all other quantizers in this comparison.*

4

Simulation Results

4.1 Introduction

In this chapter we will present numerical simulation results of the PLNC scheme described in Chapter 2. We show the BER performance of our scheme for a variety of system parameters and transmission scenarios and provide comparisons to the performance of the reference systems introduced in Section 2.6. Furthermore, we will compare the LLR quantizers described in Chapter 3 in terms of BER performance.

In Section 4.2 we consider the symmetric MARC, i.e., we assume equal SNR values on the source-relay channels and on the source-destination channels respectively. Performance results for the general case, i.e., for the non-symmetric MARC, are presented in Section 4.3.

Unless otherwise noted, we have used the following system parameters in order to obtain the BER results presented in this chapter.

- The input-output relation of the individual channels is given by (2.3), where we assume fast i.i.d. Rayleigh flat fading.
- Each source uses the RSCC depicted in Figure 1.3 with octal generator polynomial 37_8 and feedback polynomial 23_8. The sources do not employ puncturing of the encoder output and therefore the code rate $R_{c,i}$ is equal to 1/2. The convolutional codes are terminated and the resulting blocklength is 8000 bits per codeword.
- The relay always uses the same channel code as the sources.
- Each node transmits Gray labeled QPSK symbols defined according to (2.7).

- LLR quantization at the relay is performed using a scalar quantizer designed using the information bottleneck method (cf. Section 3.5).
- We do not perform data compression, i.e., lossless source coding, on the sequence of quantizer indices.
- The number of iterations of the joint network-channel decoder is fixed to 10.

Note that the plots in the following sections show the BER versus the average source-destination channel SNR. This is especially important to note in case both source use a different code rate, since this will change the E_b/N_0 ratio, but will, of course, leave the channel SNR defined in (2.5) unchanged.

In order to allow for a fair comparison between our scheme and the reference systems from Section 2.6, we compensate for the possibly different transmission (frame) durations, i.e., for the different time sharing factors α, by adapting the SNR according to (2.20). Moreover, note that the transmit energy per frame of the relay is fixed to E_r. Therefore, depending on the number of channel uses M_r, the average transmit power of the relay is scaled by the factor β according to (2.6).

4.2 The Symmetric Case

In this section we consider the case of the symmetric MARC. This means that we have $\rho_{sr} = \rho_{1r} = \rho_{2r}$ and $\rho_{sd} = \rho_{1d} = \rho_{2d}$. Therefore, under the assumption that both sources use the same channel code, the average BER is equal for both sources. The figures in this section show the BER versus the source-destination SNR ρ_{sd}. In most cases the BER curves are parametrized by the relay-destination SNR ρ_{rd}.

Coded LLR transmission versus analog LLR forwarding. In Figure 4.1 we compare our PLNC scheme to the system proposed in [9]. Note that we use coded transmission of the LLRs from the relay to the destination whereas the LLR forwarding in [9] is modeled as an analog transmission according to (2.1). Here we assume AWGN channels instead of fast fading channels and we fix the source-relay SNR to $\rho_{sr} = 5\,\text{dB}$. For the LLR quantization we use only $Q = 2$ quantizer levels, i.e., 1 bit/LLR. From Figure 4.1 we can see that the coded LLR transmission (solid lines) outperforms the analog LLR transmission (dashed lines). However, the gap between the two schemes is reduced as ρ_{rd} increases. The blue line in Figure 4.1 shows the BER performance achieved when the relay is present but silent, yielding $\rho_{rd} = -\infty\,\text{dB}$. In this case the performance is of course equal for both systems.

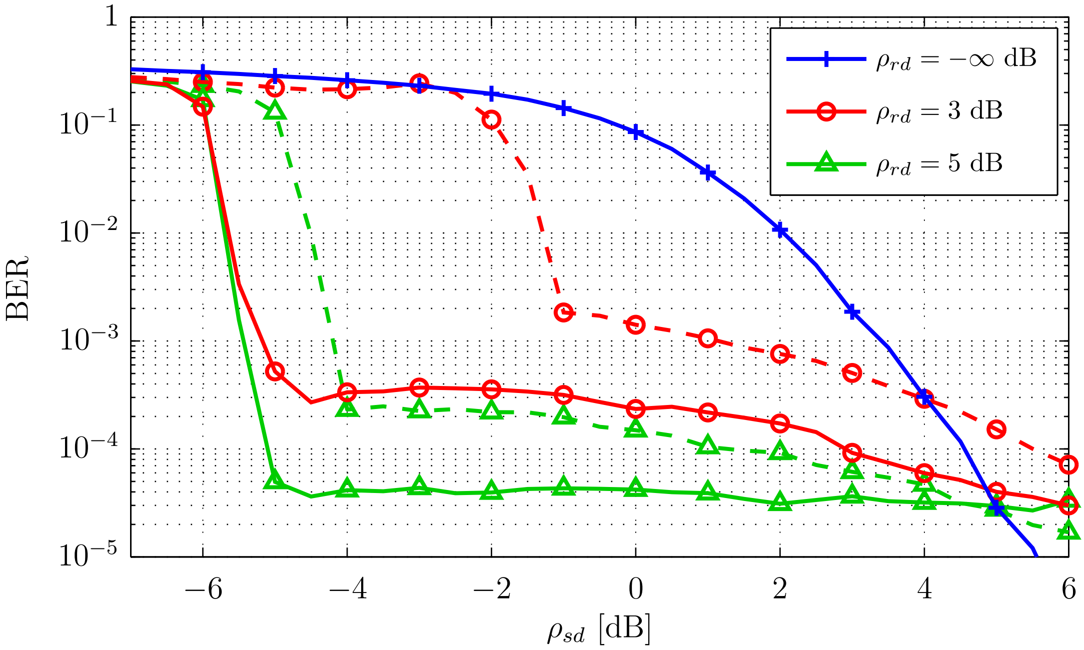


Figure 4.1: *Comparison between coded LLR transmission (solid lines) and analog LLR forwarding (dashed lines). Here, the channels are modeled as AWGN channels, ρ_{sr} is fixed to 5 dB and we use $Q = 2$ quantizer levels.*

Error floor behavior. We can observe from Figure 4.1 that both systems exhibit an error floor. This error floor is due to the imperfect coupling between the channel decoders in the iterative joint network-channel decoder. Perfect coupling between the channel decoders would imply that $\widetilde{\Lambda}_r = \Lambda_r \rightarrow \pm\infty$. However, the LLRs Λ_r are possibly small, i.e., $|\Lambda_r| \neq \infty$, depending on the quality of the source-relay channels, and furthermore they are not available perfectly at the destination, i.e., $\widetilde{\Lambda}_r \neq \Lambda_r$. These "wrong" LLRs are subsequently "injected" into the iterative decoder via the network decoder (cf. Figure 2.11). Thus the noisy transmission from the sources to the relay and the noisy transmission from the relay to the destination are the two main reasons for the presence of the error floor. In addition, the LLRs are distorted by the quantization. Clearly, LLR quantization is unavoidable for coded LLR transmission and therefore the additional quantization noise can only be reduced by increasing the number of quantizer levels. The noise on the source-relay and relay-destination channels should be combated by the use of suitable channel codes and modulation schemes.

The above reasoning is supported by the fact that the error floor of the analog LLR forwarding scheme vanishes as $\rho_{rd} \rightarrow \infty$, if the source-relay SNR ρ_{sr} is sufficiently high. In that case the iterative decoder at the destination performs like a turbo decoder, where the constituent codes are those used by the two sources. Note that the error floor does not vanish as $\rho_{rd} \rightarrow \infty$ in the case of coded (quantized) LLR transmission, since the quantization noise is independent of the relay-destination SNR ρ_{rd}. For a fixed quantizer choice and a given ρ_{sr}, the error floor can only be lowered by using a strong channel code on the relay-destination link and by taking care that ρ_{rd} is large enough for reliable transmission of the coded LLRs from the relay to the destination, e.g., by clever deployment of the relay.

Decoding threshold. From Figure 4.1 we can see that the decoding threshold, in terms of the source-destination SNR ρ_{sd}, is located at very low values of ρ_{sd}. Communication at a reasonable BER is possible using our PLNC scheme at source-destination SNRs as low as $\rho_{sd} = -5$dB, under the assumptions that lead to the results in Figure 4.1. Furthermore note that in our scheme the decoding threshold is not (significantly) shifted to higher SNR values as ρ_{rd} decreases, as long as the BER on the relay-destination link is not too high (which depends on the channel code that is employed by the relay). This is a further advantage of coded LLR transmission when compared to analog LLR forwarding.

In case the destination is confident about the sum of the transmitted code bits $c_1 \oplus c_2'$, i.e., $I(C_1, C_2'; \hat{Z}_r)$ is large, it seems that only very little information from the sources is necessary at the destination in order to induce coupling between the individual channel decoders in the iterative joint network-channel decoder and thereby obtaining turbo decoding gains. Thus, given a sufficiently high source-relay SNR ρ_{sr}, the decoding threshold is mainly determined by the reliability of the LLRs coming directly from the sources, whereas the error floor is determined by the quality of the LLRs transmitted by the relay, i.e., by the number of quantizer levels and the BER on the relay-destination link. This reasoning is backed up by results shown in Section 4.3.

The effect of decoder iterations. Figure 4.2 shows the effect of the iterations of our joint network-channel decoder. Here, the system parameters are the same as for Figure 4.1 and we have fixed ρ_{rd} to 5 dB. From Figure 4.2 we note that the first iteration yields the largest gain, whereas after five iterations the gains are already small. Moreover we see that (at least in this SNR regime) the iterative decoding procedure converges sufficiently well after a relatively small number of iterations. Furthermore it is interesting to see that the error floor is established after only two iterations.

The black curve in Figure 4.2 is equal to the solid green curve in Figure 4.1 since we run 10 decoder iterations in total. However, the blue curve in Figure 4.2 is different from the blue curve in Figure 4.1, since the former is obtained after one (half) iteration.

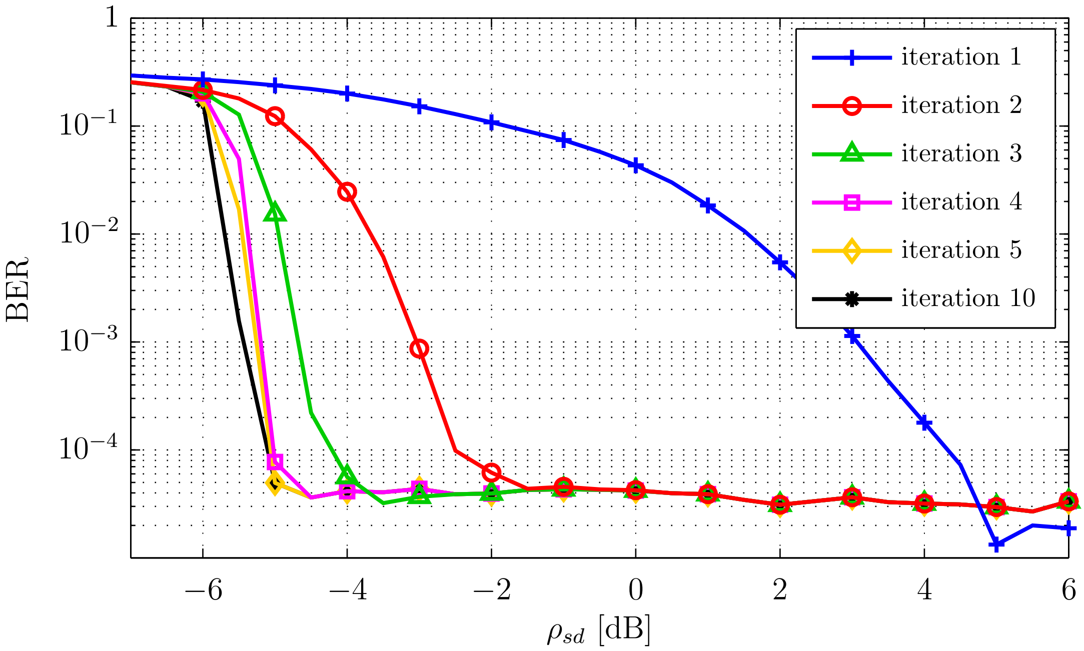


Figure 4.2: *The effect of decoder iterations. The system parameters are the same as in Figure 4.1. Here, the relay-destination SNR is fixed to 5 dB.*

Note that in the first iteration of our joint network-channel decoder, only the second channel decoder obtains prior information from the network decoder. The first channel decoder has to decode using only the direct observation from the respective source in the first iteration. Therefore the average BER after one iteration is only slightly better than in the case where the destination receives no information from the relay. As shown in Figure 4.2 the average BER drops significantly after the second iteration, i.e., after *both* channel decoders have exchanged information (via the network decoder) for the first time.

Comparison to a transmission without relay. Let us now investigate the performance gain of our PLNC scheme compared to a transmission without relay. Figure 4.3 shows the corresponding results for a source-relay SNR of $\rho_{sr} = 10$ dB and $Q = 2$ quantizer levels. Here, we assume fast i.i.d. Rayleigh flat fading channels (as we will henceforth always do). The comparison in Figure 4.3 is fair in the sense that we compensate for the different time sharing factors α, i.e., for the different transmission rates. For the case without relay, both source equally share the available transmission time, yielding $\alpha_{\text{ref}} = 1/2$. For our PLNC scheme we have $\alpha = 1/4$ and therefore, according to (2.20), the BER curve for the case without relay is shifted by 3 dB towards lower

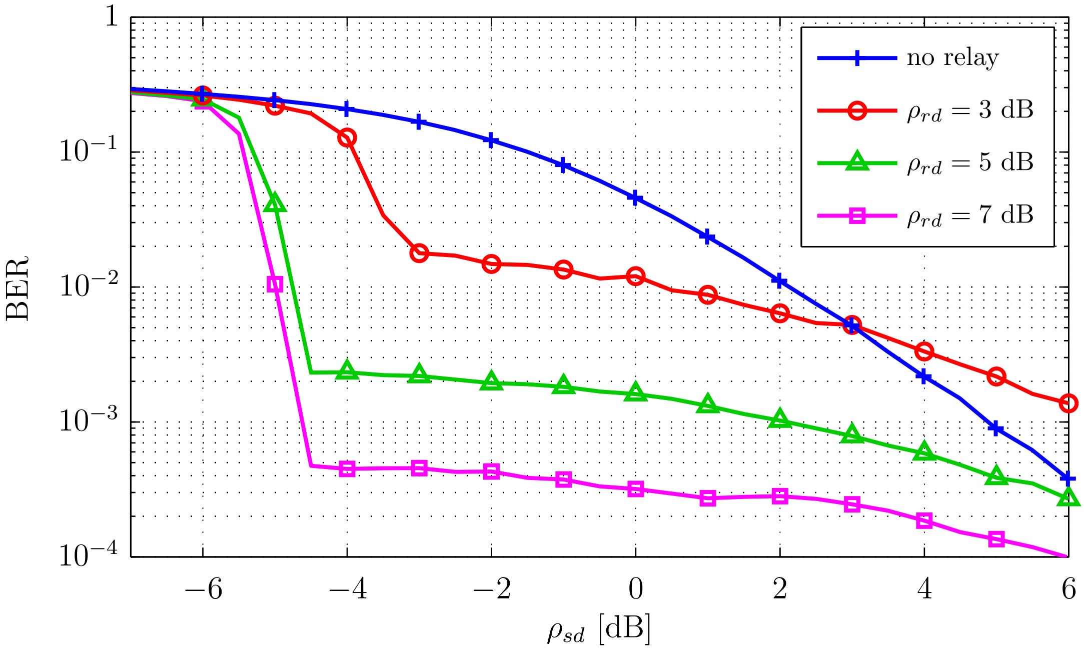


Figure 4.3: *Comparison between our PLNC scheme and a transmission without relay. The source-relay SNR ρ_{sr} is fixed to 10 dB and we use $Q = 2$ quantizer levels. We observe an SNR gain of up to 9.5 dB in this example.*

SNRs. The average transmit power of the relay is P_A since $\beta = 1$ according to (2.6).

Figure 4.3 shows that the we can achieve large gains with our PLNC scheme. Here, we obtain an SNR gain of approximately 9.5 dB (for $\rho_{rd} = 7$ dB) compared to the case without relay. We can furthermore see that the decoding threshold is shifted by approximately 1 dB compared to a transmission over AWGN channels. This is intuitive since for fading channels we need a higher SNR than in the AWGN case to convey the same amount of information to the destination. The same is true with respect to the error floor. In order to lower the error floor we would need higher SNRs ρ_{rd} and ρ_{sr} or we could use stronger channel codes.

Quantizer comparison in terms of BER. In Figure 4.4 we show the BER performance of our PLNC scheme using different quantizers for a source-relay SNR of $\rho_{sr} = 7$ dB and $Q = 4$ quantizer levels, i.e., 2 bits/LLR. We compare the quantizers resulting from the quantizer design methods presented in Chapter 3. We can see that the IBM quantizer (solid lines) is clearly superior to the other quantizers. Furthermore the equiprobable output quantizer (dashed lines) performs better than the Lloyd quantizer (dash-dotted lines). Thus we see that the results in terms of mutual information

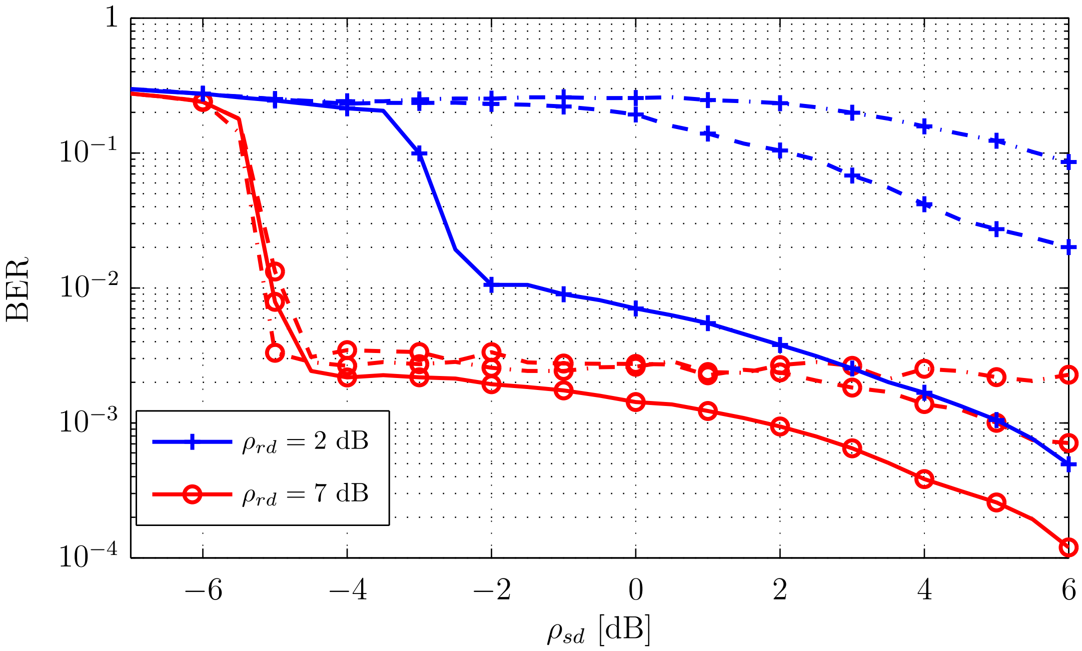


Figure 4.4: *Quantizer comparison in terms of BER for $Q = 4$ quantizer levels and a source-relay SNR of $\rho_{sr} = 7$ dB. The IBM quantizer (solid lines) is superior and the equiprobable output quantizer (dashed lines) performs better than the Lloyd quantizer (dash-dotted lines).*

loss from Chapter 3 are reflected directly in this BER comparison. This means that mutual information loss is a proper measure for the design of quantizers in digital communication systems. As shown in Figure 4.4 the IBM quantizer performs best in terms of the error floor and the decoding threshold (although in this example the decoding threshold is the same for all three quantizers if ρ_{rd} is large enough).

Since we have seen that the IBM quantizer yields the better BER performance compared to the other quantizers, we will henceforth use only the IBM quantizer. Similar results than those shown in Figure 4.4 are obtained for the case of $Q = 8$ quantizer levels.

Source coding at the relay. As already mentioned in Chapter 2, the quantizer output will in general not be uniformly distributed for $Q > 2$. Therefore lossless source coding can be applied to the sequence of quantizer indices z_r in order to save bits in the transmission from the relay to the destination. However, this implies that the destination will discard the data received from the relay in case the it was unable to correctly decode the source-coded message.

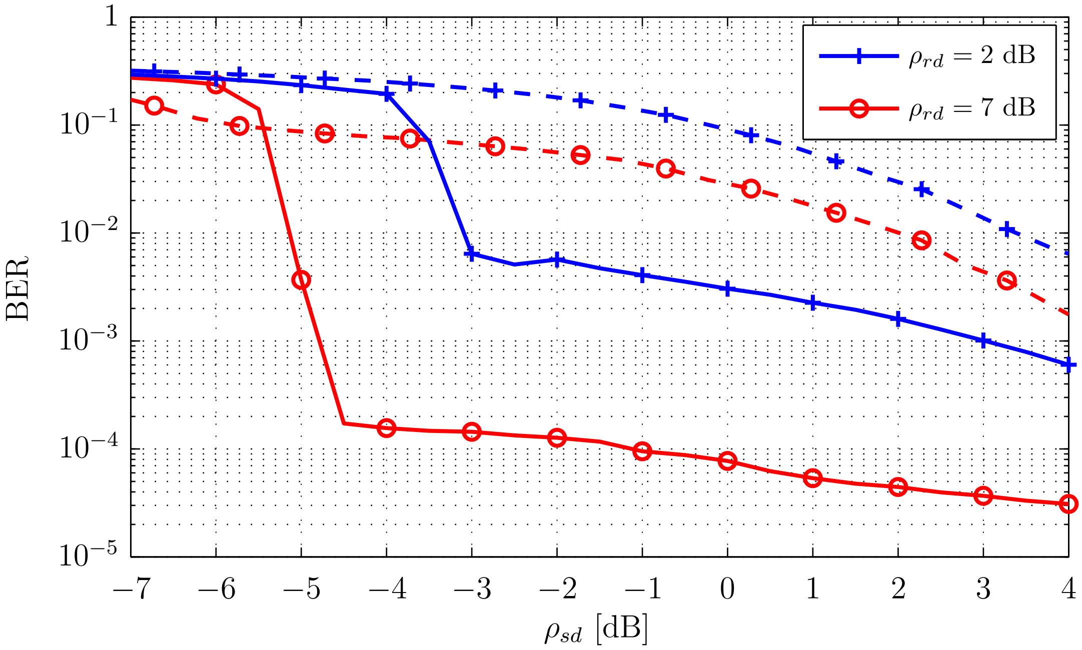


Figure 4.5: *Comparison between transmission with source code (dashed lines) and without source code (solid lines). The source-relay SNR ρ_{sr} is fixed to 10 dB and we use $Q = 4$ quantizer levels.*

In Figure 4.5 we compare the performance of our PLNC scheme for the case without source coding (solid lines) and for the case with source coding at the relay (dashed lines). In order to obtain these results we used $Q = 4$ quantizer levels and we fixed the source-relay SNR to $\rho_{sr} = 10$ dB. In this comparison we compensate for the unequal time sharing parameters and the different number of relay channel uses in both cases. This implies that the relay-destination SNR is roughly 3 dB better for the case with source coding (the legend in Figure 4.5 shows the values of ρ_{rd} for the case without source coding). However, even with a relatively high SNR of $\rho_{rd} = 10$ dB many source-coded messages have to be discarded at the destination due to decoding errors. Therefore the BER performance with source coding at the relay is poor in this SNR regime. Nevertheless, for very high values of the relay-destination SNR ρ_{rd} it is a viable option to reduce the number of channel uses required by the relay. We note that it should be investigated whether it is really necessary to always discard the complete data transmitted by the relay in case the destination was unable to decode the message without error.

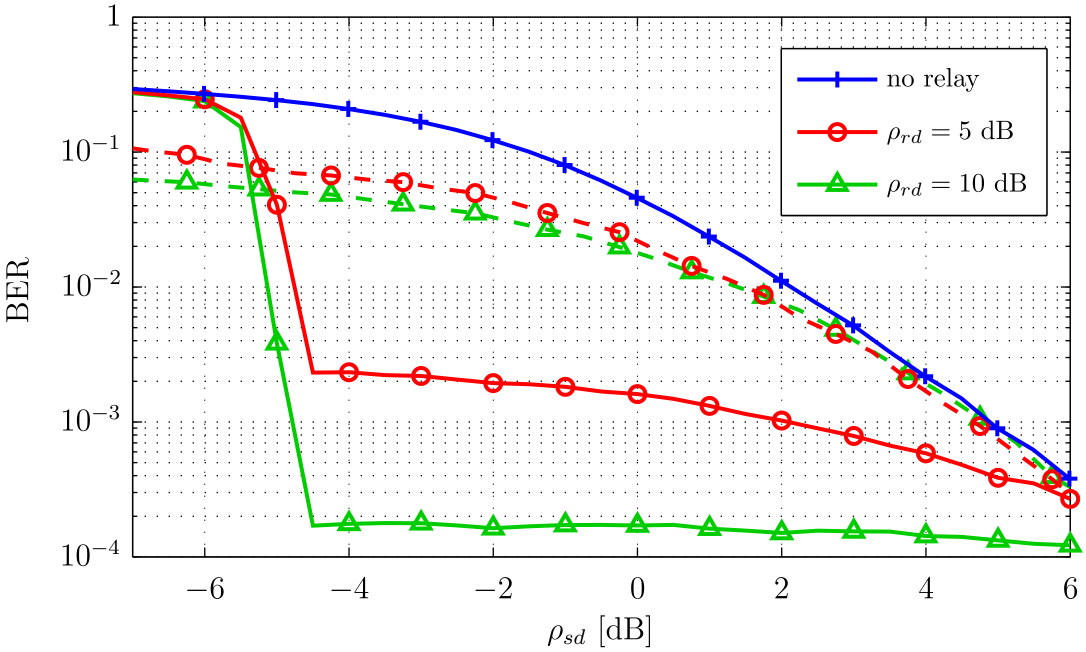


Figure 4.6: *Comparison between our PLNC scheme (solid lines) and a decode-and-forward scheme (dashed lines). The source-relay SNR is equal to 10 dB and we use $Q = 2$ quantizer levels. In this example we obtain an SNR gain of almost 10 dB.*

Comparison to a decode-and-forward scheme. Let us compare the performance of our PLNC scheme to the simple DF scheme described in Section 2.6. In this DF scheme the relay decodes both source messages and forwards half of the bits from each codeword to the destination without re-encoding the received data. At the destination the LLRs corresponding to the respective codewords are combined and then both messages are decoded separately. In case the relay can only decode one of the two source messages, it spends its complete timeslot for that source. If the relay was unable to decode any message it will remain silent. In any case, the time sharing parameter α is fixed to 1/3.

The performance comparison between our PLNC scheme (solid lines) and the DF scheme described above (dashed lines) is depicted in Figure 4.6. Here, the source-relay SNR is equal to 10 dB and we use $Q = 2$ quantizer levels. As a reference we also plot the BER for the case of no relay in Figure 4.6. In all cases we have compensated for the difference in the transmission time in order to ensure a fair comparison. From Figure 4.6 we can see that the SNR gain of our scheme compared to the reference DF scheme is almost 10 dB at a BER of 10^{-3} (for $\rho_{rd} = 10$ dB). Furthermore we observe that such

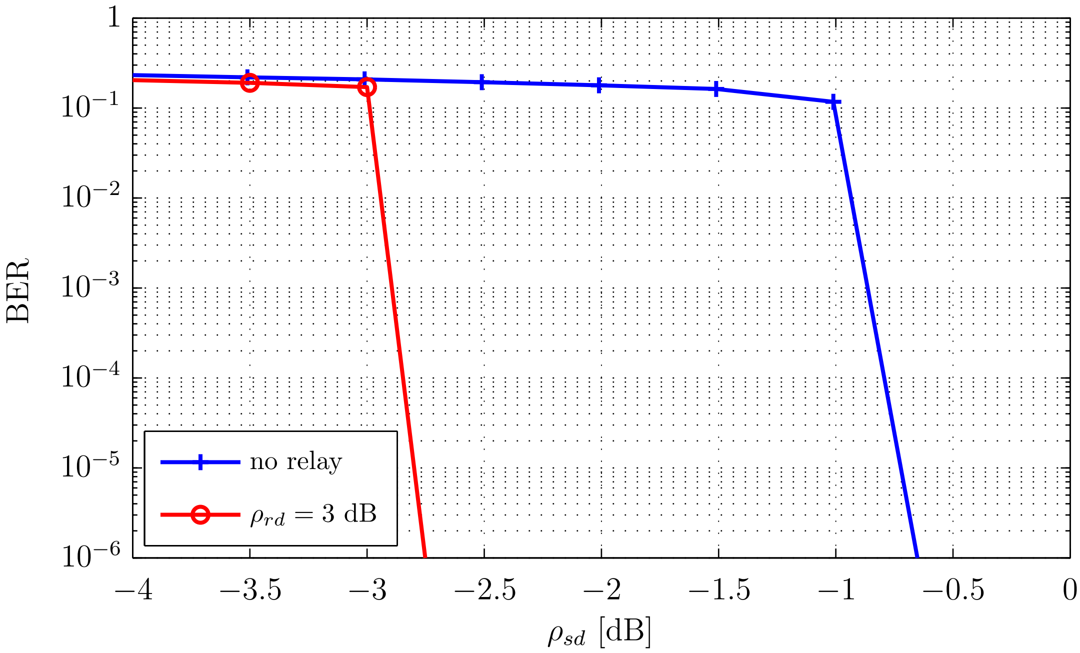


Figure 4.7: *BER performance with LDPC codes and a comparison to the case without relay. Here, the SNRs are given by $\rho_{sr} = \rho_{rd} = 3$ dB and we use $Q = 2$ quantizer levels. In this example we obtain an SNR gain of 2 dB compared to a transmission without relay.*

a DF scheme is not able to significantly improve the BER performance of two sources simultaneously. Moreover, for the DF scheme without re-encoding at the relay, the SNR gain compared to the case without relay vanishes as ρ_{sd} increases. For $\rho_{sr} < 10\,\text{dB}$ the difference between the DF scheme and the case without relay is even less (also for lower ρ_{sd}), since the probability of successful decoding at the relay decreases for with decreasing ρ_{sr}.

BER performance with LDPC codes. In the previous examples we have seen that the error floors in our scheme were relatively high in the SNR regime we considered. In Figure 4.7 we show BER results for the case where each source (and also the relay) uses an LDPC code. The LDPC code we use has a code rate of 1/2 and the blocklength is 64 000 bits. Using this channel code we did not observe any error floor. It is very likely that there is still an error floor, however, according to our simulation results this error floor should be below 10^{-8}. Since LDPC codes exhibit a pronounced threshold behavior, the BER performance does not change significantly once the SNR is above the decoding threshold. Therefore we choose $\rho_{sr} = \rho_{rd} = 3$ dB and we use $Q = 2$

quantizer levels.

From Figure 4.7 we can see that our PLNC scheme with LDPC codes yields an SNR gain of 2 dB compared to the case without relay. Note that the decoding threshold when using LDPC codes is at an approximately 2 dB higher SNR compared to the decoding threshold obtained with the RSCC which is used in the other examples. For the iterative decoder at the destination we have performed up to 50 inner iterations of the sum-product algorithm to decode the constituent codes and we have again used 10 outer iterations in which the channel decoders exchange information via the network decoder. We note that it might be possible to improve the result shown in Figure 4.7 by considering advanced scheduling algorithms for the iterative joint network-channel decoder.

4.3 The General Case

In this section we extend our analysis to the non-symmetric MARC. This means that in general we have $\rho_{1r} \neq \rho_{2r}$ and $\rho_{1d} \neq \rho_{2d}$. In this case both sources will have different BERs, denoted by BER_1 and BER_2. In the plots, the BER of source 1 will be indicated by solid lines and the BER of source 2 will be indicated by dashed lines. In order to be able to present our results in two-dimensional plots, we will not consider independent SNRs ρ_{ij}. Instead, we will consider a fixed offset between the source-relay SNRs and/or the source-destination SNRs. We show that the non-symmetries can be compensated by adapting the code rate of the sources.

In this section, unless otherwise noted, we assume $\rho_{1r} = 7$ dB and we use $Q = 4$ levels for the LLR quantization.

Non-symmetric source-relay channels: $\rho_{1r} > \rho_{2r}$, $\rho_{1d} = \rho_{2d}$. In Figure 4.8 we show BER performance of our system in case of non-symmetric source-relay channels and equal code rates $R_{c,1} = R_{c,2} = 1/2$. The source-relay SNRs are given by $\rho_{1r} = 7\,\text{dB}$ and $\rho_{2r} = 4\,\text{dB}$. From Figure 4.8 we can see that the error floor of source 2 is increased by an order of magnitude due to its lower source-relay SNR, while the performance of source 1 is almost unchanged compared to the symmetric case. The decoding threshold remains unchanged, as it is basically determined by the quality of the source-destination links in this SNR regime. We note that the difference in the error floors between the two sources decreases with increasing source-relay SNR (for a constant SNR offset $\rho_{1r} - \rho_{2r}$). For very high source-relay SNR, i.e., when $I(C_1, C_2'; \Lambda_r) \approx 1$, we obtain the performance of the symmetric case.

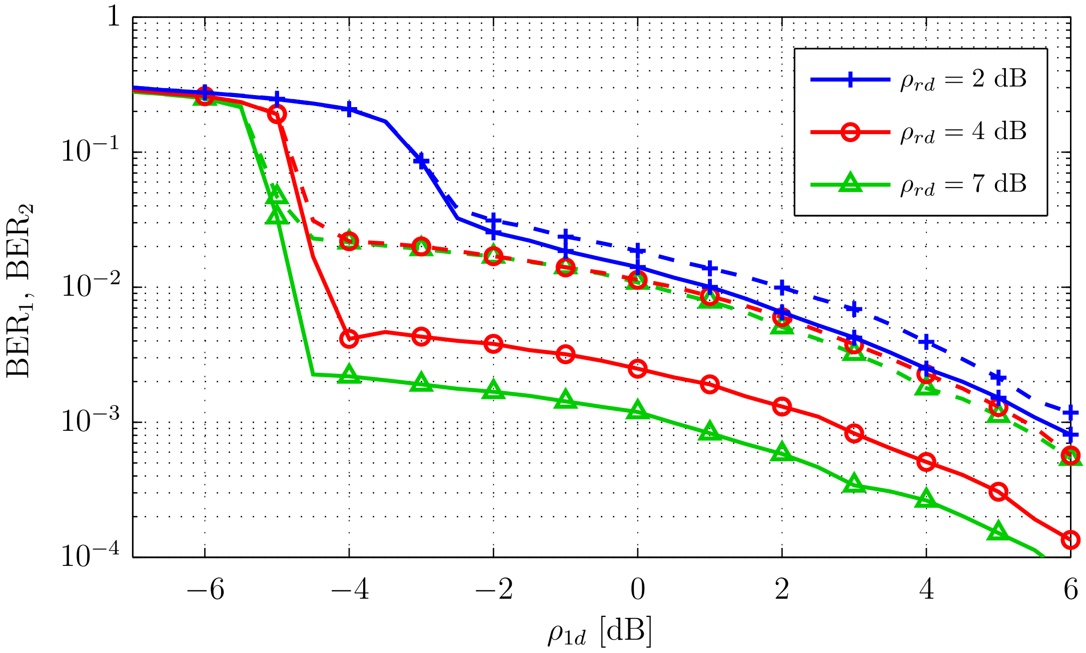


Figure 4.8: *Non-symmetric source-relay channels. Here, both sources use the same code rate and the source-relay SNRs are given by $\rho_{1r} = 7$ dB, $\rho_{2r} = 4$ dB.*

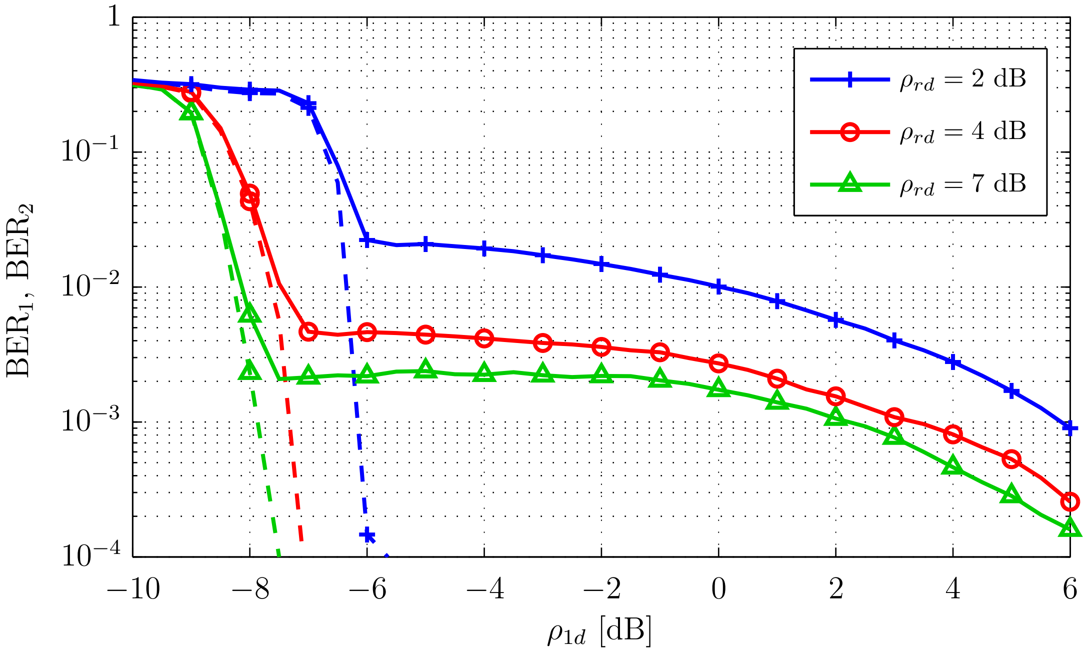


Figure 4.9: *Non-symmetric source-relay channels. Here, the code rates are given by $R_{c,1} = 1/2, R_{c,2} = 1/4$. The source-relay SNRs are $\rho_{1r} = 7$ dB, $\rho_{2r} = 4$ dB.*

In order to compensate for the worse BER performance in the non-symmetric case, source 2 can lower its code rate, e.g., to $R_{c,2} = 1/4$ while the parameters of source 1 remain unchanged. The performance in this case is depicted in Figure 4.9. We observe that the BER performance of source 2 has improved dramatically with an error floor of less than 10^{-5}. Here, both sources benefit from a decoding threshold which is more than 3 dB lower than in the case of equal code rates. This is due to the fact that source 2 provides a sufficient amount of information in order to obtain turbo gains already at a lower source-destination SNR. The difference of 3 dB corresponds to the difference in E_b/N_0 when decreasing the code rate from 1/2 to 1/4. Except for the decoding threshold, the performance of source 1 again remains largely unchanged.

Non-symmetric source-destination channels: $\rho_{1d} > \rho_{2d}$, $\rho_{1r} = \rho_{2r}$. The BER performance of our system in case of non-symmetric source-destination channels is depicted in Figure 4.10. Here, both sources use the same code rate, $R_{c,1} = R_{c,2} = 1/2$, and the source-destination SNRs are related by $\rho_{1d} - \rho_{2d} = 3$ dB. The result in Figure 4.10 shows that the error floor of source 2 is slightly increased in case of higher relay-destination SNRs ρ_{rd}. On the other hand, the error floor of source 2 is smaller than the one of source 1 for $\rho_{rd} = 2$ dB. We conjecture that this is the case, because the LLRs corresponding to the code bits transmitted by source 2 are (on average) smaller than the code bit LLRs corresponding to source 1, and the smaller LLRs are influenced less by distorted LLRs coming from the relay due to the properties of the boxplus operation. Furthermore we can see that the decoding threshold is increased, because source 2 provides less information about its code bits to the iterative decoder at the destination. Again, the performance of source 1 remains almost unchanged compared to the symmetric case.

When source 2 lowers its code rate to 1/4, we obtain the results depicted in Figure 4.11. We note that this result is similar to the result for the non-symmetric source-relay channels. The BER performance of source 2 is significantly improved and the corresponding error floor is lower than 10^{-5}. The decoding threshold is again improved by 3 dB when compared to the non-symmetric case, while otherwise the performance of source 1 remains basically unchanged.

Non-symmetric source-relay and source-destination channels. In this case both the source-relay channel and the source-destination channel of one source have an SNR offset compared to the respective channels of the other source. We will not discuss this case in detail since it is just a combination of the two previous examples and there are no additional noteworthy effects in this case.

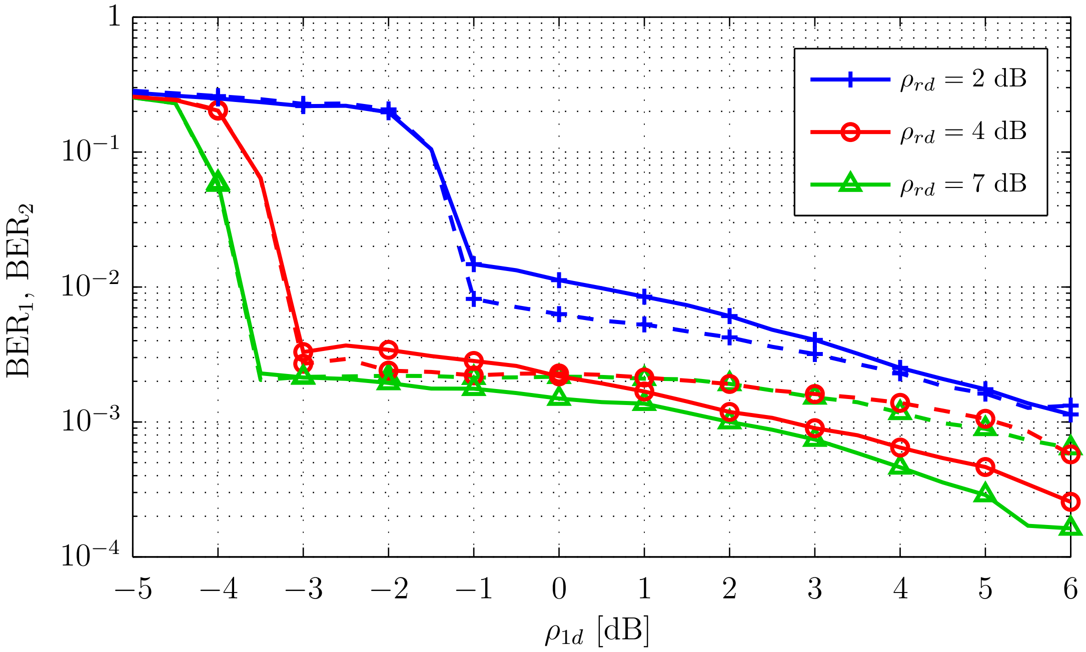


Figure 4.10: *Non-symmetric source-destination channels. In this example, the source-destination SNRs are related by $\rho_{1d} - \rho_{2d} = 3$ dB and both sources use the same code rate. The source-relay channels are symmetric, i.e., $\rho_{1r} = \rho_{2r}$.*

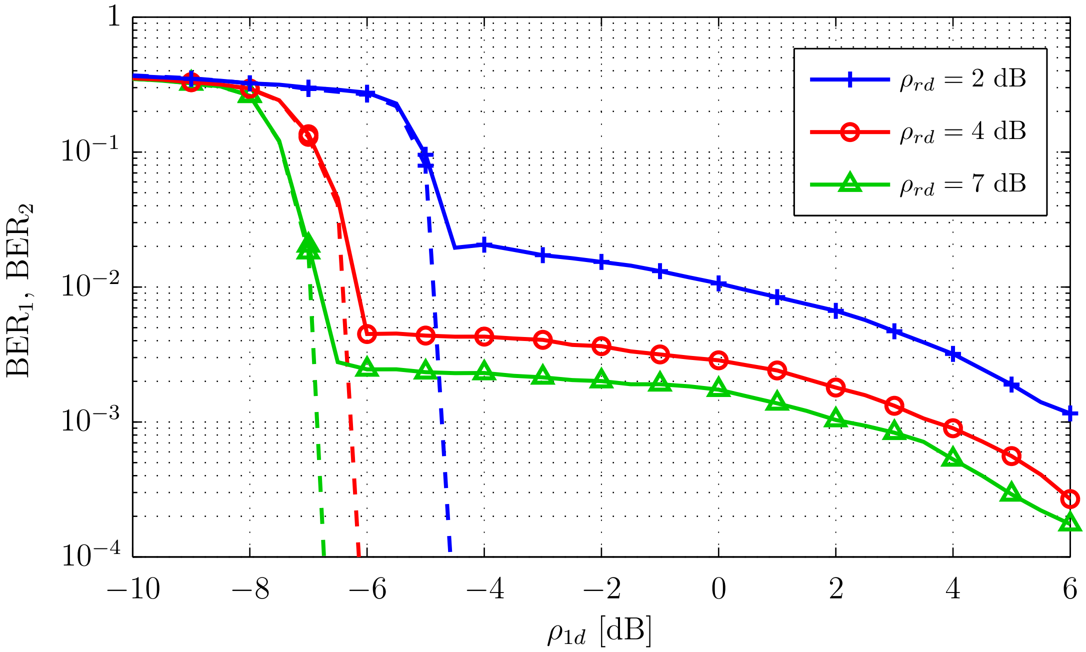


Figure 4.11: *Non-symmetric source-destination channels. Here, the code rates are given by $R_{c,1} = 1/2, R_{c,2} = 1/4$ and the source-destination SNRs are related by $\rho_{1d} - \rho_{2d} = 3$ dB. The source-relay channels are symmetric, i.e., $\rho_{1r} = \rho_{2r}$.*

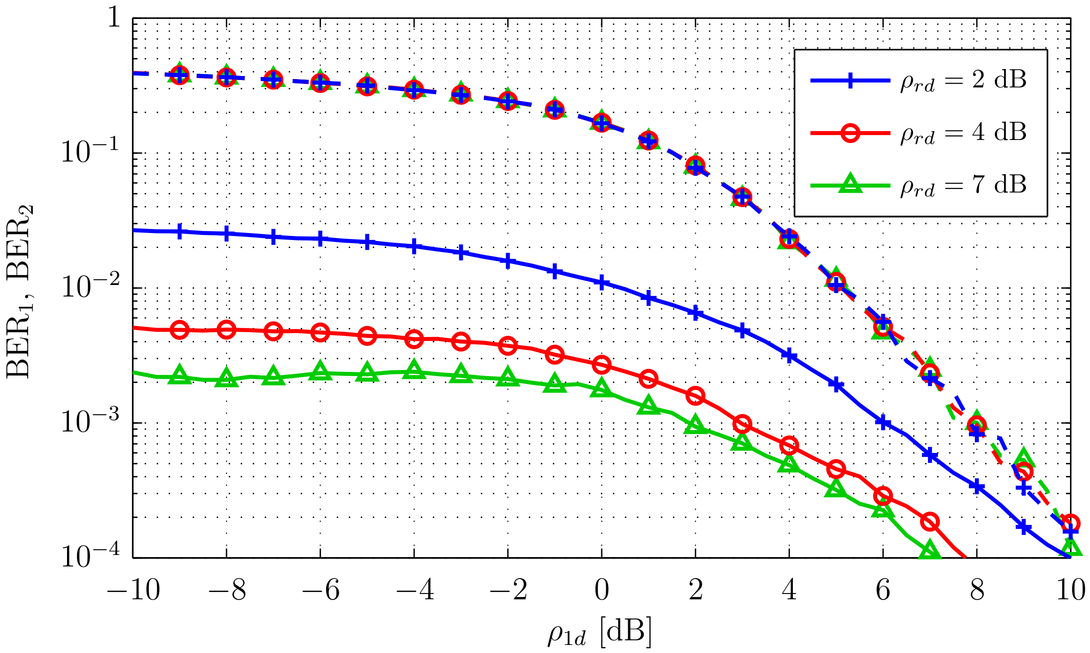


Figure 4.12: *The case of a broken source-relay link. Here the source-relay and relay-destination SNRs are given by $\rho_{1r} = 7$ dB, $\rho_{2r} = -\infty$ and $\rho_{1d} = \rho_{2d}$.*

The case of a broken source-relay link: $\rho_{2r} = -\infty$, $\rho_{1d} = \rho_{2d}$. If one of the two source-relay links is broken, then the relay has to cease its support for the source in question. Otherwise, all LLRs Λ_r would be zero and thus the relay would be unable to help the other source. This means that in case $\rho_{2r} = -\infty$, the MARC will degrade to a relay channel and a point-to-point link. We can see this behavior in Figure 4.12. Here, the BER performance of source 2 is independent of ρ_{rd} and equal to the performance of a point-to-point link. In Figure 4.12 we have $\rho_{1r} = 7$ dB and we quantize the LLRs at the relay using 2 bits/LLR. We can observe that the BER performance of source 1 improves as ρ_{rd} increases. However, we again see an error floor in the regime of low SNR ρ_{1d}. Here, the BER performance is limited by the SNR ρ_{1r} even if the relay-destination SNR $\rho_{rd} \rightarrow \infty$. Clearly, as the source-destination SNR ρ_{1d} increases the support from the relay becomes less important.

The case of a broken source-destination link: $\rho_{2d} = -\infty$, $\rho_{1r} = \rho_{2r}$. Figure 4.13 shows that in our PLNC scheme it is possible to decode the data of both users even if one source-destination channel is broken. The decoding threshold is equal for both sources, however compared to the symmetric case the decoding threshold is significantly increased, because only one of the two direct links conveys information to the

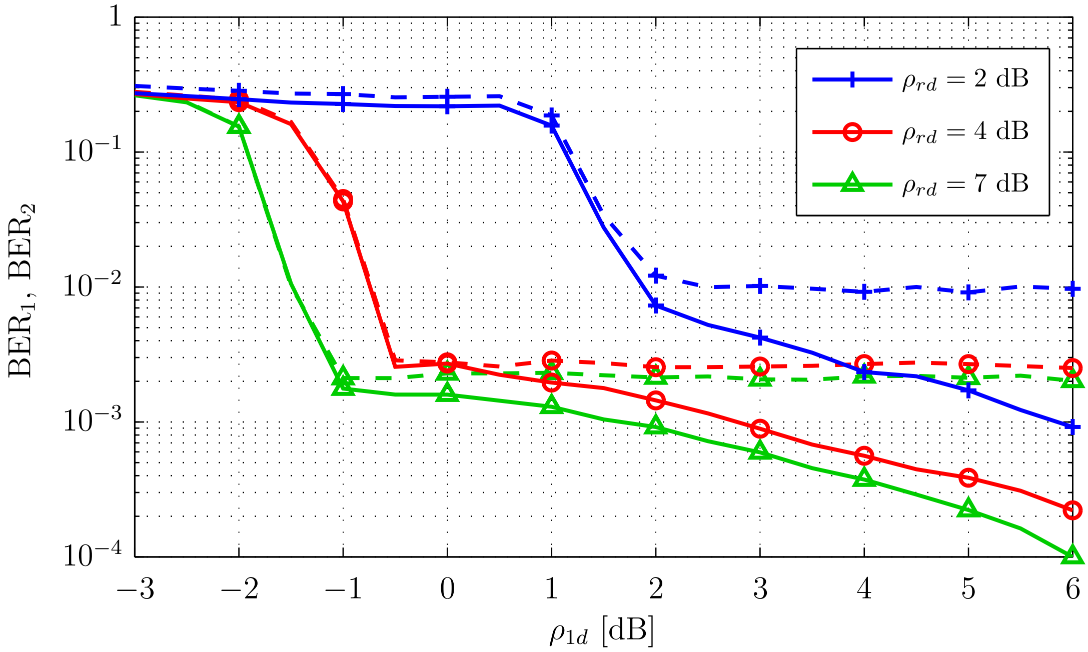


Figure 4.13: *The case of a broken source-destination link. Here the SNRs are given by $\rho_{2d} = -\infty$ and $\rho_{1r} = \rho_{2r} = 7$ dB.*

destination. Furthermore we note that source 2 exhibits an increased error floor. As before, the performance of source 1 remains largely unchanged.

We conclude that our PLNC scheme shows good performance even if the source-destination channels are non-symmetric. Moreover the system can compensate for a broken source-destination link. In addition we have seen that the impact of non-symmetric source-relay channels is more severe than the effect of non-symmetric source-destination channels. We propose to compensate for non-symmetric source-relay channels by adapting the code rate of the sources.

5

Summary and Outlook

5.1 Summary

In this thesis, we have presented a PLNC scheme for the MARC. An important advantage of this transmission scheme is that error-free decoding of the source messages at the relay node is not required. The relay can therefore make use of all the information it has obtained from its received signals. At the destination node, the network code and the channel codes are decoded jointly by exchanging soft information between the network decoder and the channel decoders. The joint network-channel decoder of our PLNC scheme shows "turbo-like" performance for a wide range of SNR values. The overall system offers significant SNR gains compared to legacy relaying schemes.

In our PLNC scheme the relay forwards soft information to the destination. In order to allow for digital transmission, the soft values have to be quantized at the relay. We have analyzed different quantizer design approaches and we have found that the information bottleneck method provides a framework for optimum quantizer design in terms of mutual information loss. Furthermore we have shown that proper LLR quantization is vital for the performance our PLNC scheme, i.e., the metric of mutual information loss is indeed relevant for the performance of the proposed system.

Finally we have provided numerical simulation results for a variety of system parameters and transmission scenarios. We have shown the performance gain that can be achieved using our PLNC scheme by comparing it to a number of reference systems. Moreover we have gained insight on which parameters determine the decoding threshold and the error floor of our system. A detailed analysis of the system performance for the case of the non-symmetric MARC has shown that we can compensate for different channel conditions by adapting the code rate of the sources.

5.2 Outlook

In this section we propose several ideas for future research. Most of these ideas concern extensions to the PLNC scheme proposed in this thesis.

- In our PLNC scheme for the MARC we use TDMA with three timeslots in order to obtain orthogonal channels. Therefore the sources may use only a fraction of the total frame duration for their transmission. This approach helps to simplify the receivers and the synchronization between the nodes, however, it also decreases the overall rate at which the sources transmit their data. We should therefore try to reduce the number of timeslots in order to increase the transmission rate of the sources. Clearly, a full-duplex relay is required if only one timeslot is used, i.e., if both sources and the relay constantly transmit data. Since wireless full-duplex relays are infeasible at the moment, we do not consider non-orthogonal multiple-access of all nodes for our PLNC scheme. Nevertheless, we can combine the timeslots of the two sources and still use a half-duplex relay, thereby increasing the transmission rate of the sources. It remains to be seen if such a non-orthogonal multiple-access scheme with two timeslots is a viable option for our PLNC scheme in terms of computational complexity and system performance.

- So far we have only considered offline design of the LLR quantizer employed in our PLNC scheme. This is because optimum quantizer design requires knowledge of the conditional LLR statistics which can only be obtained using Monte Carlo simulations since we lack an analytical model of the LLRs obtained at the relay. Of course, it would be more practical if the quantizer design would happen “on-the-fly”, i.e., during data transmission. The quantizer levels and the reproducer values of the IBM quantizer discussed in Section 3.5 seem to scale with increasing source-relay SNR in a way that should allow on-the-fly quantizer design. Further investigations would then be necessary in order to assess the impact of (suboptimal) on-the-fly quantizer design on the system performance.

- The network coding operation at the relay is performed in terms of soft values in our PLNC scheme. We have seen that the LLR of the modulo-2 sum of two independent code bits is given by the “boxplus” of the two code bit LLRs. We thereby combine two individual LLRs to one LLR about the network-coded bits, which is subsequently quantized by a scalar quantizer. Using this approach the rate of the relay-destination link is equally divided between the two sources. Whereas this is optimum in case of symmetric source-relay channels, i.e., when $\rho_{1r} = \rho_{2r}$, a different rate allocation, preferring the user with the better source-relay channel, should be used in case of non-symmetric source-relay channels. This could be realized by applying two-dimensional vector quantization directly to the code bit LLRs, instead of first combining them and performing a scalar quantization afterwards. This would make our scheme more suitable for non-symmetric channel conditions.

- In our scheme, the sequence of quantizer output indices is channel encoded and modulated at the relay prior to the transmission to the destination. We conjecture that it would be beneficial to design the coding and modulation at the relay such that the sign of the LLRs is better protected than the magnitude information. This could be achieved by applying unequal error protection techniques. However, it remains an open question if the performance of our system can be improved significantly by considering unequal error protection.

- Up to now we have investigated PLNC with soft information forwarding only for the MARC. It would be interesting to see if the basic ideas of our PLNC scheme for the MARC can also be applied to other wireless relay networks. In particular, PLNC for the TWRC might be studied in future work.

- Finally, we think that PLNC in conjunction with rateless codes is an important topic that should be studied in detail.

Bibliography

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, pp. 1204–1216, July 2000.

[2] B. Bollobás, *Graph Theory, An Introductory Course.* New York: Springer-Verlag, 1979.

[3] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *IEEE International Symposium on Information Theory*, pp. 442–442, 2003.

[4] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *Proceedings IEEE INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 4, 2005.

[5] C. Gkantsidis, J. Miller, and P. Rodriguez, "Anatomy of a p2p content distribution system with network coding," in *Proc. of the 5th International Workshop on Peer-to-Peer Systems (IPTPS 2006)*, 2006.

[6] M. Effros, M. Médard, T. Ho, S. Ray, D. Karger, R. Koetter, and B. Hassibi, "Linear network codes: A unified framework for source, channel, and network coding," in *Advances in network information theory: DIMACS Workshop Network Information Theory, March 17-19, 2003, Piscataway, New Jersey*, p. 197, 2004.

[7] L. Xiao, T. Fuja, J. Kliewer, and D. Costello, "Cooperative diversity based on code superposition," in *2006 IEEE International Symposium on Information Theory*, pp. 2456–2460, 2006.

[8] 3GPP, "TR 36.814 Further advancements for E-UTRA: Physical layer aspects." www.3gpp.org, TR 36.814 v. 9.0.0, March 2010.

[9] S. Yang and R. Koetter, "Network coding over a noisy relay: a belief propagation approach," in *IEEE International Symposium on Information Theory, 2007. ISIT 2007*, pp. 801–804, 2007.

[10] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J*, vol. 27, pp. 379–423, 1948.

[11] J. G. Proakis, *Digital Communications.* New York: McGraw-Hill, 3rd ed., 1995.

[12] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inf. Theory*, vol. 42, pp. 429–445, March 1996.

[13] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[14] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proc. IEEE GLOBECOM-89*, (Dallas, TX), pp. 1680–1686, June 1989.

[15] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. 20, pp. 284–287, March 1974.

[16] F. Jelinek, "Fast sequential decoding algorithm using a stack," *IBM Journal of Research and Development*, vol. 13, no. 6, pp. 675–685, 1969.

[17] C. Berrou, A. Glavieux, and P. Thitimajshime, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE ICC-1993*, (Geneva, Switzerland), pp. 1064–1070, May 1993.

[18] H. R. Sadjadpour, N. J. A. Sloane, and G. Nebe, "Interleaver design for turbo codes," *IEEE J. Sel. Areas Comm.*, vol. 19, pp. 831–837, May 2001.

[19] C. Heegard and S. B. Wicker, *Turbo Coding.* Boston, MA: Kluwer, 1999.

[20] R. G. Gallager, "Low density parity check codes," *IRE Trans. Info. Theory*, vol. 8, pp. 21–28, Jan. 1962.

[21] R. Neal and D. Mackay, "Good Codes based on Very Sparse Matrices," in *Cryptography and Coding. 5th IMA Conference ed. Colin Boyd, Lecture Notes in Computer Science*, vol. 1025, pp. 100–111, 1995.

[22] N. Alon and M. Luby, “A linear time erasure-resilient code with nearly optimal recovery,” *IEEE Transactions on Information Theory*, vol. 42, no. 6 Part 1, pp. 1732–1736, 1996.

[23] D. J. C. MacKay, “Good error correcting codes based on very sparse matrices,” *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, 1999.

[24] W. Ryan and S. Lin, *Channel Codes: Classical and Modern*. Cambridge University Press, 2009.

[25] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[26] M. Valenti and B. Zhao, “Distributed turbo codes: towards the capacity of the relay channel,” in *2003 IEEE 58th Vehicular Technology Conference, 2003. VTC 2003-Fall*, vol. 1, 2003.

[27] H. Sneesens and L. Vandendorpe, “Soft decode and forward improves cooperative communications,” in *1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, pp. 157–160, 2005.

[28] Y. Li, B. Vucetic, Y. Tang, Z. Zhou, and M. Dohler, “Practical distributed turbo coding through soft information relaying,” in *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications, 2005. PIMRC 2005*, vol. 4, 2005.

[29] G. Kramer, M. Gastpar, and P. Gupta, “Cooperative strategies and capacity theorems for relay networks,” *IEEE Trans. Inform. Theory*, vol. 51, pp. 3037–3063, Sept. 2005.

[30] T. M. Cover and A. A. E. Gamal, “Capacity theorems for the relay channel,” *IEEE Trans. Inf. Theory*, vol. 25, pp. 572–584, Sept. 1979.

[31] G. Kramer and A. Van Wijngaarden, “On the white Gaussian multiple-access relay channel,” in *IEEE International Symposium on Information Theory, 2000. Proceedings*, 2000.

[32] B. Rankov and A. Wittneben, “Achievable rate regions for the two-way relay channel,” in *2006 IEEE International Symposium on Information Theory*, pp. 1668–1672, 2006.

[33] C. Hausl and J. Hagenauer, "Iterative network and channel decoding for the two-way relay channel," in *IEEE International Conference on Communications, 2006. ICC'06*, vol. 4, 2006.

[34] C. Hausl and P. Dupraz, "Joint network-channel coding for the multiple-access relay channel," *3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, vol. 3, pp. 817–822, 2006.

[35] K. Witzke and C. Leung, "A comparison of some error detecting CRC code standards," *IEEE Transactions on Communications*, vol. 33, no. 9, pp. 996–998, 1985.

[36] G. Castagnoli, S. Bräuer, and M. Herrmann, "Optimization of Cyclic Redundancy-Check Codes with 24 and 32 Parity Bits," *IEEE Transactions on Communications*, vol. 41, no. 6, pp. 883–892, 1993.

[37] D. Duyck, D. Capirone, M. Moeneclaey, and J. Boutros, "A full-diversity joint network-channel code construction for cooperative communications," in *20th Annual IEEE International symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '09)*, 2009.

[38] G. Zeitler, R. Koetter, G. Bauch, and J. Widmer, "On quantizer design for soft values in the multiple-access relay channel," in *Proc. IEEE Int. Conf. Comm*, 2009.

[39] K. Sayood, *Introduction to data compression.* Morgan Kaufmann, 2000.

[40] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* New York: Wiley, 1991.

[41] J. Max, "Quantization for minimum distortion," *IEEE Trans. Inform. Theory*, vol. 6, no. 2, pp. 7–12, 1960.

[42] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[43] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, 1980.

[44] C. Novak, P. Fertl, and G. Matz, "Quantization for soft-output demodulators in bit-interleaved coded modulation systems," in *Proc. of ISIT 2009*, (Seoul, Korea), pp. 1070–1074, IEEE, June 2009.

[45] W. Rave, "Quantization of log-likelihood ratios to maximize mutual information," *IEEE Signal Processing Letters*, vol. 16, pp. 283–286, 2009.

[46] S. Schwandter, C. Novak, P. Fertl, and G. Matz, “Log-likelihood ratio clipping in MIMO-BICM systems: Information geometric analysis and impact on system capacity,” in *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, (Taipei, Taiwan), pp. 2433–2436, April 2009.

[47] N. Tishby, F. Pereira, and W. Bialek, “The information bottleneck method,” in *In Proc. of the 37-th Allerton Conference on Communication and Computation*, 1999.

[48] R. E. Blahut, “Computation of channel capacity and rate-distortion functions,” *IEEE Trans. Inf. Theory*, vol. 18, pp. 460–473, 1972.

Notation

Throughout this thesis, vectors are denoted by boldface letters. Time dependent quantities are denoted by $x(t)$ or $x[n]$ in the continuous time and discrete time case, respectively. Unless noted otherwise, the meaning of the following symbols is as stated below.

Symbol	*Meaning*
$\mathcal{A}$	Symbol alphabet
a	Transmit symbol, $a \in \mathcal{A}$
$\mathcal{CN}(\mu, \sigma^2)$	Circularly symmetric, complex Gaussian distribution with mean μ and variance σ^2
c_i	Sequence of code bits transmitted by node i
$\hat{c}_i$	Estimate of the sequence of code bits c_i transmitted by node i
$\mathrm{E}\{X\}$	Expectation of the random variable X
E_i	Transmit energy of node i per codeword
$\mathrm{GF}(q)$	Galois field with q elements
h_{ij}	Channel coefficient of channel from node i to node j
K_i	Dimension of the channel code used by node i, i.e., the number of information bits per codeword
l	Number of bits carried by one symbol, $l = \log_2(M_a)$
M_a	Cardinality of $\mathcal{A}$, $M_a = \|\mathcal{A}\|$
M_i	Number of symbols transmitted by node i per codeword
M	Total number of symbols transmitted per frame
N_i	Blocklength of the channel code used by node i
$\mathrm{P}\{\mathcal{E}\}$	Probability of the event $\mathcal{E}$
P_i	Transmit power of node i

$R_{c,i}$	Rate of channel code used by node i, $R_i^{\mathrm{c}} = K_i/N_i$
R_i	Transmission rate of node i in bit per channel use, $R_i = K_i/M_i$
u_i	Sequence of information bits transmitted by node i
$\hat{u}_i$	Estimate of the sequence of information bits $\mathbf{u}_i$ transmitted by node i
w_i	Sequence of noise samples observed at node i
x_i	Sequence of symbols transmitted by node i
y_{ij}	Sequence of noisy symbols received at node j, transmitted by node i
α	Time sharing parameter (for the ith source we have $M_i = \alpha M$)
γ_{ij}	Instantaneous SNR on the channel from node i to node j
$\Lambda(\cdot)$	A posteriori log-likelihood ratio
$\Lambda_p(\cdot)$	A priori LLR
$\Lambda_e(\cdot)$	Extrinsic LLR
$\Pi(\cdot)$	Interleaver
$\Pi^{\text{-1}}(\cdot)$	Deinterleaver
ρ_{ij}	Average SNR on the channel from node i to node j
$\oplus$	Modulo-2 addition, bit-wise XOR operation
$\boxplus$	Boxplus operation [12]

List of Abbreviations

3GPP	**3**rd **G**eneration **P**artnership **P**roject
APP	**a** **p**osteriori **p**robability
ARQ	**a**utomatic **r**epeat re**q**uest
AWGN	**a**dditive **w**hite **G**aussian **n**oise
BCJR algorithm	**B**ahl, **C**ocke, **J**elinek and **R**aviv algorithm [15]
BER	**b**it **e**rror **r**atio
BPSK	**b**inary **p**hase **s**hift **k**eying
CRC	**c**yclic **r**edundancy **c**heck
CSI	**c**hannel **s**tate **i**nformation
FCS	**f**rame **c**heck **s**equence
FEC	**f**orward **e**rror **c**orrection
i.i.d.	**i**ndependent and **i**dentically **d**istributed
JNCC	**j**oint **n**etwork-**c**hannel **c**oding
LDPC code	**l**ow **d**ensity **p**arity **c**heck code
LLR	**l**og-**l**ikelihood **r**atio
MARC	**m**ultiple-**a**ccess **r**elay **c**hannel
PCCC	**p**arallel **c**oncatenated **c**onvolutional **c**ode
pdf	**p**robability **d**ensity **f**unction
PLNC	**p**hysical **l**ayer **n**etwork **c**oding
QPSK	**q**uadrature **p**hase **s**hift **k**eying
RF	**r**adio **f**requency
RSCC	**r**ecursive **s**ystematic **c**onvolutional **c**ode
SISO decoder	**s**oft-**i**nput **s**oft-**o**utput decoder
SNR	**s**ignal-to-**n**oise **r**atio
TDMA	**t**ime **d**ivision **m**ultiple-**a**ccess
TWRC	**t**wo-**w**ay **r**elay **c**hannel