UNIVERSITA' DEGLI STUDI DI PAVIA

Dottorato di Ricerca in Ingegneria Elettronica, Informatica ed Elettrica

Design and decoding techniques of q-ary codes

Dott. Ing. Andrea Marinoni

Tutor:

Prof. Eugenio Costamagna

Coordinatore:

Prof. Vittorio Degiorgio

XXIII ciclo (2007-2010)

Contents

Chapter 1

Introduction

Since 1950, when Hamming published the first work on parity bits [86], a lot of coding theories have been developed. As the technology enhancements took place, the range of coding architectures grew up as the variety of transmission requirements as well. Thus, modern coding theory has to face a lot of challenges that can be summarized asin the following:performance has to approach as much as possible the channel capacity, minimizing at the same time some parameters such as decoding complexity (and therefore decoding delays), area occupation, non-linear behaviors, bandwidth waste, power consumption, etc.

Among the families of capacity approaching codes, low-density parity-check (LDPC) codes are especially attractive and play a fundamental role in modern communications. LDPC codes [1] are linear codes characterized by a sparse parity-check (PC) matrix, H, having M rows and N columns. A LDPC code is either regular or irregular depending on its row and column degree-distributions. Regular LDPC codes have a PC matrix in which all rows (and columns) present equal weight, while irregular LDPC codes do not exhibit this property.

Non-binary (or q-ary) LDPC codes have codewords (and therefore a PC matrix) whose symbols are elements of the Galois field $GF(q)$, with $q > 2$. These q-ary LDPC codes typically have steeper bit-error-rate waterfall curves, while the decoding complexity is $O(Ntq^2)$, where N is the blocklength, t is the average column weight, and q is the alphabet size [3], [4].

q-ary LDPC codes are definitely interesting since they can be employed in a wide range of applications, merging benefits and performance of both inner and outer codes used in more traditional architectures. Further, given their peculiar graphical structure and decoding method, they have a great potential in order to improve the error-rate performance in the waterfall region, to lower the error floor and to generate powerful structured codes. Therefore, q-ary LDPC codes have represented the main course of the whole Ph.D. program. In fact, design and decoding algorithms for q-ary LDPC codes have been developed, implemented and simulated during the past three years.

The thesis is structured as follows. Chapter 2 introduces some q-ary LDPC code design methods that can improve performance for specific requirements such as spectral efficiency, error floor control and burst error correction capability. Chapter 3 provides a comparison between the most relevant detection-and-decoding architectures for q-ary LDPC codes over partial response (PR) channels. Further, it presents a brand new receiving structure for joint detection and decoding over the aforesaid channels. Chapter 4 introduces prospective scenarios of practical employment of q-ary LDPC codes, according to the developed activity during the Ph.D. program. Chapter 5 delivers the final remarks.

Chapter 2

Design of q-ary codes

It is well known how a proper design of LDPC codes strongly infer on the performance of the given codes itself. Several construction algorithms (for instance, [18], [44], [31]) aim at fitting the given code to specific requirements, such as error-rate performance in the low and the high signal-to-noise ratio (SNR) region, error burst correction, etc. Given the peculiar graph structure and the decoding method, designing q-ary LDPC codes is fundamental to achieve the desired results. The next Sections introduce several construction algorithms that aim at facing some important communication problems.

Specifically, in recent times the need of spectral efficiency has become a relevant topic for many communication systems, especially for wireless services. In order to achieve the best trade-off between bandwidth occupancy and error-rate performance, several structures that involve large constellations have been proposed in literature. Section 2.1 focuses on LDPC-coded systems using 16-QAM constellations on a channel affected by additive white Gaussian noise (AWGN). The LDPC codes that have been used include both binary and non-binary systems. In order to be compared, they have been designed such that they are equivalent in terms of blocklength, rate and average column weight. Simulation results show how the structure that involves a q-ary LDPC code outperforms the other schemes: new possible scenarios to be analyzed are then introduced.

Further, Section 2.2 introduces a protograph-based method for designing q-ary LDPC codes for use with modulations larger than quadrature phase shift keying (QPSK). Simulations focus on a $GF(16)$, 16-QAM example. The proposed construction method achieves the maximum gain when the average column weight is chosen so that the linear minimum distance growth property is satisfied. In this region, the benefit of a protograph-based design over a standard progressive edge-growth (PEG) approach was 0.3 dB. It is shown that a careful field-element selection algorithm provides about 0.1 dB of improvement over random field-element selection. Overall, the proposed improvements yielded 0.4 dB of gain over a PEG-based $GF(16)$ code with randomly selected Galois field elements. The performance of this baseline $GF(16)$ code was comparable to the best known binary LDPC code for 16-QAM, so that the proposed improvements allow the $GF(16)$ LDPC code to outperform known binary approaches.

On the other hand, Section 2.3 introduces construction methods for designing q-ary LDPC codes with low error floors and moderate field order. These algorithms are based on a deep geometrical and graphical analysis of the q-ary LDPC decoding problem. Specifically, decoding failures and characteristic topological structures can be related. Thus, enlarging the size of the aforesaid topological structures provides substancial improvements in terms of error-rate performance over other construction algorithms. Simulation results show how q-ary LDPC codes constructed by means of the proposed algorithms have error floors that are orders of magnitude lower than those of codes based on other known algorithms.

Finally, an optimized design of q-ary LDPC codes that takes into account their burst error correction capability has been considered. In recent works, the performance of LDPC decoding in presence of noise bursts has been related to the structure of the parity-check matrix. In particular, two approaches to characterize the burst error correction capabilities have been proposed in the literature. Following these ideas, different matrix designs are compared in Section 2.4 in order to choose the best matrix constraints to be maximized in a PEG construction. Several non-binary LDPC codes generated with the proposed design methods are compared. Their performance are analyzed in the context of magnetic recording channels, where they are considered a promising alternative to the Reed-Solomon (RS) codes.

2.1 Spectrally efficient LDPC coded modulations

In order to achieve the best possible error-rate performance, capacity-approaching codes such as Turbo-Codes (TC) [2] and LDPC [1] codes have been adopted by a multitude of systems - with applications ranging from storage to optical communications.

Using their bipartite graph representation, [5] and [23] showed that LDPC codes may perform very close to capacity on AWGN channels and achieve capacity on binary erasure channels. Therefore, it is natural to ask if LDPC codes can improve the bit-error-rate performance of a code in a communication system that has several requirements from high bandwidth efficiency to high coding rate.

In fact, with an ever-increasing demand for wireless services, the need for spectral efficiency in data communications has become an important topic. To alleviate the crowding of the radio-frequency spectrum, it is desirable to make more efficient use of currently allocated frequency bands. Historically, the most popular scheme to improve bandwidth-efficiency has been to utilize higher-order modulation. This approach allows more bits per transmitted symbol, but the higher symbol density requires increased SNR to achieve acceptable bit-error-rate (BER) performance.

In order to find the system that guarantees the best trade-off between error-rate performance and spectral efficiency, three different LDPC code architectures have been analyzed on the AWGN channel, paying particular attention to the properties of the LDPC code selected for each one. These systems are the Turbo-like architecture, the Multilevel Coding architecture, and the q-ary LDPC Coded architecture. In each of these architectures, the basic system model is as follows: the input to the modulator is encoded by a LDPC code whose properties depend on the particular architecture under consideration. At the receiver, the received signal is sent to the LDPC decoder. Depending on the transmitter architecture that was used, the receiver decodes the signal in an appropriate manner. In the three different architectures that are discussed, the first two are based on binary LDPC codes, while the last one is based on a q-ary LDPC code.

All the LDPC codes used in this section have been constructed using quasi-regular PC matrices [20], [22] generated by the PEG algorithm [18]. Quasi-regular PC matrices are defined as follows: given a rate R, and the average column weight (i.e. the average variable-node degree in the Tanner graph), $\overline{d}_v$, the average row weight (i.e. the average check-node degree), $\overline{d}_c$, is computed as:

$$\overline{d}_c = \frac{\overline{d}_v}{1-R}. \tag{2.1}$$

Furthermore, the variable-node degree distribution is computed as:

$$f(d_v = j) = \begin{cases} \lfloor \overline{d}_v \rfloor - \overline{d}_v + 1 & \text{if } j = \lfloor \overline{d}_v \rfloor \\ \lfloor \overline{d}_v \rfloor - \overline{d}_v & \text{if } j = \lfloor \overline{d}_v \rfloor + 1 \\ 0 & \text{otherwise,} \end{cases} \tag{2.2}$$

where $f(d_v = j)$ represents the fraction of columns with weight j in the given PC matrix, and $\lfloor z \rfloor$ is defined as the largest integer less than or equal to z.

Analogously, the check-node degree distribution can be computed as follows:

$$f(d_c = j) = \begin{cases} \lfloor \overline{d}_c \rfloor - \overline{d}_c + 1 & \text{if } j = \lfloor \overline{d}_c \rfloor \\ \lfloor \overline{d}_c \rfloor - \overline{d}_c & \text{if } j = \lfloor \overline{d}_c \rfloor + 1 \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

Here, $f(d_c = j)$ represents the fraction of rows with weight j in the given PC matrix.

The next three subsections introduce the three coding architectures considered in this section: the Turbo-like, Multilevel Coding, and q-ary LDPC Code architectures respectively.

2.1.1 The turbo-like architecture

In this subsection, an LDPC-coding architecture that employs a turbo-like receiver (TLR) is considered. A block diagram of the receiver is given in Figure 2.1. In this turbo-like architecture, the transmitted signal is a binary LDPC codeword that has been properly mapped to the constellation associated with the given higher-order modulation scheme. At the receiver, a soft detector incorporates extrinsic information provided by the binary LDPC decoder, and the LDPC decoder incorporates soft information provided by the detector. Extrinsic information between the detector and decoder is exchanged in an iterative way until an LDPC codeword is found or a maximum number of iterations are performed [4], [12]. With LDPC codes, convergence to a codeword is easily detected by the receiver when the parity check equations are satisfied. The decoding employs the message passing (MP) algorithm, which is described in detail in [3], [4].

In this turbo-like architecture, the received vector, $\mathbf{y}$, is demapped by a log-likelihood ratio (LLR) calculation for each of the coded bits included in the transmitted vector $\mathbf{x}$. The extrinsic information provided by the detector is the difference of the soft-input and soft-output LLR values for the coded bits. For the κ-th coded bit of $\mathbf{x}$, x_κ, the extrinsic LLR value of the estimate

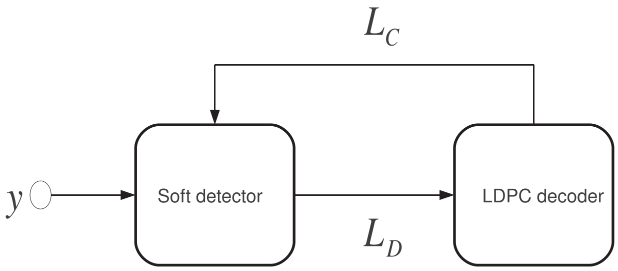

Figure 2.1: Turbo iterative detection-and-decoding receiver for an LDPC coded system

of x_κ is computed as follows:

$$\begin{aligned} L_D(x_\kappa) &= \log \frac{P(x_\kappa = +1|\mathbf{y})}{P(x_\kappa = -1|\mathbf{y})} - \log \frac{P(x_\kappa = +1)}{P(x_\kappa = -1)} \\ &= \log \frac{P(x_\kappa = +1|\mathbf{y})}{P(x_\kappa = -1|\mathbf{y})} - L_C(x_\kappa), \end{aligned} \tag{2.4}$$

where $L_C(x_\kappa)$ is the extrinsic information of x_κ computed by the LDPC decoder in the previous turbo iteration. Note that the decoder sets $L_C(x_\kappa) = 0$ at the first iteration.

Assuming the bits associated with $\mathbf{x}$ are statistically independent of one another, the *a priori* probability $P(\mathbf{x})$ can be expressed in the following way:

$$P(\mathbf{x}) = \prod_{i=1}^{N} P(x_i) = \prod_{i=1}^{N} \left[1 + \exp(-\mathbf{x}^{x_i} \cdot L_C(x_i))\right], \tag{2.5}$$

where $\mathbf{x}^{x_i}$ corresponds to the value (either +1 or -1) of the i-th bit in the vector $\mathbf{x}$.

2.1.2 The multilevel coding architecture

This subsection considers a multilevel coding architecture. In [11], the authors' idea of multilevel coding (MLC) is to protect each address bit, x_i, of the constellation points by an individual binary code, ξ_i, at level i. At the receiver, each code ξ_i is decoded individually starting from the lowest level: if the decisions from prior decoding stages are taken into account, this procedure is called multistage decoding (MSD), otherwise, it is called parallel independent decoding (PID). Figures 2.2 and 2.3 show MSD and PID respectively. In terms of transmission rate flexibility, the MLC approach outperforms Ungerboeck's trellis coded modulation (TCM) [8,9], since the size of the signal constellation is separated from the code rate. Moreover, any kind of code can be used as constituent code.

A generalization of the approach in [11] is to use q-ary constituent codes based on non-binary partitioning of the signal set. However, using binary codes in conjunction with multilevel codes turns out to be asymptotically optimal. In order to choose the best rate for each constituent code, Huber et al. [13] and Kofman et al. [14] proved that the capacity of the adopted modulation scheme can be achieved by multilevel codes together with MSD if and only if the individual rates of the constituent codes are chosen properly. In these papers, it is assumed that the signal points are equiprobable and the partitioning is regular. In [10], the authors generalized these results to arbitrary signaling and labeling of signal points by means of the chain rule for mutual information. In this manner, it is possible to create a model with virtually independent parallel channels for each address bit at the different partitioning levels. These partitioning levels are called equivalent channels.

In order to better understand the idea beneath this concept, consider the previously described modulation scheme with $L = 2^{\lambda}$ signal points. Since each of the signal points exists in a L-dimensional signal space, every signal point is taken from the signal set $T = \{\tau_0, \tau_1, \ldots, \tau_{L-1}\}$ where $T \subset \mathbf{R}^L$ ($\mathbf{R}$ being the field of real numbers). The channel output signal points come from the alphabet $Y = \mathbf{R}^L$.

Each possible input binary vector $\mathbf{x} = [x_0, \ldots, x_{\lambda-1}]$ is then mapped into a signal point. Since the mapping has to be bijective in order to create an

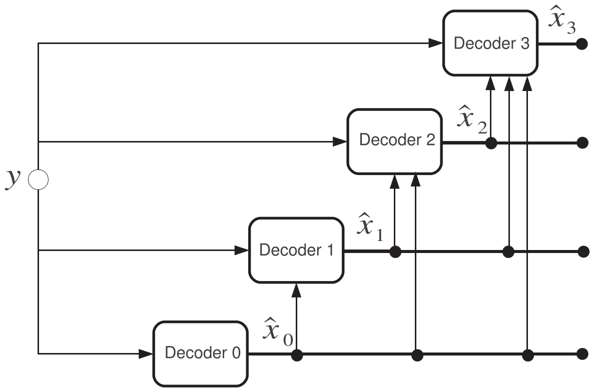

Figure 2.2: Multistage decoding for 16-ary modulation

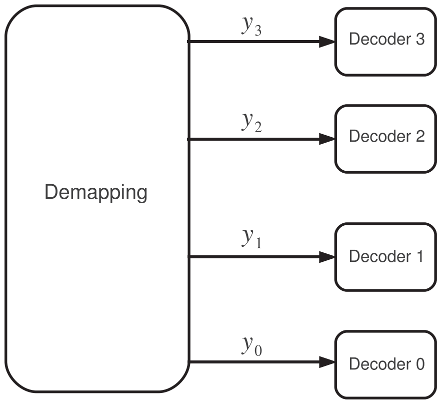

Figure 2.3: Parallel independent decoding for 16-ary modulation

effective error-correcting code, the mutual information, $I(Y;T)$, between the transmitted signal point $\tau \in T$ and the received signal point $y \in Y$ equals the mutual information, $I(Y;X_0^{\lambda-1})$, between the mapper binary input $\mathbf{x} \in \{0,1\}^\lambda$ and the received signal point $y \in Y$. X_a^b is $[X_a, X_{a+1}, \ldots, X_b]$.

Applying the chain rule of mutual information, it is possible to obtain the following:

$$\begin{aligned} I(Y;T) &= I(Y;X_0^{\lambda-1}) \\ &= I(Y;X_0) + I(Y;X_1|X_0) + \ldots \\ &+ I(Y;X_{\lambda-1}|X_0^{\lambda-2}). \end{aligned} \tag{2.6}$$

Essentially, this shows that the transmission of binary vectors over the physical channel can be separated into the parallel transmission of each single bit x_i over λ equivalent channels with $x_0, \ldots, x_{i-1}$ known. In other words, the mutual information $I(Y;X_\kappa|X_0^{\kappa-1})$ of the κ-th equivalent channel can be easily calculated as the following:

$$I(Y;X_\kappa|X_0^{\kappa-1}) = \qquad I(Y;X_\kappa^{\lambda-1}|X_0^{\kappa-1}) - I(Y;X_{\kappa+1}^{\lambda-1}|X_0^{\kappa}). \tag{2.7}$$

Since the different subsets at one partitioning level may not be symmetric, the mutual information $I(Y;X_\kappa,\ldots,X_{\lambda-1})$ is calculated by averaging over all possible combinations of $x_0^{\kappa-1} = x_0, \ldots, x_{\kappa-1}$. Specifically:

$$I(Y;X_\kappa^{\lambda-1}|X_0^{\kappa-1}) = E_{x_0^{\kappa-1}\in\{0,1\}^\kappa}\left[I(Y;X_\kappa^{\lambda-1}|x_0^{\kappa-1})\right]. \tag{2.8}$$

Since the physical channel is characterized by the set $\{f_Y(y|\tau)|\tau \in T\}$ of conditional probability density functions of the received point y given the transmitted signal point τ, assuming the bits in the lower levels, $x_0^{\kappa-1}$, are fixed, the κ-th equivalent channel is characterized by the pdf $f_y(y|x_\kappa, x_0^{\kappa-1})$.

The underlying signal subset for the equivalent κ-th modulator is given by $T(x_0^{\kappa-1})$, which denotes the partition of the signal set with the set of bits $x_0^{\kappa-1}$ in common. Since the binary symbol x_κ is potentially represented several times in this subset, the signal point τ is in effect chosen uniformly from the subset $T(x_0^\kappa)$. Therefore, $f_Y(y|x_\kappa, x_0^{\kappa-1})$ is given by the expected value of the pdf $f_Y(y|\tau)$ over all signal points τ out of the subset $T(x_0^\kappa)$, as follows:

$$\begin{aligned} f_Y(y|x_\kappa, x_0^{\kappa-1}) &= E_{\tau \in T(x_0^\kappa)}\left[f_Y(y|\tau)\right] \\ &= \frac{1}{P(T(x_0^\kappa))} \sum_{\tau \in T(x_0^\kappa)} P(\tau) \cdot f_Y(y|\tau). \end{aligned} \tag{2.9}$$

The κ-th equivalent channel is completely characterized by a set of probability density functions $\mathbf{f}_Y(y|x_\kappa)$ of the received point y if the binary symbol x_κ is transmitted. Moreover, since the subset for transmission of symbol x_κ depends on the symbols at levels 0 through $\kappa - 1$, the set of pdf's, $\mathbf{f}_Y(y|x_\kappa)$, is the set of $f_Y(y|x_\kappa, x_0^{\kappa-1})$ for each possible combination of $x_0^{\kappa-1}$. Specifically:

$$\mathbf{f}_Y(y|x_\kappa) = \left\{ f_Y(y|x_\kappa, x_0^{\kappa-1}) | x_0^{\kappa-1} \in \{0,1\}^\kappa \right\}. \tag{2.10}$$

The multilevel coding approach together with its multistage decoding procedure is a consequence of the chain rule described in (2.6). The binary symbols x_i, $i = 0, \ldots, \lambda - 1$, come from independently encoding different data symbols. Each binary encoder generates words $\mathbf{x}_i = [x_{i_1}, \ldots, x_{i_N}]$ of the component code ξ_i, where $x_{i_j} \in \{0,1\}\, \forall j \in \{1, \ldots N\}$.

From a theoretical point of view, MLC allows the constituent codes to differ in form and even blocklength, as long as each is operating sufficiently close to the capacity of its layer. However, in thi section each constituent code is assumed to be a LDPC code and each layer uses a constituent code with N output bits. Regardless of blocklength, it is still possible to define different rates for every ξ_i, resulting in different lengths of the encoder inputs, denoted K_i.

Using this notation, the rate of the i-th encoder is defined as $R_i = K_i/N$. The codeword symbols, $x_{i_j} \in \mathbf{x}_i$, form the the binary address $\mathbf{x}_j = \left[x_{0_j}, \ldots, x_{\lambda-1_j}\right]$, which is mapped to the signal point $\tau_j \in T$, with $|T| = 2^\lambda$. The code rate, R, of this scheme is equal to the sum of the individual code rates, R_i, as follows:

$$R = \sum_{i=0}^{\lambda-1} R_i = \sum_{i=0}^{\lambda-1} \frac{K_i}{N}. \tag{2.11}$$

As determined by the MSD procedure, the constituent codes ξ_i are succesively decoded by the corresponding decoders, D_i (Figure 2.2). At the i-th stage, D_i processes the block, $\mathbf{y} = [y_1, \ldots, y_N]$ $(y_j \in Y)$, of received signal points using the decisions, $\widehat{\mathbf{x}}_l$, from the i previous decoding stages (i.e. $l = 0, \ldots, i-1$).

This procedure necessarily introduces delays in the decoding process: here their effects associated with such a structure are not considered. In order to satisfy the chain rule (2.6) and preserve the mutual information, the estimated symbol, $\widehat{\mathbf{x}}_l$, is required to equal to the transmitted symbol, $\mathbf{x}_l$. Therefore, if it is assumed that error free decisions are generated by the decoders, D_i, MSD can be interpreted as an implementation of the chain rule (2.6), and hence is mutual information preserving.

In order to approach channel capacity, it is necessary to maximize the mutual information over all controllable parameters. Usually, these are the *a priori* probabilities of the signal points. Therefore, a specific channel-input probability distribution, $P(\tau)$, is required in order to achieve the channel capacity, C. These probabilities can not be optimized independently for each individual level, and hence the entire signal set must be considered. The capacity of the i-th equivalent channel, C_i, is given by the respective mutual informations, $I(Y; X_i|X_0^{i-1})$, corresponding to the channel input probabilities. C_i is then given as follows:

$$\begin{aligned} C_i &= I(Y; X_i|X_0^{i-1}) \\ &= E_{x_0^{i-1}}\left[C(T(x_0^{i-1}))\right] - E_{x_0^i}\left[C(T(x_0^i))\right], \end{aligned} \tag{2.12}$$

where $C(T(x_0^i))$ denotes the capacity when using the subset $T(x_0^i)$ with *a priori* probabilities $P(\tau)/P(T(x_0^i))$. At this point, it is possible to determine the capacity $C = C(T)$ for a 2^λ-ary digital modulation scheme given the *a priori* probability distribution, $P(\tau)$, of the signal points $\tau \in T$. In particular, C is equal to the sum of the capacities of the equivalent channels, C_i, in the MLC scheme:

$$C = \sum_{i=0}^{\lambda-1} C_i. \tag{2.13}$$

The capacity, C, can be approached via MLC-MSD if the individual rates, R_i, are chosen to be arbitrarily close to (but not greater than) the capacities of the equivalent channels C_i.

In order to lower the latency of the MLC system, a different decoding scheme has been studied in [7] and [10]. In the MLC with parallel indepedent decoding structure, each decoder D_i does not use the decisions of the other levels $j \neq i$ (Figure 2.3). In [10], the authors showed how the mutual information of the modulation scheme can be approached with MLC-PID if and only if the rate R_i of each code is set in order to fulfill $R_i = I(Y; X_i)$. Moreover, they showed that the MLC-PID approach represents a suboptimal solution of an optimum coded modulation scheme and that the capactiy of such a scheme strongly depends on the particular labeling of signal points. However, they also showed how the gap to an optimum scheme can be very small using a Gray labeling of the signal points.

2.1.3 Combination of q-ary LDPC codes and q-ary Modulation

In this final architecture that is analyzed, LDPC codes over $GF(q)$ ($q = 2^p$, p a positive integer) are combined with q-ary modulation to achieve bandwidth-efficient transmission (Figure 2.4). For a chosen code rate, R, and a block-length, N, it is necessary to find a PC matrix, $H = \{h_{ij}\}_{i=1,\dots,M, j=1,\dots,N}$, where $h_{ij} \in GF(q)$ and $R = 1 - \frac{M}{N}$. In this manner, the $K = NR$ information symbols and the M parity symbols are encoded into a q-ary vector $\mathbf{x} \in GF(q)^N$.

After q-ary LDPC encoding, the N elements of $\mathbf{x}$ are mapped into the modulated sequence $\mathbf{s} = \{s_j\}_{j=1,\ldots,N}$. This sequence depends on the address given by $\mathbf{x}_b = \left\{\overline{x}_b^j\right\}_{j=1,\ldots,N}$, where $\overline{x}_b^j = \left\{x_{b_k}^j\right\}_{k=0,\ldots,p-1}$ is the binary representation of the non-binary codeword symbol x_j. Therefore, the bandwidth efficiency of this structure is equal to $R \cdot p$.

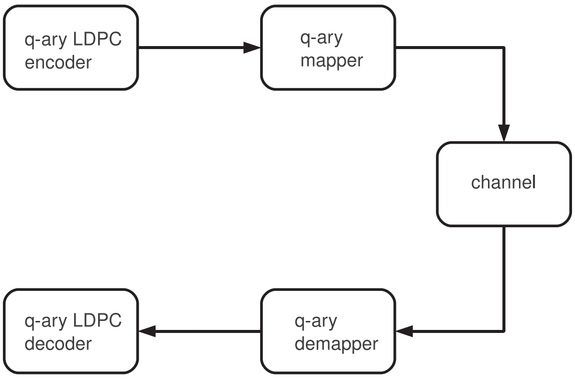

Figure 2.4: Block diagram of the structure that combinates q-ary LDPC code and q-ary modulation

At the receiver, the output of the AWGN channel may be expressed as:

$$y_\kappa = s_\kappa + n_\kappa = (s_{\kappa_I} + js_{\kappa_Q}) + (n_{\kappa_I} + jn_{\kappa_Q}) = y_{\kappa_I} + jy_{\kappa_Q}, \tag{2.14}$$

where $\kappa = 1, \ldots, N$ and $n_{\kappa_I}, n_{\kappa_Q}$ are two independent noises with the same variance, σ^2, related to the in-phase and quadrature component of the modulated signal. Starting with $P(y_\kappa|s_\kappa)$, and using the Bayes' theorem [20], the *a posteriori* probability distribution can be written as:

$$P(s_\kappa|y_\kappa) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(y_{\kappa_I} - s_{\kappa_I})^2 + (y_{\kappa_Q} - s_{\kappa_Q})^2}{2\sigma^2}\right). \tag{2.15}$$

The probabilities in (2.15) are used to initialize the message passing algorithm in the decoder [3]. It is worth to note that the computational complexity of the algorithm provided by [3] may be reduced by employing Fast Fourier Transform (FFT) or the Fast Hadamard Transform (FHT) approach [20].

2.1.4 Simulation results

In this section, simulation results obtained by implementing the three structures introduced in the previous sections are discussed. In each of these implementions, a Gray-mapped 16-QAM modulation and a global bandwidth efficiency of 2 bits/symbol (i.e. a coding rate equal to 0.5) are taken into account. The next subsections introduce the simulation results obtained depending on the input blocklength.

Input blocklength set to 5000 bits

The input blocklength was initially set to 5000 bits per codeword. For the system described in subsection 2.1.1, the binary LDPC code had then blocklength $N = 10000$ and rate 0.5. The variable-node degree distribution, following the notation introduced in [21] and [22] and according to (2.2) and [20] , was $\lambda_8 = 0.2$ and $\lambda_9 = 0.8$, where $\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1}$, and d_v is the maximum symbol-node degree. In what follows, the maximum number of iterations between the soft-detector and the LDPC decoder is set to 30 [12].

In order to make a fair comparison between architectures, the PC matrix of the 16-ary LDPC code used in the architecture introduced in subsection 2.1.3 also had a rate equal to 0.5, while the blocklength N was set to 2500 symbols, and the variable-node degree distribution was $\lambda_2 = 0.8$ and $\lambda_3 = 0.2$ [20]. For this decoding architecture and the MSD architecture, the maximum number iterations performed by the LDPC decoder has been set to 25.

The MLC structure is defined by $4 = \log_2(16)$ binary LDPC codes corresponding to each address bit. They each had initially blocklength $N = 2500$

and variable-node degree distribution λ_2=0.8 and λ_3=0.2. Each rate is defined to be $[R_0, R_1, R_2, R_3] = [0.337, 0.663, 0.337, 0.663]$ in the MSD case and $[R_0, R_1, R_2, R_3] = [0.349, 0.651, 0.349, 0.651]$ in the PID case. These values agree with the ones in [7], since 16-QAM can be interpreted as product of two indepedent 4-PAM costellations.

The simulation results plotted in Figure 2.5, 2.6 with the dashed line show how the q-ary LDPC code architecture from subsection 2.1.3 outperforms the binary LDPC turbo-like architecture of subsection 2.1.1. In particular, the gain is about 2.5 dB in terms of signal-to-noise ratio. Moreover, the MLC architectures outperforms the turbo-like architecture, however they do not perform as close to capacity as the q-ary LDPC coded architecture introduced in subsection 2.1.3.

Further, for sake of an overall view on the spectrally efficient coded modulations, in Figure 2.5 the performance of the LDPC coded MLC-MSD structure on a 4-PAM modulation exposed in [7] have been plotted in dash-dot line.

The authors in [7] proposed two architectures having a coding rate equal to 1/2, an input blocklength fixed to 5000 bits and based respectively on an optimized LDPC code and a quasi-regular LDPC code.

In order to better exploit the influence of the codeword length of each subcode in the layered structures, a MLC-MSD architecture involving two subcodes with rates $[R_0, R_1] = [0.337, 0.663]$ and input blocklengths fixed at 2500 bits per layer has been set up as well.

Given the simulation results (Figure 2.5), the QR structure proposed in [7] and the 2 layer MLC-MSD architecture outperform the 4 layer MLC-MSD architecture. The gain is about 0.35 dB and 0.15 dB in terms of SNR respectively. On the other hand, the optimized structure ("opt") proposed in [7] performs really close to the q-ary LDPC code architecture from subsection 2.1.3.

Input blocklength set to 20000 bits

In order to exploit the capacity-approaching behaviour of the structures described in subsection 2.1.2 and 2.1.3, the input blocklength was raised up to 20000 bits per codeword. Thus, the blocklength of the 16-ary LDPC code used in the architecture of subsection 2.1.3 was set to 10000 as that of each subcode of the MLC architectures as well. The other characteristic parameters have not been changed. The simulation results (Figure 2.5, 2.6 in solid line) show how the MLC-MSD structure gets closer to the performance of the one using the 16-ary LDPC code and gains about 0.2 dB on the MLC-PID one. Nevertheless, the architecture described in subsection 2.1.3 still shows the best error-rate performance.

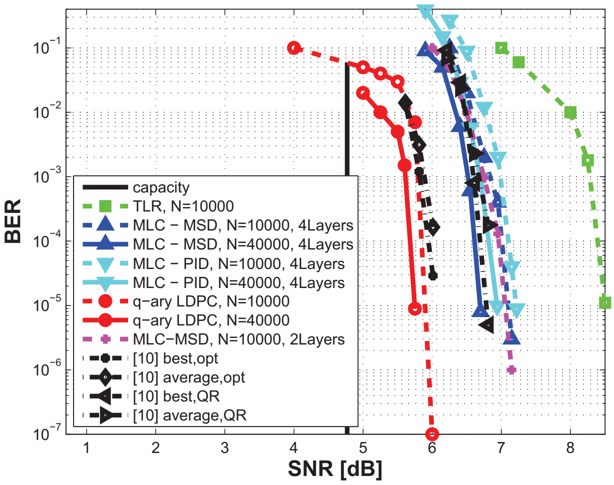

Figure 2.5: Bit-error rate performance of the architectures discussed in subsections 2.1.1, 2.1.2 and 2.1.3 on the AWGN channel: turbo-like receiver (TLR), multilevel coding with multistage decoding (MLC-MSD) and parallel independent decoding (MLC-PID), q-ary LDPC code combined with $q$ary modulation. N represents the codeword blocklength in bits.

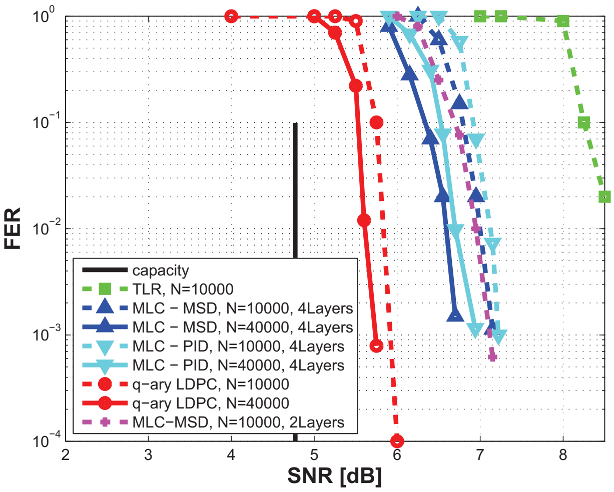

Figure 2.6: Frame-error rate performance of the architectures discussed in subsections 2.1.1, 2.1.2 and 2.1.3 on the AWGN channel: turbo-like receiver (TLR), multilevel coding with multi-stage decoding (MLC-MSD) and parallel independent decoding (MLC-PID), q-ary LDPC code combined with $q$ary modulation. N represents the codeword blocklength in bits.

2.1.5 Conclusions

Three higher-order coded modulations employing LDPC codes were introduced and analyzed in order to study their corresponding trade-offs between bandwidth-effciency and bit-error-rate performance.

Simulation results for 16-QAM modulation schemes showed that the best performance can be achieved by using a code whose alphabet size matches the modulation order. Consequently, using such an architecture, associating each non-binary coded symbol to a modulated symbol appears to be have the best performance in an environment (such the wireless one) where high bandwidth-efficiency and good error-correction capability is desirable. However, this performance does not come at a cost of increased decoding complexity.

Future directions for research could focus on the behavior of the proposed architectures over different channels and with different modulation schemes, as well as different LDPC codes, having different codeword length or degree-distribution profile as in [35]. Further, since a complete analysis of decoding architectures in terms of latency and complexity is lacking in the literature, future works could potentially highlight such features.

Moreover, in order to complete the analysis on the spectrally efficient coded modulations, an optimization of the construction of q-ary LDPC codes has to be taken into account, starting from the results provided in [26] and [28]. Specifically, the next section introduces a brand new construction method that provides an important improvement of the error-rate performance in the waterfall region.

2.2 Protograph q-ary LDPC codes

This section presents a protograph-based design approach for LDPC codes using modulation larger than QPSK. Several papers [3], [4], [20], [27], [33] - [35] have investigated the optimization of q-ary LDPC codes. Among these papers, the primary goals have been the following:

- improving the waterfall region;
- lowering the error floor;
- controlling the trade off between decoding complexity and error-rate performance.

For Galois fields with more than two elements, there has been limited work on optimizing the left and right LDPC degree distributions in the context of a code ensemble analysis, such as density evolution [31] has done for $GF(2)$. Most notable in this regard is the recent work of Urbanke [27] that provides degree distributions for field sizes up to 8.

The common design approach has begun with the design of a binary mother PC matrix using a binary degree distribution obtained through density evolution and a specific PC matrix satisfying that distribution obtained using well known algorithms such as [18] or [19]. This design approach then replaces each binary one in that matrix with a non-zero $GF(q)$ element and each binary zero with a $GF(q)$ zero.

A key issue is the choice of the distribution of the non-zero $GF(q)$ elements within a row. The $GF(q)$ elements chosen to replace each non-zero entry in the binary mother matrix typically have to fulfill the row minimum distance condition [33], [34]. Specifically, in [33] MacKay found several valid sequences for 16-ary and 64-ary LDPC codes using a Monte Carlo approach, choosing $GF(q)$ values in order to maximize the marginal entropy of the syndrome related to the output sequences. In [34] the authors proposed a method based on the binary images of each element living in the Galois Field $GF(q)$, considering therefore the binary row minimum distance to be optimized and providing results for 64-ary and 256-ary LDPC codes.

In this section, a new design technique for q-ary LDPC codes that maximizes the row minimum distance by using proper sequences of q-ary elements in a protograph-based binary PC mother matrix [26], [29] is provided. This method aims to improve the waterfall region for q-ary LDPC codes. It does not address the decoding complexity of q-ary LDPC codes. The next subsection introduces the protograph-based construction scheme. Following that, the procedure to properly choose the non-zero entries in the PC matrix is described.

2.2.1 Protograph LDPC codes

Several papers have investigated the design of LDPC codes with imposed substructures, from multi-edge-type codes [32] to quasi-cyclic (QC) codes [30]. Protograph-based codes are structured codes as well [29], [26].

A protograph is a Tanner graph with a relatively small number of nodes [29]. Given a set of variable nodes $V = \{v_i\}_{i=1,\ldots,|V|}$ and a set of check nodes $C = \{c_j\}_{j=1,\ldots,|C|}$, each edge of the protograph has to connect a variable node

v_i to a check node c_j; parallel edges are permitted [29].

In order to obtain larger derived graphs of various sizes, a "copy and permute" operation can be applied to the protograph. This operation consists of first making G copies of the protograph and then permuting the endpoints of each edge among the G variable nodes and the G check nodes connected to the set of G edges copied from the same edge in the protograph. The derived graph is the graph of a code G times as large as the code corresponding to the protograph, with the same rate $R_P = 1 - \frac{|C|}{|V|}$ and the same distribution of variable and check node degrees as the protograph. In fact, the derived graph contains $G \cdot N_P$ transmitted nodes, and $G \cdot M_P$ check nodes.

The local neighborhood of a node in the derived graph is completely determined by the protograph [29]. The local neighborhood to depth d consists of all nodes and edges connected to a given node by a path of length d or less. This neighborhood is a tree if there is at most one path of length d or less to any other node. If the neighborhood of a variable node in the derived graph is a tree, the connections among the nodes are still determined by the protograph structure. As a result, density evolution analysis [31] can be applied on the protograph to determine whether or not decoding will yield arbitrarily small bit-error probability on a large derived graph.

In [26], the authors discuss protograph codes that benefit from both capacity-approaching thresholds and linear minimum distance growth. They provide methods to compute iterative decoding thresholds and asymptotic ensemble weight enumerators for protograph-based LDPC codes as well.

The analysis they provide aims to compute the ensemble weight enumerator for an LDPC code ensemble built from a protograph. The normalized weight distribution is used to obtain an upper bound on the threshold of E_b/N_0 when the code ensemble is used on an AWGN channel with maximum-likelihood (ML) decoding and to determine whether or not the minimum distance of typical codes in the ensemble increases linearly with the code length. Thus, the asymptotic ensemble weight enumerator is used to determine whether the code ensemble achieves linear minimum distance growth, i.e. whether the minimum distance of most codes in the ensemble increases linearly with the blocklength.

Also in [26], the authors provide design methods that guarantee code ensembles constructed from certain protographs have linear minimum distance growth. This property holds for protographs having all variable node degrees equal to 3 or higher. However, it is known from the analysis in [22] and the results provided in [20] that good iterative decoding thresholds for LDPC codes can require a substantial fraction of degree-2 variable nodes. To resolve this conflict, methods in [26] allow the addition of degree-2 and degree-1 nodes to improve the iterative decoding threshold while preserving the linear minimum distance growth property. Specifically, the "check node splitting technique" in [26] accomplishes this.

In this section, the analysis provided in [26] is used to construct protograph LDPC codes that have the linear minimum distance growth property. A binary mother matrix from a protograph having the linear minimum distance growth property by using the "copy and permute" operation is derived.

Let P be a protograph containing only transmitted variable nodes having degree equal to 2 or higher and let S_P be the subgraph of P containing only its degree-2 variable nodes and their attached edges and checks. In case the subgraph is not connected, decompose it into its disjoint connected pieces $S_P(j)$. Thus, $S_P = \bigcup_j S_P(j)$. Each connected subgraph $S_P(j)$ has n_j degree-2 variable nodes. The protograph P satisfies the check node splitting condition (that is, has the minimum distance that grows linearly with the blocklength) if each subgraph $S_P(j)$ involves at least $n_j + 1$ check nodes. Verifying this condition provides binary mother matrices suitable to be optimized with a proper choice of the q-ary elements, that will be introduced in the next subsection.

2.2.2 Selecting the q-ary elements in the PC matrix

In this subsection a method to optimize the error-rate performance of a q-ary LDPC code having a binary mother PC matrix constructed as in subsection 2.2.1 is considered. The optimization of q-ary LDPC codes has been studied since their introduction in the late 90's [3], [4]. MacKay [33] and Poulliat et al. [34] have addressed the problem of the proper selection of specific $GF(q)$ elements to replace the ones in a given binary mother PC matrix.

In [33], the author considered the PC matrix as constructed using the algorithm presented in [18]. He examined the marginal entropy of a single element of the syndrome vector for each choice of the κ non-zero entries in a row of the PC matrix. He chose this metric because if the entropy of the syndrome increases, then an optimal decoder can get closer to the Shannon limit.

For a specified number κ of ones in a row of the binary mother PC matrix, the entropy-maximizing algorithm of [33] provides a set of κ distinct field elements that can be used in any order to replace the κ ones in that row. In [33] results are shown for LDPC codes over $GF(16)$ and $GF(64)$ having 4 and 5 non-zero elements per row.

In [34], the authors propose a method for selecting the specific $GF(q)$ elements to replace the ones in a binary mother PC matrix constructed by using either PEG [18] or the ACE algorithm [19]. This optimization scheme is based on the binary image representation of the code and it aims to improve the waterfall region. It intends to lower the computational cost of the method proposed in [33] for high order fields.

The idea behind that scheme is as follows: the higher the minimum distance computed on the binary image, the more distinguishable (that is, reliable) the messages involved in the MP algorithm. Therefore, the algorithm seeks to maximize the minimum distance of the associated binary code, selecting the κ-tuples with the maximum minimum distance and, among them, those with the smallest weight enumerator coefficient [34], [35].

By comparing the results of both the aforesaid methods, the authors in [34] observed that the sets they obtained may include the sets given in [33]. Specifically, they found that the method proposed in [33] could provide the set of κ-tuples with the smallest weight enumerator coefficient for $GF(16)$ and for $GF(64)$ too, if κ is set to 5. On the other hand, the method of Poulliat et al. could optimize rows, while MacKay's is too computationally intensive for larger field sizes.

Thus, it is proper to select the non-zero entries as follows for a given binary mother PC matrix derived from a protograph as in subsection 2.2.1. Let H_b be the binary mother PC matrix derived from a protograph by using the "copy and permute" operation. H_b has N columns and M rows: each element of H_b lives in $GF(2)$. Let $\Omega(\kappa|q)$ be the set of length-κ sequences $\Omega_j(\kappa|q)$ (where $j = 1, \ldots, |\Omega(\kappa|q)|$) having maximum minimum distance computed on the binary image of the related q-ary LDPC code.

To obtain the optimized PC matrix of the q-ary LDPC code, each row of H_b is considered. Then, the sequence of d_{c_i} non-zero entries of H_b is replaced with a d_{c_i}-tuple picked from the set $\Omega(d_{c_i}|q)$ provided in [33] and [34], where d_{c_i} represents the degree of the i-th check node, $i = 1, \ldots, M$.

In case the weight of a given row did not match the length of the provided sequences, the following procedure is used. As previously mentioned in this subsection, the i-th row of the PC matrix of the q-ary LDPC code can be a permutation or multiplication by a constant of d_{c_i}-tuples of the set $\Omega(d_{c_i}|q)$. Therefore, each element $\Omega_j(d_{c_i}|q) \in \Omega(d_{c_i}|q)$ can be written as an ordered sequence as $\Omega_j(d_{c_i}|q) = \alpha_r^v = [\alpha^r, \alpha^s, \ldots, \alpha^u, \alpha^v]$ where α is the primitive element of the Galois field $GF(q)$, $|r - v| = d_{c_i}$, $r < s < \ldots < u < v$ and $\{r < s < \ldots < u < v\} \in \{0, \ldots, q-2\}$.

Then, it is possible to derive a $(d_{c_i} + 1)$-tuple from $\Omega(d_{c_i}|q)$ by adding an element [34] as $\Omega_l(d_{c_i} + 1|q) = [\Omega_j(d_{c_i}|q), \alpha^a] = [\alpha_r^v, \alpha^a]$ where $v < a$, $a \in \{0, \ldots, q-2\}$, $j = 1, \ldots, |\Omega(d_{c_i}|q)|$, $l = 1, \ldots, |\Omega(d_{c_i} + 1|q)|$. The matrix obtained after this operation is called H_r.

To completely define the PC matrix of the q-ary LDPC code, a law that multiplies every non-zero element in each row of H_r by a constant is applied. The law $L(\gamma_i)$ that multiplies each element in the i-th row for a factor α^{γ_i} is applied, as follows: $L(\gamma_i) : \alpha^{u_j} \longmapsto \alpha^{u_j} \cdot \alpha^{\gamma_i}$ where $\{u_j, \gamma_i\} \in \{0, \dots, q-2\}$, $j = 1, \dots, d_{c_i}$ and α represents the primitive element of $GF(q)$.

In order to avoid the occurrence of cycles in the PC matrix, $L(\gamma_i)$ has to satisfy the full rank condition (FRC) [34]. Therefore, the value of γ_i for each $L(\gamma_i)$ has to be chosen such that, given two rows k and l having the same degree $d_{c_k} = d_{c_l}$, $\gamma_k \neq \gamma_l$. Finally, once $L(\gamma_i)$ has been applied to all the rows, the PC matrix H for the considered q-ary LDPC code is achieved: $H = \{h_{ij}\}_{i=1,\dots,M; j=1,\dots,N}$, $h_{ij} \in GF(q)$. .

2.2.3 Simulation results

This section presents simulation results obtained by implementing 16-ary modulation using 16-ary LDPC codes. Thus, the system model has been exposed in subsection 2.1.3. In each implementation, a symmetric ultracomposite [6] Gray-labeled 16-QAM modulation with a bandwidth efficiency of 2 bits/symbol (i.e. a coding rate R equal to 0.5) is used. For each 16-ary LDPC code that is considered, the blocklength N was set to 2500 symbols (10000 bits). In this section quasi-regular LDPC codes [20], [23] are considered.

The codes that are taken into account are designed and simulated using four approaches for a variety of average column weights. First, two binary mother PC matrices are constructed: one by using the PEG algorithm [18] and one by using the protograph-based algorithm introduced in subsection 2.2.1. For the protograph-based PC matrix, the number of transmitted variable nodes N_P of the protograph has been set to 10, while the number of check nodes M_P of the protograph has been set to 5. The protograph is copied and permuted 250 times to produce the 2500-symbol LDPC code. For both the PEG and the protograph-based PC matrices the specific $GF(q)$ elements to replace the ones in the binary mother PC matrix are chosen both randomly and by using the

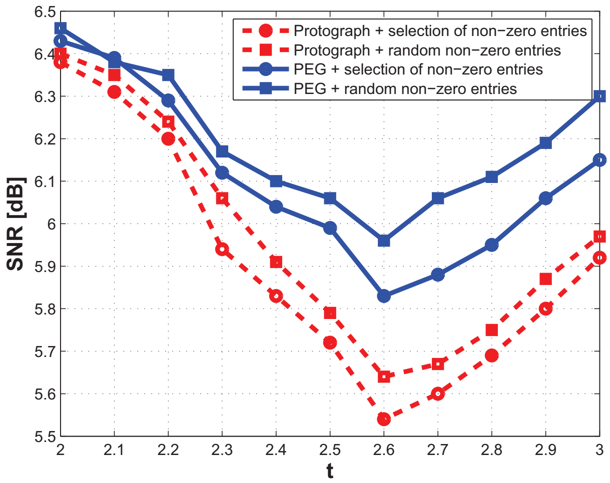

Figure 2.7: SNR at which each considered q-ary LDPC code has BER=10^{-5} for different values of the average column weight t. The codes taken into account are PEG-based and protograph-based, with random or careful selection of the non-zero entries in the PC matrix.

selection method introduced in subsection 2.2.2. The primitive polynomial of the considered 16-ary Galois Field is $p(x) = 1 + x + x^4$.

Figure 2.7 plots the SNR required to achieve a BER=10^{-5} for codes with a variety of average column weights t. Performance of the codes whose binary mother PC matrix was constructed using the PEG algorithm are plotted as blue solid lines, while performance of those constructed by using the protograph-based algorithm introduced in subsection 2.2.1 are plotted as red dashed lines. Performance of codes produced by the random selection of $GF(q)$ elements are identified by a square marker, and performance of codes produced by careful selection of the non-zero entries according to 2.2.2 are identified by a circle marker.

This figure shows how the linear minimum distance growth property, which holds for $t > 2.6$, influences the performance of the protograph-based codes. As t decreases from 2.6 to 2 the benefit provided by the protograph-based approach over the PEG approach for the same (either random or selected) $GF(q)$ element selection algorithm decreases from about 0.3 dB to less than

0.05 dB. For values of t above 2.6 the benefit provided by the protograph-based design is relatively constant. Also, Figure 2.7 clearly shows that the performance is best for $t = 2.6$, when it is just large enough to provide the linear minimum distance growth [26].

Let us look more closely at the $t = 2.6$ design, which has the variable-node degree distribution [22] $\lambda_2 = 0.4$ and $\lambda_3 = 0.6$. In that case, the binary adjacency matrix H_P of the protograph is as follows:

$$H_P = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{2.16}$$

Figures 2.8 and 2.9 show the bit-error-rate and frame-error-rate (FER) performance respectively from our simulation results for $t = 2.6$ codes. The proposed q-ary LDPC code constructed by the combination of the protograph-based mother code construction of subsection 2.2.1 and the $GF(q)$ selection algorithm of section 2.2.2 outperforms the other codes. By itself, the selection of the non-zero entries according to 2.2.2 provides about 0.1 dB of gain in terms of SNR w.r.t. the random selection for both the PEG-based and the protograph-based LDPC codes. By itself, the protograph-based mother code construction of subsection 2.2.1 outperforms the PEG-based non-protograph construction algorithm by about 0.3 dB for either $GF(q)$ selection algorithm. Overall, the benefit of a protograph-based design and the $GF(q)$ selection algorithm of section 2.2.2 provides a 0.4 dB benefit.

In order to compare the proposed construction method to the existing state of the art of the bandwidth-efficient coded modulation schemes, in Figure 2.8 the performance of the optimized LDPC coded structure on a 4-PAM modulation exposed in [7] have been plotted as a black dash-dot line. The architecture proposed in [7] is based on a multilevel coding approach and it uses a multistage decoding system. Further, it has a coding rate equal to 1/2 and an input blocklength fixed to 5000 bits. The protograph-based LDPC codes proposed in this section outperform that proposed in [7] by about 0.4 dB. Moreover, the 16-ary PEG-based LDPC code with selected field elements outperforms by about 0.15 dB the structure proposed in [7].

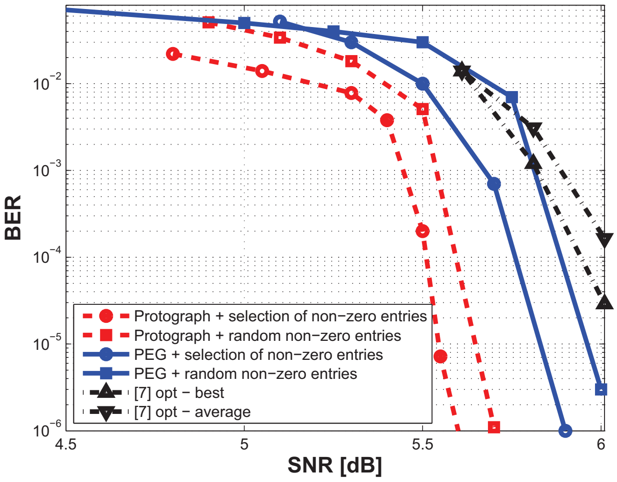

Figure 2.8: Bit-error rate performance of the considered q-ary LDPC codes on the AWGN channel: PEG-based and protograph-based, with random or careful selection of the non-zero entries in the PC matrix. The channel capacity is 4.77 dB. Average and best BER performance of the optimized ("opt") LDPC coded structure proposed in [7] are provided as well.

2.2.4 Conclusions

This section introduced a protograph-based method for designing q-ary LDPC codes for use with modulations larger than QPSK. Simulations focused on a $GF(16)$, 16-QAM example. The proposed construction method achieves the maximum gain when the average column weight is chosen so that the linear minimum distance growth property is satisfied. In this region, the benefit of a protograph-based design over a standard PEG approach was 0.3 dB. For selection of the specific Galois field elements, a careful selection algorithm

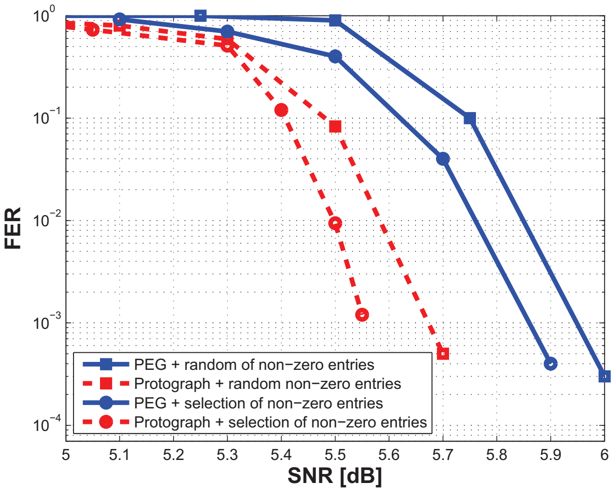

Figure 2.9: Frame-error rate performance of the considered q-ary LDPC codes on the AWGN channel: PEG-based and protograph-based, with random or careful selection of the non-zero entries in the PC matrix. The channel capacity is 4.77 dB.

provides about 0.1 dB of improvement over a random selection algorithm.

2.3 q-ary LDPC codes with low error floor

Among linear codes, LDPC codes [1] definitely play a fundamental role in approaching the channel capacity. Specifically, proper degree distribution design [22, 23, 31] allows the LDPC codes to approach the Shannon limit. Moreover, it has been shown [3,4] that the error-rate performance may be increased in terms of coding rate, blocklength and intersymbol interference (ISI) resilience (see for instance [38] and references therein) by designing LDPC codes over $GF(q > 2)$. These codes are called q-ary (or non-binary) LDPC codes.

However, decoding complexity is one of the major issues of the optimization of q-ary LDPC codes. Several papers have addressed the problem, by using modified versions of the message passing algorithm proposed in [3] (see for instance [37]) or by a proper code design. Specifically, it has been shown that regular [35], quasi-regular [20] and moderate field order ($q \leq 16$) protograph-based [39] q-ary LDPC codes may have good error-rate performance with an acceptable decoder computational complexity. Quasi-cyclic [30] q-ary LDPC codes might further lower the decoding complexity.

Recently LDPC codes have become strongly attractive for emerging storage technologies such as Flash memories and Phase Change Memories (PCMs) (see [40] and references therein). In fact, these devices require very good error-rate performance at very high code rate. The readout channel model is typically memoryless and it can be binary-input (e.g, for standard PCMs [40, 41]) or q-ary-input (e.g., for multilevel cell PCMs [42]) either.

Further, it is important to note that solid state memory devices typically work in very high SNR region. In fact, information is basically stored in resistivity levels that are separated by orders of magnitude. Therefore, LDPC codes that aim at being employed in reading architectures for the aforesaid storage techonologies must have extremely low error floors. Specifically, q-ary LDPC codes that combine very low error floors and acceptable decoding complexity may definitely play an important role in solid state device readout systems, since they typically provide very good performance in the waterfall region at high coding rate as well.

[44] shows how not all short cycles are equally harmful. Furthermore, the authors prove how small stopping sets [45] in binary LDPC codes lead to high error floors under belief propagation decoding [5]. Moreover, a very effective construction algorithm for irregular binary LDPC codes with low error floor is provided. Error floors in q-ary LDPC codes have been related to small minimum distance too [35], [27], [36]. However, designing a degree distribution for large minimum distance is computationally cumbersome. In fact, density evolution algorithm for q-ary LDPC codes is prohibitive for alphabet size larger than eight. Further, careful selections of the non-zero entries [33], [35] in the q-ary PC matrix provide good results in terms of error floors especially for very high field order, leading to a large computational complexity.

This section aims to provide efficient construction methods for q-ary LDPC codes with low error floors and moderate field order. These algorithms enlarge the size of q-ary stopping sets and q-ary linearly dependent sets. The aforesaid topological structures have been related to specific failures in q-ary LDPC decoding by a deep geometrical and graphical analysis of the q-ary LDPC decoding problem. Thus, providing large q-ary stopping sets and q-ary linearly dependent sets can alleviate the error floor of the given q-ary LDPC code.

The section is organized as follows. Section 2.3.1 describes the system model. Section 2.3.2 reports a geometrical representation of the q-ary decoding problem and some practical examples of the decoding failures that may occur. Section 2.3.3 provides a graphical analysis of the q-ary decoding problem. Specifically, definitions, properties and relationships of characteristic topological structures such as q-ary cycle sets, q-ary stopping sets and q-ary linearly dependent sets are provided. In Section 2.3.4 the correspondences between the decoder failures introduced in Section 2.3.2 and the topological structures presented in 2.3.3 are motivated. Furthermore, this section also provides the low error floor q-ary LDPC code construction algorithms. Section 2.3.5 presents the simulation results. This section also discusses the practical aspects of the code construction and decoding. Section 2.3.6 delivers the conclusions.

2.3.1 System model

This section uses LDPC codes over $GF(q = 2^p)$, with p an integer. For a chosen code rate R, and blocklength N, the LDPC code requires a PC matrix, $H = \{H_{ij}\}_{i=1,\ldots,M;j=1,\ldots,N}$, where $H_{ij} \in GF(q)$ and $R = 1 - \dfrac{M}{N}$.

In this manner, the $K = NR$ information symbols and the M parity symbols are encoded into a q-ary vector $\mathbf{x} = \{x_j\}_{j=1,\ldots,N}$, $x_j \in GF(q)$. $\mathbf{x}$ is associated with the sequence of addresses given by $\mathbf{x}_b = \{\overline{x}_b^j\}_{j=1,\ldots,N}$, where $\overline{x}_b^j = \{x_{b_k}^j\}_{k=1,\ldots,p}$ is the binary vector representation of the non-binary codeword symbol x_j.

In case a binary-input channel is considered, $\mathbf{x}_b$ is mapped into a sequence

$\mathbf{s}_b = \{\overline{s}_b^j\}_{j=1,\ldots,N}$, where $\overline{s}_b^j = \{s_{b_k}^j\}_{k=1,\ldots,p}$ is associated with $\overline{x}_b^j$. On the other hand, in case a q-ary-input channel is considered, the N elements of $\mathbf{x}$ are mapped into the modulated sequence $\mathbf{s} = \{s_j\}_{j=1,\ldots,N}$ according to the sequence of addresses $\mathbf{x}_b$.

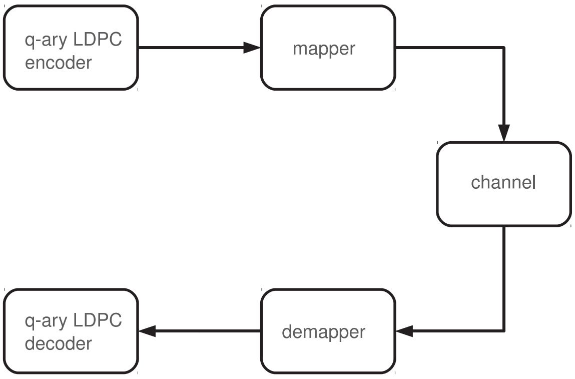

Figure 2.10: Block diagram of the system model.

At the receiver, the channel outputs are properly demapped and the probabilities that are used to initialize the message passing algorithm in the decoder [3] are computed.

Specifically, in case a binary-input channel is considered, the channel output is $\mathbf{y}_b = \{\overline{y}_b^j\}_{j=1,\ldots,N}$, where $\overline{y}_b^j = \{y_{b_k}^j\}_{k=1,\ldots,p}$. Each $\overline{y}_b^j$ is associated with the binary representation $\overline{x}_b^j$ of the j-th transmitted symbol. Moreover, since the channels that have been taken into account are memoryless, the likelihood of the j-th symbol being equal to a is $f_j^a = \prod_{k=1}^p P(x_{b_k}^j = a_k | y_{b_k}^j)$ for each $j = 1, \ldots, N$, $a \in GF(q)$ where a_k is the k-th bit of the binary representation of a.

On the other hand, in case a q-ary-input channel is considered the channel output is $\mathbf{y} = \{y_j\}_{j=1,\ldots,N}$. Each y_j is associated with the j-th transmitted symbol x_j. Further, each f_j^a can be properly computed directly from the respective channel output.

Finally, it is worth to introduce some definitions that will be used throughout the whole section. Let $\chi(H) = \{\chi(H_{ij})\}_{i=1,\ldots,M;j=1,\ldots,N}$, $\chi(H_{ij}) \in GF(2)$ be the *binary mother* matrix associated with the q-ary parity-check matrix H. In other terms, $\chi(H_{ij}) = 1 \leftrightarrow H_{ij} > 0$, while $\chi(H_{ij}) = 0 \leftrightarrow H_{ij} = 0$. Therefore, $\chi(H)$ identifies the positions of the non-zero entries in H.

Further, let $\psi(H) = \{\psi(H_{ij})\}_{i=1,\ldots,M;j=1,\ldots,N}$ be the *binary image* associated with the q-ary parity-check matrix H [35]. Each $\psi(H_{ij})$ is represented by a p x p matrix over $GF(2)$, i.e., $\psi(H_{ij}) \in GF(2)^{p\text{x}p}$. Therefore, $\psi(H)$ is a pM x pN matrix.

Specifically, let $p(x) = \alpha_0 + \alpha_1 x + \ldots + x^p$, $\alpha_i \in GF(2)\ \forall i = 0, \ldots, p-1$ be the primitive polynomial of the Galois Field $GF(q = 2^p)$. Let α be the primitive element of $GF(q)$. Finally, let A be the primitive element of $GF(q)$ under a matrix representation. Thus, A is as follows:

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & 0 \\ 0 & 0 & 0 & \ddots & 1 \\ \alpha_0 & \alpha_1 & \alpha_2 & \cdots & \alpha_{p-1} \end{bmatrix} \tag{2.17}$$

Therefore, the powers of A are the non-zero elements of $GF(q)$ as the powers of α as well. That is, $\alpha^\kappa \Leftrightarrow A^\kappa$, where $\kappa \in \{0, \ldots, q-2\}$. Thus, $\psi(H_{ij} = \alpha^t) = A^t \leftrightarrow H_{ij} > 0$, while $\psi(H_{ij}) = \underline{\underline{0}} \leftrightarrow H_{ij} = 0$. $\underline{\underline{0}}$ is the p x p all-zero matrix. It is proper to remind that additions and multiplications in $GF(q)$ correspond to additions and multiplications modulo 2 of the aforesaid matrices.

2.3.2 Geometrical representation

From a geometrical point of view, a length-N codeword of a LDPC code represents a point in a N-dimensional discrete space. Therefore, the value assigned to the i-th variable node represents the i-th coordinate in the aforesaid space.

For a q-ary LDPC code each coordinate can assume q values, from 0 to $q-1$.

Each row in the M x N parity-check matrix represents a parity-check equation, that is, a constraint that a codeword has to satisfy. In the N-dimensional space, each constraint represents in general a hypersolid. Every vertex of the i-th hypersolid is a combination of the feasible values of the variable nodes involved in the i-th parity-check equation. Therefore, every codeword is a vertex that is common to each hypersolid.

The decoder's input probabilities (hence, the channel output) define a feasibility region that binds the decoder behavior. In general, the feasibility region is the set of vertices that form a hypersolid in the N-dimensional space. Thus, the decoder aims to find the correct point in the N-dimensional space within the aforesaid feasibility hypersolid.

The erasures that occur on the data transmission strongly infer on the geometry of the feasibility region. Specifically, in case a q-ary LDPC code is employed, it is possible to make a distinction between full erasures and partial erasures. A symbol is fully erased if each element of its binary representation is erased. On the other hand, a symbol is ϕ-partially erased if only ϕ components of its binary representation are erased.

In case the i-th symbol is fully erased, the feasibility hypersolid spans all over the i-th dimension-axis. On the other hand, if the i-th symbol is ϕ-partially erased, the feasibility hypersolid spans over just $2^{p-\phi}$ possible values on the i-th dimension-axis, where $p = \log_2 q$.

In case of erasure channels, there are two possible reasons for a decoding failure to occur. Let $X = \{x_\kappa\}_{\kappa=1,\ldots,|X|}$ be the set of the codewords of the given q-ary LDPC code. Let $F = \{f_\kappa\}_{\kappa=1,\ldots,|F|}$ be the set of the vertices of the feasibility region. Let $C_k = \{c_{k_\kappa}\}_{\kappa=1,\ldots,|C_k|}$ be the set of the points in the N-dimensional space that belong to the hypersolid related to the k-th constraint, that is, satisfy the k-th parity check equation of the PC matrix. Thus, $\bigcap_{k=1}^{M} C_k = X$. Furthermore, let $\pi_i(z)$ be the set of the projection on the i-th dimension-axis of each element living in z.

Decoder failure - type I

Let $\Gamma(F, k)$ be the set of the cardinalities of the projections of the intersection between the feasibility region F and the hypersolid related to the k-th constraint C_k on each dimension of the N-dimensional space: that is, $\Gamma(F, k) = \{\Gamma_i(F, k) = |\pi_i(F \cap C_k)| \, |i \in \{1, \ldots, N\}\}$. Thus, let $\Gamma^+(F, k)$ be the set of the elements of $\Gamma(F, k)$ that are equal to the alphabet size of the q-ary LDPC code, i.e., $\Gamma^+(F, k) = \{\Gamma_i(F, k) : \Gamma_i(F, k) = q\}$. Further, let $I^+(F)$ be the set of the dimensions related to the elements of $\Gamma^+(F, k)$ for each $k = 1, \ldots, M$, i.e., $I^+(F) = \{i : \Gamma_i(F, k) = q, \forall k = 1, \ldots, M\}$. Finally, let $E^+(F)$ be the set of the constraints that insist over each element in $I^+(F)$, i.e., $E^+(F) = \{k : \Gamma_i(F, k) \in \Gamma^+(F, k) | i \in I^+(F)\}$.

In other terms, $\Gamma(F, k)$ takes into account the width of the feasibility region on each dimension when the k-th constraint of the q-ary LDPC code is considered. Analogously, $\Gamma^+(F, k)$ counts the symbols that are fully erased and that insist on the k-th constraint of the q-ary LDPC code at the same time. Finally, $I^+(F)$ contains the coordinates of the aforesaid symbols and $E^+(F)$ is the set of the constraints that infer on the value of the coordinates in $I^+(F)$.

Therefore, if $|\Gamma^+(F, k)| \geq 2 \; \forall k \in E^+(F)$, the decoder fails. In fact, if it is not possible to recover any information about the coordinates of at least two dimensions from any constraint infering on them, there is no way for the decoder to converge to any codeword.

Decoder failure - type II

Decoder failures of the first type occur when the decoder can not converge to any possible codeword of the given q-ary LDPC code. However, a decoding failure may occur also when the decoder can not choose among the possible codewords which is the correct one. In other terms, if $|X \cap F| \geq 2$ the decoder fails as well. In fact, there are at least two possible codewords belonging to the feasibility region. Thus, since the messages from check node to variable node and viceversa are set to the same value for two or more elements of X, the decoder keeps oscillating between at least two of the possible solutions of the system.

Examples

Decoder failures of type I and type II are related to specific characteristics of the q-ary parity-check matrix H, of the binary mother matrix $\chi(H)$ and of the binary image $\psi(H)$ of the q-ary PC matrix. In order to point out what those types of decoding failures and, in general, the geometrical representation are meant to, this subsection reports examples of decoder behaviours for several transmissions of a simple 4-ary LDPC code.

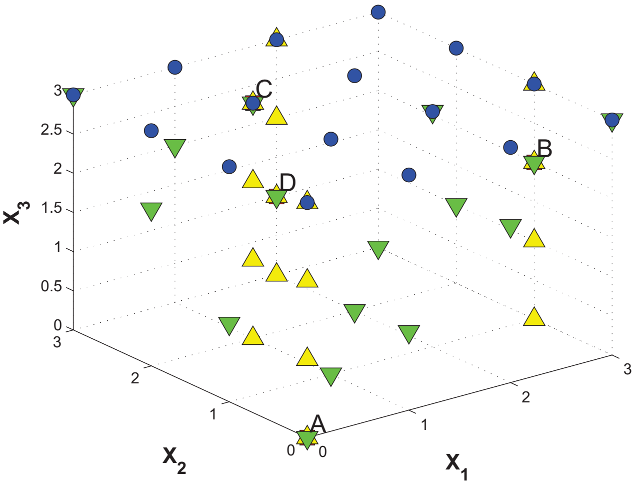

Figure 2.11: Example of decoder failure of the first type. Yellow upward-pointing triangles and green downward-pointing triangles represent the 3-tuples that satisfy the first and the second constraint in the code associated with the PC matrix in (2.18) respectively. Blue circles represent the points of the feasibility region associated with the channel output $Y = [\Theta, \Theta, 3] = [(\theta, \theta), (\theta, \theta), (1, 1)]$.

Let us consider a rate-1/3 q-ary code where $q = 4$: the number of variable nodes is $N = 3$, while the number of check nodes is $M = 2$. The related q-ary PC matrix H is as follows:

$$H = \begin{bmatrix} 2 & 1 & 0 \\ 3 & 3 & 3 \end{bmatrix} \tag{2.18}$$

Thus, the related binary mother matrix $\chi(H)$ is as follows:

$$\chi(H) = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.19}$$

On the other hand, the binary image $\psi(H)$ of (2.18) is as follows:

$$\psi(H) = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{2.20}$$

Each codeword is a 3-tuple $[x_1, x_2, x_3]$. Each symbol x_i, $i = 1, \ldots, N$ is represented by $p = \log_2 q = 2$ bits, i.e., $x_i = (x_{i_1}, x_{i_2})$, x_{i_1} being the least significant bit. Specifically, the set of the codewords of the 4-ary LDPC code related to (2.18) $X = \{A, B, C, D\}$ is exposed in Table 2.1.

Table 2.1: Codewords of the 4-ary LDPC code associated with the PC matrix in (2.18)

Codeword	**q-ary representation**	**binary representation**
A	[0,0,0]	[(0,0), (0,0), (0,0)]
B	[3,1,2]	[(1,1), (1,0), (0,1)]
C	[1,2,3]	[(1,0), (0,1), (1,1)]
D	[2,3,1]	[(0,1), (1,1), (1,0)]

Let us consider a binary erasure channel. Let $Y = [y_i]_{i=1,\ldots,N}$ be the q-ary representation of a given channel output and let $(y_{i_j})_{j=1,\ldots,p}$ be the binary representation of y_i. Further, each y_i lives in $\{0, \ldots, q-1, \Theta, \{\Theta_{\phi_J}\}_{J \in \{1,\ldots,p\}, |J|=\phi}\}$. Θ represents a symbol that has been fully erased, while Θ_{ϕ_J} represents a symbol that has been ϕ-erased on the bits defined by J. Let θ represent a bit erasure in the binary representation of a given symbol. Therefore, each element of the binary representation of Θ is θ. Furthermore, in case the i-th symbol is ϕ-erased, i.e., $y_i = \Theta_{\phi_J}$, the j-th element of the binary representation of y_i is θ if $j \in J$, while $y_{i_j} = x_{i_j}$ if $j \notin J$.

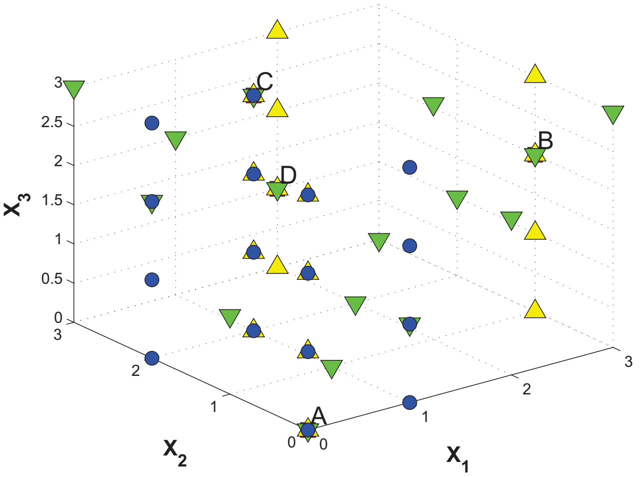

Figure 2.12: Example of decoder failure of the second type. Yellow upward-pointing triangles and green downward-pointing triangles represent the 3-tuples that satisfy the first and the second constraint in the code associated with the PC matrix in (2.18) respectively. Blue circles represent the points of the feasibility region associated with the channel output $Y = [\Theta_{1_1}, \Theta_{1_2}, \Theta] = [(\theta, 0), (0, \theta), (\theta, \theta)]$.

Therefore, given the 4-ary LDPC code associated with the PC matrix of (2.18), $y_i \in \{0, \ldots, 3, \Theta, \Theta_{1_1}, \Theta_{1_2}\}$, $i = 1, \ldots, 3$. Further, for the given i-th symbol, the binary representations of Θ, Θ_{1_1} and Θ_{1_2} are (θ, θ), (θ, x_{i_2}) and (x_{i_1}, θ) respectively.

Figures 2.11, 2.12 and 2.13 show the feasibility region of the code associated with (2.18) for three different channel outputs. Each point of a given feasibility region is plotted as a blue circle. The aforesaid figures also provide a graphical representation of the 3-tuples that satisfy the first constraint (yellow upward-pointing triangles) and the second constraint (green downward-pointing triangles) as well. Each 3-tuple that satisfies both the constraints is a codeword.

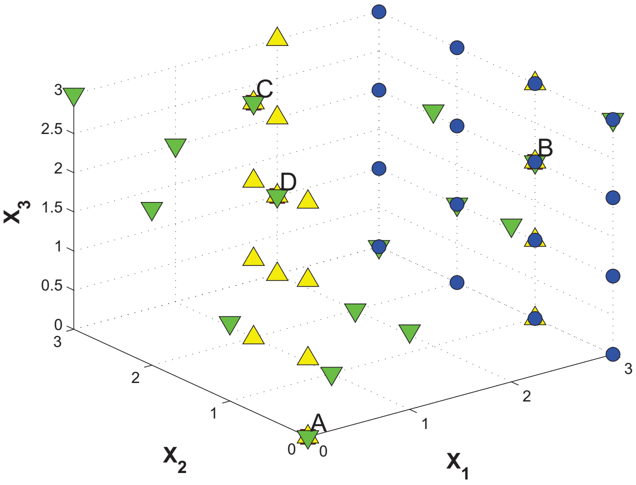

Figure 2.13: Example of decoder success. Yellow upward-pointing triangles and green downward-pointing triangles represent the 3-tuples that satisfy the first and the second constraint in the code associated with the PC matrix in (2.18) respectively. Blue circles represent the points of the feasibility region associated with the channel output $Y = [3, \Theta, \Theta] = [(1, 1), (\theta, \theta), (\theta, \theta)]$.

Example 1: The channel output associated with the feasibility region in Figure 2.11 is $Y = [\Theta, \Theta, 3] = [(\theta, \theta), (\theta, \theta), (1, 1)]$. Therefore, the feasibility region is a plan that cuts the 3-dimensional space for a value of the third coordinate set to 3. Although there is only a codeword (C) within the feasibility region, the decoder can not converge to it since it can not recover any information about x_1 and x_2 from any constraint of the code. In fact, the feasibility region spans through all the first dimension-axis and the second dimension-axis as well. Following the notation of subsection 2.3.2, $\Gamma_1(F, k) = \Gamma_2(F, k) = q$, $\Gamma_3(F, k) = 1 \, \forall k = 1, 2$. Thus, since $\Gamma^+(F, k) = 2 \, \forall k = 1, 2$, Figure 2.11 is an example of a decoder failure of the first type.

Example 2: The feasibility region in Figure 2.12 is related to the channel output $Y = [\Theta_{1_1}, \Theta_{1_2}, \Theta]$, where the codeword that is supposed to be transmitted is $C = [1, 2, 3] = [(1, 0), (0, 1), (1, 1)]$. Therefore, the binary representation of the channel output is $[(\theta, 0), (0, \theta), (\theta, \theta)]$. Such a configuration brings to a decoder failure of the second type. In fact, two codewords (A and C) fall within the feasibility region, i.e., $X \cap F = \{A, C\}$. Thus, the decoder can not choose whether A or C has been transmitted and a decoding failure of subsection 2.3.2 occurs.

Example 3: Finally, let us assume that the codeword $B = [3, 1, 2] = [(1, 1), (1, 0), (0, 1)]$ has been transmitted. Moreover, let us assume that x_2 and x_3 got fully erased. Therefore, the associated channel output Y is $[3, \Theta, \Theta] = [(1, 1), (\theta, \theta), (\theta, \theta)]$. Figure 2.13 shows the feasibility region associated with the aforesaid channel output. This configuration can not bring to a decoder failure of the second type, since $X \cap F = \{B\}$, so $|X \cap F| = 1$. Moreover, a decoder failure of the first type can not occur as well. The projections of the intersections of the feasibility region F and the hypersolid associated with the k-th constraint of the PC matrix in (2.18) are exposed in Table 2.2. Thus, following the notation of subsection 2.3.2, $\Gamma^+(F, 1) = \{3\}$ and $\Gamma^+(F, 2) = \{2, 3\}$, hence, $|\Gamma^+(F, k)| = 2$ if and only if $k = 2$. Specifically, given the knowledge of $x_1 = 3$, the first constraint provides the correct information about $x_2 = 1$. On the next iteration, the second constraint of (2.18) provides information about $x_3 = 2$. Therefore, Figure 2.13 is an example of a decoder success for the 4-ary code related to the PC matrix in (2.18).

Table 2.2: Projections of $F \cap C_k$ over the 3 dimensions of the 4-ary LDPC code associated with the PC matrix in (2.18), when the channel output is $Y = [3, \Theta, \Theta]$

$\pi_i(F \cap C_k)$	$i = 1$	$i = 2$	$i = 3$
$k = 1$	$\{3\}$	$\{1\}$	$\{0, \dots, 3\}$
$k = 2$	$\{3\}$	$\{0, \dots, 3\}$	$\{0, \dots, 3\}$

2.3.3 Graph analysis

As previuosly introduced, decoder failures of type I and type II are related to specific characteristics of the q-ary parity-check matrix H, of the binary mother matrix $\chi(H)$ and of the binary image $\psi(H)$ of the q-ary PC matrix. In fact, it is possible to define some characteristic structures over the graphs associated with the aforesaid matrices that can be related to error events in the low error probability region. Further, construction algorithms can be implemented taking into account those relationships in order to optimize the q-ary LDPC code design.

Specifically, it is worthwhile to find a correspondence between a decoder failure and a given graphical topology, in order to implement algorithms that help in improving the performance of q-ary LDPC codes in the low error-rate region.

This analysis has been already accomplished for binary LDPC codes [44], [45]. However, there is no paper in literature dealing with this task for LDPC codes over $GF(q)$. In order to highlight the differences and the correspondences with their binary counterparts, some of the notations introduced in [44] have been extended to the q-ary case.

Definition 1: A set of d q-ary variable nodes and d q-ary constraint nodes forms a **q-ary cycle** if they are connected by edges induced by a *binary mother* matrix such that a path exists that travels through every node in the set and connects each node to itself without traversing a node twice.

Definition 2: A set of q-ary variable nodes is a **q-ary cycle set** ξ_d if it has d elements and if one or more q-ary cycles are formed between this set and its neighboring constraint set in a bipartite graph induced by the *binary mother* matrix $\chi(H)$.

Definition 3: A set of q-ary variable nodes is a **q-ary stopping set** S_d if it has d elements and all its neighboring constraint nodes are connected to it at least twice in the bipartite graph induced by the *binary mother* matrix $\chi(H)$.

Definition 4: A set of q-ary variable nodes is a **q-ary linearly dependent set** $\Lambda_{d,l}$ if it has d elements and l columns associated with those nodes in the *binary image* $\psi(H)$ of the q-ary PC matrix H form a *binary* linearly dependent set [44].

The relationships among q-ary cycle sets, q-ary stopping sets and q-ary linearly dependent sets provide useful informations about the graph structure of a given q-ary LDPC code. The following lemmas and theorems explain the nature of the relationships that intercur between ξ_d, S_d and $\Lambda_{d,l}$.

Lemma 1: In a bipartite graph induced by a *binary mother* matrix $\chi(H)$ without singly connected variable nodes, every q-ary stopping set contains q-ary cycles ($\{\xi_d\} \supset \{S_d\}$).

Proof: Since q-ary cycle sets and q-ary stopping sets are basically defined over the bipartite graph induced by a *binary mother* matrix, Lemma 1 can be proved as Lemma 1 in [44]. In fact, traversing the bipartite graph induced by a *binary mother* matrix leaving a node on a different edge than that used to enter that node indefinetely, it is possible to visit a node twice, that is, to form a q-ary cycle. □

Lemma 2: In a q-ary linearly dependent set $\Lambda_{d,l}$, l is at least equal to d, i.e., $l \geq d$.

Proof: The binary image of each non-zero entry in the q-ary PC matrix is a p x p submatrix ($p = \log_2 q$) that has no linearly dependent columns. Thus, a set of d q-ary variable nodes takes at least d columns of its binary image to form a binary linearly dependent set. □

Theorem 1: A q-ary stopping set is a q-ary linearly dependent set if and only if the rank of the submatrix formed by the associated columns in the q-ary PC matrix is not full.

Proof: A set of d q-ary variable nodes forms a q-ary linearly dependent set $\Lambda_{d,l}$ if the binary sum of l columns of the *binary image* $\psi(H)$ associated with those nodes is the all-zero vector. The d q-ary variable nodes forms a q-ary stopping set if any neighbor (q-ary constraint nodes) is shared by each variable node in the set at least twice. Therefore, the submatrix formed by the columns in $\psi(H)$ associated with those q-ary variable nodes must have at least a couple of linearly dependent columns. Thus, the rank of the aforesaid submatrix is not full, as the rank of the submatrix formed by the columns associated with those q-ary variable nodes in the q-ary PC matrix as well. □

Corollary to Theorem 1: A q-ary linearly dependent set that contains q-ary cycles has to be a q-ary stopping set ($\{\Lambda_{d,l} \cap \xi_d\} = \{\Lambda_{d,l} \cap S_d\}$).

Proof: The binary sum of l columns of the binary image associated with the nodes in $\Lambda_{d,l}$ is the all-zero vector. Thus, each neighbor (q-ary constraint node) has to properly insist on at least two q-ary variable nodes in the set. If $\Lambda_{d,l}$ contains q-ary cycles, then any neighbor is shared by each variable node in the set at least twice. □

Theorem 2: A q-ary linearly dependent set does not contain any q-ary cycle if and only if the number of the q-ary variable nodes is less than the number of their neighbors.

Proof: Each q-ary variable node in a q-ary linearly dependent set has to be shared by every neighbor (q-ary constraint node) at least twice. Thus, the minimal cycle-free structure that fulfills this condition may be represented as follows:

$$\begin{bmatrix} \zeta_0 & \zeta_1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & \zeta_2 & \zeta_3 & 0 & 0 & \dots & 0 & 0 \\ \vdots & & & \ddots & \ddots & & & \vdots \\ \vdots & \dots & 0 & \zeta_{2i} & \zeta_{2i+1} & 0 & \dots & \vdots \\ \vdots & & & \ddots & \ddots & & & \vdots \\ 0 & 0 & \dots & \dots & 0 & \zeta_{2(m-1)} & \zeta_{2(m-1)+1} & 0 \\ 0 & 0 & \dots & \dots & 0 & 0 & \zeta_{2m} & \zeta_{2m+1} \end{bmatrix} \tag{2.21}$$

Each ζ_i is an element of H living in $GF(q)$. Therefore, if m is the number of constraint nodes that are involved in the minimal cycle-free structure, the number of q-ary variable nodes has to be at least $m + 1$. □

Theorem 3: If d columns of the q-ary PC matrix are linearly dependent over $GF(q)$, the related q-ary variable nodes belong to a $\Lambda_{d,l}$ set. Specifically, $l = d$ and the columns associated with those q-ary variable nodes in the binary

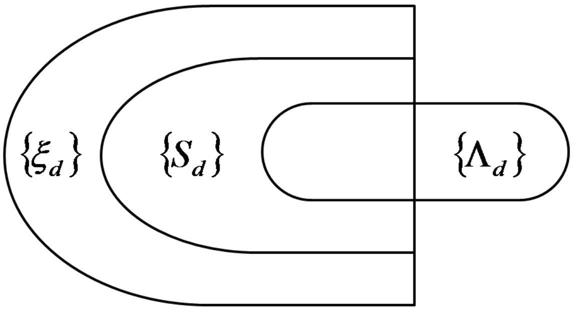

Figure 2.14: Venn diagram of the relationships among q-ary cycle sets ξ_d, q-ary stopping sets S_d and q-ary linearly dependent sets $\Lambda_{d,l}$.

image of the q-ary PC matrix form p binary linearly dependent sets, where $p = \log_2 q$.

Proof: Let D be the set of the indexes of the d columns of the q-ary PC matrix that are linearly dependent over $GF(q)$. The sum over $GF(q)$ of the M x N q-ary PC matrix columns associated with the d q-ary variable nodes that are taken into account is the all-zero vector. That is, $\sum_{j \in D} H_{ij} = 0$ *in* $GF(q)$ $\forall i = 1, \ldots, M$. It is worth to remind that additions of elements in $GF(q)$ correspond to additions modulo 2 of their binary images. The columns of the binary image of the q-ary PC matrix columns associated with the d q-ary variable nodes have to be binary linearly dependent as well. Therefore, the aforesaid d q-ary variable nodes form a q-ary linearly dependent set $\Lambda_{d,l}$. Further, the sum of the k-th column of the binary images of the d columns of the q-ary PC matrix associated with the q-ary variable nodes in $\Lambda_{d,l}$ has to be an all-zero vector. Thus, $\bigoplus_{j:V_j \in \Lambda_{d,l}} \psi(H_{ij})_{tk} = 0 \; \forall k = 1, \ldots, p, \; \forall (i,t) \in \{1, \ldots, M\}$ x $\{1, \ldots, p\}$. That is, the binary sum is performed over d columns of $\psi(H)$ $\forall k = 1, \ldots, p$. Therefore, $l = d$ and the submatrix $\{\psi(H_{ij})\}_{i=1,\ldots,M; j:V_j \in \Lambda_{d,d}}$ contains p binary linearly dependent sets. □

Figure 2.14 shows the relationships among q-ary cycle sets, q-ary stopping sets and q-ary linearly dependent sets.

Given the aforesaid definitions and relationships, it is possible to define proper correspondences between decoder failures and topological structures induced by the code graph. The next section introduces those correspondences and provides algorithms that aim to counteract the effect of such graphical structures on the error-rate performance of the q-ary LDPC codes.

2.3.4 Low error floor design

This section introduces two algorithms that aim to mitigate the effects of the topological structures exposed in the previous section. Specifically, the combination of those algorithms guarantee the q-ary LDPC code to have as large q-ary stopping sets and q-ary linearly dependent sets as possible for a given degree distribution.

The algorithms are based on the correspondences that may be found between topological structures and decoder failures. Once those have been defined, the design methods can be exploited and properly implemented as well. The next subsections are dedicated to expose the aforesaid correspondences first and the proper design algorithms then.

Correspondences

Among the topological structures introduced in Section 2.3.3, q-ary stopping sets and q-ary linearly dependent sets bring the main contribution to the error-rate performance.

Specifically, it is possible to relate q-ary stopping sets and q-ary linearly dependent sets to decoder failures of type I and decoder failures of type II respectively.

Let us assume it is not possible to get any information about the q-ary variable nodes in a q-ary stopping set S_d from the channel output. Therefore, the likelihoods of each symbol in the set of being equal to every $a \in GF(q)$ are set to the same value. Following the notation in subsection 2.3.2, this corresponds to the case for which the feasibility hypersolid spans over the dimensions defined by the variable nodes in the q-ary stopping set. Therefore, the indexes of the variable nodes involved in the q-ary stopping sets correspond to the elements of the sets $I^{+}(F)$. Analogously, the indexes of the constraints infering over the values of the symbols in the q-ary stopping set are the elements of $E^{+}(F)$.

Each element in a q-ary stopping set has to be shared by any constraint node linked to them at least twice. That is, the cardinalities of the sets $\Gamma^{+}(F, k)$ where $k \in E^{+}(F)$ are at least equal to 2. Thus, any constraint connected to the symbols in S_d can not help in recovering the values of the q-ary variable nodes. Therefore, it is not possible for the decoder to converge to a codeword, hence, the decoder fails.

It is worth to note that the correspondence between decoder failure of type I and q-ary stopping set is strongly related to the concept of full erasure. In fact, if there is at least a symbol in the q-ary stopping set that has not been fully erased by the channel, the q-ary decoder can successfully converge to the transmitted codeword as its maximum number of iterations is large enough. Further, since the decoding is operated over $GF(q)$ symbols, a binary stopping set [44] in the binary image $\psi(H)$ of a q-ary PC matrix H does not bring to a decoder failure.

For instance, let us consider the 4-ary code of subsection 2.3.2. Let V be the set of the 4-ary variable nodes, so that $V = \{V_1, V_2, V_3\}$. Let (v_{i_1}, v_{i_2}) be the binary representation of the i-th 4-ary variable node, V_i. Thus, given the Definition 3 in the previous Section, $\{V_1, V_2\}$ is a 4-ary stopping set for the code associated with the q-ary PC matrix in (2.18). Further, in case those q-ary variable nodes are fully erased by the channel, decoder failure occurs. Example 1 in subsection 2.3.2 shows this situation.

Let us look to the binary image $\psi(H)$ in (2.20). It is possible to note that the set $\{v_{1_1}, v_{1_2}, v_{2_1}\}$ (that is, the first three columns in $\psi(H)$) would represent a binary stopping set [44] if a binary decoder would be used over $\psi(H)$. However, it is possible to prove that even if those variable nodes in the binary representation got binary erased, the q-ary decoder can always successfully recover the transmitted codeword.

Partial erasures over the q-ary variable nodes in a q-ary stopping set can cause a decoder failure. This happens when those binary erasures occur over the binary representation of those variable nodes forming a binary linearly dependent set in the binary image of the q-ary PC matrix. That is, partial erasures over the q-ary variable nodes in a q-ary stopping set may bring to a decoder failure if that set is also a q-ary linearly dependent set.

For instance, let us consider another 4-ary code, whose related q-ary PC matrix H is as follows:

$$H = \begin{bmatrix} 2 & 2 & 0 \\ 3 & 3 & 3 \end{bmatrix} \tag{2.22}$$

Thus, the related binary mother matrix $\chi(H)$ is as follows:

$$\chi(H) = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.23}$$

The binary image $\psi(H)$ of (2.22) is as follows:

$$\psi(H) = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{2.24}$$

$\{V_1, V_2\}$ is a 4-ary stopping set S_2 and a 4-ary linearly dependent set $\Lambda_{2,2}$ as well. In case, if only v_{1_1} and v_{2_2} are binary erased, the decoder recovers the transmitted codeword. On the other hand, if only v_{1_1} and v_{2_1} are binary erased, the decoder fails.

Partial erasures are therefore fundamental when q-ary linearly dependent sets are taken into account. This is because q-ary linearly dependent sets are directly related to the concept of minimum distance of the code. In fact, the distance among the codewords increases as the number of linearly independent columns in the binary image of the q-ary PC matrix increases. Moreover, a code with minimum distance l_{min} has at least one $\Lambda_{d_{min},l_{min}}$ set but no $\Lambda_{d_{min},l}$ sets where $l < l_{min}$.

q-ary linearly dependent sets influence the shape of the feasibility region introduced in Section 2.3.2. In fact, $\Lambda_{d,l}$ sets make the feasibility region involve at least two possible codewords, i.e., following the notation of subsection 2.3.2, the cardinality of $\{X \cap F\}$ is at least 2. Therefore, q-ary linearly dependent sets are strictly related to decoder failure of type II. Example 2 in subsection 2.3.2 shows a $\Lambda_{3,4}$ set that causes a decoder failure of the second type, since the columns associated with $\{v_{1_1}, v_{2_2}, v_{3_1}, v_{3_2}\}$ in (2.20) form a binary linearly dependent set.

Finally, it is worth to remind that the role of q-ary stopping sets and q-ary linearly dependent sets can be translated to binary and q-ary non-erasure scenarios. In those cases, variables with poor observation reliability are analogous to erasures. Thus, increasing the minimum q-ary stopping set size and the minimum q-ary linearly dependent set size should represent an effective method for generating q-ary LDPC codes suited for Message-Passing decoding.

Construction algorithms

In order to counteract the effect of decoder failures of first and second type, proper q-ary LDPC code design algorithms can be implemented. As previously introduced in this Section, decoder failures of type I and type II are related to characteristic topological structures. Therefore, limiting the occurrence of such topological structures, i.e., increasing the size of the minimum q-ary stopping set and q-ary linearly dependent set, should also limit the occurrence of the aforesaid decoder failures.

Specifically, q-ary stopping sets are strictly related to the configuration of the binary mother matrix associated with the q-ary LDPC code. A proper construction of the binary mother matrix should guarantee the q-ary LDPC code to be able to limit the effect of q-ary stopping sets on the error-rate performance of the code. Thus, ACE algorithm [44] represents a very efficient method to increase the minimum q-ary stopping set size.

Once the positions of the non-zero entries in the q-ary PC matrix are defined, the value of each $H_{ij} \in H$ has to be chosen. In order to limit the decoder failures of type II, an algorithm that aims at providing large size q-ary linearly dependent sets can be implemented. q-ary linearly dependent set maximization (LDSM) algorithm performs the choice of the non-zero entries in each column of the q-ary PC matrix such that the number of binary linearly dependent columns in the binary image is maximized. Specifically, LDSM algorithm is performed sequentially. In fact, the values of the non-zero entries of the columns that insist on most of the q-ary check nodes of the last column of the q-ary PC matrix that has been taken into account are chosen.

Let N be the number of the q-ary variable nodes and M be the number of the q-ary check nodes of the q-ary LDPC code. Let H, $\chi(H)$ and $\psi(H)$ be the q-ary PC matrix, the binary mother matrix and the binary image of H respectively. Let $B = \{B_i\}_{i=1,\dots,|B|}$, $R = \{R_i\}_{i=1,\dots,|R|}$, $T = \{T_i\}_{i=1,\dots,|T|}$ and $U = \{U_i\}_{i=1,\dots,|U|}$ be sets of column indexes, i.e., $B_i, R_i, T_i, U_i \in \{1, \dots, N\}$. Further, let $c_1(z)$ be the number of ones in the binary sequence z. Finally, let $G(t)$ be the set of the row indexes of the non-zero entries of the t-th column of the q-ary PC matrix, i.e., $G(t) = \{i \in \{1, \dots, M\} : \chi(H_{it}) = 1\}$. The LDSM algorithm is as follows:

```
initialize:
  $B = \varnothing$;
  $R = \varnothing$;
  $U = \varnothing$;
  $T = \{1\}$;
end initialize
```

```
while $B \subset \{1, \ldots, N\}$ do
begin
 for $k = 1$ to $|T|$ do
 begin
        $R = findcol(B, T_k)$;
        for $n = 1$ to $|R|$ do
        begin
             $B \leftarrow B \cup R_n$;
             $\{H_{iR_n}\}_{i \in G(R_n)} = choose(B, q)$;
             if $B = \{1, \ldots, N\}$
                 exit;
             end if
        end for
        $U \leftarrow U \cup R$;
 end for
$T \leftarrow U$;
end while

function: $R = findcol(B, T_k)$
 $R_i \in \{1, \ldots, N\} \setminus B$ s.t. $c_1(\{\chi(H_{uT_k}) \oplus \chi(H_{uR_i})\}_{u=1,\ldots,M})$ is minimized.
end function

function: $\{H_{iR_n}\}_{i \in G(R_n)} = choose(B, q)$
```

```
    $\{H_{iR_n}\}_{i\in G(R_n)} \in GF(q)^{|G(R_n)|}$ s.t. the number of linearly dependent columns
        in $\{\psi(H_{ij})\}_{i=1,\dots,M;j\in B}$ is maximized.
end function
```

As previously introduced, preventing small q-ary linearly dependent sets prevents the q-ary LDPC code to have a small minimum distance. Specifically, LDSM algorithm aims to construct q-ary LDPC codes having q-ary linearly dependent sets $\Lambda_{d,l}$ where $d_{min} = \min\{d\}$ and $l_{min} = \min\{l\}$ are as large as possible. The values of d_{min} and l_{min} are strictly related to the binary mother matrix $\chi(H)$. In fact, the topological structure of the q-ary PC matrix depends on the edge connections imposed by $\chi(H)$ too. Therefore, it is worth to remind that an effective maximization of the q-ary linearly dependent set size also requires a proper construction of the binary mother matrix.

2.3.5 Simulation results

This section presents simulation results by implementing transmissions over binary-input channels and q-ary-input channels using q-ary LDPC codes. Each code that has been constructed is characterized by $(N, R, q) = (2500, 1/2, 16)$, where N is the number of symbols for each codeword, R is the code rate and q is the alphabet size. In this section quasi-regular LDPC codes [20], [23] have been considered. Further, each code is characterized by an average column weight $t = 2.6$ in order to have a fair comparison in terms of error-rate performance. The maximum number of iterations of the decoding message passing algorithm [3, 4] has been set to 50.

The primitive polynomial of the considered 16-ary Galois field is $p(x) = 1+x+x^4$. Thus, the primitive element of $GF(16)$ under a matrix representation in (2.17) is as follows:

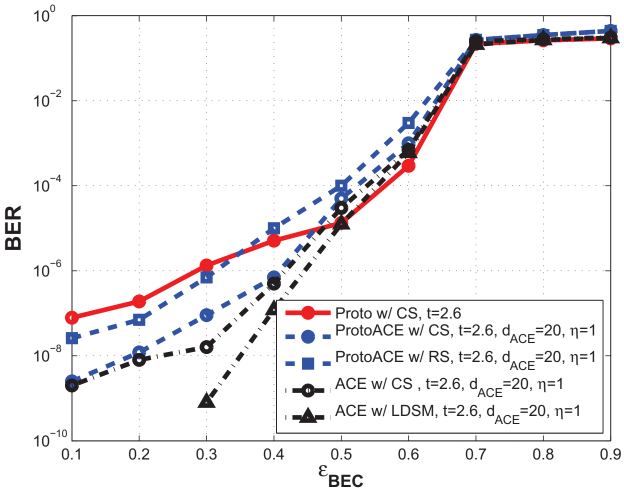

Figure 2.15: Bit Error Rate performance of the considered q-ary LDPC codes on the BEC channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

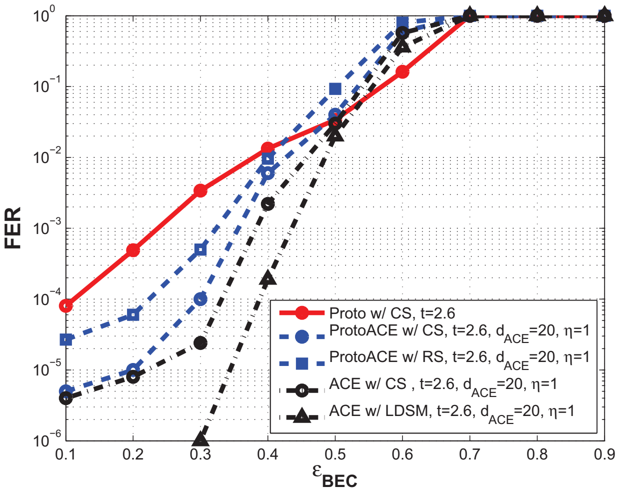

Figure 2.16: Frame Error Rate performance of the considered q-ary LDPC codes on the BEC channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

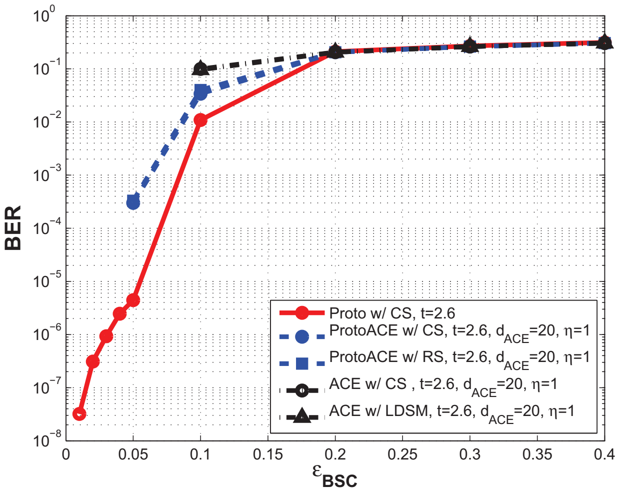

Figure 2.17: Bit Error Rate performance of the considered q-ary LDPC codes on the BSC channel. The binary mother matrix is constructed by using protograph-based algorithm ("proto") [39], ACE algorithm ("ACE") [44] and protograph-based ACE algorithm ("protoACE"). The non-zero entries in the q-ary PC matrix are randomly selected ("RS"), carefully selected ("CS") or selected by means of the LDSM algorithm ("LDSM"). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

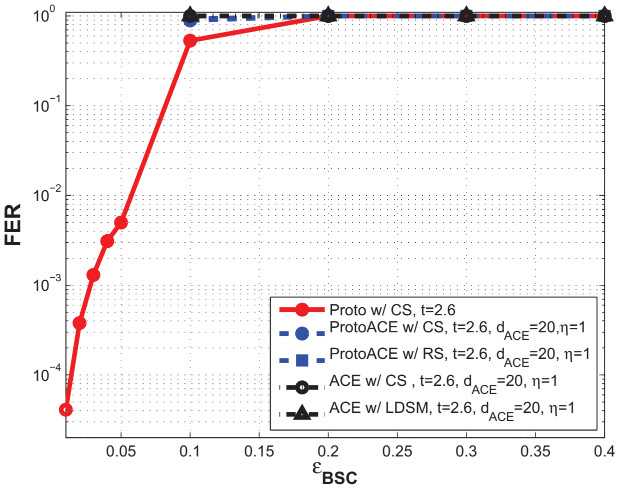

Figure 2.18: Frame Error Rate performance of the considered q-ary LDPC codes on the BSC channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

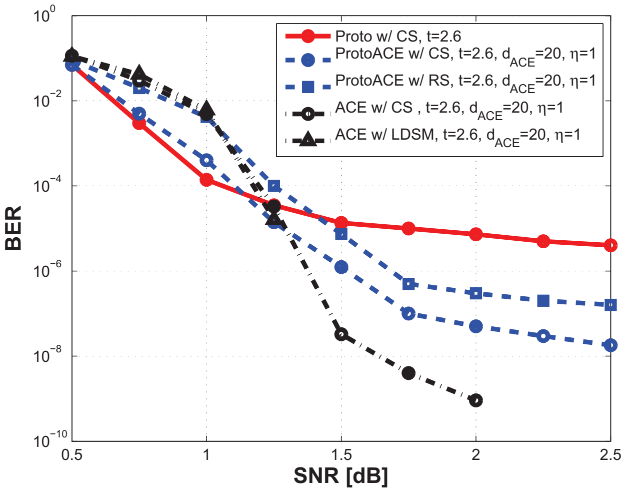

Figure 2.19: Bit Error Rate performance of the considered q-ary LDPC codes on the binary AWGN channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

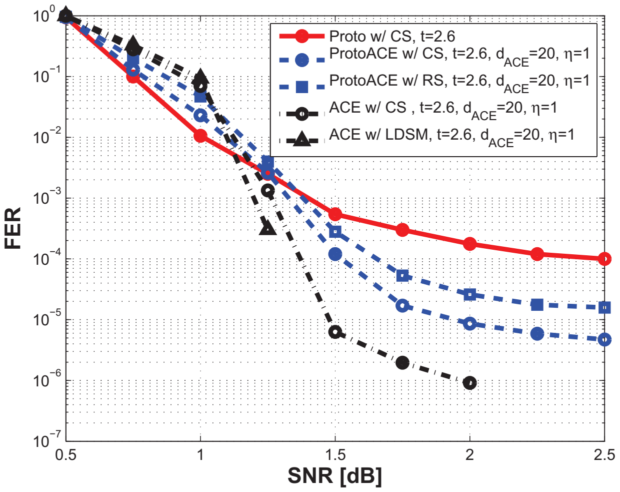

Figure 2.20: Frame Error Rate performance of the considered q-ary LDPC codes on the binary AWGN channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \tag{2.25}$$

The 16-ary LDPC codes have been constructed in a disjoint manner. First, the binary mother matrices are defined by using the following algorithms:

- protograph-based algorithm [39];
- ACE algorithm [44];
- protograph-based ACE algorithm. The connections of the binary mother matrix are defined by using an ACE algorithm [44] that is constrained by the protograph that is considered.

Specifically, for the protograph-based PC matrices and the protograph-based ACE PC matrices the number of transmitted variable nodes of the protograph has been set to 10, while the number of check nodes of the protograph has been set to 5. For the protograph-based codes, the protograph is copied and permuted 250 times to produce the 2500-symbol LDPC codes. On the other hand, the ACE PC matrices and the protograph-based ACE PC matrices have been constructed by setting the d_{ACE} and η parameters [44] to 20 and 1 respectively. The value of η is strictly related to the average column weight that has been taken into account.

The non-zero entries of the q-ary PC matrices are then chosen by using the following algorithms:

- random selection. The non-zero elements are randomly selected from the non-zero elements in $GF(q)$;
- careful selection. The non-zero elements in each row of the q-ary PC matrix are selected by using the method introduced in [39];

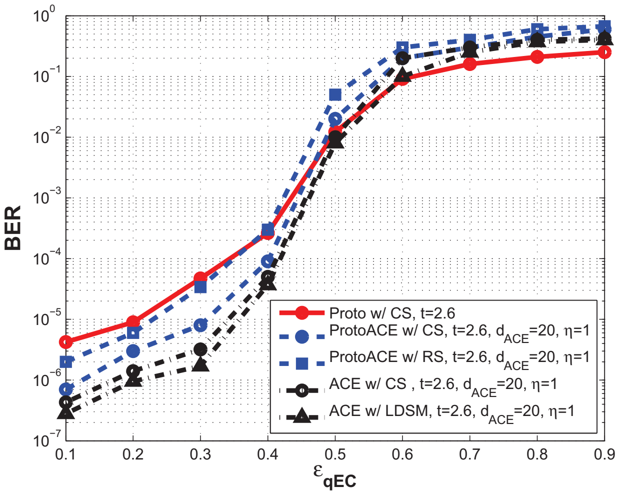

Figure 2.21: Bit Error Rate performance of the considered q-ary LDPC codes on the qEC channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

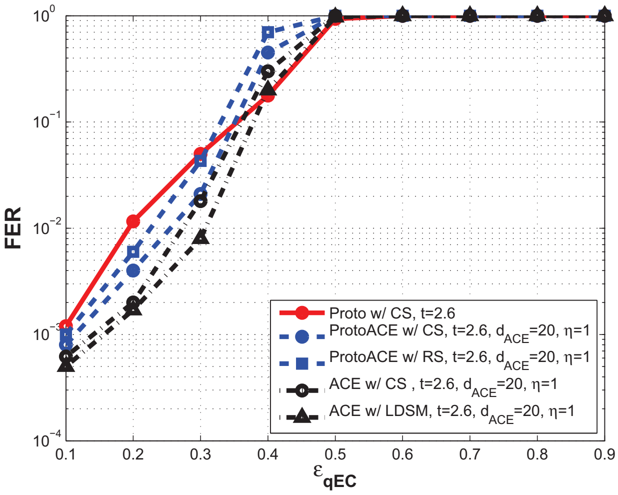

Figure 2.22: Frame Error Rate performance of the considered q-ary LDPC codes on the qEC channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

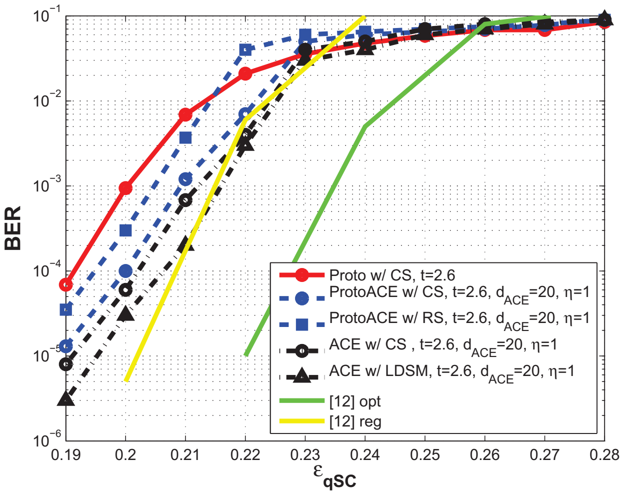

Figure 2.23: Bit Error Rate performance of the considered q-ary LDPC codes on the qSC channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm. BER performance of the regular (”reg”) and optimized (“opt”) LDPC codes proposed in [43] are provided as well.

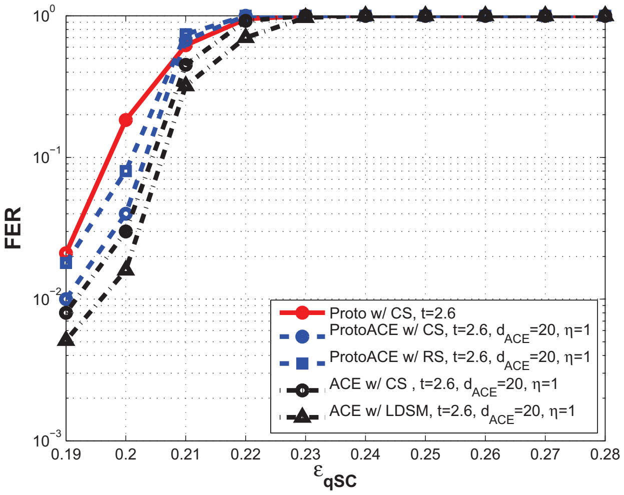

Figure 2.24: Frame Error Rate performance of the considered q-ary LDPC codes on the qSC channel. The binary mother matrix is constructed by using protograph-based algorithm ("proto") [39], ACE algorithm ("ACE") [44] and protograph-based ACE algorithm ("protoACE"). The non-zero entries in the q-ary PC matrix are randomly selected ("RS"), carefully selected ("CS") or selected by means of the LDSM algorithm ("LDSM"). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

- linearly dependent set maximization algorithm of subsection 2.3.4.

The channels that have been considered are binary-input channels and q-ary-input channels. Moreover, the aforesaid channels are discrete-output channels and continuous-output channels. Let X_c be the channel input alphabet and let Y_c be the channel output alphabet. Thus, each binary-input (q-ary-input) channel output in Section 2.3.1 lives in a proper Y_c, i.e., $y^j_{b_k}\,(y_k) \in Y_c$. Specifically, the binary-input channels are characterized by a channel input alphabet $X_c = \{0,1\}$, while the q-ary-input channels are characterized by $X_c = \{0,\dots,q-1\}$. The channels are discrete-output if $|Y_c| < +\infty$. On the other hand, if $Y_c = \mathbf{C}$, the channel output is continuous, where $\mathbf{C}$ is the field of the complex numbers. In case the channel is discrete-output, it is possible to describe the given channel by a matrix $Q = \{Q_{ij}\}_{i=0,\dots,|Y_c|-1; j=0,\dots,|X_c|-1}$, called transition probability matrix. Each term Q_{ij} represents the probability that the given readout symbol is $y_i \in Y_c$, while the stored data is $x_j \in X_c$: that is, $q_{ij} = P(y = y_i|x = x_j)$.

The binary-input channels that have been taken into account in this section are the following:

- *binary erasure channel* (BEC). The related channel output alphabet is $Y_c = \{0, ?, 1\}$. ? represents the uncertainty state. The transition probability matrix Q associated with the BEC can be written as follows:

$$Q = \begin{bmatrix} 1-\epsilon_{BEC} & 0 \\ \epsilon_{BEC} & \epsilon_{BEC} \\ 0 & 1-\epsilon_{BEC} \end{bmatrix} \tag{2.26}$$

  In other terms, since the transmitted data live in $GF(2)$, $P(y = i|x = i) = 1 - \epsilon_{BEC}$, $P(y = i|x = j) = 0$, $P(y = ?|x = i) = \epsilon_{BEC}$ $\forall i, j \in \{0,1\}, i \neq j$.

- *binary symmetric channel* (BSC). The related channel output alphabet is $Y_c = \{0,1\}$. The transition probability matrix Q associated with the

BSC can be written as follows:

$$Q = \begin{bmatrix} 1-\epsilon_{BSC} & \epsilon_{BSC} \\ \epsilon_{BSC} & 1-\epsilon_{BSC} \end{bmatrix} \tag{2.27}$$

In other terms, since the transmitted data live in $GF(2)$, $P(y = i|x = i) = 1 - \epsilon_{BSC}$, $P(y = j|x = i) = \epsilon_{BSC}$ $\forall i, j \in \{0, 1\}, i \neq j$.

- *binary AWGN channel.* Each codeword is binary phase shift keying (BPSK) modulated. The channel output associated with the k-th bit of the binary representation of the j-th symbol may be expressed as $y^j_{b_k} = s^j_{b_k} + n^j_{b_k}$. The noise component $n^j_{b_k}$ has variance σ^2. Thus, $P(s^j_{b_k}|y^j_{b_k}) = P(x^j_{b_k}|y^j_{b_k}) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{(y^j_{b_k} - s^j_{b_k})^2}{2\sigma^2}\right)$.

The q-ary-input channels that have been taken into account in this section are the following:

- *q-ary erasure channel* (qEC). The related channel output alphabet is $Y_c = \{0, 1, \ldots, q-1, ?\}$. ? represents the uncertainty state. The transition probability matrix Q associated with the qEC can be written as follows:

$$Q = \begin{bmatrix} 1-\epsilon_{qEC} & 0 & \ldots & 0 & 0 \\ 0 & 1-\epsilon_{qEC} & \ldots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \ldots & 0 & 1-\epsilon_{qEC} \\ \epsilon_{qEC} & \epsilon_{qEC} & \ldots & \epsilon_{qEC} & \epsilon_{qEC} \end{bmatrix} \tag{2.28}$$

In other terms, since the transmitted data live in $GF(q)$, $P(y = i|x = i) = 1 - \epsilon_{qEC}$, $P(y = i|x = j) = 0$, $P(y = ?|x = i) = \epsilon_{qEC}$ $\forall i, j \in GF(q), i \neq j$.

- *q-ary symmetric channel* (qSC). The related channel output alphabet is $Y_c = \{0, \ldots, q-1\}$. The transition probability matrix Q associated with the qSC can be written as follows:

$$Q = \begin{bmatrix} 1-\epsilon_{qSC} & \dfrac{\epsilon_{qSC}}{q-1} & \dots & \dfrac{\epsilon_{qSC}}{q-1} & \dfrac{\epsilon_{qSC}}{q-1} \\ \dfrac{\epsilon_{qSC}}{q-1} & 1-\epsilon_{qSC} & \dots & \dfrac{\epsilon_{qSC}}{q-1} & \dfrac{\epsilon_{qSC}}{q-1} \\ \vdots & & \ddots & & \vdots \\ \dfrac{\epsilon_{qSC}}{q-1} & \dfrac{\epsilon_{qSC}}{q-1} & \dots & \dfrac{\epsilon_{qSC}}{q-1} & 1-\epsilon_{qSC} \end{bmatrix} \tag{2.29}$$

In other terms, since the transmitted data live in $GF(q)$, $P(y = i|x = i) = 1 - \epsilon_{qSC}$, $P(y = j|x = i) = \dfrac{\epsilon_{qSC}}{q-1}$ $\forall i, j \in GF(q), i \neq j$.

- *q-ary AWGN channel.* Each symbol is symmetric ultracomposite [6] Gray-labeled 16-QAM modulated with a bandwidth efficiency of 2 bits/symbol (i.e. a coding rate R equal to 0.5). The channel output in Section 2.3.1 may be expressed as $y_\kappa = s_\kappa + n_\kappa = (s_{\kappa_I} + js_{\kappa_Q}) + (n_{\kappa_I} + jn_{\kappa_Q}) = y_{\kappa_I} + jy_{\kappa_Q}$, where subscripts I and Q correspond to the in-phase and quadrature components and $\kappa = 1, \dots, N$. The in-phase and quadrature noise components $n_{\kappa_I}, n_{\kappa_Q}$ are independent with the same variance σ^2. Thus, $P(s_\kappa|y_\kappa) = P(x_\kappa|y_\kappa) = \dfrac{1}{2\pi\sigma^2}\exp\left(-\dfrac{(y_{\kappa_I} - s_{\kappa_I})^2 + (y_{\kappa_Q} - s_{\kappa_Q})^2}{2\sigma^2}\right)$.

Figures from 2.15 up to 2.26 show the BER and FER performance of the 16-ary LDPC codes that have been constructed over the aforesaid channels. The correspondences between figures and considered channels are listed in Table 2.3. The red solid line, the blue dashed line and the black dash-dot line represent a LDPC code whose binary mother matrix has been constructed by means of protograph-based algorithm, protograph-based ACE algorithm and ACE algorithm respectively. Moreover, the square marker, the circle marker and triangle marker represent a LDPC code whose non-zero entries binary in the q-ary PC matrix mother matrix are randomly selected, carefully selected and selected by using the LDSM algorithm respectively. The yellow and green solid lines in Figure 2.23 represent the regular and optimized codes exposed in [43] respectively.

The code that has been constructed by means of the protograph-based

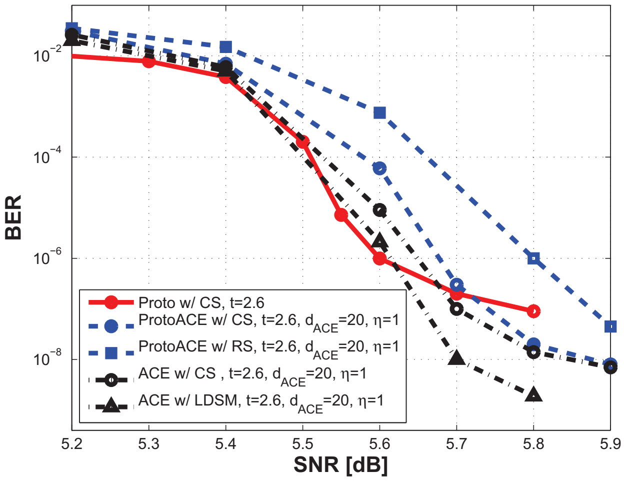

Figure 2.25: Bit Error Rate performance of the considered q-ary LDPC codes on the q-ary AWGN channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

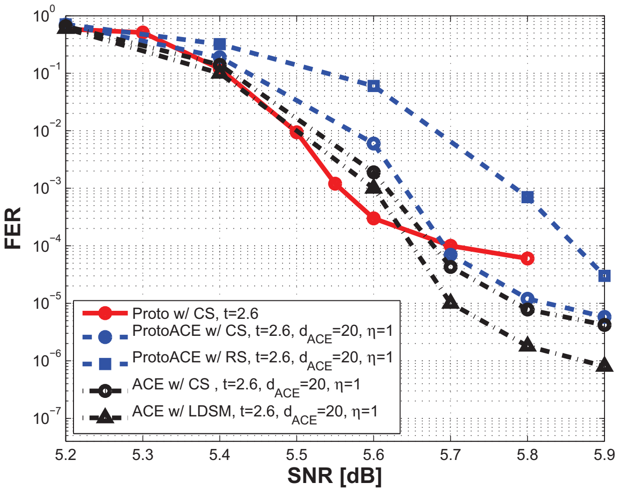

Figure 2.26: Frame Error Rate performance of the considered q-ary LDPC codes on the q-ary AWGN channel. The binary mother matrix is constructed by using protograph-based algorithm (“proto“) [39], ACE algorithm (“ACE“) [44] and protograph-based ACE algorithm (“protoACE“). The non-zero entries in the q-ary PC matrix are randomly selected (”RS”), carefully selected (”CS”) or selected by means of the LDSM algorithm (”LDSM“). t is the average column weight, whereas d_{ACE} and η are the setup parameters for the ACE algorithm.

Table 2.3: Summary of the figure-channel correspondences

Figures	**Channels**
2.15, 2.16	BEC
2.17, 2.18	BSC
2.19, 2.20	BAWGNC
2.21, 2.22	qEC
2.23, 2.24	qSC
2.25, 2.26	qAWGNC

algorithm in [39] show the best performance in the waterfall region. However, it also shows the worst performance in terms of error floor on every channel that has been considered.

The codes whose binary mother matrix has been constructed using the protograph-based ACE algorithm have the closer error-rate performance in the waterfall region to those of the protograph-based q-ary LDPC code. Further, they show lower error floors than the protograph-based q-ary LDPC code. Specifically, the selection of the non-zero entries in the q-ary PC matrix delivered by the algorithm in [39] provides an effective improvement of the performance w.r.t. the random selection algorithm. This benefit is shown in the waterfall region as in the error floor region as well.

On the other hand, the q-ary LDPC codes whose binary mother matrix is constructed by means of the ACE algorithm show the best error floor performance over each considered channel. The error floors of these codes are significantly lower than those related to protograph-based ACE constructed codes. In fact, the protograph constraints induce a sort of "super-structure“ on the bipartite graph. Therefore, the q-ary stopping set mitigation of the protograph-based ACE algorithm might not be as effective as that delivered by the standard ACE algorithm for quasi-regular and irregular codes.

Further, the selection of the non-zero entries performed by using the LDSM algorithm yields orders of magnitude of improvement in the error floor region for every channel that has been taken into account w.r.t. the selection algorithm in [39]. The gain provided by LDSM algorithm is higher in case of BEC, binary AWGN channel and q-ary AWGN channel, while it gets tinier as q-ary-input discrete-output channels (that is, qEC and qSC) are considered. In fact, the effect of q-ary linearly dependent sets gets reduced as partial erasures are avoided. Thus, the benefit provided by LDSM algorithm gets lower as well.

Error-rate performance over binary-input and q-ary-input symmetric channels show how the shape of the feasibility region strongly influences the decoder behavior. In fact, as full and partial erasures are avoided, the error events induce several constraints over the feasibility region so that the decoder can hardly converge to the correct codeword. Thus, the MP algorithm would take a very large maximum number of iterations to try to solve the decoding problem, leading to a very expensive computational cost. On the other hand, customed designed front-end architectures might be designed to counteract the aforesaid effect with an acceptable computational complexity. The observations delivered by [43] might be useful in that sense.

2.3.6 Conclusions

In this section, a deep geometrical and graphical analysis of the q-ary LDPC decoding problem has been addressed. Further, construction methods for designing q-ary LDPC codes with low error floors and moderate field order have been proposed. It has been shown how enlarging the size of the aforesaid topological structures provides substancial improvements in terms of error-rate performance over other construction algorithms. In fact, the relationships between decoding failures and specific topological structures such as q-ary stopping sets and q-ary linearly dependent sets can be efficiently counteracted by the proposed algorithms. Specifically, combining ACE algorithm [44] and LDSM algorithm leads to a benefit by orders of magnitude in terms of error floor over the other construction algorithms that have been taken into account. Finally, simulation results over BEC and binary AWGN channel are very interesting as those channels can model the readout channel of emerging storage technologies [41, 42].

2.4 q-ary LDPC codes robust to error bursts

Recent research findings propose LDPC as a possible alternative to RS coding in systems such as magnetic recording channels for hard-disk drives (HDD) [46], [48]. Near capacity performance and the ability to correct large error bursts make these codes a good alternative to powerful algebraic decoding of non-binary RS codes with large symbols. In this sense, parallel research works go toward the direction of improving RS performance introducing the soft decoding concept of LDPC (see [47] and references therein). However, the RS parity-check matrix is not suitable for the conventional MP algorithm; in fact, RS soft decoding is still an open problem, especially when complexity is considered.

Non-binary, or q-ary, LDPC codes may be used in HDD in order to increase the robustness against error bursts, caused by media defects and thermal asperities [48]. The performance of these codes outperforms both binary LDPC and RS, while the major drawback is the complexity which increases with Ntq^2, where q is the number of symbols defined in the finite field $GF(q)$, N is the coded block length and t is the average weight of PC matrix columns, i.e. the number of non-zero elements per column [3]. In this section decoder complexity is neglected. However, recent works indicate that it can be reduced either using the Fast Fourier Transform [48], [4] or simplifying the decoding rules for high rate codes, considering the dual codes [49].

In [46] the capability to correct an erasure burst, when no errors are expected in the guard band (i.e. all the symbols outside the burst are perfectly known at the receiver), is measured by means of a compact algorithm which provides the "maximum resolvable erasure-burst length", MREBL, defined as "the longest string of erasures that the codeï¿½s decoder is guaranteed to fill in with the correct bit values (or symbol values ...), no matter where the burst starts in". The algorithm inspects the matrix H, looking for the error locations in a burst which can be resolved by checks at each iteration. Here, MREBL will be considered in conjunction with the number of iterations the algorithm requires to find MREBL. The capability to correct erasure bursts has been related to the parity-check matrix by means of the minimum space distance (mSD), that is the minimum distance between non-zero entries of parity-check matrix rows [48]. mSD represents also a lower bound for MREBL. In [48], the authors include the mSD maximization in a PEG construction algorithm for the PC matrix. A similar approach has been reported in [50] for binary LDPC. Anyway, there is no difference between the binary and non-binary case as long as the usual way to build a q-ary PC matrix is to multiply the non-zero entries of a binary mother matrix H by random elements taken in $GF(q)$ [48], [3], [4]. It's obvious that the MREBL and mSD as measured on the mother parity-check are to be multiplied by $log_2(q)$ in order to obtain the effective burst length in bits. A similar burst-error-correction capability estimation is reported in [15], [16] for structured LDPC Gilbert codes which are very similar to the so-called array codes [17]. In reality, channels are often affected by random noise and erasures. In this sense, in all the cited works performance has been evaluated by adding error bursts in an AWGN channel or in magnetic recording channel.

The main results presented in this section are:

1. the optimization of the PC matrix, monitoring in a PEG algorithm the MREBL and the mSD measures introduced in [46], [48];
2. the comparison among different matrix constructions, optimizing the mSD and the MREBL;
3. validation of the most performable scheme by simulations in burst noise AWGN and magnetic recording channels.

Section 2.4.1 is devoted to the description of the modified PEG algorithms for the PC matrix design. The mSD has been inserted in the PEG search in a way similar to [46], while the MREBL has been included progressively in order to reduce the computation time and maximize the probability of finding a good LDPC matrix without cycles. The subsequent section describes the modeled system with particular emphasis on the perpendicular magnetic channel. Sections 2.4.3 and 2.4.4 show simulation results on AWGN and perpendicular magnetic recording (PMR) channels, followed by some conclusion and perspective of future works which conclude this section.

2.4.1 Modified PEG algorithms

An empiric procedure to build a Tanner graph maximizing the girth g_0 is suggested in [16], [18]: the PEG algorithm is based on a pre-determined weight distribution of symbol and check nodes. The graph construction is based on iterative edge-by-edge steps, maximizing the local girth for given nodes. In the present section, girth ≥ 6 is imposed. The obtained matrix may be either regular or irregular. In the seminal work [4], the author suggests a method, based on Montecarlo simulations, to find the best average column weight of a non-binary LDPC code. Following this reasoning, for code-rate $8/9$ $GF(16)$ codes, it has been observed that an average column weight equaling 2.88 is the best choice. This is the result of an empirical optimization based on AWGN channel simulations results obtained with matrices characterized by different average column weights. Future works might be devoted to implementing a more general search algorithm as suggested in [4], [20]. According to this result, the matrices in this section share an average column weight 2.88; the rate is $8/9$ and size is (128,1152) $GF(16)$ symbols. In the following two modified PEG algorithms which include the mSD and the MREBL constraints are described.

PEG with the mSD constraint

Let the Tanner graph consists of n symbol nodes s_j, $0 \leq j \leq n-1$, and m check nodes c_i, $0 \leq i \leq m-1$. Let d_{s_j}, d_{c_i} be the degree of symbol node s_j and of check node c_i, respectively, where node degree means the number of edges incident to it (this value is defined by weights distributions $\lambda(x)$ and $\rho(x)$), E_x is the set of edges incident to node x, while E_x^y represents the y^{th} edge incident to node x. Let $N_{s_j}^l$ be the set of check nodes that can be reached from symbol node s_j by l edges or less and $\overline{N_{s_j}^l}$ the complementary set of $N_{s_j}^l$: in other words, $N_{s_j}^l \cup \overline{N_{s_j}^l} = V_c$, where V_c is the set of all check nodes in the graph. Finally, let B_{s_j} be the set of check nodes $\{c_p\}$ that can be reached from symbol node s_j such that, given two edges incident to c_p, i.e. $E_{c_p}^u \leftarrow (c_p, s_a)$ and $E_{c_p}^v \leftarrow (c_p, s_j)$ $\forall u, v \in \{1, ..., d_{c_p}\}$, $u \neq v$ and $a \in \{0, ..., n-1\}$, $a \neq j$, $|a-j| \geq L$, with fixed L: at this point, it is possible to define $\overline{N_{s_j}^l} \cap B_{s_j} = B_{s_j}^l$ and L means the desired mSD length. The procedure of constructing the Tanner graph is the following:

```
for $j = 0$ to $n-1$ do
begin

 for $k = 0$ to $d_{s_j} - 1$ do

 begin

      if $j = 0$

          $A_{s_j} = V_c$
          $A_{s_j}^l = \overline{N_{s_j}^l}$
          $A_{s_j}^{l+1} = \overline{N_{s_j}^{l+1}}$

      else

          $A_{s_j} = B_{s_j}$
          $A_{s_j}^l = B_{s_j}^l$
          $A_{s_j}^{l+1} = B_{s_j}^{l+1}$
```

```
            end
            if $k = 0$
                $E^0_{s_j} \leftarrow edge(c_i, s_j)$ where $E^0_{s_j}$ is the first edge incident to $s_j$ and $c_i$
                    is one check node picked from the set $A_{s_j}$ such that it has the
                    lowest check degree under the current graph setting $\bigcup_{r=0}^{j-1} E_{s_r}$
            else
                expanding a tree from symbol $s_j$ up to depth $l$ under the cur-
                    rent graph setting such that $A^l_{s_j} \neq \emptyset$ but $A^{l+1}_{s_j} = \emptyset$, or the
                    cardinality of $A^l_{s_j}$ stops increasing but is less than $m$, then
                    $E^k_{s_j} \leftarrow edge(c_i, s_j)$, where $E^k_{s_j}$ is the $k^{th}$ edge incident to $s_j$
                    and $c_i$ is one check node picked from the set $A^l_{s_j}$ having the
                    lowest check-node degree
            end
    end
end
```

PEG with the MREBL constraint

The construction procedure of the Tanner graph is similar to the one described in the paragraph above, except for the meaning of some definitions: B_{s_j} represents the set of check nodes $\{c_q\}$ that can be reached from symbol node s_j such that $c_q \neq \tilde{c}$, $|B_{s_j}| = m - 1$, where $\tilde{c} \in \{c_i, i = 0, ..., m-1\}$ is the checkï¿$\frac{1}{2}$node with the lowest degree under the current graph setting that includes the symbol nodes s_a, $a \in I^L_j$, where L is fixed and I^L_j is defined as the following:

$$I^L_j = \begin{cases} [0, j] & if\ j < L \\ [j - L, j] & otherwise \end{cases} \qquad (2.30)$$

According to these assumptions, it is possible to define $\overline{N_{s_j}^l} \cap B_{s_j} = B_{s_j}^l$ and L as the desired MREBL length.

Parity-check matrices

In order to test different way of optimizing the PC matrix with respect to the burst error correction capabilities, the five LDPC code matrices reported in table 2.4 have been considered. The first matrix represents the result of a mild mSD maximization which results also in a good MREBL value, the second and the third ones have been obtained by respectively optimizing the MREBL and the mSD, while the last two have been constructed with a standard PEG search algorithm without burst error constraints. For all the matrices the mSD, MREBL and the number of maximum iterations requested for correcting a burst of length equal to MREBL have been computed. The $GF(16)$ q-ary codes are obtained by multiplying each non zero element of the mother PC matrix by a random number defined in the non-binary field. The error burst length correction capabilities, reported in table 2.4, are given in bits (i.e. MREBL x $\log_2(q)$). For sake of reader's convenience the MREBL of the RS code adopted for the comparison is given. For each considered PC matrix the variable distributions is $\lambda_2 = 0.112$ and $\lambda_3 = 0.8889$, where $\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1}$ (where d_v represent the maximum value of symbol node degree), in agree with notation introduced in [21] and proposed in [22].

2.4.2 System model description

A non-binary LDPC coded scheme has been tested on AWGN channel and a perpendicular magnetic recording channel. The LDPC decoder runs a maximum of 25 iterations of a standard MP algorithm ([48], [3], [4]) in both cases.

Table 2.4: PC matrices

PC matrix	**mSD**	**MREBL**	**MREBL number of iterations**
mat1	20	100x4	23
mat2	1	104x4	28
mat3	38	90x4	14
mat4	1	85x4	10
mat5	1	94x4	12
RS	-	25x10	-

The AWGN channel

On AWGN channel a simple BPSK modulation has been adopted. In order to evaluate the performance of the different codes in presence of long bursts, an error burst where N_b BPSK symbols are null is injected.

The perpendicular magnetic recording channel (PMRC)

User data in a HDD channels are generally encoded with a run-length-limited code (RLL) followed by a Reed-Solomon code. In this section RLL encoding is neglected. The Reed-Solomon is defined over $GF(2^{10})$; in the simulation results the power correction is set equal to 25 symbols. User data are separated in blocks, namely sectors, 512 bytes long. Here, the non-binary LDPC are supposed to replace the RS code. As demonstrated later, it may be beneficial to consider also an interleaver to separate channel and LDPC code. In this case, the write path is modified according to the Figure 2.27. User bits are first scrambled by the interleaver and then encoded. Parity bits are then evenly distributed along user bit stream. This approach preserves the RLL constraints [51], possibly applied to the bits u_k. Encoded bits are used to generate the write currents which determine, through the inductive write disk head, the magnetization of the platter magnetic media; a different magnetization is applied whenever a bit differs from the previous one.

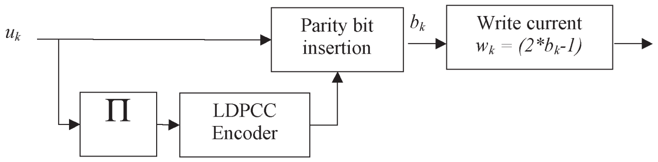

Figure 2.27: Write path schematic with interleaver.

The readback signal $s(t)$ of the PMR channel can be described by the following equation [52]:

$$s(t) = \frac{1}{2} \sum a_k h(t - kT + \tau_k) + n(t) \tag{2.31}$$

where $a_k = (w_k - w_{k-1})$, $w_k \in \{+1, -1\}$ are the recorded data and $n(t)$ is the electronic noise. $h(t)$ is the transition waveform; it corresponds to the current modification induced when the magneto-resistive read head detects a magnetic field change. The transition waveform model is:

$$\alpha \cdot erf\left(\frac{2\sqrt{ln2}}{T \cdot CBD} t\right) + (1 - \alpha) \cdot \tanh\left(\frac{\ln(3 + 2\sqrt{2})}{T \cdot CBD} t\right) \tag{2.32}$$

with $\alpha = 0.7$. The coded bit density CBD is defined as PW50/T, where PW50 is the width of the transition waveform derivative at half the maximum amplitude. CBD will be derived as CBD = UBD/Rate where UBD is the user bit density (the uncoded bits) and Rate takes into account the redundancy of Reed-Solomon or LDPC code. The jitter transition noise (or media noise) is generated by the uncertainty of the transition position τ_k which is a zero-mean Gaussian random variable. The SNR definition is given by:

$$SNR_{dB} = 10 \log_{10}\left(\frac{1}{N_0 + M_0}\right) \tag{2.33}$$

The noise power is split into media noise M_0 and thermal noise N_0 according to the parameter $0 \leq mix \leq 1$, so that:

$$\begin{aligned} N_0 &= mix \cdot 10^{-SNR_{dB}/10} \\ M_0 &= (1 - mix) \cdot 10^{-SNR_{dB}/10} \end{aligned} \tag{2.34}$$

The jitter noise power is given by:

$$\frac{M_0}{2} \cong P_t \cdot \sigma_\tau^2 \int_{-\infty}^{+\infty} \left(\frac{dh(t)}{dt} \right)^2 dt \tag{2.35}$$

where σ_τ is the variance of the random variable τ and $P_t = 0.5$, is the transition probability.

The read path described in Figure 2.28 consists of an anti-alias continuous time filter and an analog-to-digital converter (ADC) followed by a digital adaptive 10 taps finite impulse response (FIR) filter which equalizes the read signal to match a generic partial response target. A maximum likelihood sequence detection (MLSD) is performed with a data-dependent-noise-predictive (DDNP) [53] soft-output Viterbi algorithm (SOVA) [54]. The SOVA algorithm includes the modification proposed in [55] which makes the performance equivalent to the Max-Log-MAP detection algorithm. The SOVA trellis has 8 states. The branch whitening filters have three taps. For this section simulations the partial response target is [4 10 7 -2]. The LDPC decoder is possibly separated by the interleaving block. Error bursts are simulated with an attenuation of N_b consecutive readback signal samples in a random location along the sector; thermal and media noise are added after sample attenuation.

2.4.3 AWGN Simulation Results

Performance has been evaluated by comparing the proposed scheme with a non-binary RS coding scheme with the same redundancy, designed and shortened to work in HDD sector of 512 bytes as already mentioned in section 2.4.2.

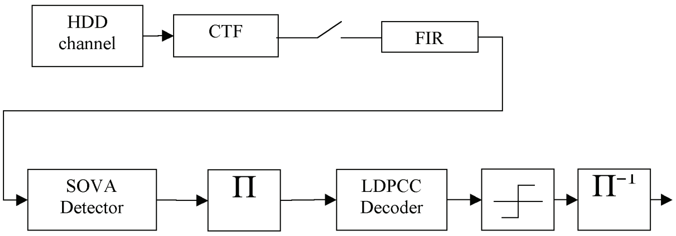

Figure 2.28: Read path schematic.

In Figure 2.29 the performance for the sector-failure rate (SFR) versus the average signal-to-noise ratio per bit (E_b/N_0), is shown in presence of only AWGN noise. According to the main literature results, q-ary LDPC outperform RS of about 2 dB. Furthermore all the PC matrices considered show the same performance according to the fact that they have all been constructed by a PEG search with the same average column weight $t = 2.88$.

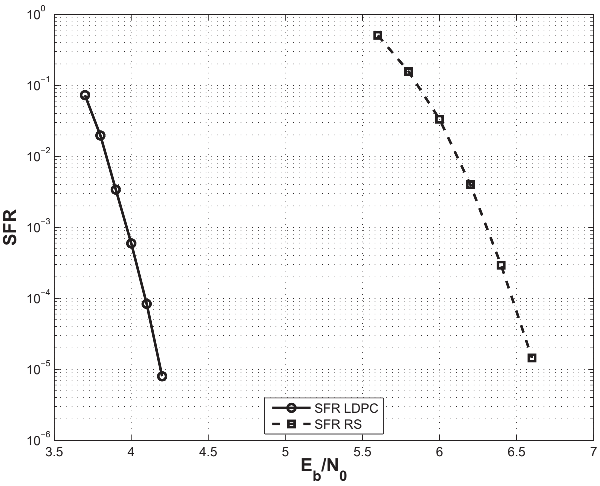

Figure 2.29: Performance results on AWGN channel.

Figure 2.30 and 2.31 show performance results on AWGN channel with burst erasure of length respectively of 100 and 150 bits. It seems that the PC matrix optimized for maximizing the mSD exhibits the best performance. Actually, the MREBL measure assumes an ideal erasure bursts where perfect knowledge of the other bits is given. In real burst-error channel situations, erasure bursts are observed in presence of other noise sources.

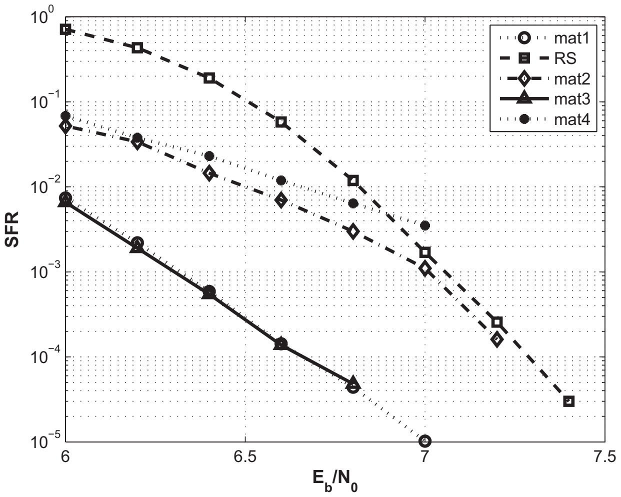

Figure 2.30: Performance results on AWGN channel in presence of bursts of length = 100 bits.

2.4.4 PMRC Simulation Results

Results in Figure 2.32 show that in absence of error burst, codes generated with (mat1) or without (mat5) mSD constraint performs similarly. When an error burst of 150 bits with attenuation 0.5 affects the sector, it is confirmed that the mSD optimization provides with better performance. Additionally, the performance of the designed code with the largest mSD (mat3) is reported. Its burst robustness is further confirmed. Performance can be compared with a RS code with same redundancy. The LDPC gain is about 1dB. However, RS slopes looks steeper compared to LDPC. The present day explanation is given by the absence of an interleaver separating the ISI detector and the LDPC decoder. In fact, ISI detector generates short burst of correlated errors; most of them affect two to six consecutive bits. Correlated error can results in a suboptimal estimation of the GF symbol likelihoods which are the LDPC decoder input. To prove this point, the same LDPC codes have been simulated with a random interleaver as already described in subsection 2.4.2. It is obvious that, in this

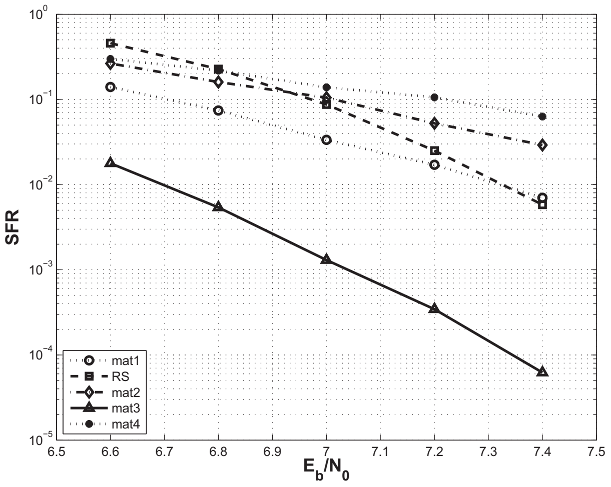

Figure 2.31: Performance results on AWGN channel in presence of bursts of length = 150 bits.

case, the optimization of the code against error burst is lost. Nevertheless, their performance are significantly improved as demonstrated in Figure 2.33. Burst error optimization gain is no longer apparent. As a consequence, further work is necessary in order to optimize the codes separated by an interleaver.

2.4.5 Conclusions

The presented results highlight some interesting design parameters to take into account for the construction of efficient burst error correction LDPC codes. Although the MREBL is a valid performance indicator of a code under ideal noise burst, mSD optimization looks more promising when burst error correction robustness is required in random noise channels like AWGN and PMRC. Performance on AWGN channel demonstrate that non-binary LDPC codes outperform RS codes in presence of long bursts. On the other hand, on PMR channels, non-binary codes suffers from performance degradation due the ISI detector correlated outputs. Two interesting perspectives of development are both the joint optimization of the PC matrix and interleaver and a more so-

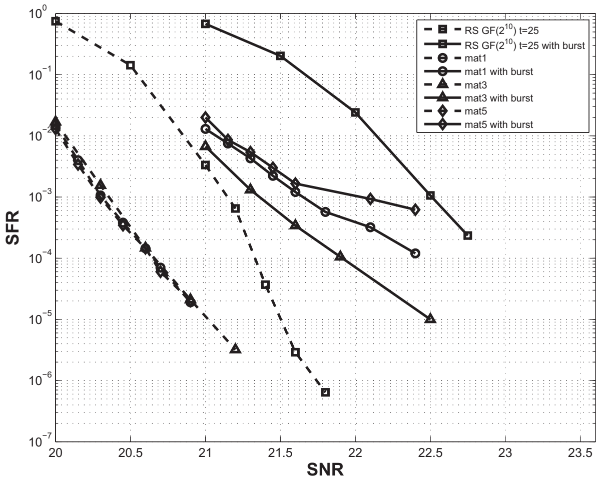

Figure 2.32: Performance results on PMR channel: results without and with a bursts of length = 150 bits are compared with the RS code.

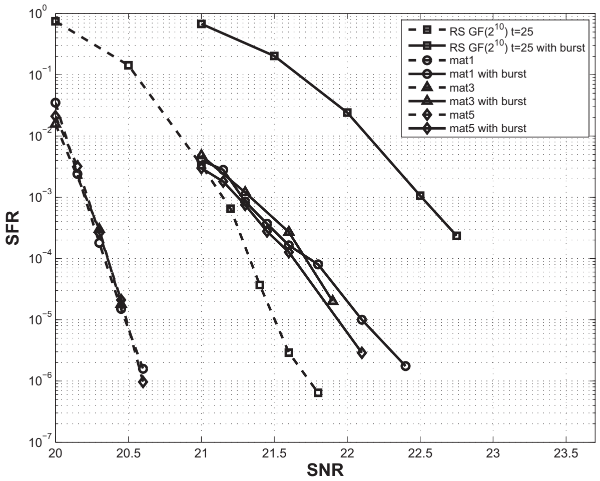

Figure 2.33: Performance results on PMR channel: results without and with a bursts of length = 150 bits are compared with the RS code; an interleaver separates ISI detector and LDPC decoder.

phisticated method for computing the likelihoods of the non-binary symbols in order to match the detector soft outputs to the LDPC decoder.

Chapter 3

Efficient decoding of q-ary LDPC codes

Recently q-ary LDPC codes have been used to achieve performance close to the channel capacity in different channel environments, from satellite communications to magnetic data storage systems. Design of receivers employing these codes for transmissions over channels affected by intersymbol interference is still an open issue. In fact, detection-and-decoding systems have to face the trade-off between error-rate performance and complexity. In the next Section some receiver architectures for 16-ary LDPC codes are compared: serial and turbo concatenated schemes and a joint message-passing based receiver as well. Performance of these systems are evaluated over three different partial response (PR) channels, using simulations. Finally, future directions for research are discussed.

3.1 Efficient receivers for q-ary LDPC coded signals over Partial Response channels

Intersymbol interference affects the most of communication systems in which noise implies frequency shifts in the channel output, like long-haul optical transmissions and magnetic recording systems. Partial response channels represent a discrete model useful in order to approximate the ISI channel. The optimal receiver for partial response channels may be realized by a maximum likelihood sequence detector, using the estimated channel impulse response (CIR) in a Viterbi algorithm [70] [68]. A Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [66] can replace a Viterbi processor if the soft output information is employed in a concatenated scheme, feeding an outer channel decoder [67]. Recently turbo codes [2] and LDPC [1] codes have been used to achieve BER lower than other typical error-correcting codes (like Reed-Solomon) in applications where ISI has to be very efficiently counteracted.

Recent works reveal that q-ary LDPC may be applied to magnetic recording channels, allowing reduced complexity schemes [48] and robustness to burst errors [38]. Historically, considering a LDPC coded system, the most popular scheme to improve error-correction performance over channels affected by ISI has been to serially concatenate a soft-output detection algorithm and the binary LDPC decoder. However, a greater performance improvement can be achieved incorporating the channel detector in the iterative decoder: this implies a turbo concatenation of the two system blocks and several papers in literature call that configuration “turbo equalization”. Further, in [59] [63] [65] an LDPC coded detection-and-decoding system implemented by a joint algorithm based on the MP algorithm is addressed.

In this section some different q-ary LDPC coded architectures for partial response channels are compared, paying particular attention to the properties of the detector and the decoder selected for each one. Specifically, architectures that employ a serial and a turbo concatenation of detector and decoder are analyzed. Moreover, the joint detection-and-decoding scheme is extended to q-ary LDPC codes.

The section is organized as follows. Section 3.1.1 introduces the system model for the different architectures that are discussed in Section 3.1.2: the serially concatenated architecture, the turbo concatenated architecture and the joint MP based architecture. Moreover, Section 3.1.2 discusses the computational complexity of each receiver architecture. In Section 3.1.3 the simulation results are given, as the practical aspects of the system implementing as well. Conclusions about future research development conclude the section.

3.1.1 System Model

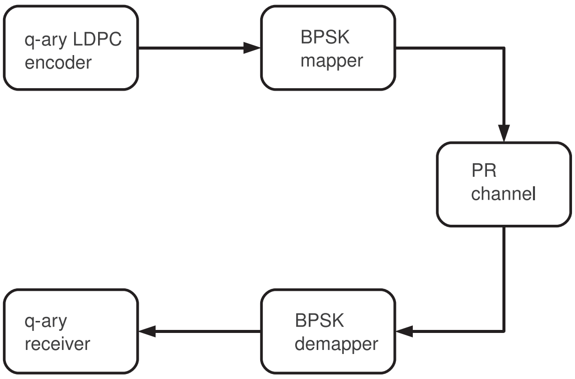

Figure 3.1: Block diagram of the basic system model for q-ary LDPC coded signals over binary-input PR channels

In this section, the performance of three receivers for q-ary LDPC coded signals over PR channels are analyzed. These systems are the serially concatenated architecture, the turbo concatenated architecture and the joint message passing based architecture.

For each of these architectures, the basic system model is shown in Figure

3.1. The q-ary LDPC encoder output is a length-N codeword $\underline{\xi} = [\xi_i]_{i=1,\ldots,N}$, $\xi_i \in GF(q = 2^p)$ $\forall i = 1, \ldots, N$ such that

$$H\underline{\xi} = \underline{0} \tag{3.1}$$

where $H = \{H_{ij}\}_{i=1,\ldots,M,j=1,\ldots,N}$,$H_{ij} \in GF(q = 2^p)$ is the Parity Check matrix. The codeword $\underline{\xi}$ represents the input to the BPSK modulator. After the binary-input length-ν memory PR channel, additive white Gaussian noise is added. The received signal, properly demapped, is then sent to the detection-and-decoding system.

Depending on the receiver architecture, the signal is detected and decoded in an appropriate manner. Each receiver architecture that has been taken into account employes a q-ary LDPC decoder. Moreover, the detector utilizes a BCJR algorithm that can be bit-based or symbol-based as well.

The next subsections introduce the two detectors for the architectures considered in this section: the Bit-Based and the Symbol-Based detectors respectively.

Bit-based detector

The Bit-based (BB) detector is represented by a standard BCJR algorithm [66]. The symbol-wise *a posteriori* probabilities (APPs) are approximated applying the BCJR algorithm to the PR channel and then multiplying the APPs that form a symbol [58], [56]. The symbol-wise APPs are passed to the q-ary LDPC decoder to initialize the *a priori* probabilities.

Symbol-based detector

The symbol-based (SB) detector [58], [69] modifies the way the probability functions are updated when compared to the original BCJR algorithm. In [69], the authors develop a method, called optimal subblock-by-subblock detector (OBBD), in order to calculate the APP of a block of p consecutive bits. In [58], the authors show that simplifications of the algorithm can be made for the case of the binary-input ISI channels, specifically for $p \geq \nu$.

The SB detector is as follows.

Let $a \in GF(q = 2^p)$ be an information symbol. It is possible to map each symbol in $GF(q = 2^p)$ to a distinct sequence of p bits; in other terms, $a = (b_{p-1}, b_{p-2}, \ldots, b_1, b_0) = b_{p-1}^{0}$, where $b_i \in GF(2)$. Let ζ_τ be the state at time τ. The *a posteriori* probability that the information symbol ξ equals a conditioned on the length-$N_b(= Np)$ received bit sequence $r_1^{N_b}$ is:

$$\begin{aligned} P(\xi = a | r_1^{N_b}) &= \frac{P(\xi = a, r_1^{N_b})}{P(r_1^{N_b})} \\ &= \frac{1}{P(r_1^{N_b})} \sum_{\zeta} \sum_{\zeta'} P(\xi = a, \zeta_\tau = \zeta, \zeta_{\tau-p} = \zeta', r_1^{N_b}) \end{aligned} \tag{3.2}$$

Equation (3.2) is obtained using Bayes' rule and the principle of total probability. By applying Bayes' rule and the Markov property that events after time τ only depend on the current state ζ_τ and are independent of past observations [58], it is possible to write (3.2) as follows:

$$\begin{aligned} P(\xi = a, \zeta_\tau = \zeta, \zeta_{\tau-p} = \zeta', r_1^{N_b}) &= P(r_1^{N_b} | \zeta_\tau = \zeta) \cdot \\ \cdot P(\xi = a, \zeta_\tau = \zeta, r_{\tau-(p-1)}^{\tau} | \zeta_{\tau-p} = \zeta') &\cdot P(\zeta_{\tau-p} = \zeta', r_1^{\tau-p}) \\ &= \beta_\tau(\zeta) \cdot \gamma_{(\tau-(p-1),\tau)}^{a}(\zeta, \zeta') \cdot \alpha_{\tau-p}(\zeta') \end{aligned} \tag{3.3}$$

The term $\beta_\tau(\zeta)$ is the backward state metric, the term $\gamma_{(\tau-(p-1),\tau)}^{a}(\zeta, \zeta')$ is the branch transition probability and the term $\alpha_{\tau-p}(\zeta')$ is the forward state metric. Using the Bayes' rule and the fact that the symbol *apriori* probabilities are state-independent, the branch transition metric can be written as follows:

$$\begin{aligned} \gamma_{(\tau-(p-1),\tau)}^{a}(\zeta, \zeta') &= P(\xi = a) \cdot \\ \cdot P(\zeta_\tau = \zeta | \xi = a, \zeta_{\tau-p} = \zeta') &\cdot \end{aligned} \tag{3.4}$$

$$\cdot P(r^{\tau}_{\tau-(p-1)}|\zeta_\tau = \zeta, \xi = a, \zeta_{\tau-p} = \zeta')$$

If state ζ at time τ is connected to state ζ' at time $\tau - p$ via the input sequence $\xi = a$ then $P(\zeta_\tau = \zeta|\xi = a, \zeta_{\tau-p} = \zeta') = 1$, otherwise $P(\zeta_\tau = \zeta|\xi = a, \zeta_{\tau-p} = \zeta') = 0$. The term $P(r^{\tau}_{\tau-(p-1)}|\zeta_\tau = \zeta, \xi = a, \zeta_{\tau-p} = \zeta')$ is a function of the channel characteristic; in a PR channel with AWGN the pdf can be calculated as

$$(\frac{1}{\sqrt{\pi N_0}}) \exp(\frac{-\sum_{i=0}^{p-1}(r_{\tau-i} - y_i)^2}{N_0}) \tag{3.5}$$

In (3.5) $N_0/2$ is the noise variance and y^0_{p-1} is the PR channel output sequence that corresponds to the input sequence $a = b^0_{p-1}$.

The forward state metric $\alpha_\tau(\zeta)$ and the backward state metric $\beta_\tau(\zeta)$ can be updated as in the original BCJR algorithm [66].

In the case of binary-input length-ν memory ISI channels, since the states represent subsequences of the input sequence [58], the equation (3.3) can be expressed as follows:

$$\begin{aligned} & P(\xi = a, \zeta_\tau = \zeta, \zeta_{\tau-p} = \zeta', r_1^{N_b}) = \\ &= P(\xi = b^0_{p-(\nu+1)}, \zeta_\tau = \zeta, \zeta_{\tau-(p-\nu)} = \zeta'', r_1^{N_b}) \\ &= \alpha_{\tau-(p-\nu)}(\zeta'')\gamma^{b^0_{p-(\nu+1)}}_{(\tau-(p-(\nu+1)),k)}(\zeta'', \zeta)\beta_\tau(\zeta) \end{aligned} \tag{3.6}$$

In (3.6) ζ'' is the state that corresponds to the shift register configuration after an input of ν bits. This simplification is valid when $p \geq \nu$. Therefore, for sake of generality, this section refers the SB detecting structure described in [69].

3.1.2 Receiver architectures

The next three subsections introduce the three detection-and-decoding architectures considered in this section: the serially concatenated, turbo concatenated, and joint MP based architectures respectively. In the forth subsection the computational complexity of each proposed architecture is discussed.

Serially concatenated architecture

This subsection considers a serially concatenated architecture (SCA).

q-ary LDPC codewords are properly mapped to a BPSK modulated signal. At the receiver, the detector defines the *a priori* probabilities that have to be provided to the q-ary LDPC decoder. The decoding employs the message passing algorithm, which is described in detail in [3], [4]. In this section it is assumed that detection may be bit-based (subsection 3.1.1) and symbol-based (subsection 3.1.1): both detectors involve the BCJR algorithm [66].

Turbo concatenated architecture

In this subsection, an architecture that employs a turbo concatenated receiver is considered.

In this turbo concatenated architecture (TCA), the transmitted signal is a q-ary LDPC codeword that has been properly mapped in a BPSK sequence. At the receiver, a soft detector incorporates extrinsic information provided by the q-ary LDPC decoder, and the LDPC decoder incorporates soft information provided by the detector. Extrinsic information between the detector and decoder is exchanged in an iterative way until an LDPC codeword is found or a maximum number of iterations are performed [4], [67], [60], [61]. With LDPC codes, convergence to a codeword is easily detected by the receiver when the parity check equations are satisfied. The decoding employs the message passing algorithm, which is described in detail in [3], [4]. The detection may employ a bit-based and a symbol-based BCJR algorithm: these detectors are introduced in subsections 3.1.1 and 3.1.1 respectively.

Joint MP based architecture

In this subsection an architecture that employs a joint detection-and-decoding system based on a MP algorithm is considered.

In the architectures introduced in subsections 3.1.2 and 3.1.2 the channel constraints are represented by the channel trellis and the code constraints are represented by the bipartite graph. In the joint MP based architecture, the channel constraints are represented as a graph, therefore the MP detection algorithm has to be designed to operate on this graph [65]. Specifically, in this subsection both bit-based and symbol-based BCJR algorithms operating on the channel constraints are considered. The channel constraints are represented on the graph by a set of nodes called channel nodes.

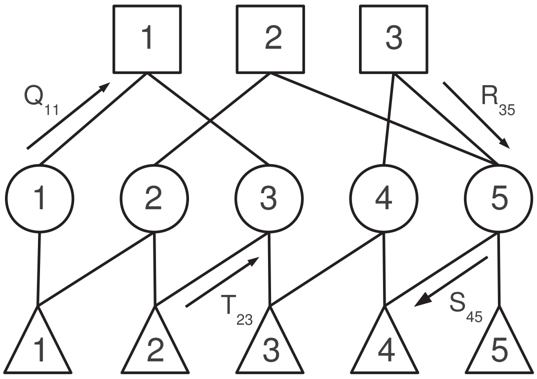

Figure 3.2: A graph that represents channel constraints and the parity check of the LDPC code. Circles represent variable nodes, squares represent parity check nodes and triangles represent channel nodes

The graph obtained by including the parity check nodes to the channel graph is tripartite, as shown in Figure 3.2: circles correspond to the variable (or symbol) nodes ξ_κ, squares to the parity-check nodes z_κ and triangles to the channel nodes ϵ_κ.

A parity check node i and a variable node j are connected if the value of

H_{ij} is not zero. A channel node k is connected to the variable node j if the channel response involves bits from the binary representation of ξ_k and ξ_j.

Following the notation in [3], let $N(i) = \{j : H_{ij} \neq 0\}$ be the set of symbols linked to check node i and let the checks linked to symbol j belong to $M(j) = \{i : H_{ij} \neq 0\}$. Moreover, let $L(k)$ be the set of the channel nodes connected to the k-th symbol node and let $I(l)$ be the set of the symbol nodes connected to the channel node l.

At this point, for every $a \in GF(q)$ two quantities are set up, Q^a_{ij} and R^a_{ij}, for each non-zero element of the PC matrix H. The first is defined as the probability that j-th symbol is a, depending on the information flowing by the whole checks except the i-th one. R^a_{ij} instead is meant to be the probability of check z_i being satisfied if ξ_j is equal to a.

Analogously, let S^a_{kl} be the probability that the symbol l is a given the information obtained by the channel nodes other than the k-th. Further, T^a_{kl} is the probability of channel node ϵ_k being satisfied if symbol l is considered fixed at a. Figure 3.2 show these quantities on the tripartite graph. Finally let $X[\tau]$ be the value of X at the τ-th iteration of the iterative algorithm.

The joint MP algorithm works as follows.

Initialization

The channel inputs are i.i.d., so all state transitions are initially equally likely [65]. The *a priori* probabilities are then initialized as:

$$f^a_j = P(\xi_j = a) = \frac{1}{q} \tag{3.7}$$

for $\forall j = 1, \ldots, N$.

Updating S_{ij}^a

For every iteration of the MP algorithm, the messages S_{ij}^a that symbol j sends to channel node i should be the belief the parent has that it is in state a, according to all other children. Therefore, $S_{ij}^a[\tau]$ can be expressed as:

$$S_{ij}^a[\tau] = f_j^a \prod_{k \in L(j) \setminus i} T_{kj}^a[\tau - 1] \prod_{l \in M(j)} R_{lj}^a[\tau - 1] \tag{3.8}$$

Updating Q_{ij}^a

The messages Q_{ij}^a that symbol node j sends to the parity-check node i should be the belief the parent has that it is in state a, according to all other children. Thus, Q_{ij}^a at the τ-th iteration is updated as follows:

$$Q_{ij}^a[\tau] = f_j^a \prod_{k \in L(j)} T_{kj}^a[\tau - 1] \prod_{l \in M(j) \setminus i} R_{lj}^a[\tau - 1] \tag{3.9}$$

Updating T_{ij}^a

The message that channel node i sends to symbol node j should be the probability of channel node i being satisfied if ξ_j was in state a. Thus, it is necessary to sum over all the configurations $\underline{\xi}$ for which the channel constraint is satisfied and the j-th symbol is in state a and add up the probability of the configuration, as follows:

$$\begin{aligned} T_{ij}^a[\tau] &= \frac{P(\xi_j = a | \{r_\mu\}_{\mu \in I(i)}, \{\xi_\mu\}_{\mu \in I(i)})}{P(r_j | \xi_j = a) \cdot P(\xi_j = a)} \\ &= \sum_{\underline{\xi} : \xi_j = a} P(\epsilon_i | \underline{\xi}) \prod_{k \in I(i) \setminus j} S_{ik}^{\xi_k}[\tau - 1] \end{aligned} \tag{3.10}$$

The probability $P(\epsilon_i|\underline{\xi})$ of the channel constraint being satisfied is either 0 or 1 for any given configuration $\underline{\xi}$.

The way the value of T_{ij}^{a} has to be computed depends on the considered detection scheme. Specifically, when a SB detector is employed, the computation of T_{ij}^{a} is based on (3.2) and (3.3). In other terms, T_{ij}^{a} can be expressed as follows:

$$T_{ij}^{a}[\tau] \sim \sum_{\{\zeta_L,\zeta_R:\zeta_L \xrightarrow{\xi_j=a} \zeta_R\}} \widetilde{\alpha}_{j-1}(\zeta_L) \cdot \gamma^{a}_{(j-1,j)} \cdot \widetilde{\beta}_j(\zeta_R) \qquad (3.11)$$

The value of $\widetilde{\alpha}$ and $\widetilde{\beta}$ can be calculated as follows:

$$\begin{aligned} \widetilde{\alpha}_{j-1}(\zeta_L) &= \sum_{t=1}^{|\Theta^L|} \alpha_{j-1}(\zeta_L) \cdot \prod_{k \in I(j-1)\backslash j} S^{\theta_t^L}_{(j-1)k}[\tau-1] \\ \widetilde{\beta}_j(\zeta_R) &= \sum_{t=1}^{|\Theta^R|} \beta_j(\zeta_R) \cdot \prod_{k \in I(j+1)\backslash j} S^{\theta_t^R}_{(j+1)k}[\tau-1] \end{aligned} \qquad (3.12)$$

θ_κ^L and θ_κ^R represent the possible configurations of the input that lead to the states ζ_L and ζ_R respectively. On the other hand, α and β represent the forward state metric and the backward state metric respectively, as previously mentioned in subsection 3.1.1.

ζ_L and ζ_R are linked convoluting the binary representation of the q-ary symbol ξ_j that has been set to a and the given channel response. Each $\theta_\kappa^L \in \Theta^L = \{\theta_\kappa^L\}_{\kappa=1,\ldots,|\Theta^L|}$ and each $\theta_\kappa^R \in \Theta^R = \{\theta_\kappa^R\}_{\kappa=1,\ldots,|\Theta^R|}$ can be constituted by one or more symbols, depending on the length ν of the channel memory.

It is easy to notice that the S contributions related to $\widetilde{\alpha}$ represent the causal part of the ISI effect, while the S contributions related to $\widetilde{\beta}$ represent the anti-causal part of it.

Since the channel constraints for a given PR channel are well defined, it is natural to ask if a more efficient way to compute the probability expressed in (3.10) could exist.

Specifically, the edges between channel nodes and the variable nodes in the tripartite graph can be represented by a square adjacency matrix, $\Lambda = \{\Lambda_{ij}\}_{i,j=1,\ldots,N}$. If Λ_{ij} is not set to zero, an edge in the tripartite graph between the i-th channel node and the j-th variable node has to be drawn. Therefore, the variable node j belongs to the set $N_\Lambda(i)$, where $N_\Lambda(i) = \{j : \Lambda_{ij} \neq 0\}$.

The expression of the adjacency matrix Λ depends on the length ν of the channel memory and on the alphabet size q of the code. E.g., the adjacency matrix for a 16-ary LDPC coded transmission over the EPR4 channel (that is, $\nu = 3$) is as follows:

$$\Lambda = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 1 & 0 & \ldots & 0 \\ \vdots & & & \ddots & \ddots & & \vdots \\ 0 & 0 & \ldots & \ldots & 0 & 1 & 1 \end{bmatrix} \tag{3.13}$$

Each element of Λ has its own binary representation depending on the channel response $h(D)$ and the alphabet size q. Specifically, the binary representation Λ^b_{ij} of Λ_{ij} is a square p x p matrix, where $p = \log_2 q$.

E.g., since the EPR4 channel response is $h_{EPR4}(D) = 1 + D - D^2 - D^3$, each Λ^b_{ij} related to the matrix in (3.13) can be written as follows $\forall i = 1, \ldots, N$:

$$\Lambda^b_{ii} = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 0 & 1 & 1 & -1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.14}$$

$$\Lambda^b_{i(i+1)} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 \\ 1 & -1 & -1 & 0 \end{bmatrix} \tag{3.15}$$

T^a_{ij} can be efficiently calculated by treating the partial sums of a parity check as the states in a Markov chain [4], therefore T_{ij}^a can be written as follows:

$$T_{ij}^a[\tau] = \sum_{\{\zeta_L,\zeta_R:\zeta_L \xrightarrow{\xi_j=a} \zeta_R\}} P(F_{i(j-1)}^T \Rightarrow \zeta_L) \cdot P(B_{i(j+1)}^T \Rightarrow \zeta_R) \tag{3.16}$$

In order to better exploit the definition of F^T and B^T, let $\varphi_u^v(X)$ be the law that transforms X (living in $GF(u)$) in its v-ary counterpart. Thus, the terms denoted by F^T and B^T are defined as follows:

$$\begin{array}{l} F_{\kappa k}^T = \varphi_2^q(\sum_{j:j\leq k} \Lambda_{\kappa j}^b \cdot \varphi_q^2(\xi_j)) \\ B_{\kappa k}^T = \varphi_2^q(\sum_{j:j\geq k} \Lambda_{\kappa j}^b \cdot \varphi_q^2(\xi_j)) \end{array} \tag{3.17}$$

The corresponding probabilities are computed as:

$$P(F_{\kappa j}^T \Rightarrow a) = \sum_{\{s,t:s \xrightarrow{\Lambda_\varphi} a\}} P(F_{\kappa i}^T \Rightarrow s) \cdot S_{\kappa j}^t[\tau - 1] \tag{3.18}$$

$$P(B_{\kappa j}^T \Rightarrow a) = \sum_{\{s,t:s \xleftarrow{\Lambda_\varphi} a\}} P(B_{\kappa i}^T \Rightarrow s) \cdot S_{\kappa j}^t[\tau - 1] \tag{3.19}$$

Λ_φ is set to $\varphi_2^q(\Lambda_{\kappa j} \cdot \varphi_q^2(t))$. That is, s and t have to be chosen such that the convolution of those values and the channel response leads the system to the state a. i, j are successive indexes living in $N_\Lambda(\kappa)$, with $j > i$ for the F^T contribution, while $j < i$ for the B^T part.

Updating R_{ij}^a

The message that check node i sends to symbol node j should be the probability of check node i being satisfied if ξ_j was in state a. As in step 3.1.2, using the laws of probability R_{ij}^a can be expressed as:

$$\begin{aligned} R_{ij}^{a}[\tau] &= P(z_i|\xi_j = a) \\ &= \sum_{\underline{\xi}:\xi_j=a} P(z_i|\underline{\xi}) \cdot P(\underline{\xi}|\xi_j = a) \\ &= \sum_{\underline{\xi}:\xi_j=a} P(z_i|\underline{\xi}) \cdot \prod_{\kappa\in N(i)\backslash j} Q_{i\kappa}^{\xi_\kappa}[\tau-1] \end{aligned} \tag{3.20}$$

The probability $P(z_i|\underline{\xi})$ of the check being satisfied is either 0 or 1 for any given configuration $\underline{\xi}$ as in the previous step. R_{ij}^{a} can be efficiently calculated by treating the partial sums of a parity check as the states in a Markov chain [4], therefore R_{ij}^{a} can be written as follows:

$$R_{ij}^{a}[\tau] = \sum_{\{s,t:s+t+H_{ij}a=0\}} P(F_{i(j-1)}^{R} = s) \cdot P(B_{i(j+1)}^{R} = t) \tag{3.21}$$

The terms denoted by F^R and B^R are defined as:

$$\begin{aligned} F_{\kappa k}^{R} &= \textstyle\sum_{j:j\leq k} H_{\kappa j} \cdot \xi_j \\ B_{\kappa k}^{R} &= \textstyle\sum_{j:j\geq k} H_{\kappa j} \cdot \xi_j \end{aligned} \tag{3.22}$$

The corresponding probabilities are computed as:

$$\begin{aligned} P(F_{\kappa j}^{R} = a) &= \textstyle\sum_{\{s,t:H_{\kappa j}t+s=a\}} P(F_{\kappa i}^{R} = s) \cdot Q_{\kappa j}^{t}[\tau-1] \\ P(B_{\kappa j}^{R} = a) &= \textstyle\sum_{\{s,t:H_{\kappa j}t+s=a\}} P(B_{\kappa i}^{R} = s) \cdot Q_{\kappa j}^{t}[\tau-1] \end{aligned} \tag{3.23}$$

where i, j are successive indexes living in $N(\kappa)$, with $j > i$ for the F^R contribution, while $j < i$ for the B^R part.

Tentative decoding

A tentative decoding codeword $\widehat{\underline{\xi}}$ be derived using the following expression:

$$\widehat{\xi}_j = \arg\max_a f_j^a \prod_{k\in L(j)} T_{kj}^{a}[\tau] \prod_{l\in M(j)} R_{lj}^{a}[\tau] \tag{3.24}$$

If $\underline{\widehat{\xi}} = \left[\widehat{\xi}_j\right]_{j=1,\dots,N}$ satisfies (3.1), then the decoding process is stopped, declaring a success, otherwise the algorithm iterates from step 3.1.2. A failure is declared if the codeword is not found after reaching a fixed maximum number of iterations.

Computational complexity

In this subsection the computational complexity of each aforesaid receiver architecture is discussed.

The number of operations required from each system is given by the complexity of the used couple of detector and decoder. Table 3.1 shows the computational complexity of each receiver for a length-N codeword of a LDPC code over $GF(q)$. Specifically, $p = \log_2 q$ and ν represents the length of the memory of the considered ISI channel.

Since a classic "flooding" decoding scheme [3] has been taken into account, the computational complexity of a q-ary LDPC decoder is $O(Ntq^2)$, as already stated in this section; t is the average column weight. It is proper to point out that there exist specific decoding schemes, such as those based on layered belief propagation (LBP) algorithm [57], that can lower the aforesaid computational cost.

However, the overall complexity of a given receiver depends on the number of operations required by the detector and the decoder separately and on the way detector and decoder are linked in the architecture that is taken into account.

The systems employing a SB detector typically show lower computational complexities with respect to the correspondent architecture employing a BB detecting scheme. In order to better exploit this point, it is proper to take into account a trellis of p stages, that correspond to an input sequence of p bits. In these conditions, the number of paths that have to be set while employing a BB detector is $p \cdot 2^{\nu+1}$. On the other hand, a SB detector in the same conditions needs only $\sum_{i=0}^{p-1} 2^{\chi(\nu-i)}$ patterns to work, where $\chi(t) = t$ if $t \geq 0$, $\chi(t) = 0$ otherwise. Therefore, the bit-based detecting scheme requires a larger number of operations to work than the symbol-based detecting scheme.

The serially concatenated architecture requires a number of operations that is simply the sum of those needed by the q-ary LDPC decoder and the employed detector.

On the other hand, the turbo concatenated architecture shows a computational complexity that is proportional to N_{TI}, that is the number of iterations between detector and decoder. Therefore, the architecture introduced in subsection 3.1.2 show a computational complexity N_{TI} times lower than that related to the TCA.

Finally, the architecture introduced in 3.1.2 requires a number of operations that is related to the sum of those necessary for the detection and decoding steps. Specifically, the computational complexity of the joint MP based scheme depends on the convergence speed of the system. That is, the number of operations required by the architecture decreases as the detection-and-decoding scheme finds a codeword within the maximum number of iterations of the MP algorithm.

In other terms, for a given detection scheme and a given q-ary decoder, joint MP based architecture shows a computational complexity that is typically lower than that of TCA and higher than that required by a SCA system.

3.1.3 Simulation results

In this section, simulation results obtained by implementing the three structures introduced in the previous section are discussed.

In each of these implementations, a 16-ary LDPC code with coding rate $R = 1 - \frac{M}{N}$ equal to 8/9 is used. The codeword blocklength is set to 1152 symbols (that is 4608 bits) and the average column weight is set to 2.88. The variable-node degree distribution, following the notation introduced in [21]

Table 3.1: Computational complexity of the proposed architectures

Receiver architecture	**Computational complexity**
SCA w/ BB det	$Np2^{\nu+1+p} + O(Ntq^2)$
TCA w/ BB det	$[Np2^{\nu+1+p} + O(Ntq^2)] \cdot N_{TI}$
BB Joint MP based	$O(Np2^{\nu+1+p} + Ntq^2)$
SCA w/ SB det	$N((p+2)2^{\nu} + 2^{\nu} - 1) + O(Ntq^2)$
TCA w/ SB det	$[N((p+2)2^{\nu} + 2^{\nu} - 1) + O(Ntq^2)] \cdot N_{TI}$
SB Joint MP based	$O(N((p+2)2^{\nu} + 2^{\nu} - 1) + Ntq^2)$

and [22], was $\lambda_2 = 0.112$ and $\lambda_3 = 0.8889$, where $\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1}$, and d_v is the maximum symbol-node degree. The q-ary LDPC code has been constructed using quasi-regular PC matrices [22] generated by a modified PEG algorithm [18] that maximizes the minimum space distance [48], [38]. The maximum number of iterations for the q-ary LDPC decoder of subsections 3.1.2 and 3.1.2 and for the message-passing algorithm of subsection 3.1.2 has been set to 25. Moreover, the maximum number of iterations between the detector and the q-ary LDPC decoder for the architecture of subsection 3.1.2 has been set to 10.

Three different PR channels having different memory length ν are considered. Each channel response $h(D)$ is expressed as follows:

$$\begin{aligned} h_{PR4}(D) &= 1 + D^2 \\ h_{EPR4}(D) &= 1 + D - D^2 - D^3 \\ h_{EEPR4}(D) &= 1 + 2D - 2D^3 - D^4 \end{aligned} \tag{3.25}$$

Figures 3.3, 3.4 and 3.5 show the simulation results, as Figures 3.7, 3.8 and 3.9 as well. The serially concatenated architecture is plotted in blue line, the turbo concatenated architecture in red line and the joint MP based architecture in black line. Moreover, the results of the architectures employing a

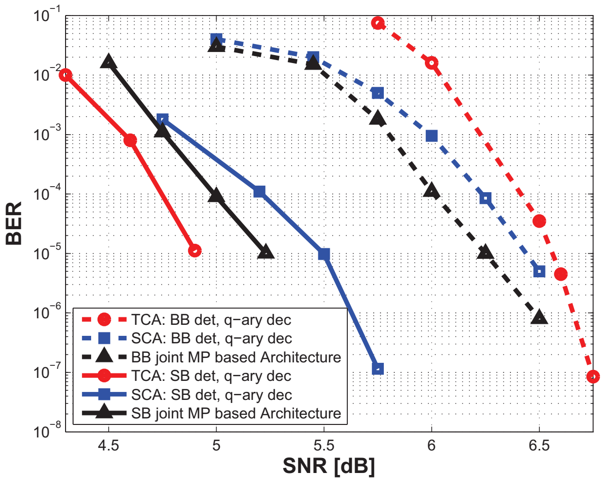

Figure 3.3: Bit Error Rate performance of the architectures discussed in Section 3.1.2 on the EPR4 channel: serially concatenated architecture (SCA), turbo concatenated architecture (TCA), joint MP based architecture. SCA and TCA employ bit-based (BB) and symbol-based (SB) detectors and q-ary LDPC decoder. Joint MP based architecture employs a BB BCJR algorithm or a SB BCJR algorithm to operate on the channel constraints.

bit-based detector are plotted in dashed line, while the results of the architectures employing a symbol-based detector are plotted in solid line.

The systems employing a symbol-based detector largely outperform the architectures employing a bit-based detector. A reason for this might be that it is possible to include invalid trellis paths in the calculation by using the bit-based detection approach [58]. Thus, the SCA employing BB detection shows better error-rate performance than TCA employing BB detection, since BB-TCA iterates the aforesaid information between detector and decoder.

Specifically, the minimum gain of the systems employing a SB detector goes from about 0.3 dB to about 0.75 dB in terms of SNR, depending on the PR channel that is taken into account. This behaviour highlights how the rate $\frac{p}{\nu+1}$ plays an important role in the error-rate performance of the described architectures.

Further, the ratio $\frac{p}{\nu+1}$ influences the error-rate performance of the joint MP

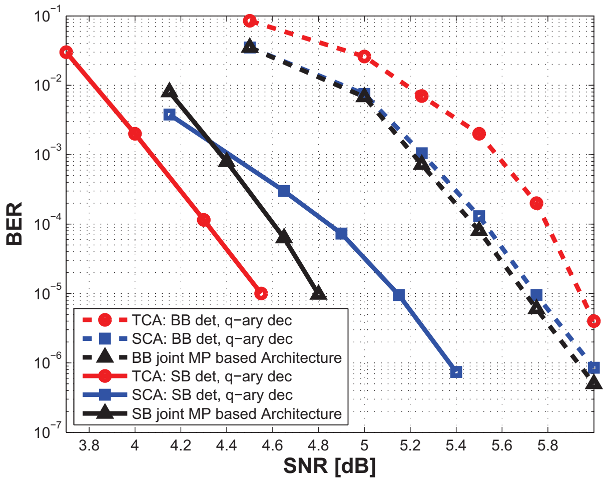

Figure 3.4: Bit Error Rate performance of the architectures discussed in Section 3.1.2 on the PR4 channel: serially concatenated architecture (SCA), turbo concatenated architecture (TCA), joint MP based architecture. SCA and TCA employ bit-based (BB) and symbol-based (SB) detectors and q-ary LDPC decoder. Joint MP based architecture employs a BB BCJR algorithm or a SB BCJR algorithm to operate on the channel constraints.

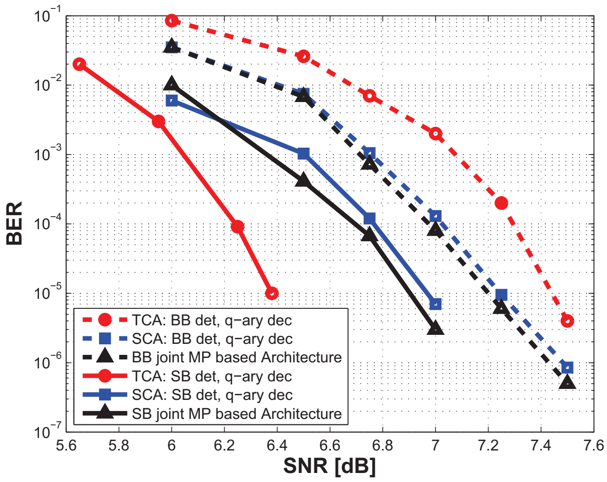

Figure 3.5: Bit Error Rate performance of the architectures discussed in Section 3.1.2 on the EEPR4 channel: serially concatenated architecture (SCA), turbo concatenated architecture (TCA), joint MP based architecture. SCA and TCA employ bit-based (BB) and symbol-based (SB) detectors and q-ary LDPC decoder. Joint MP based architecture employs a BB BCJR algorithm or a SB BCJR algorithm to operate on the channel constraints.

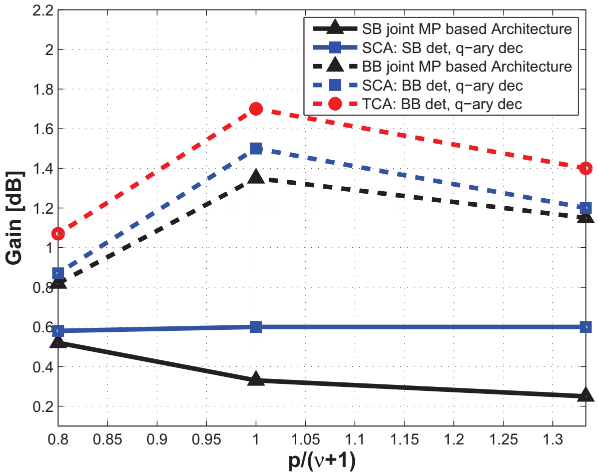

Figure 3.6: Gain of the TCA employing SB detector on the other architectures proposed in Section 3.1.2 for the PR channels taken into account. The gain is computed in terms of SNR at BER $=10^{-5}$; $p = \log_2 q$ and ν is the memory length of the given PR channel.

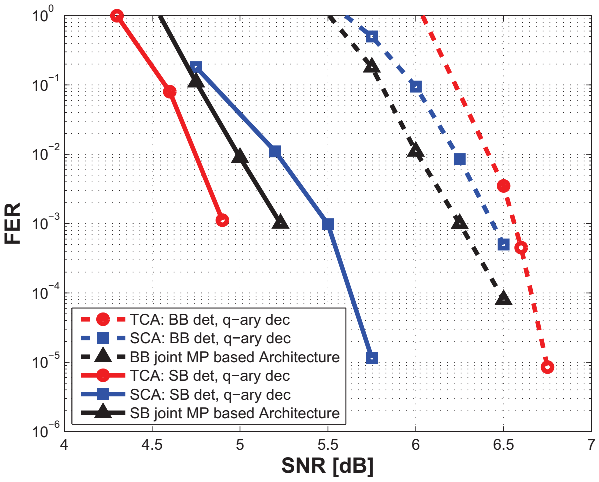

Figure 3.7: Frame Error Rate performance of the architectures discussed in Section 3.1.2 on the EPR4 channel: serially concatenated architecture (SCA), turbo concatenated architecture (TCA), joint MP based architecture. SCA and TCA employ bit-based (BB) and symbol-based (SB) detectors and q-ary LDPC decoder. Joint MP based architecture employs a BB BCJR algorithm or a SB BCJR algorithm to operate on the channel constraints.

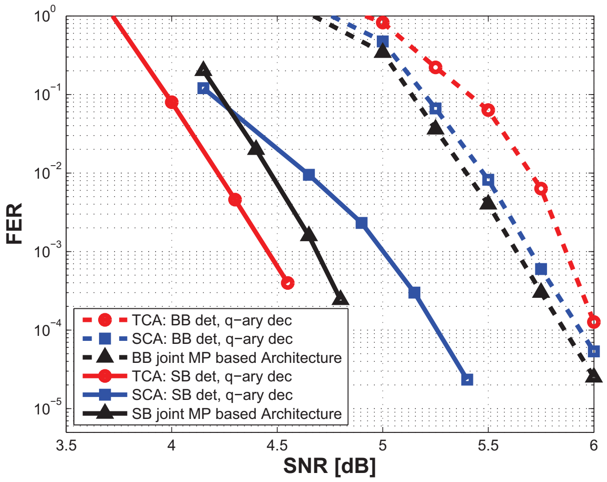

Figure 3.8: Frame Error Rate performance of the architectures discussed in Section 3.1.2 on the PR4 channel: serially concatenated architecture (SCA), turbo concatenated architecture (TCA), joint MP based architecture. SCA and TCA employ bit-based (BB) and symbol-based (SB) detectors and q-ary LDPC decoder. Joint MP based architecture employs a BB BCJR algorithm or a SB BCJR algorithm to operate on the channel constraints.

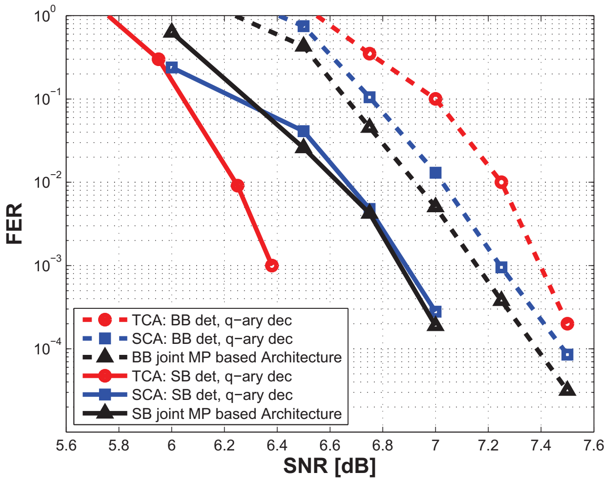

Figure 3.9: Frame Error Rate performance of the architectures discussed in Section 3.1.2 on the EEPR4 channel: serially concatenated architecture (SCA), turbo concatenated architecture (TCA), joint MP based architecture. SCA and TCA employ bit-based (BB) and symbol-based (SB) detectors and q-ary LDPC decoder. Joint MP based architecture employs a BB BCJR algorithm or a SB BCJR algorithm to operate on the channel constraints.

based architecture. In fact, the joint MP based architecture employing a BB detection outperforms both BB-TCA and BB-SCA. For the BER curves, the gain is really tiny on PR4 and EEPR4 channels, where $\frac{p}{\nu+1} \neq 1$.

Moreover, TCA employing SB detection outperforms SB-SCA and SB joint MP based architecture as well. This result furthermore highlights the influence of the computation of the soft input to the q-ary LDPC decoder on the extrinsic information required in the turbo architecture.

Figures 3.7, 3.8 and 3.9 globally confirm the aforesaid observations. The frame-error rate curves of the architectures employing a Bit Based detection scheme show how the gain of joint MP based architecture on the other receivers in terms of SNR is larger than that observed for each PR channel in Figures 3.3, 3.4 and 3.5. Moreover, SB-TCA shows the best FER performance. The SB joint MP based architecture outperforms the FER performance of SB-SCA on every PR channel but the EEPR4 channel.

Further, Figure 3.6 shows the gain of the SB-TCA with respect to the other proposed architectures in terms of SNR as a function of the ratio $\frac{p}{\nu+1}$. The gain is calculated as the given bit-error rate curve passes the threshold of 10^{-5} in Figure 3.3 (i.e., $\frac{p}{\nu+1} = 1$), Figure 3.4 (i.e., $\frac{p}{\nu+1} = 1.333$) and Figure 3.5 (i.e., $\frac{p}{\nu+1} = 0.8$).

Specifically, the gain of SB-TCA on the architectures employing a BB detection scheme is greater than 0.8 dB at least. Further, the gain is maximum on every other BB receiver for $\frac{p}{\nu+1} = 1$.

On the other hand, it looks like the more $\frac{p}{\nu+1}$ gets larger, the more SB-TCA outperforms symbol-based joint MP based architecture. Specifically, the minimum gain of SB-TCA on SB joint MP based architecture is achieved on the PR4 channel and it is about 0.25 dB.

Finally, a very interesting task is addressed by the behaviour of SB-SCA with respect to SB-TCA. In fact, SB-TCA gains about 0.6 dB on SB-SCA for each PR channel that has been taken into account. This is a result that might require a deeper investigation.

3.1.4 Conclusions

Three receiver architectures employing q-ary LDPC codes have been introduced and analyzed in order to study their bit-error-rate performance over some PR channels.

Simulation results for 16-ary LDPC code over three different partial response channels showed that the best performance can be achieved by using a turbo concatenated architecture whose detection is symbol-based. Consequently, using such an architecture appears to have the best performance in an environment affected by ISI where good error-correction capability is desirable. Moreover, it has been observed that in general a symbol-based detection provides better performance with respect to a bit-based detection.

Ongoing research that promises efficiency at the receiver includes the analysis of different structures as a joint MP based architecture employing a symbol-based BCJR algorithm operating on the channel constraints. Future directions for research could focus on the behavior of the proposed architectures over different channels (like the magnetic recording channel) and with different LDPC codes, having different codeword length or degree-distribution profile as in [35].

Moreover, in order to complete the analysis on the efficient receivers over PR channels, a study of the influence of the ratio $\frac{p}{\nu+1}$ have to be performed. Further, an investigation on the better detection method to be used has to be addressed, as a study on the improvements provided by other decoders (such as those based on LBP algorithms [57]) as well. Finally,an optimization of the construction of q-ary LDPC codes have to be taken into account, starting from the results provided in [26] and [28].

Chapter 4

Applications of q-ary codes

Recently rate-compatible (RC) LDPC codes have been used to achieve performance close to the channel capacity for different applications that require rate adaptivity. In Section 4.1 the joint q-ary LDPC decoding of correlated sources is considered, exploiting the differences of symbol puncturing patterns at the encoders. Despite the proposed algorithm simplicity, comparison with the independent decoding of such sequences, using both punctured and non-punctured codes, shows substantial coding gains. Future directions for research and possible applications are finally discussed.

Phase Change Memory definitely represents one of the most promising technologies among the non-volatile memories to be used in the next decade. Even though the noise resilience of PCMs looks to be very strong, an information theory-based design may improve the error-rate performance of the PCM reading process. In Section 4.2, a channel model for the information recovery in PCMs is introduced and discuss the related mutual information. Experimental results are provided in order to highlight relationships between information recovery performance, electrical parameters of the memory cell, and the sensitivity of the reading architecture. This information theory-based point of view represents a base for an effective optimization of the error-rate performance during the PCM reading process and opens the path to very interesting new research lines.

4.1 Exploiting Source Correlation to Improve Performance of q-ary Rate Compatible LDPC Codes

Recent technological improvements have allowed the employment of complex structures in order to reach the maximum reliability in data communications. The manifold ways information can be transmitted and received have determined the flourishing of many topologies in the communication architectures. Several papers [71], [72] have highlighted how processing as much information as possible could improve the overall performance of the systems. Specifically, some approaches have been proposed to improve the robustness of transmissions of coded correlated sources over noisy channels. In regards to this, see [73] - [75] and references therein.

The main contribution of this section is related to the possibility of exploiting punctured q-ary codes for the joint coding and decoding processes of correlated sources. The only two needed requirements are a joint coding algorithm, based on different puncturing patterns, and a joint decoding algorithm of the contemporaneous received correlated codewords.

Recent works reveal that LDPC codes may be efficiently applied to rate-adaptive systems, by employing RC codes. Proper puncturing algorithms [76], [77] and PC matrix construction methods [78], [18] have been proposed in literature.

In this section RC-LDPC codes on a multiple source transmission are employed, paying particular attention to the correlation of the sources. Different puncturing patterns are defined for the different correlated sources, starting from the algorithms provided in [76], [77], and employing non-binary LDPC codes. A novel decoding scheme exploiting the correlation among the sequences is also introduced and compared with an independent decoding strategy.

The section is organized as follows. Section 4.1.1 introduces the system model for the architectures that are discussed in Section 4.1.2: the independent stream decoding and the two versions of cross stream decoding. In Section 4.1.3 the simulation results are given, and the practical aspects of the system implementing are discussed. Conclusions about future research development conclude the section.

4.1.1 System Model

In this section, the performance of two different decoding architectures for LDPC coded correlated sources are analyzed. These systems employ the independent stream decoding and the cross stream decoding. For each of these architectures, the basic system model is shown in Figure 4.1.

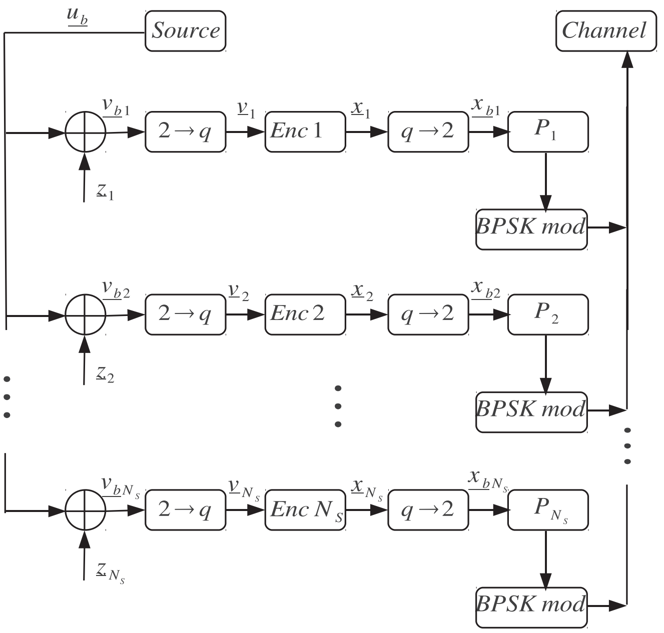

Figure 4.1: Block diagram of the transmitting system. $2 \rightarrow q$ operates the conversion from a binary alphabet to a q-ary alphabet. $q \rightarrow 2$ operates the conversion from a q-ary alphabet to a binary alphabet. P_k operates puncturing on the k-th stream.

The information source generates a sequence $\underline{u}$ of K symbols living in $GF(q)$: i.e., $\underline{u} = [u_i]_{i=1,\dots,K}$, $u_i \in GF(q)\ \forall\, i = 1,\dots,K$. Each symbol is $p = \log_2 q$ bits long: q depends on the alphabet size of the LDPC code that is meant to be used. The information message is then replicated N_S times. Each copy of the binary representation $\underline{u_b}$ of $\underline{u}$ passes through N_S blocks that operate the sum in $GF(2)$. The k-th copy of $\underline{u_b}$ is meant to be summed to the length-Kp sequence $\underline{z}_k$ for $k = 1,\dots,N_S$. The resulting sequence has the following expression:

$$\underline{v_{b_k}} = \underline{u_b} \oplus \underline{z}_k \tag{4.1}$$

where each $\underline{z}_k$ is chosen accordingly to the value of the empirical cross-correlation ρ_{mn} of the sequences that have been taken into account. For every couple of streams ρ_{mn} is defined as $\rho_{mn} = \frac{\alpha_{mn}}{Kp}$, where α_{mn} represents the number of places in which $\underline{v_{b_m}}$ and $\underline{v_{b_n}}$ agree, being $m \neq n$ [73]. In this section $\alpha_{mn} = \alpha$ for every $m, n \in \{1,\dots,N_S\}$. Therefore, ρ_{mn} is set to a given value $\rho\ \forall\, m, n \in \{1,\dots,N_S\}$, $m \neq n$ as well. Thus, ρ_{mn} can be written for every couple of $\underline{v_b}$s as follows:

$$\begin{aligned} \rho_{mn} &= \frac{\alpha_{mn}}{Kp} = \frac{K - C_1(\underline{v_{b_m}} \oplus \underline{v_{b_n}})}{Kp} \\ &= \frac{K - C_1(\underline{u_b} \oplus \underline{z}_m \oplus \underline{u_b} \oplus \underline{z}_n)}{Kp} \\ &= \frac{K - C_1(\underline{z}_m \oplus \underline{z}_n)}{Kp} = \rho \end{aligned} \tag{4.2}$$

$C_1(X)$ represents the number of ones in the sequence X.

Each $\underline{v_{b_k}}$ has its own q-ary representation $\underline{v}_k$ that is properly encoded. Therefore, for each stream a length-N codeword $\underline{x}_k = [x_{k_j}]_{j=1,\dots,N}$, $x_{k_j} \in GF(q = 2^p)\ \forall\, j = 1,\dots,N$, $\forall\, k = 1,\dots,N_S$ is generated such that

$$H(k)\underline{x}_k^T = \underline{0} \tag{4.3}$$

where $H(k) = \{H_{ij}(k)\}_{i=1,\dots,M,j=1,\dots,N}$, $H_{ij}(k) \in GF(q = 2^p)$ is the Parity Check matrix for the k-th stream. In this section each Parity Check matrix is chosen such that $H(k) = H \; \forall \, k = 1, \dots, N_S$.

Further, H is chosen such that each $\underline{x}_k$ is systematic, i.e., $\underline{x}_k = [\underline{c}_k | \underline{v}_k]$. $\underline{c}_k = [c_{k_i}]_{i=1,\dots,M}$, $c_{k_i} \in GF(q = 2^p) \; \forall \, i = 1, \dots, M$ is the parity check part of the codeword.

It is proper to remind that the empirical cross-correlation calculated on the codewords ρ_{mn}^{CW} is typically lower than ρ_{mn}. Moreover, since the expression of each $\underline{c}_k$ $k = 1, \dots, N_S$ depends on the Parity Check matrix H, it is not easy to infer on the codeword empirical cross-correlation such that $\rho_{mn}^{CW} = \rho^{CW}$ $\forall \, m, n \in \{1, \dots, N_S\}$ $m \neq n$.

The binary representation of each $\underline{x}_k$ represents the input to the block P_k that operates the puncturing. Namely each P_k can be different for any $k \in \{1, \dots, N_S\}$: however, in this section $P_k = P \; \forall \, k \in \{1, \dots, N_S\}$. After the puncturing operation, the N_{TX} unpunctured bits of each stream are BPSK modulated. $N_{TX} = (N - N_{P_l}) \cdot p$, where N_{P_l} represents the number of symbols that have to be punctured to achieve the transmission rate $R_{P_l} = \frac{K}{N_{TX}}$. Finally, additive white Gaussian noise is added to each bitstream.

At the receiver, the N_S length-N bitstreams $\underline{y}_k = [y_{k_j}]_{j=1,\dots,N}$, $k = 1, \dots, N_S$ are reconstructed: specifically, each punctured bit is replaced with a 0. The received signal, properly demapped, is then sent to the decoding system.

The next subsection introduces the proposed decoding systems with the related puncturing schemes.

4.1.2 Decoding architectures

In order to better exploit the decoding process of each architecture, let $g_j^{\kappa}(k)$ be the probability that the j-th received bit of the k-th stream is κ. That is, $g_j^{\kappa}(k) = P(y_{j_k} = 1 - 2\kappa)$ with $\kappa = \{0, 1\}$, $j = 1, \dots, N$ and $k = 1, \dots, N_S$.

Since the channel model that has been taken into account is memoryless and a q-ary LDPC code is employed, the decoder input for the k-th bitstream is $f_j^a(k) = \prod_{i=1}^{p} g_{j_i}^{a_i}(k)$,$k = 1, \ldots, N_S$ for each $a \in GF(q)$, where a_i is the i-th bit of the binary representation of a and $p = \log_2 q$.

Independent stream decoding

When independent stream decoding (ISD) is performed, puncturing patterns are derived by using the algorithms provided in [76] and [77]. Therefore, the transmission rate R_{P_l} that can be achieved after performing the puncturing operation has the following expression:

$$\begin{aligned} R_{P_l} &= \frac{R_0}{1 - \frac{N_{P_l} \cdot p}{N \cdot p}} \\ &= \frac{\frac{K}{N}}{\frac{N - N_{P_l}}{N}} \\ &= \frac{K}{N - N_{P_l}} \end{aligned} \tag{4.4}$$

R_0 is the rate of the PC matrix of the employed LDPC code over $GF(q = 2^p)$. N_{P_l} is the number of symbols that have to be punctured to achieve the transmission rate R_{P_l}. $l \in \{1, \ldots, I_H\}$, where I_H is the maximum number of iterations required to recover all punctured nodes [76]. Thus, the maximum achievable transmission rate is $R_{max} = R_{P_{I_H}} = \frac{K}{N - N_{P_{I_H}}}$.

The *a priori* probability for the j-th punctured bit in a given stream is set to $\frac{1}{2}$. Decoding is then performed by using a proper MP algorithm [3], [4].

Cross stream decoding

When cross stream decoding (CSD) is performed, a modified puncturing scheme is employed.

For a given transmission rate $R_{P_l}^{CSD}$ the algorithm proposed in [76] is modified in order to look for a set of $N_l^{CSD} = N_S \cdot N_{P_l}$ symbols suitable for puncturing. That is, the tree used to look for the best recoverable bits is extended N_S times deeper than what is described in [76].

Thus, N_{P_l} bits are punctured for each bitstream such that a bit that is punctured in a stream can not be punctured in the other $N_S - 1$ streams. Therefore, the transmission rate has the following expression:

$$\begin{aligned} R_{P_l}^{CSD} &= \frac{R_0}{1 - \frac{N_l^{CSD}}{N_S} \cdot \frac{1}{N}} \\ &= \frac{\frac{K}{N}}{\frac{N - N_l^{CSD}/N_S}{N}} \\ &= \frac{K}{N - \frac{N_l^{CSD}}{N_S}} \end{aligned} \tag{4.5}$$

R_0 is the rate of the PC matrix of the employed LDPC code over $GF(q = 2^p)$ and $l \in \{1, \ldots, I_H^{CSD}\}$. I_H^{CSD} is the maximum number of iterations required to recover all nodes that have been chosen to be punctured by using the modified puncturing algorithm. The maximum transmission rate R_{max}^{CSD} is achieved when $l = I_H^{CSD}$, that is, in this case, when all the bits of the binary representation of the codeword are punctured through all the streams. Therefore, the transmission rate obtained after puncturing is maximum if $N_l^{CSD} = N$. I.e., R_{max}^{CSD} can be written as follows:

$$\begin{aligned} R_{max}^{CSD} &= \frac{K}{N - \frac{N}{N_S}} \\ &= \frac{K}{N \cdot \frac{N_S - 1}{N_S}} \\ &= R_0 \cdot \frac{N_S}{N_S - 1}. \end{aligned} \tag{4.6}$$

At the receiver, the likelihood of the j-th punctured bit in the k-th bitstream is set to

$$g_j^{\kappa}(k) = \frac{\sum_{i \in \{1,\ldots,N_S\} \setminus k} g_j^k(i)}{N_S - 1} \tag{4.7}$$

where $\kappa = \{0, 1\}$, while $j \in S_\tau = \{\tau, \tau + 1, \ldots, N \cdot p\}$. If $\tau = 1$ the system performs the first version of CSD, called CSD-I. On the other hand, if $\tau = (M + 1) \cdot p$, the second version of CSD is used, called CSD-II.

CSD-I relies on the side information that can be recovered from the streams other than that is meant to be decoded. CSD-II relies on the error correction capability of the LDPC code. Specifically, CSD-I aims to use the statistical smoothing of the 1s in each $\underline{z}_k$ achieved operating the (4.7). On the other hand, CSD-II takes advantage of the value of empirical cross-correlation calculated on the information part of each codeword, since it is typically higher than that calculated on the whole codeword.

Decoding is finally performed by using a proper MP algorithm [3], [4].

4.1.3 Simulation results

In this section, simulation results obtained by using a 4-ary LDPC code and a 16-ary LDPC code with coding rate $R_0 = 1 - \frac{M}{N}$ equal to 1/2 are discussed. The codeword blocklength is set to 4608 bits, that is 2304 symbols over $GF(4)$ and 1152 symbols over $GF(16)$. The average column weight is set to 2.5. The variable-node degree distribution is $\lambda_2 = \lambda_3 = 0.5$, where $\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1}$, and d_v is the maximum symbol-node degree.

The LDPC codes have been constructed using quasi-regular PC matrices [23] generated by a modified PEG algorithm [18] that maximizes mSD [38]. The maximum number of iterations for the MP based decoders has been set to 50.

Figures from 4.2 up to 4.5 show the simulation results for the 16-ary LDPC coded architectures while Figures from 4.6 up to 4.9 show the simulation results for the 4-ary LDPC coded ones. Puncturing is performed in order to obtain transmission rates R_P equal to 3/5 and 2/3. The empirical cross-correlation

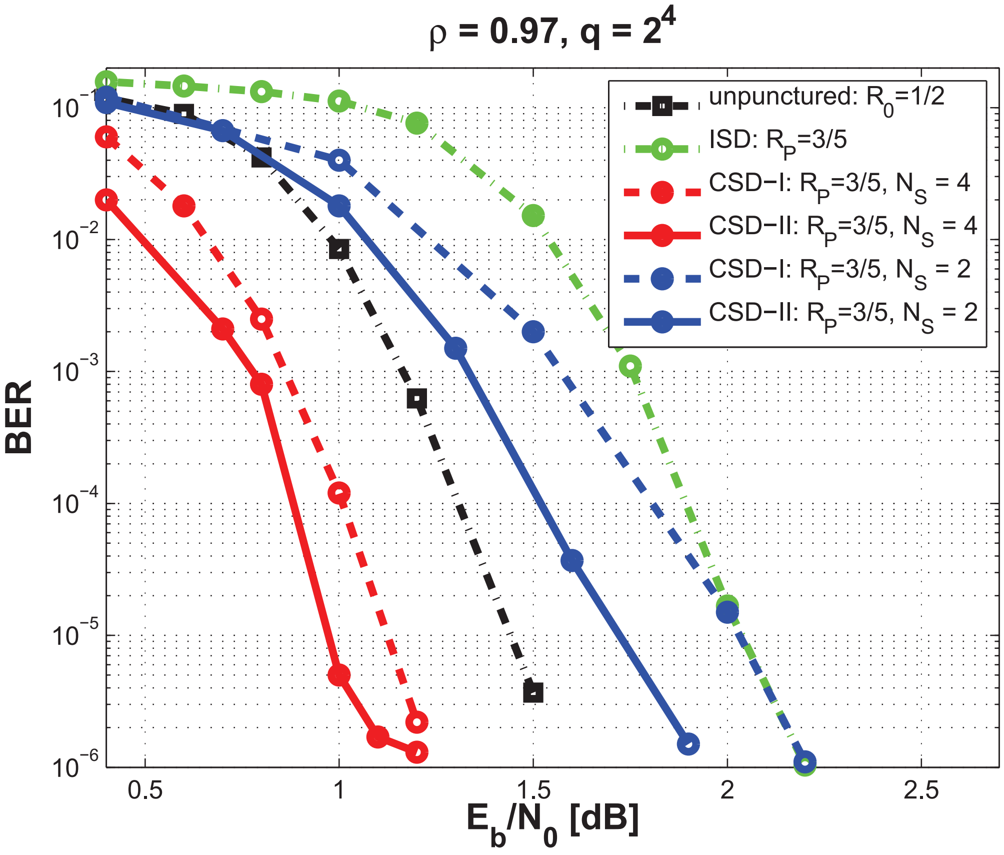

Figure 4.2: Bit Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size, ρ is the empirical cross-correlation introduced in Section 4.1.1, N_S is the number of streams that have been employed.

ρ, introduced in Section 4.1.1, is set to 0.97, that leads to an empirical cross-correlation calculated on the codewords ρ^{CW} having a value about 73 %. Figure 4.14 shows the empirical cross correlation that has been calculated on the whole codeword ρ^{CW} as a function of the empirical cross correlation that has been calculated on the information part of the codeword only ρ. It is proper to remind that the behavior of ρ^{CW} strictly depends on the PC matrix H that has been chosen. However, Figure 4.14 shows how the value of the empirical cross correlation actually decreases as it is calculated on the whole codeword.

In Figures from 4.10 up to 4.13, BER and FER simulated results, obtained

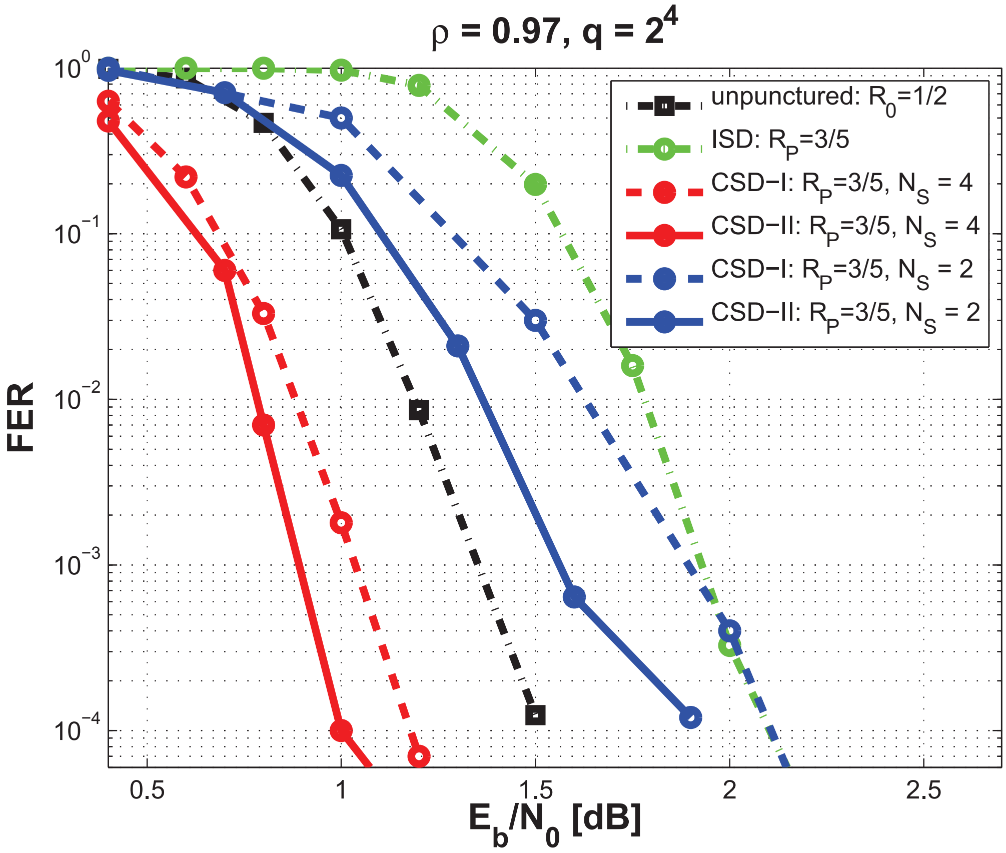

Figure 4.3: Frame Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size, ρ is the empirical cross-correlation introduced in Section 4.1.1, N_S is the number of streams that have been employed.

with different source correlations, are compared. The BER/FER curves represent the average vlaue of the N_S stream decoding error-rates.

The architectures employing CSD-II outperform those using CSD-I especially for the 16-ary case, using 4 streams. Comparing the proposed encode-and-decoding scheme with the punctured ISD, both CSD method provides significant gains and the best reults have been obtained with CSD-II with $N_s = 4$, where a significant gain, similar to those shown in [73]- [75], is reached with respect to the independent decoding of the unpunctured mother-code.

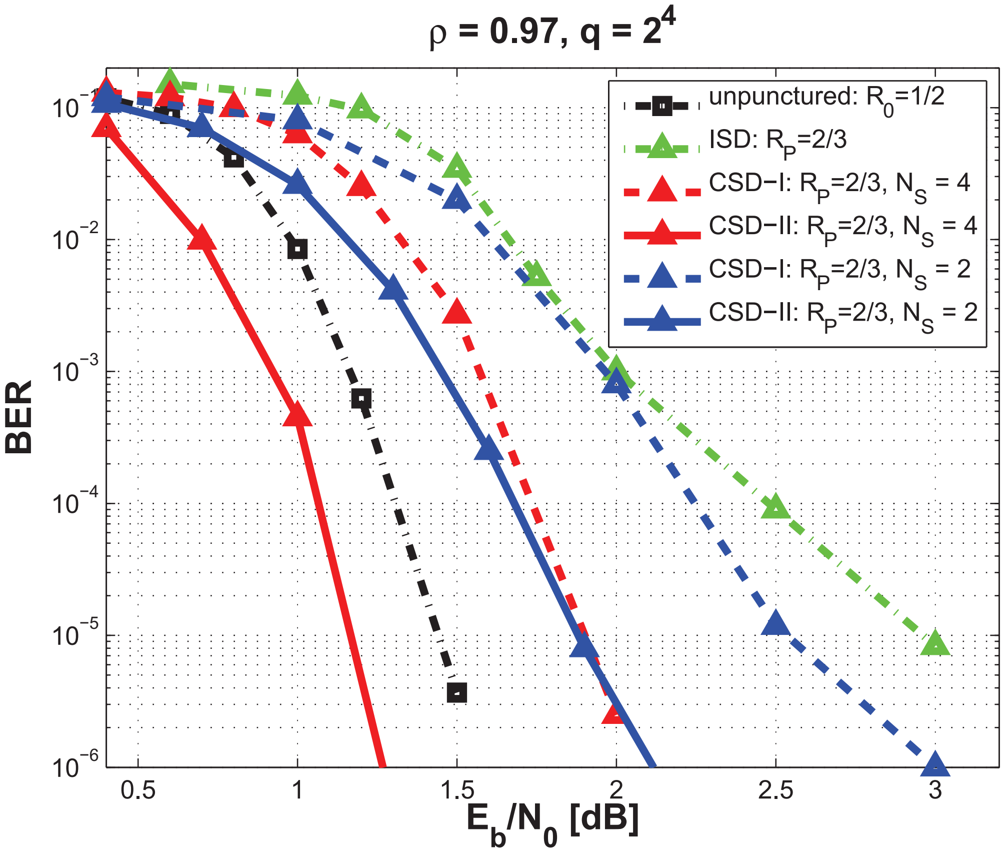

Figure 4.4: Bit Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size, ρ is the empirical cross-correlation introduced in Section 4.1.1, N_S is the number of streams that have been employed.

4.1.4 Conclusions

Two decoding structures employing RC-LDPC codes have been introduced and analyzed in order to study their bit-error-rate performance on the transmission of correlated sequences.

Simulation results for q-ary LDPC codes, with $q = 4$ and 16 show that the best performance can be achieved by using an architecture whose decoding is correlation based. Moreover, it has been observed that in general a 16-ary LDPC coding provides better performance with respect to the 4-ary case.

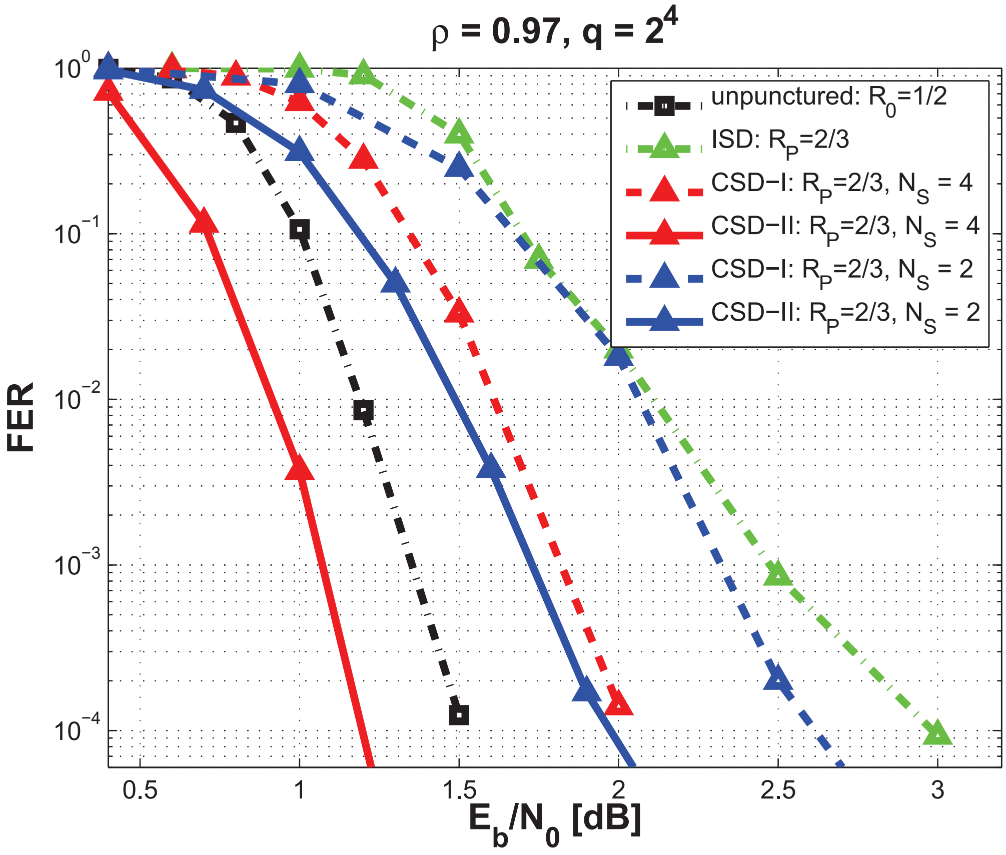

Figure 4.5: Frame Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size, ρ is the empirical cross-correlation introduced in Section 4.1.1, N_S is the number of streams that have been employed.

These interesting results have been obtained with the same decoder complexity of the standare MP decoding of non binary codes. This in an important difference with respect to previous works [73] - [75], where a turbo scheme is employed to exploiting the high mutual correlation of the received signals. This is allowed by using the joint coding with different puncturing patterns.

Ongoing research that promises efficiency at the receiver includes the analysis of novel decoders in which the correlation influence is furthermore highlighted in the MP structure. Future directions for research could focus on the behavior of the proposed architectures over different channels and with

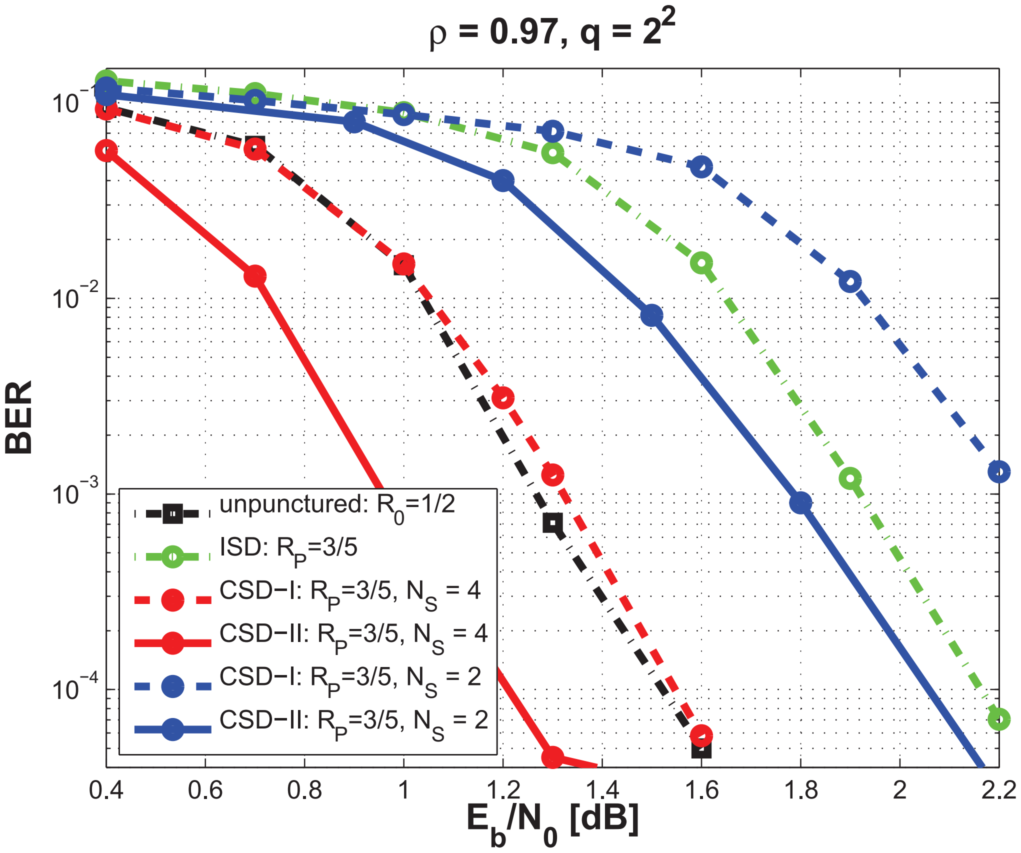

Figure 4.6: Bit Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size.

different LDPC codes, having different codeword length or degree-distribution profile as in [35]. Further, an optimization of the construction of LDPC codes has to be taken into account, starting from the results provided in [26].

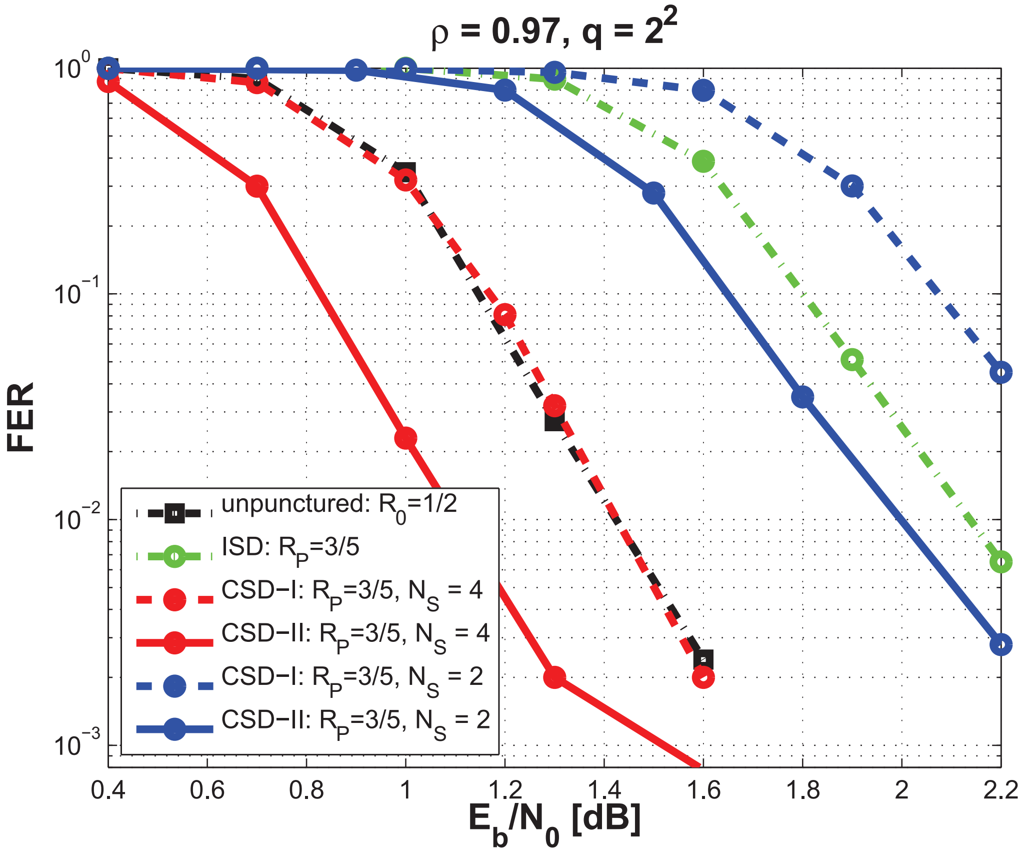

Figure 4.7: Frame Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size.

4.2 Challenges and opportunities for information theory-based design of Phase Change Memories

Technological improvements in VLSI circuits have allowed an ever increasing capability in data processing, that is closely related to the need for fast read/write storage devices. Thus, the development of non-volatile memories (NVMs) is one of the most active and challenging fields for research in microelectronics. Phase-Change Memory represents one of the most promising non-

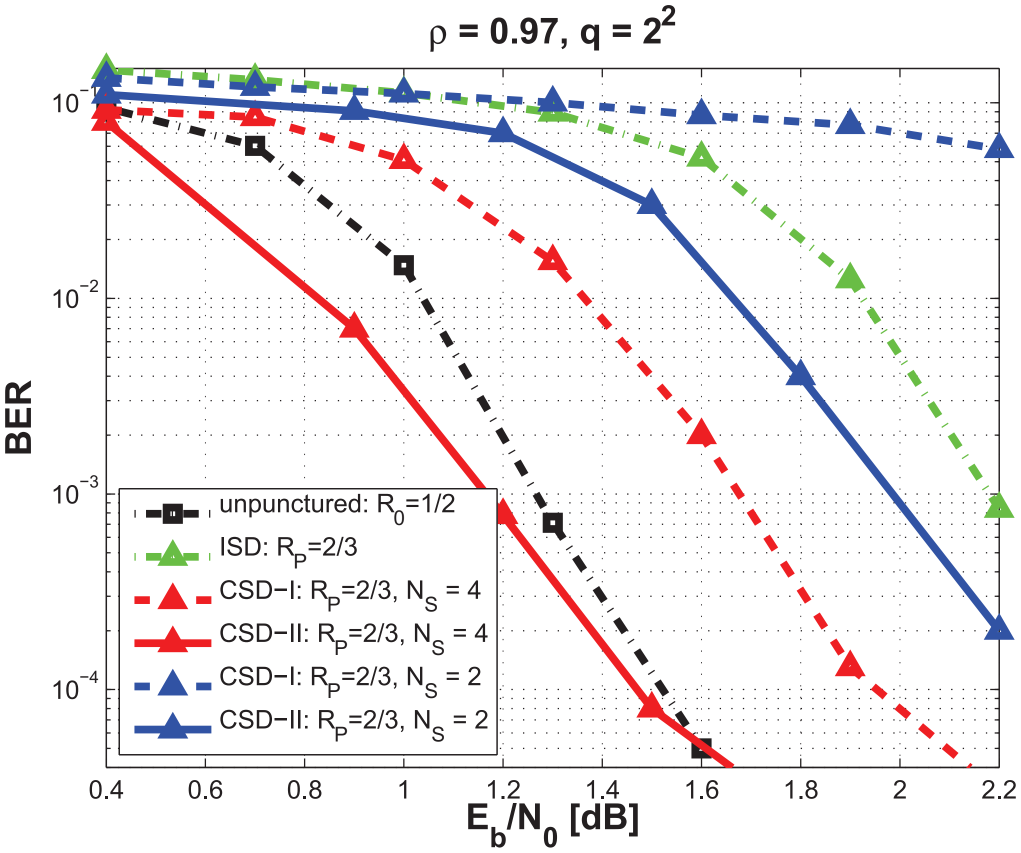

Figure 4.8: Bit Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size.

volatile semiconductor storage technologies to be used in the next decade [79]. Specifically, the microelectronics community aims at employing the aforesaid memories in order to replace standard floating-gate based Flash devices. In fact, when compared to Flash memories, PCMs can improve the performance in terms of endurance, write/read throughput, bit granularity, and scalability.

The PCM storage element is a thin chalcogenide layer contacted by a bottom semi-metallic heater and a top metal layer [80]. This storage element works as a variable resistor. Indeed, the used chalcogenide material shows a low-field resistivity that changes by orders of magnitude depending on its structural state (or, phase). Specifically, fast electrical pulses can change the

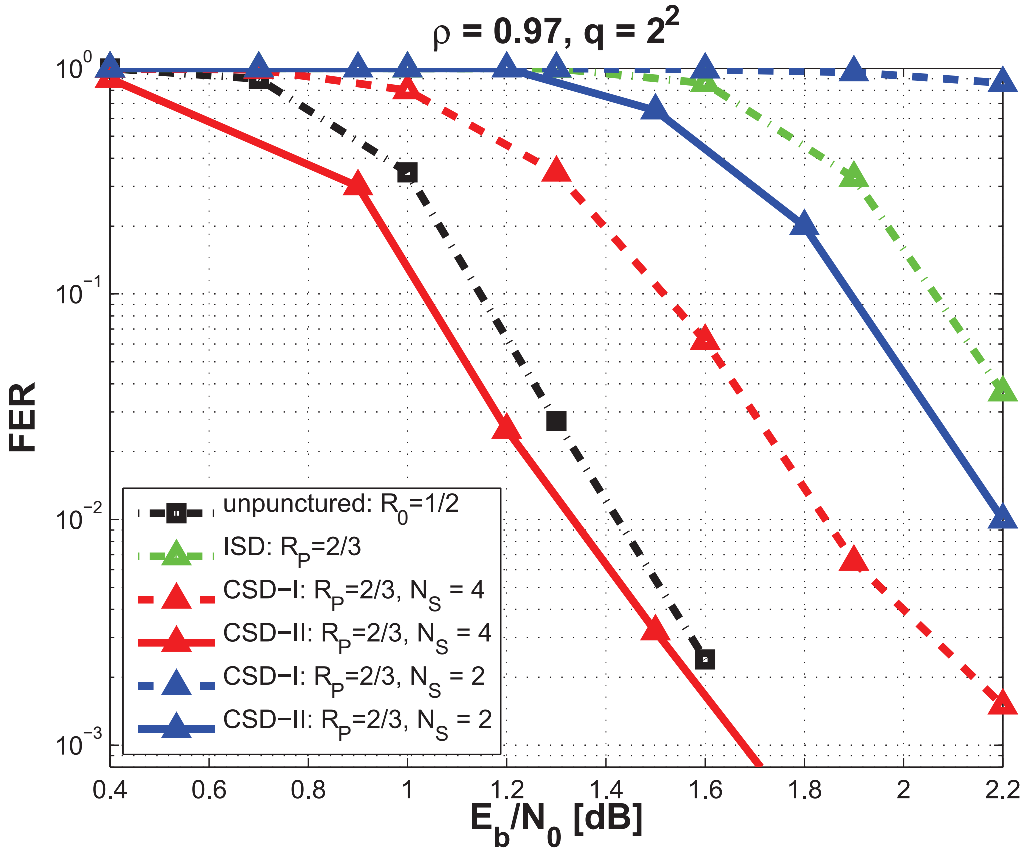

Figure 4.9: Frame Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size.

phase of a small portion of the chalocogenide layer (referred to as the active volume) between a low-conductance (that is, amorphous) state and a high-conductance (that is, crystalline) state. A complete study of PCM feasibility assessment has to take into account several items, such as:

- integration with CMOS technology;
- reduction of power consumption;
- scaling perspectives;
- reliability;

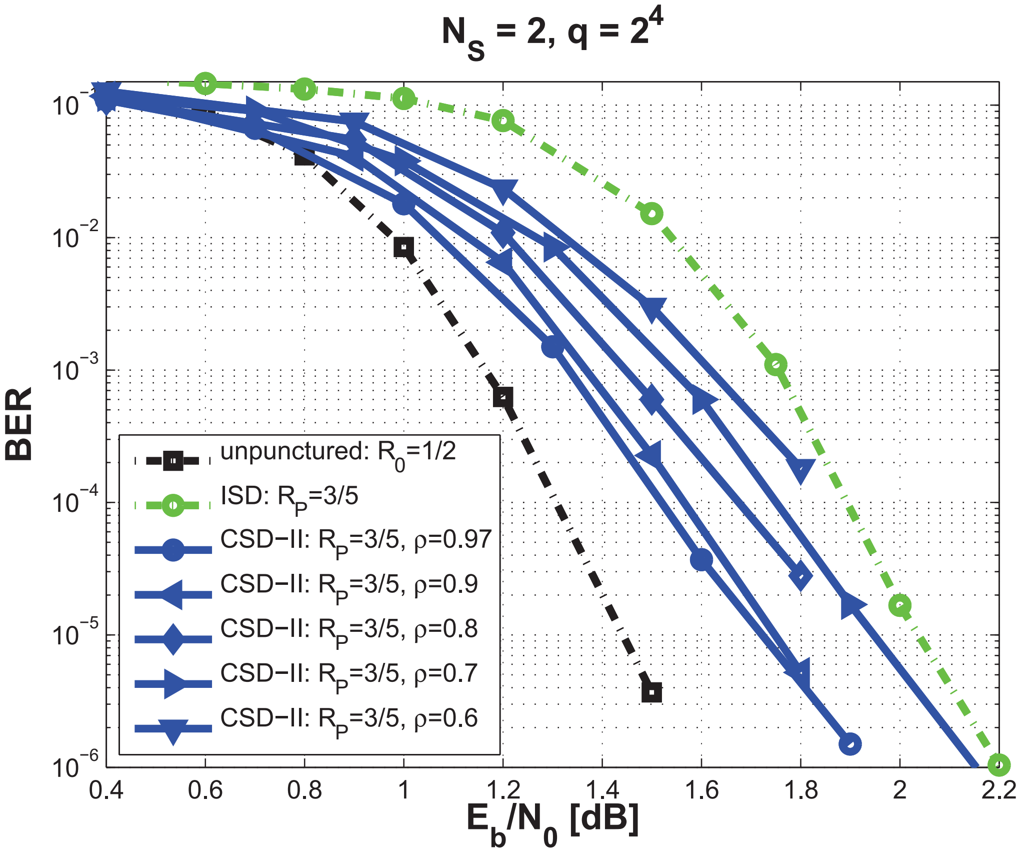

Figure 4.10: Bit Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size.

- disturbance characteristics.

The last item is definitely the most relevant and interesting from an information theory point of view. In fact, although PCMs apparently have an extremely strong noise resilience, information theory can still help improving and optimizing their performance. The quality of the interface between the heater and the chalcogenide layer and the robustness to spurious voltage transients are basic parameters to prove the manufacturability and the reliability of the memory cell. Analyzing PCMs from an information theory point of view may help in this respect.

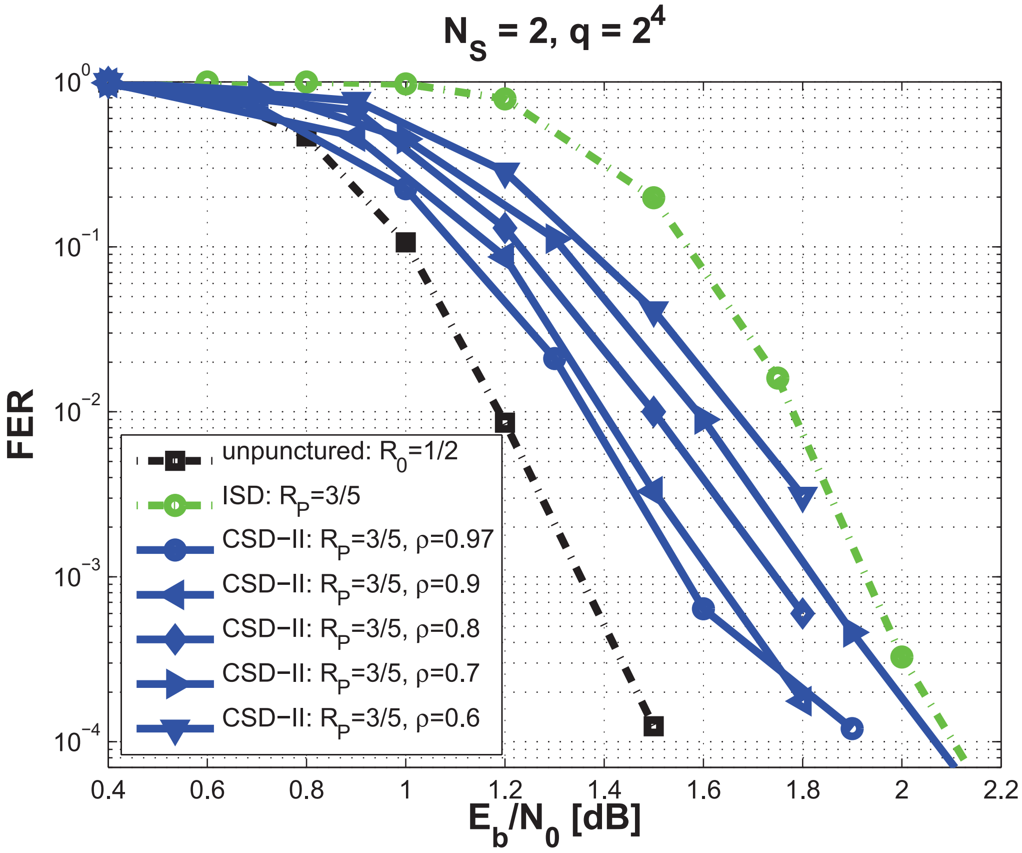

Figure 4.11: Bit Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size.

The nature of disturbances affecting PCMs may be described by means of several parameters. Among these, the following ones play a key role in the PCM capability of storing information:

- the variability of the cell geometry and the chemical composition of the phase change material;
- the non-deterministic programming behavior, or “write noise”;
- the noisiness of electrical parameters.

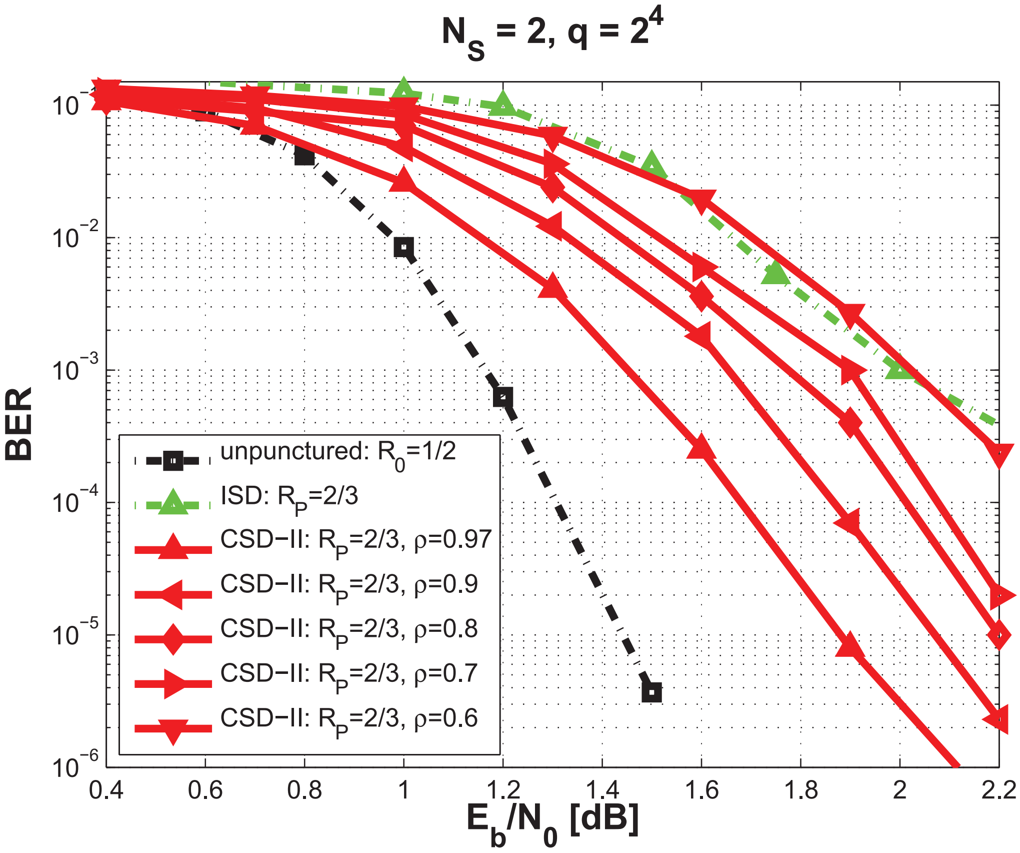

Figure 4.12: Bit Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size.

The last two points have been recently addressed in the literature. Specifically, one paper [81] investigates the relationship between programming algorithms and the amount of information that can be stored, in average, in a rewritable memory cell. Another work [82] studies the noisiness of electrical parameters in order to explain how the geometrical distribution of the amorphous material in the active chalcogenide volume affects the capability of a cell to store data.

Moreover, an information-theoretic analysis is useful as it can point out several aspects of the system, such as:

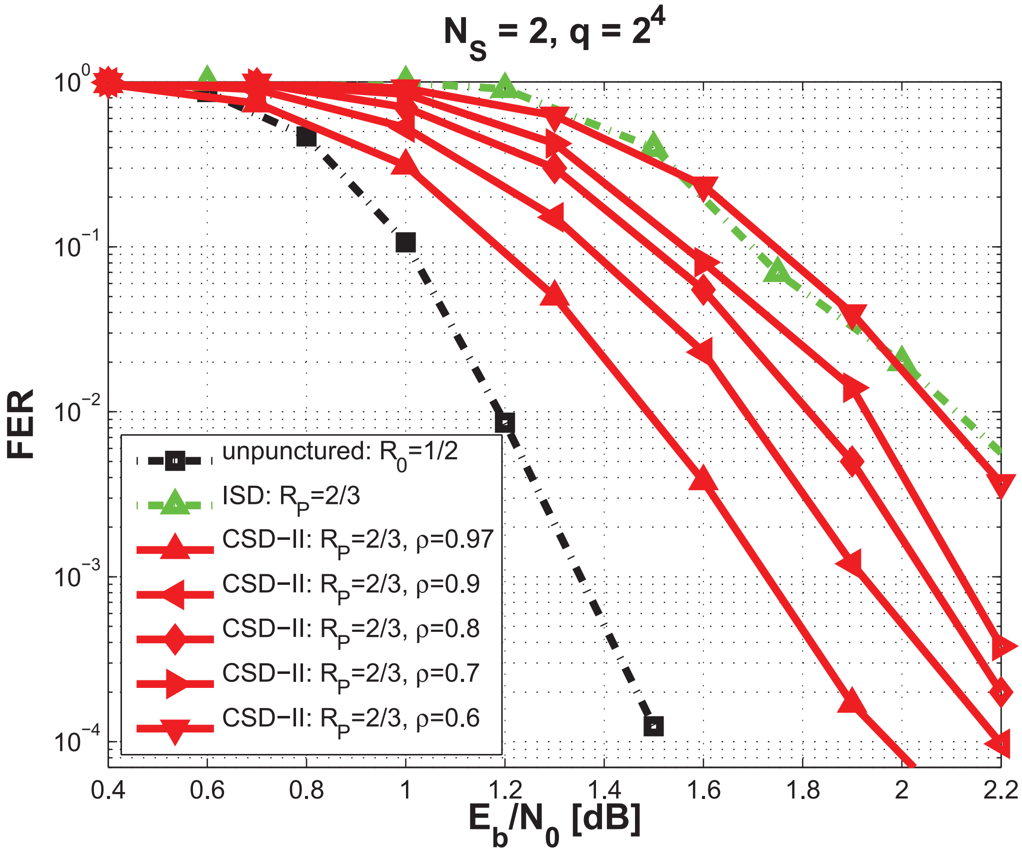

Figure 4.13: Bit Error Rate performance of the architectures discussed in Section 4.1.2: independent stream decoding (ISD) and cross stream decoding (CSD). R_P is the transmission rate, q is the alphabet size.

- definition of the channel;
- computation of the channel capacity;
- evaluation of the improvements brought by the application of error correcting codes (ECCs);
- optimization of the choices made within a modern coding theory approach (type of ECC, coding rate, alphabet size, etc.);
- proper design of the control system of the read/write interface;
- evaluation of the scalability potential.

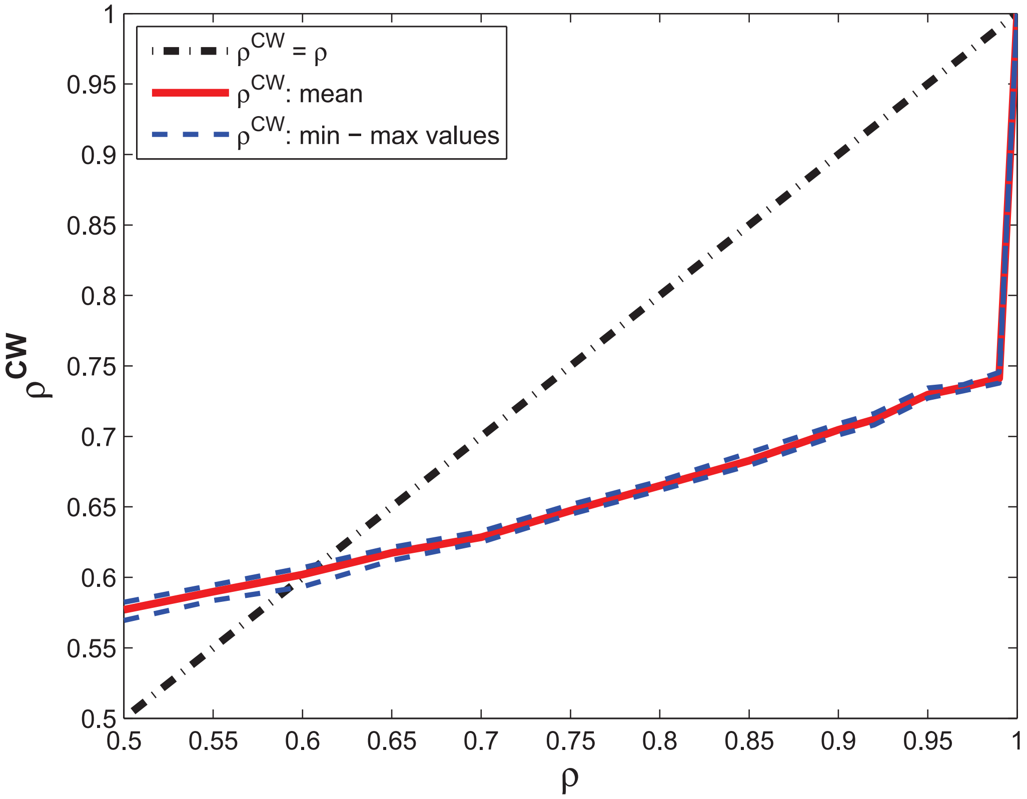

Figure 4.14: computed codeword correlation versus the information part correlation.

This section aims at providing the first results in this direction and describing challenges and opportunities for the design of the PCM cell and the reading architecture by means of an information theory-based approach. In particular, it deals with the fundamental parameters of the PCM physics, which are related to the noise mechanisms within the memory array, and with the data acquisition architecture which also contributes to the overall system noise.

First of all, it is important to determine an adequate channel model in order to analyze the system. The channel model that apparently well fits the PCM information acquisition is a generalized binary erasure channel (gBEC). Therefore, the expression of mutual information for a gBEC is provided in Section 4.2.1. Knowing this quantity is fundamental for an effective information theory-based PCM design. In fact, the definition of the relationships among the parameters affecting the mutual information is necessary to draw the directives for optimizing the PCM reading process performance. Experimental results in Section 4.2.2 provide a validation of the proposed channel model and give a major overview of the system, highlighting the key points for an information theory-based optimization. Further, these results provide a base on which a framework aiming at optimizing the design of the PCM array and providing a deeper investigation of the system can be stated. Final remarks conclude the section.

4.2.1 System model

In this section, it is assumed that the reading process is performed by sensing the value of the current through the selected cell (and, hence, through the selected bit line), which depends on the phase of the active chalcogenide volume [80].

Let I_R be the value of the current acquired in the reading process and I_P be the corresponding value in the case of an ideal PCM system. In the case of noiseless reading, let a stored bit 1 be represented by a high value of I_P, that is, $I_P = I_H$, and a stored bit 0 be represented by a low value of I_P, that is, $I_P = I_L$.

The reading process of a single cell in a Phase Change Memory can be schematically described as consisting of three different steps (Figure 4.15) [83, 84].

- In the first step (time duration T_P), a predetermined read voltage V_{rd} is applied to the addressed bit line and, hence, to the selected i-th cell. The bit line current increases from its initial value (0) towards the value V_{rd}/R_{cell}, where R_{cell} is the cell resistance. In addition, the sense circuit is equalized and preset to its optimal bias conditions.

- The read current is left free to settle for a predetermined time interval and is then acquired by the sensing circuit (total time duration T_T). Thus, the value of current I_R is detected after a time $T_P + T_T$. The content of the selected cell is determined by comparing I_R with a threshold current I_{th}, which is usually chosen as the mean of the nominal values associated to state 1 (I_H) and state 0 (I_L), that is, $I_{th} = (I_H + I_L)/2$.

- The reading current is finally set back to 0 and the sense circuit is disabled in a time T_B. After that, the reading process can restart for the next ((i+1)-th) cell. Therefore, the reading process of the i-th cell takes a time $T_{R_i} = T_R = T_P + T_T + T_B$.

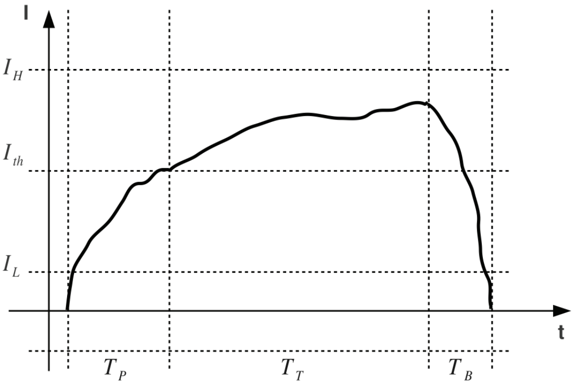

Figure 4.15: Schematic waveform for the reading process of Phase Change Memories.

Given the previous description of the PCM reading process that has been considered in this section, the PCM readout signal is not affected by resistance drift [85] or intersymbol interference. Therefore, the related channel model has to be memoryless. However, it is necessary to properly define the signal-to-noise ratio in terms of physical parameters that can affect the output waveform. It is worth to highlight that the definition of SNR for PCMs is a really challenging topic for the data storage community. To the best of the author's knowledge, there is no paper dealing with this task. In this section, the goal is not to find the expression of SNR but rather to define a simplified channel model for PCMs, in order to give the first directives for an effective information theory-based design of the PCM reading architecture. It is possible to define a channel model by taking the non idealities of the reading process into account. Specifically, the output of the reading process expressed in terms of bits, $y(t)$, could be more manageable from an information theory point of view by introducing an uncertainty zone in the acquired current value. Let I_{R_i} be the acquired current when reading the i-th cell. In the case when a hard decision is performed, the i-th sample, y_i, of the readout signal $y(t)$ corresponding to the stored data can be written as follows:

$$y_i = \begin{cases} 1 & \text{if } I_{R_i} \geq I_{th} + \Delta_I \\ \epsilon & \text{if } I_{th} - \Delta_I < I_{R_i} < I_{th} + \Delta_I \\ 0 & \text{if } I_{R_i} \leq I_{th} - \Delta_I \end{cases} \tag{4.8}$$

where ϵ represents the state of uncertainty: if the output is in the ϵ state, the reading system can not safely choose which value to assign to that given sample y_i. If the reading process is noiseless, Δ_I is set to 0. Thus, $2\Delta_I$ can be considered as the width of the uncertainty state, or, in other terms, as the sensitivity of the reading architecture.

Given the aforesaid assumptions, the readout signal expressed in (4.8) can be associated to a generalized binary erasure channel. gBECs have been studied especially for the decoding analysis of LDPC codes [23]. Let X be the input alphabet and Y be the output alphabet. In this case, $X = \{0, 1\}$ and $Y = \{0, \epsilon, 1\}$. Moreover, given $x \in X$, let $p_0 = p(x = 0)$ and $p_1 = p(x = 1)$, $p_0 = 1 - p_1$, be the input probabilities. The transition probability matrix $Q = \{q_{ij}\}_{i=0,\ldots,|Y|-1; j=0,\ldots,|X|-1}$ associated to the gBEC can be written as follows:

$$Q = \begin{bmatrix} 1-\phi_0 & g_1 \\ f_0 & f_1 \\ g_0 & 1-\phi_1 \end{bmatrix} \tag{4.9}$$

Each term q_{ij} represents the probability that the given readout symbol is

$y_i \in Y$, while the stored data is $x_j \in X$: that is, $q_{ij} = p(y = y_i|x = x_j)$. In other terms, since the stored data live in $GF(2)$, $p(y = i|x = i) = 1 - \phi_i$, $p(y = i|x = j) = g_j$, $p(y = \epsilon|x = i) = f_i$ $\forall i, j \in \{0, 1\}, i \neq j$. ϕ_i is set to $f_i + g_i$ $\forall i \in \{0, 1\}$.

Thus, the mutual information $I(X; Y)$ [24] of the gBEC is the following:

$$\begin{aligned} I(X;Y) = H(X) - H(X|Y) = \\ = H_2(p_0) - \sum_{(i,j)\in\{0,1\}^2, i\neq j} p_i[(1-\phi_i)\log_2(1+\frac{p_jg_j}{p_i(1-\phi_i)})+ \\ +g_i\log_2(1+\frac{p_j(1-\phi_j)}{p_ig_i}) + f_i\log_2(1+\frac{p_jf_j}{p_if_i})] \end{aligned} \tag{4.10}$$

where $H_2(z) = -[z\log_2(z) + (1-z)\log_2(1-z)]$.

The influence of the transition probabilities is therefore very strong on the gBEC capacity. Even when each f_i and each g_i assume small (non zero) values, the maximum fraction of information that can be reliably recovered can be much degraded. Specifically, (4.10) shows that the mutual information decreases faster with increasing g than with increasing f.

Furthermore, in the case when the channel is symmetric, i.e. $f_0 = f_1 = f$, $g_0 = g_1 = g$ and $\phi_0 = \phi_1 = \phi$, the mutual information expressed in (4.10) is maximized when $p_0 = p_1 = \frac{1}{2}$. In that case, the channel capacity C can be written as follows:

$$C = 1 - [(1-\phi)\log_2(1+\frac{g}{1-\phi}) + g\log_2(1+\frac{1-\phi}{g}) + f] \tag{4.11}$$

Figure 4.16 shows C as a function of f and g. Specifically, in case $g = 0$, i.e. the channel becomes a standard binary erasure channel, $C = 1 - f$. On the other hand, if $f = 0$, the channel becomes a binary symmetric channel, that is $C = 1 - H_2(g)$. Further, $C = 0$ if $g = \frac{1}{2}(1 - f)$.

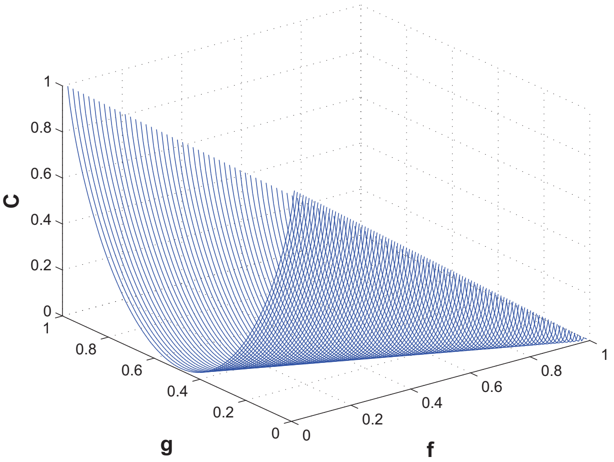

Figure 4.16: Generalized symmetric binary erasure channel capacity C as a function of $f = p(y = \epsilon|x = 0) = p(y = \epsilon|x = 1)$ and $g = p(y = 1|x = 0) = p(y = 0|x = 1)$

4.2.2 Experimental results

In this Section, the results described in Section 4.2.1 are used to analyze an experimental PCM cell array.

In particular, the read current tracks presented in this Section were obtained from a PCM test chip integrated in 180-nm CMOS technology [80]. The simplified read path is shown in Figure 4.17. NMOS transistor Y_O is characterized by its threshold voltage, V_{th}, and its transconductance parameter β, which takes its sizes as well as its electrical properties into account. Specifically, as $\beta \downarrow 0$, the area of Y_O gets smaller, modeling the scaling process. During a read operation, Y_O works in its saturation region, where the following relationship between its drain current I_D and its gate-to-source voltage $V_{GS} = V_G - V_S$ (V_G and V_S being its gate and source voltage, respectively) holds [25]

$$I_D = \frac{\beta}{2}(V_G - V_S - V_{th})^2 \tag{4.12}$$

Transistor Y_O then operates as a source follower, thereby determining the

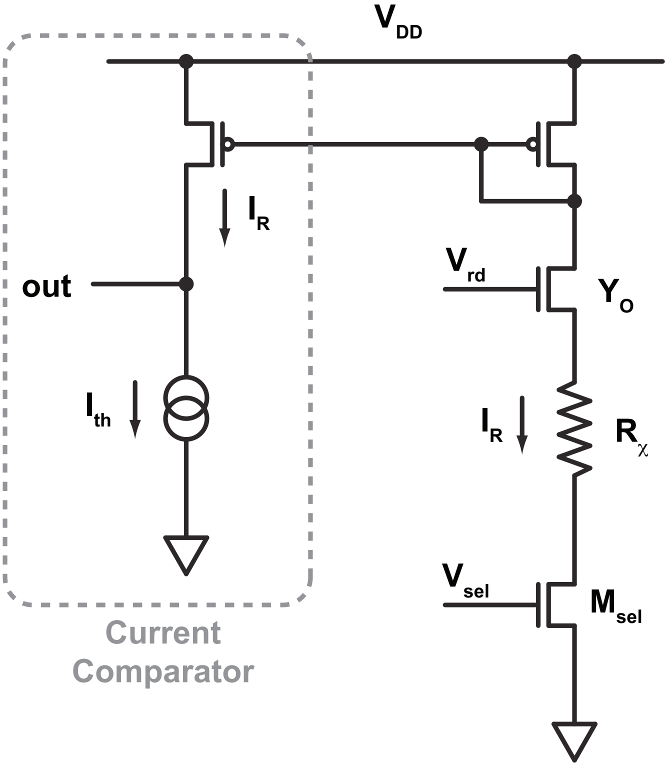

Figure 4.17: Simplified read path for the considered PCM chip. The bit-line select devices are assumed to have a negligible on-resistance and are not shown in the figure for simplicity. NMOS transistor M_{sel} is the word-line select device, and also has a negligible on-resistance. NMOS transistor Y_O works as a source follower, thus determining the voltage across the selected cell. I_R is the readout current.

voltage across the selected cell as

$$V_{rd} = V_G - V_{th} - \sqrt{\frac{2I_R}{\beta}} \tag{4.13}$$

When the PCM cell has to be read, the value of the voltage on the gate of Y_O is set to an adequate value V_G, and the ensuing read current $I_R = V_{rd}/R_{cell}$ is then acquired by the sense circuit.

In the following discussion, the readout signals are analyzed according to the observations provided in Section 4.2.1.

The parameters required to represent the PCM channel as a gBEC may

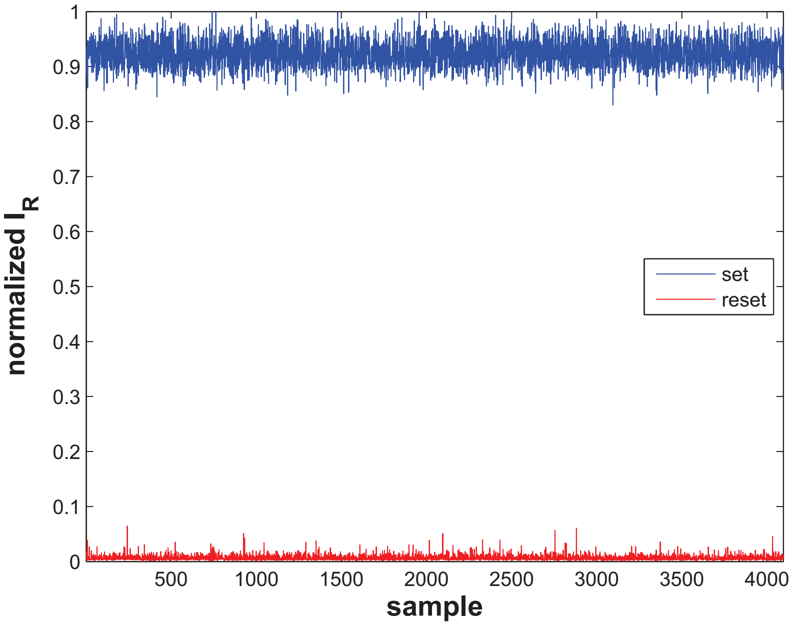

Figure 4.18: Samples of acquired current I_R obtained from the considered PCM array integrated in 180-nm CMOS technology. The reset track (lower, red) is related to cells programmed to 0 (high resistance), while the set track (upper, blue) is related to cells programmed to 1 (low resistance).

be obtained by a long enough sequence of readings in a PCM cell array. The values of f_i and g_i in (4.10) can be calculated as:

$$\begin{aligned} f_0 &= \int_{I_{th}-\Delta_I}^{I_{th}+\Delta_I} p(I_R = I'|I_P = I_L)dI' \\ f_1 &= \int_{I_{th}+\Delta_I}^{I_{th}-\Delta_I} p(I_R = I'|I_P = I_H)dI' \\ g_0 &= \int_{I_{th}+\Delta_I}^{+\infty} p(I_R = I'|I_P = I_L)dI' \\ g_1 &= \int_{-\infty}^{I_{th}-\Delta_I} p(I_R = I'|I_P = I_H)dI' \end{aligned} \tag{4.14}$$

It is worth to remind that the minimum value of Δ_I is 0, whereas its maximum value is $I_H - I_{th} = I_{th} - I_L = \frac{I_H - I_L}{2}$. Moreover, it is important to note that the channel symmetry condition does not hold for any value of Δ_I so far. In fact, from Figure 3, the variance of the set readout track is much higher than that of the reset readout track. Therefore, the values of f_0

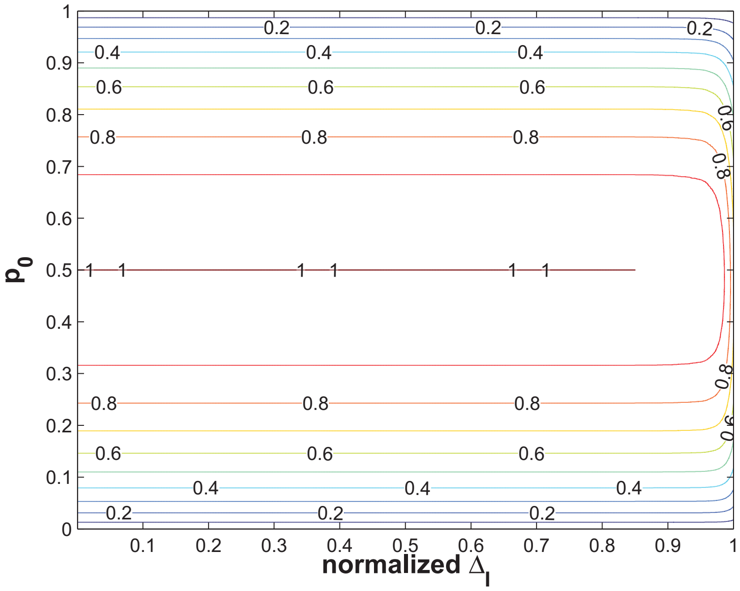

Figure 4.19: Contour plot of the mutual information as a function of the bit-0 input probability, p_0, and the normalized half-width of the uncertainty state, Δ_I.

and g_0 are usually largely different from the values of f_1 and g_1, respectively. Thus, depending on the value of the half-width, Δ_I, of the uncertainty state, the expression of the transition matrix in (4.9) may vary. Depending on the form of the transition matrix, the expression of the mutual information will vary according to the value of Δ_I as well. The mutual information in (4.10) is a function of the input probability profile and, hence, from (4.12) and (4.14), assumes different values for each pair (Δ_I, β) depending on the values of p_0 and p_1. Moreover, as Δ_I and β vary, the maximum value of the achievable mutual information can be approached by adequately modifying the input probability profile.

The mutual information behavior can give useful directives for a performance optimization of PCM technology. For instance, it can provide information on the width $2\Delta_I$ that can be tolerated to achieve an acceptable error rate. Since the values of I_H and I_L depend on the gate-to-source voltage of source follower device Y_O, the value of β strongly affects the fraction of information that can be reliably recovered.

As β increases, the PCM technology noise resilience gets stronger. This property is highlighted in Figure 3, where samples of set and reset readout tracks of a PCM array having a value of β large enough are shown. The reset track is plotted in red color, while the set track is plotted in blue color. It is worth to remind that a cell programmed to the reset state (hereinafter referred to as reset-programmed cell) corresponds to a stored bit equal to 0, while a cell programmed to the set state (hereinafter referred to as set-programmed cell) corresponds to a stored bit equal to 1. That is, according to the notation introduced in Section II, $I_P = I_L$ for a reset-programmed cell, while $I_P = I_H$ for a set-programmed cell. Figure 4 shows the contour plot of the mutual information as a function of the normalized half-width of the uncertainty state, Δ_I, and the bit-0 input probability, p_0.

Figure 5 shows the pair (Δ_I, p_0) for which the maximum mutual information is reached. The aforesaid figures further highlight the noise resilience of Phase Change Memories.

However, the value of the readout current I_R depends on the parameters of Y_O that drives the selected bit line. In fact, from (4.12), after simple algebraic manipulations, the value of the readout current, I_P, in the case of a noiseless system can be expressed as follows:

$$I_P(\beta, R_\chi) = \frac{V_G - V_{th}}{R_\chi} + \frac{1}{\beta R_\chi^2}(1 - \sqrt{1 + 2\beta R_\chi (V_G - V_{th})}) \qquad (4.15)$$

where R_χ, $\chi \in \{set, reset\}$ is the resistance of a set-programmed and a reset-programmed PCM cell, respectively. Therefore, for a given value of β, I_P assumes two different values, depending on whether the PCM cell has been programmed to the set or the reset state. From (4.13), it is apparent that the value of V_{rd} decreases with increasing values of I_P (i.e., with decreasing values of R_χ), which leads to a smaller spacing between the read currents corresponding to the two programmed states.

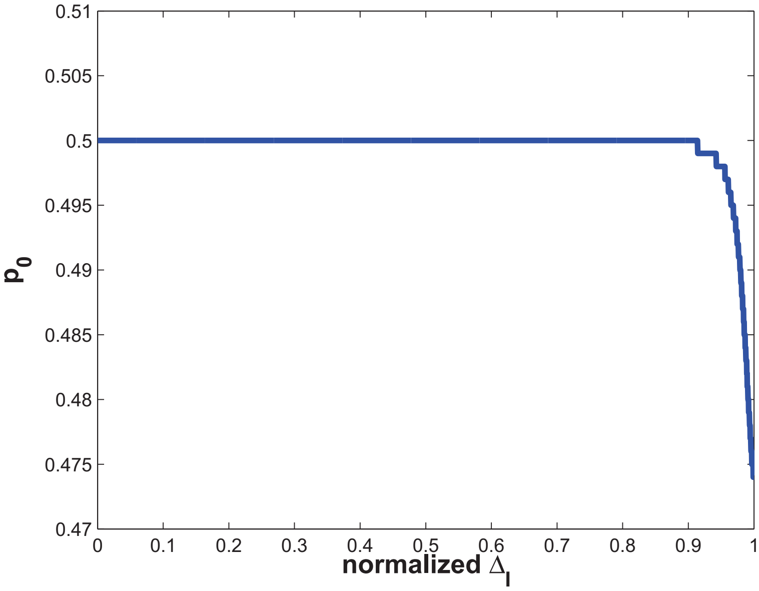

Figure 4.20: Plot of the pair (Δ_I, p_0) for which the maximum value of the mutual information in (4.10) is reached. Δ_I is the half-width of the uncertainty state, whereas p_0 is the bit-0 input probability.

Since the noisy system is assumed to be memoryless, I_R is then computed by adding a zero-mean Gaussian noise. As previously introduced in this Section (Figure 3), the noise variance depends on R_χ, being smaller for larger values of the latter. Furthermore, since $|I_P(\beta, R_\chi)| \xrightarrow{\beta\downarrow 0} 0$, it is possible to assume that the noise amplitude depends on β as well.

From (4.15), as the value of β decreases, the readout current associated to a set-programmed cell gets closer to that associated to a reset-programmed cell. This effect implies a dramatic degradation of the maximum fraction of information that can be reliably recovered, as it is apparent in Figure 4.21. This figure shows that, as β approaches 0, $I(X;Y)$ is maximum for specific values of the half-width, Δ_I, of the uncertainty state. In other terms, Δ_I has to be set to a value such that $g_i \to 0$ and $1 - \phi_i > \frac{1}{2}$ $\forall i \in \{0,1\}$. Furthermore, it is worth to note that the range of values of Δ_I for which the maximum value of $I(X;Y)$ is reached gets smaller as $\beta \to 0$.

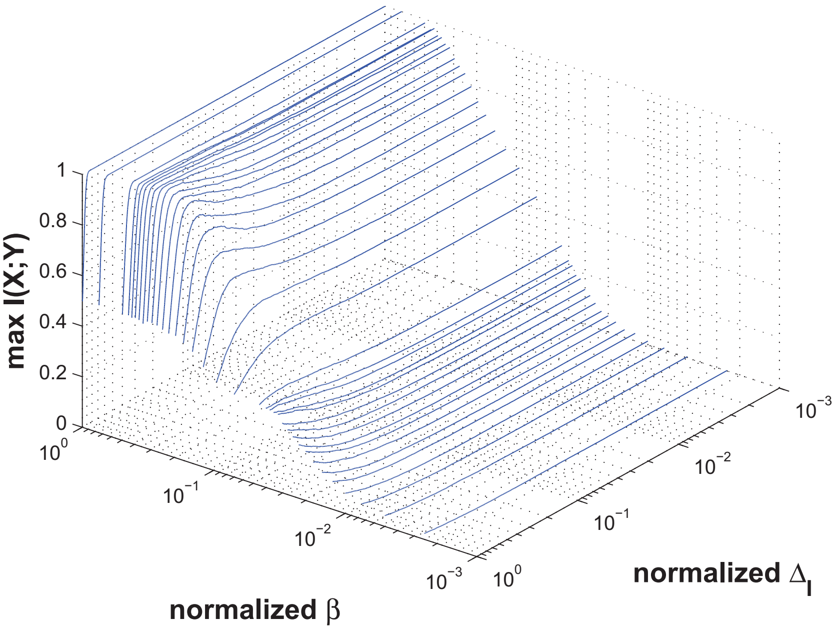

Figure 4.21: Maximum value of the mutual information as a function of the normalized half-width Δ_I of the uncertainty state and the normalized value of the parameter β characteristic of the NMOS transistor Y_O that drives the bit line.

Figure 4.22 shows the value of the 0-bit input probability p_0 for which the maximum of $I(X;Y)$ is reached for given values of β and Δ_I.

It is important to note that, as β decreases from its maximum value, p_0 increases in average for every Δ_I. This behavior highlights that a reset-programmed PCM cell provides a smaller contribution to the noise of the system than a set-programmed PCM cell. However, for β lower than a given $\overline{\beta}$, the average value of p_0 approaches 0.5 for every value of Δ_I. In fact, since $|I_P(\beta, R_{set}) - I_P(\beta, R_{reset})| = |I_H(\beta) - I_L(\beta)| \xrightarrow{\beta\downarrow 0} 0$, the values of I_P associated to a set-programmed cell and a reset-programmed cell get closer to each other. Therefore, $1 - \phi_i \xrightarrow{\beta\downarrow 0} 0 \; \forall i \in \{0, 1\}$, that is, $I(X;Y) \xrightarrow{\beta\downarrow 0} 0$ as well. Thus, when β is really close to 0, $I(X;Y)$ is maximized if and only if the input probabilities are chosen as uniform as possible. It is worth to point out that the analysis of the PCM system for $\beta \downarrow 0$ is important from a scaling perspective.

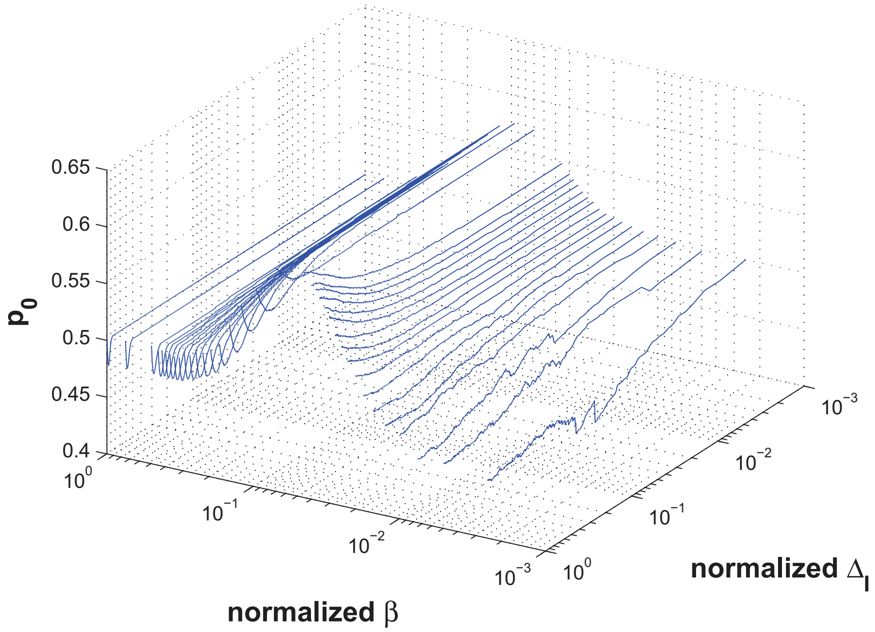

Figure 4.22: Bit-0 input probability for which the maximum value of the mutual information is reached as a function of the normalized half-width Δ_I of the uncertainty state and the normalized value of the parameter β characteristic of the NMOS transistor Y_O that drives the bit line.

Furthermore, the above results provide useful directives for the PCM system design. In fact, for a given PCM array, the maximum of the mutual information expressed in (4.10) (i.e., the maximum of the reading process reliability) can be approached by means of a proper combination of the electrical characteristics of the reading architecture, the sensitivity of the electronic devices involved in the process, and the input probability profile of the stored data, even in a resistance drift-free and ISI-free environment.

4.2.3 Conclusions

This section introduces an information theory-based analysis of Phase Change Memory technology. According to the procedure for reading PCM cells, the generalized binary erasure channel apparently fits the requirements and the properties of the PCM reading process. Closed-form expressions of the gBEC mutual information and experimental results are given as well.

Specifically, relationships between information recovery performance, electrical parameters, and sensitivity of the reading architecture are provided. The analysis that has been performed may play an important role for future developments of an information theory-based design of PCMs, especially in a scaling perspective.

Further, q-ary LDPC codes can definitely be employed as ECC for PCMs. In fact, an ECC employed in PCM control system has to exhibit the following properties:

- very high coding rate;
- extremely low error floor;
- very good performance in the waterfall region.

Thus, the q-ary LDPC code design introduced in sections 2.2 and 2.3 fits to the aforesaid requirements.

Chapter 5

Conclusions

The main course of this Ph.D. program has been represented by proper development of design and decoding techinques of q-ary LDPC codes.

In Chapter 2 specific design methods for q-ary LDPC codes have been introduced. Specifically, it has been shown as q-ary LDPC codes may be suitable for applications with particular requirements as spectral efficiency, very good bit-error rate performance at low SNRs, extremely low error floor and powerful burst error correction capabilities.

Chapter 3 describes the main architectures for q-ary LDPC decoding. Specifically, it focuses on decoding performance in terms of convergence and computational complexity for PR channels. Further, it provides a brand new structure that aims at jointly perform detection and decoding.

Chapter 4 introduces some prospective scenarios for q-ary LDPC codes. However, wireless cooperative systems and PCM control system are not the only fields q-ary LDPC codes might be suitable for. In fact, it has been proved how q-ary LDPC codes may be extremely versatile and how proper q-ary LDPC code design and decoding may bring to very powerful performance.

Therefore, q-ary LDPC codes can definitely play a fundamental role in the future communication systems, from both academical and industrial points of view. On the other hand, optimization of q-ary LDPC code performance is still an open issue. Thus, q-ary LDPC codes definitely represent one of the most interesting research fields in communication engineering.

Appendix A

List of acronyms

ADC Analog-to-digital converter

APP A posteriori probability

AWGN Additive white Gaussian noise

BB Bit based

BCJR Bahl-Cocke-Jelinek-Raviv

BEC Binary erasure channel

BER Bit-error rate

BP Belief propagation

BPSK Binary phase shift keying

BSC Binary symmetric channel

CBD Coded bit density

CIR Channel impulse response

CSD Cross stream decoding

DDNP Data-dependent-noise-predictive

ECC Error correction code

EEPR4 Extended EPR4

EPR4 Extended class-4 partial response

FER Frame-error rate

FFT Fast Fourier Transform

FHT Fast Hadamard Transform

FIR Finite impulse response

FRC Full rank condition

gBEC Generalized binary erasure channel

HDD Hard-disk drive

ISD Independent stream decoding

ISI Intersymbol interference

LDPC Low-density parity-check

LDSM Linearly dependent set maximization

LLR Log-likelihood ratio

MAP Maximum a posteriori

ML Maximum likelihood

MLC Multilevel coding

MLSD Maximum likelihood sequence detector

MP Message-passing

MREBL Maximum resolvable erasure-burst length

mSD Minimum space distance

MSD Multistage decoding

NVM Non-volatile memory

OBBD Optimal subblock-by-subblock detector

PC Parity-check

PCM Phase Change Memory

PEG Progressive edge-growth

PID Parallel independent decoding

PMR perpendicular magnetic recording

PR Partial response

PR4 Class-4 partial response

QC Quasi-cyclic

qEC q-ary erasure channel

QPSK Quadrature phase shift keying

QR Quasi-regular

qSC q-ary symmetric channel

RC Rate-compatible

RLL Run-length-limited

RS Reed-Solomon

SB Symbol based

SCA Serially concatenated architecture

SFR Sector failure rate

SNR Signal-to-noise ratio

SOVA Soft-output Viterbi algorithm

TC Turbo code

TCA Turbo concatenated architecture

TCM Trellis coded modulation

TLR Turbo-like receiver

UBD User bit density

List of Figures

List of Tables

Bibliography

[1] R.G. Gallager, *Low-Density Parity-Check Codes,* Cambridge, MA, MIT Press, 1963.

[2] C. Berrou, A. Glavieux, P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in proc. *IEEE International Conference on Communications (ICC) 1993*, Geneve, CH, pp. 1064-1070.

[3] M.C. Davey, D. MacKay, "Low-Density Parity-Check Codes over GF(q)," *IEEE Communications Letters*, vol. 2, no. 6, June 1998.

[4] M.C. Davey, '"Error-Correction Using Low-Density Parity-Check Codes," Ph.D. Thesis, University of Cambridge, UK, December 1999.

[5] N. Wiberg, '"Code and decoding on general graphs," Dissertation no. 440, Dept. Elec. Eng., Linkoping Univ., Linkoping, Sweden, 1996.

[6] R.D. Wesel, X. Liu, C. Komninakis, and J.M. Cioffi, "Constellation Labeling for Linear Encoders," *IEEE Transactions on Information Theory*, vol. 47, no. 6, Sept. 2001, pp. 1572-1581.

[7] J. Hou, P.H. Siegel, L.B. Milstein, H.D. Pfister, "Capacity-Approaching Bandwidth-Efficient Coded Modulation Schemes Based on Low-Density Parity-Check Codes," *IEEE Trans. Inform. Theory,* vol. 49, no. 9, Sept. 2003.

[8] G. Ungerboeck, "Trellis coded modulation with redundant signal sets, part I," *IEEE Commun. Mag.,* vol. 25, pp. 5-11, Feb 1987.

[9] G. Ungerboeck, “Trellis coded modulation with redundant signal sets, part II,” *IEEE Commun. Mag.,* vol. 25, pp. 12-21, Feb 1987.

[10] U. Wachsmann, R.F.H. Fischer, J.B. Huber, “Multilevel Codes: Theoretical Concepts and Practical Design Rules,” *IEEE Trans. Inform. Theory,* vol. 45, no. 5, July 1999.

[11] H. Imai, S. Hirakawa, “A new multilevel coding method using error correctin codes,” *IEEE Trans. Inform. Theory* vol IT-23, pp.371-377, May 1977.

[12] A. Matache, C. Jones, R.D. Wesel, “Reduced Complexity MIMO Detectors for LDPC Coded Systems,” in proc. *2004 IEEE Military Communications Conference*, 31 Oct.-3 Nov. 2004, pp 1073-1079 .

[13] J. Huber, U. Wachsmann, “Capacities of equivalent channels in multilevel coding systems,” *Electron. Lett.*, vol. 30, pp. 557-558, Mar. 1994.

[14] Y. Kofman, E. Zehavi, S. Shamai, “Analysis of a multilevel coded modulation system,” in proc. *1990 Bilkent Int. Cond. New Trends in Communications, Control and Signal Processing*, pp. 376-382, Ankara, Turkey, July 1990.

[15] E.A. Krouk, S.V. Semenov, ”Low-Density Parity-Check Burst Error-Correcting Codes,” in proc. *2nd International Workshop Algebraic and combinatorial coding theory*, pp. 121-4, 1990.

[16] E.A. Krouk, S.V. Semenov, *Error Correcting Coding and Security for Data Networks*, Mac-Graw Hill.

[17] E. Eleftheriou, S. Õlçer, ”Low-Density Parity-Check Codes for Digital Subscriber Lines,” in proc. *IEEE International Conference on Communications, ICC 2002*, April 28-May 2, 2002.

[18] X.Y. Hu, E. Eleftheriou, D.M. Arnold, ”Regular and Irregular Progressive Edge-Growth Tanner Graphs,” *IEEE Trans. on Information Theory*, vol. 51, no.1, Jan. 2005.

[19] T. Tian, C. Jones, J. Villasenor, R.D. Wesel, “Selective avoidance of cycles in irregular LDPC code construction,” *IEEE Trans. on Comm.*, Aug. 2004, vol. 52, no. 8, pp. 1242-1247.

[20] B. Rong, T. Jiang, X. Li, M.R. Soleymani,”Combine LDPC Codes Over GF(q) With q-ary Modulations for Bandwidth Efficient Transmission,” *IEEE Trans. on Broadcasting*, vol. 54, no. 1, March 2008.

[21] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, V. Stemann, ”Practical Loss-Resilient Codes,” in proc. 29^{th} *Annu. ACM Symp. Theory of Computing*, 1997, pp. 150-159.

[22] T. Richardson, A. Shokrollahi, R. Urbanke, ”Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes,” *IEEE Trans. Inform. Theory*, vol. 47, no. 2, Feb. 2001.

[23] T. J. Richardson and R. Urbanke, “The capacity of low-density parity-check codes under message-passing decoding,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 599-618, Feb. 2001.

[24] T. Cover, J. Thomas, *Elements of Information Theory,* Wiley, 2006.

[25] P. R. Gray, P. J. Hurst, S. H. Lewis, R. G. Meyer, *Analysis and Design of Analog Integrated Circuits*, 4^{th} ed., Wiley, New York (USA), 2001.

[26] D. Divsalar, S. Dolinar, C. Jones, K. Andrews, “Capacity-Approaching Protograph Codes,” *IEEE Journal on Selected Areas of Communications*, vol. 27, issue 6, pp. 876-888, Aug. 2009.

[27] V. Rathi, R. Urbanke, “Density evolution, thresholds and the stability condition for non-binary LDPC codes,” in *IEE Proceedings - Communications*, vol. 152, num. 6, 2005, p. 1069-1074

[28] X.Y. Hu, M. Fossorier, E. Eleftheriou, "On the Computation of the Minimum Distance of Low-Density Parity-Check Codes," *IEEE International Conference on Communications 2004*, vol. 2, pp. 767-771, June 2004.

[29] J. Thorpe, "Low-Density Parity-Check (LDPC) Codes constructed from Protographs," JPL IPN Progress Report 42-154, Aug. 15, 2003.

[30] M. Fossorier, "Quasi-Cyclic Low-Density Parity-Check codes from circulant permutation matrices," *IEEE Trans. on Inf. Theory*, vol. 50, no. 8, pp. 1788-1793, Aug. 2004.

[31] S.Y. Chung, "On the construction of some capacity-approaching coding schemes," Ph.D dissertation, Massachusetts Institute of Technology, Cambridge, MA, Sept. 2000.

[32] T. Richardson, "Multi-Edge Type LDPC codes," presented at the Workshop honoring prof. R. McEliece on his 60^{th} birthday, California Institute of Technology, Pasadena, CA, May 24-25, 2002.

[33] D. MacKay, "Optimizing Sparse Graph Codes over GF(q)," available at http://www.inference.phy.ca.ac.uk/mackay/CodesGallager.html, Aug. 2003.

[34] C. Poulliat, M. Fossorier, D. Declercq, "Using binary images of non binary LDPC codes to improve overall performance," in proc. *4th International Symposium on Turbo-codes and related topics*, April 2006, Munich, Germany.

[35] C. Poulliat, M. Fossorier, D. Declercq, "Design of regular $(2, d_c)$-LDPC codes over $GF(q)$ using their binary images," *IEEE Trans. Comm.*, vol. 56, pp. 1626-1635, Oct. 2008.

[36] V. Rathi, "Conditional entropy of non-binary LDPC codes over the BEC," in proc. *IEEE International Symposium on Information Theory (ISIT) 2008*, Toronto, ON, July 6-11, 2008.

[37] L. Barnault, D. Declercq, “Fast decoding algorithm for LDPC over $GF(2^q)$,” in proc. *IEEE Inform. Theory Workshop (ITW) 2003*, Paris, F, March 2003.

[38] A. Marinoni, P. Savazzi, S. Valle, ”Efficient Design of Non-Binary LDPC Codes for Magnetic Recording Channels, Robust to Error Bursts,” in proc. 5^{th} *International Symposium on Turbo Codes and Related Topics,* Lausanne, CH, Sept. 1-5, 2008.

[39] A. Marinoni, P. Savazzi, R.D. Wesel, “Protograph-based q-ary LDPC codes for higher-order modulation,” in proc. 6^{th} *International Symposium on Turbo Codes and Iterative Information Processing,* Brest, F, Sept. 6-10, 2010.

[40] A. Marinoni, A. Cabrini, E. Costamagna, G. Torelli, P. Gamba, “Challenges and opportunities for information theory-based design of Phase Change Memories,” in proc. *IEEE Global Communications Conference (GLOBECOM) 2010*, Miami, FL, Dec. 6-10, 2010.

[41] A. Pirovano, A. Redaelli, F. Pellizzer, F. Ottogalli, M. Tosi, D. Ielmini, A.L. Lacaita, R. Bez, “Reliability study of Phase-Change Nonvolatile Memories,” *IEEE Trans. on Device and Materials Reliability*, vol. 4, no. 3, pp. 422-427, Sept. 2004

[42] A. Pantazi, A. Sebastian, N. Papandreou, M.J. Breitwisch, C. Lam, H. Pozidis, E. Eleftheriou, “Multilevel Phase-Change Memory modeling and experimental characterization,” in proc. *EPCOS 2009*

[43] G. Lechner, C. Weidmann, “Optimization of binary LDPC codes for the q-ary Symmetric Channel with moderate q,” in proc. 5^{th} *International Symposium on Turbo Codes and Related Topics,* Lausanne, CH, Sept. 1-5, 2008.

[44] T. Tian, C. Jones, J.D. Villasenor, R.D. Wesel, “Construction of irregular LDPC codes with low error floors,” in proc. *IEEE International Conference on Communications (ICC) 2003*

[45] C. Di, D. Proietti, E. Telatar, T. Richardson, R. Urbanke, “Finite length analysis of low-density parity-check codes on the binary erasure channel,” *IEEE Trans. on Inform. Theory*, vol. 48, pp. 1570-1579, June 2002.

[46] M. Yang, W.E. Ryan,”Performance of Efficiently Encodable Low-Density Parity-Check Codes in Noise Bursts on the EPR4 Channel,” *IEEE Trans. on MAG,* vol. 40, no. 2, March, 2004.

[47] J. Bellorado, A. Kavčić, L. Ping, ”Soft-Input, Iterative, Reed-Solomon Decoding using Redundant Parity-Check Equations,” in proc. *ITW 2007*

[48] H. Song, J.R. Cruz, ”Reduced-Complexity Decoding of Q-ary LDPC Codes for Magnetic Recording,” *IEEE Trans. on MAG,* vol. 39, no. 2, March 2003.

[49] A. Ghaith, J.B. Boutros, Y. Yuan-Wu, Fast and Reduced-Complexity Decoding Rule for q-ary LDPC Codes by Using the Dualty properties,” in proc. *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, 2006. (WiMob’2006)* June 19-21, 2006, Montreal, Que.

[50] G. Hosoya, H. Yagi, S. Hirasawa, ”Modification Methods for Construction and Performance Analysis of Low-Density Parity-Check Codes over the Markov-Modulated Channel,” in proc. *International Symposium on Information Theory and its Applications, ISITA 2004* , October 10-13 2004, Parma, Italy.

[51] Jin Lu and Keith G. Boyer, ”Novel RLL-ECC Concatenation Scheme for High-Density Magnetic Recording,” *IEEE Trans. on Magnetics*, vol. 43, no. 6, pp. 2271-2273, June 2007.

[52] B. Vasic (Editor), E. M. Kurtas (Editor), *Coding and Signal Processing for Magnetic Recording Systems.* CRC Press.

[53] A. Kavcic and J. F. Moura, "The Viterbi algorithm and Markov noise memory," *IEEE Trans.Inform. Theory*, vol. 46, no. 1, pp. 291-301, Jan. 2000.

[54] J. Hagenauer, P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in proc. *IEEE GLOBECOM*, pp. 47.11-47.17, Dallas, TX, Nov 1989.

[55] M. P. C. Fossorier, F. Burkert, S. Lin, and J. Hagenauer, "On the Equivalence Between SOVA and MaxLogMAP Decodings", *IEEE Comm. Letters*, vol. 2, no. 5, pp. 137-139, May 1998.

[56] R. Koetter, A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes," in proc. *IEEE Int. Symposium Information Theory 2000 (ISIT 2000)* , Sorrento, Italy, June 2000.

[57] A.I. Vila Casado, M. Griot, R.D. Wesel,"Informed Dymanic Scheduling for Belief-Propagation decoding of LDPC codes," in proc. *IEEE International Conference on Communications (ICC) 2007*, June 24-28, 2007, Glasgow, Scotland.

[58] M.K. Cheng, J. Campello, P.H. Siegel, "Soft-Decision Reed-Solomon Decoding on Partial Response Channels," in proc. *IEEE Global Telecommunications Conference,* pp. 1026-1030, Taipei, Taiwan, Nov. 2002.

[59] B.M. Kurkoski, P.H. Siegel, J.K. Wolf, "Joint Message-Passing Decoding of LDPC Codes and Partial-Response Channels," *IEEE Trans. on Information Theory,* vol. 48, no. 6, June 2002.

[60] W. E. Ryan, L. L. McPheters, and S. W. McLaughlin, "Combined turbo coding and turbo equalization for PR4-equalized Lorentzian channels, in Proc. Conf. Information Sciences and Systems, Mar. 1998.

[61] W. E. Ryan, "Performance of high rate turbo codes on a PR4-equalized magnetic recording channel," in Proc. IEEE Int. Conf. Communications. Atlanta, GA, June 1998, pp. 947-951.

[62] T. Souvignier, M. Berg, P. H. Siegel, R. E. Swanson, and J. K. Wolf, “Turbo decoding for partial response channels,” *IEEE Trans. Commun.,* vol. 48, pp. 1297-1308, Aug. 2000.

[63] G. Colavolpe, G. Germi, “On the application of factor graphs and the sum product algorithm to ISI channels,” *IEEE Trans. Commun.,* vol. 53, no. 5, pp. 818-825, May 2005.

[64] G. Colavolpe, “On LDPC codes over channels with memory,” *IEEE Trans. Wireless Commun.,* vol. 5, no. 7, pp. 1757-1756, July 2006.

[65] R. Radhakrishnan, B. Vasic, “Joint message-passing symbol-decoding of LDPC coded signals over partial-response channels,” in Proc. IEEE Int. Conf. Communications, Dresden, Germany, June 14-18, 2009.

[66] L. Bahl, J. Cocke, F. Jelinek, J. Raviv, ”Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate,” *IEEE Trans. on Information Theory,*, Vol IT-20, pp 284-287, March 1974.

[67] C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, “Iterative correction of intersymbol interference: turbo equalization,” *Eur. Trans. Telecommun.*, pp. 507-511, Sept./Oct. 1995.

[68] G. Ungerboeck, “Adaptive Maximum-Likelihood Receiver for Carrier-Modulated Data-Transmission Systems,” *IEEE Trans. on Communications,* vol. 22, no. 5, May, 1974.

[69] P. Hoeher, “Optimal subblock-by subblock detection,” *IEEE Trans. on Comm.*, vol. 43, pp. 714-717, Feb. 1995.

[70] G.D. Forney, “Maximum-Likelihood Sequence Estimation of Digital Sequences in the Presence of Intersymbol Interference,” *IEEE Trans. on Information Theory,* vol. IT-18, no. 3, May 1972.

[71] S.M. Alamouti, “A simple transmit diversity technique for wireless communications,” *IEEE Journal on Selected Areas of Communications*, vol. 16, no.8, Oct. 1998.

[72] E. Biglieri, R. Calderbank, A. Constantinides, A. Goldsmith, A. Paulraj, H.V. Poor, *MIMO Wireless Communications*, Cambridge University Press

[73] F. Daneshgaran, M. Laddomada, M. Mondin, “LDPC-based channel coding of correlated sources with iterative joint decoding,” *IEEE Trans. on Comm.*, vol. 54, no. 4, Apr. 2006.

[74] F. Daneshgaran, M. Laddomada, M. Mondin, “Iterative joint channel decoding of correlated sources,” *IEEE Trans. on Wireless Comm.*, vol. 5, no. 10, October 2006.

[75] F. Daneshgaran, M. Laddomada, M. Mondin, “IIterative joint channel decoding of correlated sources employing serially concatenated convolutional codes,” *IEEE Trans. on Information Theory.*, vol. 51, no. 7, July 2005.

[76] J. Ha, J. Kim, D. Klinc, S.W. McLaughlin, “Rate-Compatible punctured Low-Density Parity-Check codes with short block lengths,” *IEEE Trans. on Information Theory*, vol. 52, no.2, Feb. 2006

[77] D. Klinc, J. Ha, S.W. McLaughlin, “On rate-adaptability of nonbinary LDPC codes,” in proc. 5th *International Symposium on Turbo Codes and Related Topics,* Lausanne, CH, 1-5 Sept. 2008.

[78] A.I. Vila Casado, W.-Y. Weng, S. Valle, R.D. Wesel, “Multiple-rate Low-Density Parity-Check Codes with constant blocklength,” *IEEE Trans. on Comm.*, vol. 57, no. 1, Jan. 2009.

[79] G. Atwood and R. Bez, “Current status of chalcogenide phase change memory,” *in Device Research Conf. Dig.*, Jun. 2005, vol. 1, pp. 29-33.

[80] A. Pirovano *et al.*,“Reliability study of Phase-Change Nonvolatile Memories,” *IEEE Trans. on Device and Materials Reliability*, vol. 4, no. 3, pp. 422-427, Sept. 2004

[81] L.A. Lastras-Montaño *et al.*, “Rewritable storage channels,” in proc. *Int. Symp. on Information Theory and its Applications (ISITA)*, pp. 1-6, Dec. 7-10, 2008, Auckland, NZ.

[82] M. Franceschini *et al.*, “Information theory based design of phase-change memories,” in proc. *Information Theory and Applications (ITA) Workshop 2010*, pp. 1-7, UCSD, La Jolla, CA.

[83] T. N. Blalock, R. C. Jaeger, ”A high-speed clamped bit-line current-mode sense amplifier,” *IEEE J. of Solid-State Circuits*, vol. 26, no. 4, pp. 542-548, Apr 1991.

[84] F. Bedeschi, *et al.* ”A Bipolar-Selected Phase Change Memory featuring multi-level cell storage,” *IEEE J. of Solid-State Circuits*, vol. 44, no. 1, pp. 217-227, Jan. 2009.

[85] M. Franceschini *et al.*, “A communication-theoretic approach to Phase Change Storage,” in proc. *IEEE Int. Conf. on Communications (ICC) 2010*, May, 23-27, 2010, Cape Town, ZA.

[86] T.K. Moon, *Error Correction Coding*, Wiley & Sons, 2005.