Design of Bandwidth-Efficient Unequal Error Protection LDPC Codes

Sara Sandberg and Neele von Deetzen

Abstract—**This paper presents a strategy for the design of bandwidth-efficient LDPC codes with unequal error protection. Bandwidth efficiency is obtained by appropriately designing the codes for higher order constellations, assuming an AWGN channel. The irregularities of the LDPC code are designed, using the Gaussian approximation of the density evolution, to enhance the unequal error protection property of the code as well as account for the different bit error probabilities given by the higher order constellation. The proposed code design algorithm is flexible in terms of the number and proportions of protection classes. It also allows arbitrary modulation schemes. Our method combines the design of unequal error protection LDPC codes for the binary input AWGN channel with the code design for higher order constellations by dividing the variable node degree distribution into sub-degree distributions for each protection class and each level of protection from the modulation. The results show that appropriate code design for higher order constellations reduces the overall bit-error rate significantly. Furthermore, the unequal error protection capability of the code is increased, especially for high SNR.**

Index Terms—**LDPC, unequal error protection, higher order constellations, bandwidth efficiency.**

I. INTRODUCTION

MULTIMEDIA applications require large amounts of data to be transmitted through networks and wireless transmission systems with reasonable delay and error performance. Therefore, resources like power, time, or bandwidth have to be used economically. Multimedia data usually have heterogeneous sensitivity against transmission errors. They often contain a header for data management in higher layers, some essential payload as well as additional payload used for enhanced quality. Hence, the transmission system should provide unequal error protection (UEP) in order to account for the different properties. Suitable UEP may increase the perceived performance for applications where different bits have different sensitivity to errors, such as transmission of progressively encoded images or a frame with header information. For packet-based transmissions with no possibility of retransmission, an error in the header is critical and may lead

to rejection of the packet or even a crash of the source decoder, while errors in the payload are often tolerable.

In order to allow for bandwidth-efficient transmission, it is desirable to use higher order modulation techniques, such as M-QAM, M-PSK, $M > 2$, or more advanced constellations. Modulation with higher order constellations (HOCs) may imply that different bits in the symbol have different error probabilities, i.e., the modulation already provides some UEP. Coded modulation is a well-known strategy to optimize the coding scheme given the modulation to improve the performance of transmission systems in terms of overall bit-error rate (BER), [1]–[4]. In multilevel coding [5], the modulation alphabet is successively partitioned into smaller subsets, where each partitioning level is assigned a label. These labels are protected by separate channel codes with certain protection capabilities. In such a strategy, the codes have to be designed carefully, depending on the modulation scheme and its partitioning or labeling strategy. However, applying a shorter code on each level may have drawbacks (e.g. higher BER) compared to designing one long code whose output bits are appropriately assigned to the levels. Therefore, we employ the latter, taking advantage of the longer code.

This paper focuses on low-density parity-check (LDPC) codes, originally presented by Gallager in [6]. They exhibit a performance very close to the capacity for the binary-input additive white Gaussian noise (BI-AWGN) channel, [7]. The close-to-optimal performance of LDPC codes for the BI-AWGN channel suggests the use of LDPC codes also for other channels. The aim of this paper is to design LDPC codes for transmission using HOCs in applications where the source bits have different sensitivities to errors and UEP is desired. As far as we know, such a code design has not been proposed before. However, a code design using turbo codes for coded modulation with UEP has recently been suggested, [8].

LDPC codes are block codes with a sparse parity-check matrix H of dimension $(n - k) \times n$, where $R = k/n$ denotes the code rate and k and n are the lengths of the information word and the codeword. The codes can be represented by a bipartite graph, called a Tanner graph [9], which facilitates a decoding algorithm known as the message-passing algorithm [10]. The graph consists of two types of nodes, variable nodes and check nodes, which correspond to the bits of the codeword and to the parity-check constraints, respectively. A variable node is connected to a check node if the bit is included in the parity-check constraint. For regular LDPC codes, all variable nodes have the same degree and all check nodes have another common degree. However, irregular LDPC codes are known to approach capacity more closely than regular

LDPC codes. The irregular variable node and check node degree distributions may be defined by the polynomials [7] $\lambda(x) = \sum_{i=2}^{d_{v_{max}}} \lambda_i x^{i-1}$ and $\rho(x) = \sum_{i=2}^{d_{c_{max}}} \rho_i x^{i-1}$ respectively, where $d_{v_{max}}$ is the maximum variable node degree and $d_{c_{max}}$ is the maximum check node degree of the code. The coefficients of the degree distributions describe the proportion of edges connected to nodes with a certain degree. In order to optimize the degree distribution of an irregular LDPC code, the decoding behavior has to be investigated. Using a message-passing algorithm, the messages along the edges of the graph are updated iteratively. The messages at the input of a variable node and a check node at each iteration represent mutual information and can be computed by means of density evolution using a Gaussian approximation [11]. Thereby, the decoding behavior of a code ensemble with a specific degree distribution may be predicted. Density evolution is an asymptotic tool, but is commonly used to design codes of finite length.

LDPC codes have been designed for larger constellation sizes in previous works. In [12], separate codes were designed for each level in a multilevel coding scheme. On the other hand, bit-interleaved coded modulation (BICM), [13], [14] employs only one code for all levels of the modulation. A design method for discrete multitone modulation (DMT) is proposed in [15]. This method may also be directly applied to HOCs and is very similar to the design approach suggested in [14]. In [14] and [15], the codes are designed to have local properties that match the HOCs and bit positions of the modulation scheme are assigned to the codeword bits. Reliability mappings of bits from the modulation to codeword bits have also been suggested, [16], [17].

In many applications, the desired UEP properties are low BER within one or several classes of bits, while the performance of the remaining classes should be comparable to non-UEP codes. In the following we address such codes as codes with good UEP capability. UEP is commonly provided by multilevel coding or adapted code rates, for example by puncturing. These methods have in common that different codes are used for each level of protection. However, for most applications the more important bits are fewer than the less important bits. This implies that even if the code rate used for the more important bits is low, this codeword will usually be short. To avoid the use of short codewords (or a very long delay), LDPC codes that provide UEP within one codeword may be designed instead. Such codes can for example be constructed by an algebraic method based on the Plotkin construction, [18]. However, since it is widely observed that the connection degree of the variable nodes affects the bit-error rate for a limited number of decoder iterations, it is more typical to design the variable and/or check node degree distribution of the code in an irregular way using density evolution, [19]–[23]. In this case the codeword bits are divided into several protection classes with different protection depending on the connection degrees of their bits. In [21], the check node degree distribution is adapted, keeping the variable node degree distribution fixed, whereas the authors of [22] optimize the irregular variable node degree distribution while keeping the check node degree distribution fixed. The basic idea in [20]–[23] is to achieve UEP by dividing the de-gree distributions into sub-distributions. Such sub-distributions have also been employed for systems without UEP capability to account for HOCs [14], [15], [24]. In [14], the different amount of protection for each modulation level is taken into account in the initialization of the density evolution algorithm that is employed to optimize the sub-degree distributions of the code. Both [15] and [24] designed LDPC codes for a set of parallel subchannels.

In this paper, we propose a UEP-LDPC code design for HOCs that is based on optimizing the variable node degree distribution $\lambda(x)$. Apart from designing the code to account for the UEP provided by the modulation itself and thereby reducing the overall BER, the aim is to provide a flexible design method that can use the UEP from the modulation to create a code with other UEP properties which are usually specified by the source coding unit. We design UEP-LDPC codes using sub-degree distributions both for the protection classes and for the classes of bits with different protection resulting from the modulation. This allows for a very flexible code design where any conventional modulation scheme, like M-QAM or M-PSK as well as more complex schemes like hierarchical constellations [25], may be used. The separation of the variable node degree distribution into sub-degree distributions significantly increases the number of design parameters. Therefore it is important to note that the code design is solved by iterative linear programming (LP), which enables an efficient optimization of the sub-degree distributions. Our code design is based on the design method for UEP-LDPC codes suggested in [22] that employs iterative LP.

The paper is organized as follows. Section II presents the overall system and modulator models. Section III contains the main part of this paper which introduces the code optimization of UEP capable codes used with HOCs and provides an algorithm for the code design. In Section IV, some simulation results for 8-PSK and 64-QAM are presented and compared to BPSK-optimized codes.

II. SYSTEM MODEL

A. Model Description

Each codeword bit is assigned to a protection class and a modulation class. We apply only one UEP-LDPC code, providing N_c protection classes at its output (see Fig. 1). The independent and identically distributed (i.i.d.) information bits u_i are divided into N_c-1 protection classes $C_1 \ldots C_{N_c-1}$. The least protected class C_{N_c} contains the parity bits. Each bit is protected by the UEP-LDPC code according to the protection class it belongs to, where C_1 has the best protection. The sizes of the protection classes (they do not have to be of equal size) as well as the allocation of information bits to protection classes are usually defined by the source coding unit.

The bits of the protection classes are remultiplexed and assigned to certain bit positions of the modulator, corresponding to modulation classes $M_1, \ldots, M_{N_s}$. A bit that belongs to modulation class M_j will be mapped to the bit position in the symbol that corresponds to that modulation class, i.e., it will be naturally protected according to modulation class M_j. Bits belonging to one modulation class are affected by the same (channel) bit error probability. The bit assignment will
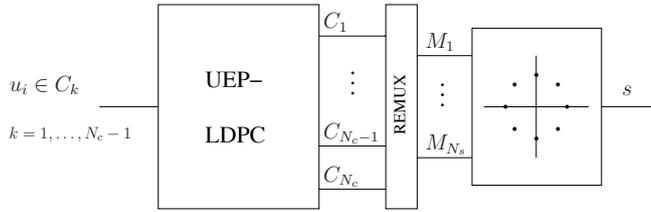
Fig. 1. Schematic description of the proposed scheme. The source bits are encoded by a UEP-LDPC code and the coded bits are assigned to modulation classes before modulation.

be described in Section III-A. In the following, we assume an AWGN channel with noise variance σ^2.

B. Modulation

Let us assume a modulation scheme with $M = 2^{l_m}$ symbols, labeled by binary vectors $\mathbf{d} = (d_{l_m}, \ldots d_2, d_1)$. In order to design codes for HOCs, we investigate the error probabilities of the individual bit positions in the symbol. Depending on the signal constellation and the labeling strategy, the error probabilities of the individual bit positions may be determined.

One may use traditional constellations like 8-PSK or 64-QAM as well as non-uniform constellations, so-called hierarchical modulation [26] or multiresolution modulation [25]. When UEP is desired, non-uniform constellations have the advantage that they can provide greater differentiation in error probabilities of the bit positions in the symbol than traditional constellations. Furthermore, the design methods in the above references allow for a controllable amount of UEP through variations of the distances between the signal points.

Generally, one can approximate the symbol-error probability P_s as well as the bit-error probabilities $P_{b,d_1} \ldots P_{b,d_{l_m}}$ of a constellation using the union bound. For Gray labeling, the average bit-error probability is often assumed to be equal for all bit positions, i.e., $\tilde{P}_{b,d_i} \approx \frac{1}{\log_2(M)} \cdot P_s$. However, it is important to be aware of the different bit-error probabilities when designing codes for HOCs. In this work we only consider Gray labeling, since the bit-errors have low statistical dependencies and can thus be assumed to be independent [27]. This is very important for the message-passing decoder. However, for labelings other than Gray, the differences in BER may be more significant. If such labelings are of interest, we suggest the use of the improved decoding algorithm proposed in [28] that combines decoding and demodulation and takes into account the statistical dependencies among bit errors originating in the same symbol.

In [5], it was stated that a symmetric channel with an input alphabet of size 2^{l_m} can be represented by l_m equivalent binary-input channels. Depending on the labeling, the equivalent channels may not be symmetric. Since the symmetry of the channel is an important property for the density evolution of LDPC code ensembles, we use a method called *i.i.d. channel adapters* [29]. In order to force symmetric component channels, i.i.d. binary vectors $\mathbf{t} = [t_1 \ldots t_n]$ are added modulo-2 to the codeword. At the receiver, the log-likelihood ratio L_j is multiplied by -1 if $t_j = 1$. Since the equivalent component codes are now symmetric, one can
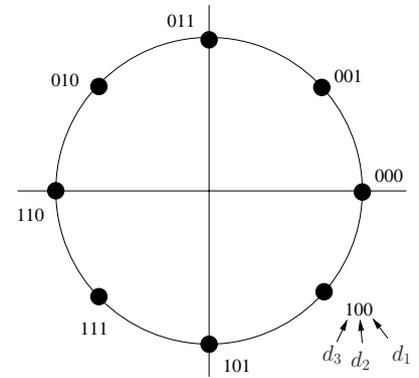

Fig. 2. 8-PSK modulation with Gray labeling.

assume their behavior being equal to BPSK signaling. From the known bit-error probabilities of the binary component channels, it is easy to determine their equivalent noise variances σ_i^2 which are required for density evolution later on. We define the noise vector $\boldsymbol{\sigma}^2 = [\sigma_1^2 \ \ldots \ \sigma_{N_s}^2]$ to be a vector that contains the equivalent noise variances for each separate bit-error rate, i.e., for each modulation class, ordered with the lowest variance first. We assume that there are N_s distinct equivalent noise variances, where $N_s \leq l_m$.

Example 2.1 (8-PSK): 8-PSK modulation with Gray labeling is shown in Fig. 2. Considering only neighboring signal points, the approximate symbol-error rate expression for this modulation is given as [30],

$$P_{s,8-PSK} = 2Q\left(\sqrt{6E_b/N_0}\sin\frac{\pi}{8}\right), \qquad (1)$$

where $Q(\cdot)$ is the Gaussian probability Q function. The average bit-error rate is $\tilde{P}_b \approx P_s/\log_2(M)$.

The expressions for the bit-error probabilities of the individual bits in the symbol are

$$P_{b,d_1} \approx Q\left(\sqrt{6E_b/N_0}\sin\frac{\pi}{8}\right), \qquad (2)$$

$$P_{b,d_2} = P_{b,d_3} \approx \frac{1}{2}Q\left(\sqrt{6E_b/N_0}\sin\frac{\pi}{8}\right). \qquad (3)$$

Note that the above expressions are obtained by applying approximations. The approximations are assumed to be appropriate for our purposes, but can be replaced by more exact formulas.   □

The equivalent noise variances of each modulation class may be determined from the different bit-error probabilities by means of an equivalent BPSK channel,

$$\sigma_j^2 = \frac{1}{\left(Q^{-1}(P_{b,d_j})\right)^2}. \qquad (4)$$

Usually, LDPC codes are designed for one noise variance and it is assumed that all bits are transmitted over the corresponding channel. In this paper we design codes using the equivalent noise variances from (4). To compare the standard code design with our new design in a fair way, we calculate an equivalent average BPSK noise variance, denoted by σ_{BPSK}^2. This is the noise variance of a BPSK channel that would give a bit-error probability equal to the average bit-error probability
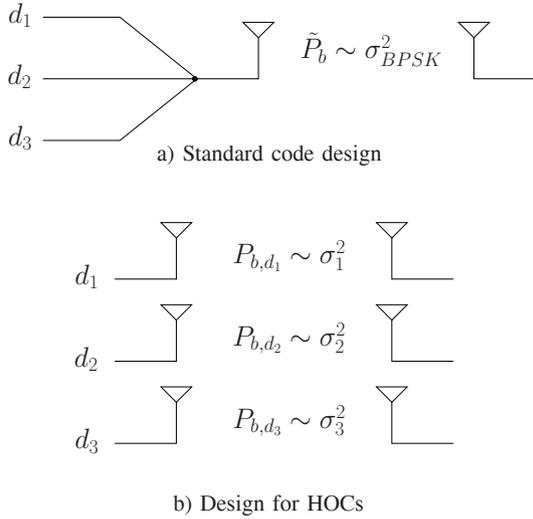
Fig. 3. Channel assumptions for a) standard code design b) design for HOCs.

of the HOC. Thus, we compare our new design for HOCs with a standard code design with σ^2_{BPSK} and use both with HOCs. Fig. 3 shows a schematic explanation of the channel assumptions made by the receiver.

In the following, we claim that approximating the HOC channel by N_s equivalent BPSK channels together with i.i.d. channel adapters meets our requirements.

C. Notations

We consider a UEP-LDPC code with N_c protection classes. The proportions of each information class, $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{N_c-1}]$, are given by the normalized lengths of each class corresponding to the information bits. α_i equals the number of bits belonging to protection class C_i divided by the total number of information bits k. The proportion distribution of the bits in the codeword belonging to the protection classes is thus given by $\mathbf{p} = [\alpha_1 R, \ldots, \alpha_{N_c-1}R, (1-R)]$. N_s is the number of different bit-error rates for the bits in a symbol. We will describe the bits with a distinct bit-error rate as belonging to one modulation class M_j, $j = 1, \ldots, N_s$. $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_{N_s}]$ defines the proportion of bits in the code-word that belongs to each modulation class.

The vector $\boldsymbol{\lambda}$ contains the overall variable node degree distribution, both for different protection classes and different modulation classes. Let $\lambda^{C_k}_{M_j,i}$ be the proportion of edges connected to variable nodes of degree i that belong to modulation class M_j and protection class C_k. Define $\boldsymbol{\lambda}^{C_k}_{M_j} = [\lambda^{C_k}_{M_j,2}, \ldots, \lambda^{C_k}_{M_j,d_{v_{max}}}]^T$ and $\boldsymbol{\lambda} = \left[\boldsymbol{\lambda}^{C_1}_{M_1}{}^T, \ldots, \boldsymbol{\lambda}^{C_{N_c}}_{M_1}{}^T, \ldots, \boldsymbol{\lambda}^{C_1}_{M_{N_s}}{}^T, \ldots, \boldsymbol{\lambda}^{C_{N_c}}_{M_{N_s}}{}^T \right]^T$, where $(\cdot)^T$ denotes the transpose. $\boldsymbol{\lambda}^{C_k}_{M_j}$ is a column vector of length $d_{v_{max}} - 1$ and $\boldsymbol{\lambda}$ is a column vector of length $(d_{v_{max}} - 1)N_c N_s$. The vector $\boldsymbol{\rho} = [\rho_2, \ldots, \rho_{d_{c_{max}}}]^T$ describes the check node degree distribution. We define $\mathbf{1}$ to be an all-ones vector of appropriate length.

III. UEP-LDPC CODES FOR HIGHER ORDER CONSTELLATIONS

A. Optimization of the Degree Distribution for HOCs

The mutual information messages from a check node to a variable node (x_{cv}) and from a variable node to a check node (x_{vc}) at iteration l, computed by means of density evolution using the Gaussian approximation [11], are given by

$$x_{cv}^{(l-1)} = 1 - \sum_{j=2}^{d_{c_{max}}} \rho_j J((j-1)J^{-1}(1-x_{vc}^{(l-1)})), \quad (5)$$

$$x_{vc}^{(l)} = \sum_{j=1}^{N_s} \sum_{k=1}^{N_c} \sum_{i=2}^{d_{v_{max}}} \lambda^{C_k}_{M_j,i} J(\frac{2}{\sigma_j^2} + (i-1)J^{-1}(x_{cv}^{(l-1)})), \quad (6)$$

with $J(\cdot)$ computing the mutual information $x = J(m)$ by

$$
\begin{aligned}
J(m) &= 1 - \mathbb{E}\{\log_2(1+e^{-z})\} \quad &(7)\\
&= 1 - \frac{1}{\sqrt{4\pi m}} \int_{\mathbb{R}} \log_2(1+e^{-z}) \cdot e^{-\frac{(z-m)^2}{4m}} dz
\end{aligned}
$$

for a consistent Gaussian random variable $z \sim \mathcal{N}(m, 2m)$ with mean m and variance $2m$. In the special case with only one protection class and one modulation class (corresponding to standard LDPC code design), i.e., $N_c = N_s = 1$, the update rule for the messages from variable nodes to check nodes is given by

$$x_{vc}^{(l)} = \sum_{i=2}^{d_{v_{max}}} \lambda_i J(\frac{2}{\sigma^2} + (i-1)J^{-1}(x_{cv}^{(l-1)})). \quad (8)$$

The update rule for the messages from check nodes to variable nodes is not affected by the division of the variable node degree distribution into sub-distributions since the check node degree distribution is constant for all protection classes and modulation classes. Equations (5) and (6) can be combined to yield the mutual information evolution of the LDPC code

$$x_{vc}^{(l)} = F(\boldsymbol{\lambda}, \boldsymbol{\rho}, \boldsymbol{\sigma}^2, x_{vc}^{(l-1)}). \quad (9)$$

If $x_{vc}^{(l)} > x_{vc}^{(l-1)}$ for any $x_{vc}^{(l-1)}$, then $\boldsymbol{\lambda}$ and $\boldsymbol{\rho}$ describe a code ensemble for which density evolution converges for the noise variance vector $\boldsymbol{\sigma}^2$. Since the variable node degree distributions give proportions of edges connected to variable nodes of certain degrees, the constraint

$$\sum_{j=1}^{N_s} \sum_{k=1}^{N_c} \sum_{i=2}^{d_{v_{max}}} \lambda^{C_k}_{M_j,i} = 1 \quad (10)$$

must be fulfilled.

When optimizing a degree distribution, there are different strategies. One could for example maximize the code rate given a certain threshold, where the threshold is defined as the lowest E_b/N_0 for which density evolution converges. In this paper, we choose to minimize the threshold given a fixed code rate. In order to do so, we rerun a linear programming routine while increasing the SNR, until the SNR is high enough so that it is possible to find a degree distribution for which density evolution converges.

The aim of the code design is to reduce the BERs of the protection classes by taking the different error probabilities of the modulation levels into account. The design algorithm

should also give the possibility to trade overall BER for UEP capability. The natural way of assigning bits from modulation classes to protection classes to achieve UEP, is to use the best protected bits from the modulation, that is, modulation class M_1, for protection class C_1 and continue like that until all bits have been assigned to a protection class. However, this assignment is not guaranteed to give a degree distribution with the lowest possible threshold. As discussed later, there is always a trade-off between a low threshold and good UEP capability. By distinguishing not only between degree distributions of different protection classes but also of different modulation classes, linear programming may be used to assign bits from the modulation classes to the protection classes.

It is well-known that a higher connectivity of a variable node leads to better protection. Thus, the optimization target is to find a variable node degree distribution for the whole code that maximizes the average variable node degree of the class being optimized. UEP capability may be obtained by running the optimization algorithm sequentially, one protection class at a time, for an E_b/N_0 slightly higher than the threshold. When the proportion of edges in the Tanner graph that are associated with a specific class is high, many messages will be passed and the local convergence of the graph is fast. However, in the limit when the number of decoder iterations tends to infinity, all messages have transited the whole graph and there should be no UEP capability left, [22]. This implies that the UEP capability theoretically should be decreasing with increasing number of iterations. However, our results show that for a reasonable number of iterations the UEP capability is not affected much by the number of decoder iterations. It has been shown that the choice of code construction algorithm is critical for remaining UEP capability after a reasonable number of iterations, [31]. Figure 8 in Section IV shows the BER as a function of the number of decoder iterations for the code design and code construction algorithm suggested in this paper.

The target function for protection class C_k can be formulated as

$$\max_{\boldsymbol{\lambda}} \sum_{j=1}^{N_s} \sum_{i=2}^{d_{v_{max}}} \lambda_{M_j,i}^{C_k} . \tag{11}$$

This target function only concerns the bits of the current protection class. The optimization constraints, such as the convergence constraint (9) or the proportion distribution constraint (10), will involve the whole set of code bits. In every step, the degree distributions of lower protection class bits are fixed and only degree distributions of higher protection classes may be changed, [22]. The target function will ensure a maximized average variable node degree, since a high value of the above sum implies that a large proportion of the edges is connected to variable nodes belonging to class C_k. In [22], the authors show that it is not only the average connection degree to be maximized but also the minimum variable node degree. Therefore, the second optimization target of our code design is the maximization of the minimum variable node degree of each protection class. Therefore, the optimization is first performed for a high minimum variable node degree. If no degree distribution for which density evolution converges can be found, the optimization is repeated for a lower minimum

variable node degree as in [22].

Repeated optimization with the target function (11) results in a degree distribution with UEP capability. However, this target function does not account for the fact that variable nodes connected to the first modulation class have lower error probability after demodulation than those from worse modulation classes. Therefore, we introduce a scaling factor k_j for each modulation class which decreases with increasing modulation class index, i.e., $k_1 > k_2 > \ldots > k_{N_s} > 0$.

$$\max_{\boldsymbol{\lambda}} \sum_{j=1}^{N_s} k_j \sum_{i=2}^{d_{v_{max}}} \lambda_{M_j,i}^{C_k} \tag{12}$$

The choice of this scaling factor affects how valuable a better modulation class is compared to a modulation class with higher equivalent noise variance. We choose $k_j = N_s - j + 1$. This choice of k_j has the effect that the linear programming algorithm, if possible while fulfilling all constraints, will use modulation classes with low noise variance for the protection class being optimized.

Besides the optimization constraints given in (9) and (10), there are some more constraints which have to be fulfilled by the optimized $\boldsymbol{\lambda}$. The variable node degree distribution is connected to the check node degree distribution and the code rate by

$$R = 1 - \frac{\sum_{j=2}^{d_{c_{max}}} \rho_j/j}{\sum_{i=2}^{d_{v_{max}}} \lambda_i/i} . \tag{13}$$

Furthermore, the proportion vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ impose the following two constraints on $\boldsymbol{\lambda}$, where n_{C_k} and n_{M_j} denote the total number of variable nodes belonging to protection class C_k and to modulation class M_j, respectively,

$$n_{C_k} = \alpha_k \cdot R \cdot n \quad k = 1, \ldots, N_c - 1, \quad \text{and} \tag{14}$$
$$n_{M_j} = \beta_j \cdot n \quad j = 1, \ldots, N_s . \tag{15}$$

Moreover, n_{C_k} and n_{M_j} are connected to $\boldsymbol{\lambda}$ by

$$n_{C_k} = \sum_{j=1}^{N_s} \sum_{i=2}^{d_{v_{max}}} \frac{\lambda_{M_j,i}^{C_k}}{i} \cdot n \cdot (1-R) / \left(\sum_{i=2}^{d_{c_{max}}} \frac{\rho_i}{i} \right) \tag{16}$$

and

$$n_{M_j} = \sum_{k=1}^{N_c} \sum_{i=2}^{d_{v_{max}}} \frac{\lambda_{M_j,i}^{C_k}}{i} \cdot n \cdot (1-R) / \left(\sum_{i=2}^{d_{c_{max}}} \frac{\rho_i}{i} \right) , \tag{17}$$

respectively.

When designing good LDPC code ensembles, the stability condition which ensures convergence of the density evolution for mutual information close to one should be fulfilled [7]. The stability condition gives an upper bound on the number of degree-2 variable nodes. For a BPSK scheme, where all bits are affected by the same noise variance σ^2, we have [7]

$$\frac{1}{\lambda'(0)\rho'(1)} > e^{-r} = \int_{\mathbb{R}} P_0(x) e^{-\frac{x}{2}} dx = e^{-\frac{1}{2\sigma^2}} \tag{18}$$

with $P_0(x)$ being the message density corresponding to the received values and $\lambda'(x)$ and $\rho'(x)$ being the derivatives of the degree polynomials. It is straightforward to see that, for our scheme, $\lambda'(0) = \sum_{j=1}^{N_s} \sum_{k=1}^{N_c} \lambda_{M_j,2}^{C_k}$ and

$\rho'(1) = \sum_{m=2}^{d_{cmax}} \rho_m \cdot (m-1)$. In our case, the bits are affected by channel noise with different equivalent noise variances σ_j^2 (see (4)) and thus different densities $P_{0,j}(x)$. The density of the whole set of bits is equivalent to the (weighted) sum of the individual densities, i.e., $\sum_j \beta_j \cdot P_{0,j}(x)$. This gives the stability condition

$$e^{-r} = \int_{\mathbb{R}} \sum_{j=1}^{N_s} \beta_j \cdot P_{0,j}(x) e^{-\frac{x}{2}} dx = \sum_{j=1}^{N_s} \beta_j \cdot e^{-\frac{1}{2\sigma_j^2}} . \quad (19)$$

All the above given constraints are rewritten to contain only $\boldsymbol{\lambda}$, $\boldsymbol{\rho}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\sigma}^2$, and R, i.e., without using n and k. This makes the algorithm independent of the code length.

B. Optimization Algorithm

The optimization algorithm proposed here is a modification of the hierarchical optimization algorithm presented in [22] for HOCs. The optimization is performed at $E_b/N_0 = \delta + \epsilon$, which will be the threshold of the optimized code, where δ is the lowest possible threshold in dB for the given $\boldsymbol{\rho}$ and $d_{v_{max}}$, and ϵ is an offset from the lowest threshold that provides more flexibility in the choice of $\boldsymbol{\lambda}$. An irregular LDPC code always provides some inherent UEP capability. This is achieved by mapping higher degree variable nodes to the more protected classes. For $\epsilon = 0$ this is the only UEP capability available. On the other hand, by introducing $\epsilon > 0$, we allow for more freedom in the code design. The LP algorithm will assign higher degrees or larger fractions of edges to the high degrees of the first protection class and accept worse properties for the less important classes. Hence, $\epsilon > 0$ leads to increased UEP capability compared to the inherent UEP capability of the 'minimum threshold code'.

The algorithm can be divided into an inner and an outer loop. For a given E_b/N_0, the outer loop runs over the protection classes, starting with the first. At this point, the degree distribution of the code is designed while optimizing the corresponding protection class. As mentioned before, there are two target functions to be maximized, i.e., the average connection degree and the minimum connection degree of the class' variable nodes. Since we are using LP with only a single-objective function, we choose it to be the maximization of the average connection degree. The maximization of the minimum variable node degree is performed by the inner loop which runs over different values for the minimum degree, starting from some maximum value and successively reducing it during the procedure. Once a valid solution is found for the current protection class, the next protection class C_k is optimized. At this point, the classes C_1, ..., C_{k-1} have already been optimized and $\boldsymbol{\lambda}_{M_j}^{C_1}$, ..., $\boldsymbol{\lambda}_{M_j}^{C_{k-1}}$, $\forall j$, are fixed. The optimization algorithm can be stated as follows.

I) Fix $E_b/N_0 = \delta + \epsilon$ and calculate $\boldsymbol{\sigma}^2$.
II) Find $\boldsymbol{\lambda}$ by performing the inner optimization loop for each protection class.

For the optimization of class C_k, a linear-programming routine is executed. It requires the definition of the check node degree distribution $\boldsymbol{\rho}$, $E_b/N_0 = \delta + \epsilon$, the maximum variable node degree $d_{v_{max}}$, the code rate R, and the proportion vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

1) Initialization $\quad\quad d_{v_{min}}^{(k)} = d_{v_{max}}$
2) While optimization failure
   a) Optimize

$$\max_{\boldsymbol{\lambda}} \sum_{j=1}^{N_s} k_j \sum_{i=2}^{d_{v_{max}}} \lambda_{M_j,i}^{C_k} \quad (20)$$

under the constraints $[C_1] - [C_6]$.
$[C_1]$ Rate constraint, see (13)

$$\sum_{j=1}^{N_s} \sum_{k=1}^{N_c} \sum_{i=2}^{d_{v_{max}}} \frac{\lambda_{M_j,i}^{C_k}}{i} = \frac{1}{1-R} \sum_{i=2}^{d_{cmax}} \frac{\rho_i}{i} \quad (21)$$

$[C_2]$ Proportion distribution constraints
   i) See (10)

$$\sum_{j=1}^{N_s} \sum_{k=1}^{N_c} \boldsymbol{\lambda}_{M_j}^{C_k \, T} \mathbf{1} = 1 \quad (22)$$

   ii) $\forall k \in \{1, \ldots, N_c - 1\}$, see (14) and (16)

$$\sum_{j=1}^{N_s} \sum_{i=2}^{d_{v_{max}}} \frac{\lambda_{M_j,i}^{C_k}}{i} = \alpha_k \frac{R}{1-R} \sum_{i=2}^{d_{cmax}} \frac{\rho_i}{i} \quad (23)$$

   iii) $\forall j \in \{1, \ldots, N_s - 1\}$, see (15) and (17)

$$\sum_{k=1}^{N_c} \sum_{i=2}^{d_{v_{max}}} \frac{\lambda_{M_j,i}^{C_k}}{i} = \beta_j \frac{1}{1-R} \sum_{i=2}^{d_{cmax}} \frac{\rho_i}{i} \quad (24)$$

$[C_3]$ Convergence constraint, see (9)

$$F(\boldsymbol{\lambda}, \boldsymbol{\rho}, \boldsymbol{\sigma}^2, x) > x \quad (25)$$

$[C_4]$ Stability condition, see (18) and (19)

$$\sum_{j=1}^{N_s} \sum_{k=1}^{N_c} \lambda_{M_j,2}^{C_k} < \left[\sum_{j=1}^{N_s} \beta_j e^{-1/2\sigma_j^2} \cdot \sum_{m=2}^{d_{cmax}} \rho_m(m-1) \right]^{-1} \quad (26)$$

$[C_5]$ Minimum variable node degree constraint

$$\forall i < d_{v_{min}}^{(k)}, \, \forall j : \lambda_{M_j,i}^{C_k} = 0 \quad (27)$$

$[C_6]$ Previous optimization constraints

$$\forall k' < k, \, \forall j : \boldsymbol{\lambda}_{M_j}^{C_{k'}} \text{ is fixed} \quad (28)$$

   b) If failure, $d_{v_{min}}^{(k)} = d_{v_{min}}^{(k)} - 1$
End (While)

C. Code Construction

When the optimal degree distribution of the variable nodes is found, a parity-check matrix is constructed by a modification of the approximate cycle extrinsic (ACE) message degree algorithm [32]. The ACE algorithm constructs a parity-check matrix following a given variable node degree distribution while selectively avoiding small cycle clusters that are isolated from the rest of the graph. For irregular LDPC codes, the ACE algorithm has good performance in the error-floor region.

The original ACE algorithm [32] only ensures a certain variable node degree distribution. However, the design algorithm optimizes the variable node degree distribution given a certain check node degree distribution and a parity-check

TABLE I
DEGREE DISTRIBUTIONS OF THE UEP SCHEME.

	C_1	C_2	C_3
$\epsilon = 0$ dB	$\lambda_7 = 0.0799$ $\lambda_8 = 0.0948$ $\lambda_{30} = 0.3029$	$\lambda_3 = 0.1790$ $\lambda_6 = 0.0737$ $\lambda_7 = 0.0414$	$\lambda_2 = 0.2103$ $\lambda_3 = 0.0181$
$\epsilon = 0.1$ dB	$\lambda_{11} = 0.1783$ $\lambda_{12} = 0.1184$ $\lambda_{30} = 0.2183$	$\lambda_3 = 0.2041$ $\lambda_4 = 0.0393$	$\lambda_2 = 0.1841$ $\lambda_3 = 0.0575$

TABLE II
DEGREE DISTRIBUTIONS OF THE HOC-UEP SCHEME.

		C_1	C_2	C_3
$\epsilon = 0$ dB	M_1	$\lambda_9 = 0.1703$ $\lambda_{10} = 0.0555$ $\lambda_{30} = 0.1811$	$\lambda_3 = 0.1673$	$\lambda_2 = 0.1240$
	M_2	$\lambda_{30} = 0.0854$	$\lambda_4 = 0.0225$ $\lambda_5 = 0.0738$ $\lambda_7 = 0.0117$	$\lambda_2 = 0.0878$ $\lambda_3 = 0.0022$ $\lambda_4 = 0.0183$
$\epsilon = 0.1$ dB	M_1	$\lambda_{12} = 0.3290$ $\lambda_{30} = 0.1782$	$\lambda_3 = 0.1070$	$\lambda_2 = 0.1585$
	M_2		$\lambda_3 = 0.0396$ $\lambda_4 = 0.1026$ $\lambda_5 = 0.0165$	$\lambda_2 = 0.0547$ $\lambda_3 = 0.0137$
$\epsilon = 0.2$ dB	M_1	$\lambda_{15} = 0.4840$ $\lambda_{30} = 0.0327$	$\lambda_3 = 0.0848$	$\lambda_2 = 0.1733$
	M_2		$\lambda_3 = 0.0782$ $\lambda_4 = 0.0940$	$\lambda_2 = 0.0414$ $\lambda_3 = 0.0115$
$\epsilon = 0.3$ dB	M_1	$\lambda_{16} = 0.4993$	$\lambda_3 = 0.0268$	$\lambda_2 = 0.2163$
	M_2	$\lambda_{16} = 0.0345$	$\lambda_3 = 0.1849$ $\lambda_4 = 0.0291$	$\lambda_2 = 0.0092$

matrix with degrees given by $\boldsymbol{\lambda}$ and $\boldsymbol{\rho}$ is desired. The modified ACE construction of the parity-check matrix used here also ensures that the check node degree distribution equals $\boldsymbol{\rho}$. Whereas the ones in a column are located at random positions in the original ACE algorithm, we allow only those positions where a one does not violate the defined check node degree distribution.

Generally, the recently proposed ACE constrained progressive edge growth (PEG) code construction algorithm [33] has been shown to perform well compared to the ACE algorithm, especially in the error-floor region. Despite this good performance, it is unsuitable for UEP applications, since it entirely looses its UEP capability after a few decoder iterations.

IV. SIMULATION RESULTS

In this section, simulation results for examples with 8-PSK and 64-QAM are presented. We denote our scheme by "HOC-UEP", which is a UEP-LDPC code optimized for the different σ_j^2 from the modulation. We compare the HOC-UEP scheme to two other schemes. The first one is a UEP-LDPC code optimized for BPSK [22] which is denoted by "UEP", designs the code for a BPSK channel with the comparable σ_{BPSK}^2 (see Section II-B) and assigns the best bits from the modulation to the first protection class and vice versa. The second comparison is a design without UEP similar to the design proposed in [14]. The variable node degree distributions are optimized for $R = 1/2$, $N_c = 3$, $\boldsymbol{\alpha} = [0.3, \; 0.7]$, $d_{v_{max}} = 30$, and $\rho(x) = 0.00749x^7 + 0.99101x^8 + 0.00150x^9$, which is found by numerical optimization in [7] to be a good check node degree distribution for $d_{v_{max}} = 30$.

A. Results for 8-PSK

For Gray-labeled 8-PSK, the noise vector $\boldsymbol{\sigma}^2$ is calculated according to (4), with $N_s = 2$ and $\boldsymbol{\beta} = [2/3, \; 1/3]$. Table I shows the degree distributions of the UEP scheme. We choose $\epsilon = 0.1$ dB to allow for some increased UEP capability. The resulting degree distributions $\boldsymbol{\lambda}^{C_k}$ are given for each protection class C_k. For comparison, the degree distribution for the minimum threshold, that is $\epsilon = 0$ dB, is also shown. This degree distribution corresponds to the code design with inherent UEP only. The degree distributions of the HOC-UEP scheme, $\boldsymbol{\lambda}_{M_j}^{C_k}$, are given in Table II.

For comparison, we also optimize a code for HOCs but without UEP capability, which is similar to the approach in [14]. Note that we separate the degree distributions for information bits and parity bits, which is not done in [14]. The degree distributions are shown in Table III.

TABLE III
HOC DESIGN WITHOUT UEP SIMILAR TO [14].

		Information bits		Parity bits	
$\epsilon = 0$ dB	M_1	$\lambda_3 = 0.1474$ $\lambda_5 = 0.0021$ $\lambda_7 = 0.0023$ $\lambda_9 = 0.2204$ $\lambda_{11} = 0.0008$ $\lambda_{13} = 0.0002$ $\lambda_{15-25} = 0.0013$ $\lambda_{27} = 0.0003$ $\lambda_{29} = 0.0011$	$\lambda_4 = 0.0008$ $\lambda_6 = 0.0008$ $\lambda_8 = 0.0005$ $\lambda_{10} = 0.0040$ $\lambda_{12} = 0.0004$ $\lambda_{14} = 0.0002$ $\lambda_{26} = 0.0002$ $\lambda_{28} = 0.0004$ $\lambda_{30} = 0.1751$	$\lambda_2 = 0.1237$ $\lambda_3 = 0.0154$	
	M_2	$\lambda_3 = 0.0019$ $\lambda_5 = 0.0581$ $\lambda_7 = 0.0008$ $\lambda_9 = 0.0017$ $\lambda_{11} = 0.0003$ $\lambda_{13} = 0.0001$ $\lambda_{15-25} = 0.0008$ $\lambda_{27} = 0.0002$ $\lambda_{29} = 0.0005$	$\lambda_4 = 0.0509$ $\lambda_6 = 0.0049$ $\lambda_8 = 0.0057$ $\lambda_{10} = 0.0006$ $\lambda_{12} = 0.0002$ $\lambda_{14} = 0.0001$ $\lambda_{26} = 0.0001$ $\lambda_{28} = 0.0003$ $\lambda_{30} = 0.0869$	$\lambda_2 = 0.0881$ $\lambda_3 = 0.0005$	

Finite-length codeword simulations with $n = 4096$ and 50 decoding iterations are performed using the equivalent BPSK channels. A soft demapper provides the message passing decoder with the channel log-likelihood ratios (LLRs) which are computed using the appropriate noise variances σ_j^2 of the modulation classes.

Fig. 4 shows the overall BER after 50 decoder iterations averaged over all protection classes for both $\epsilon = 0$ dB and $\epsilon = 0.1$ dB. It is seen that the overall BERs of the HOC-UEP codes are lower than for the UEP codes. For an overall BER of 10^{-3}, there is a gain of around 0.8 dB by the HOC-UEP scheme. This is due to the consideration of the different σ_j^2 in contrast to only assuming the average σ_{BPSK}^2. The overall BER for the design similar to the approach in [14] (HOC without UEP) is shown for comparison, even though this scheme has no UEP capability. It is expected that the overall BERs for the codes with $\epsilon > 0$ dB are higher than for the corresponding codes with $\epsilon = 0$ dB, because the thresholds of the codes are increased in order to allow an increased average variable node degree of the most protected classes. However, at high E_b/N_0 the HOC-UEP scheme shows a slightly lower overall BER for the case with $\epsilon = 0.1$ dB compared to $\epsilon = 0$ dB. Reasons for this are discussed later in this section.
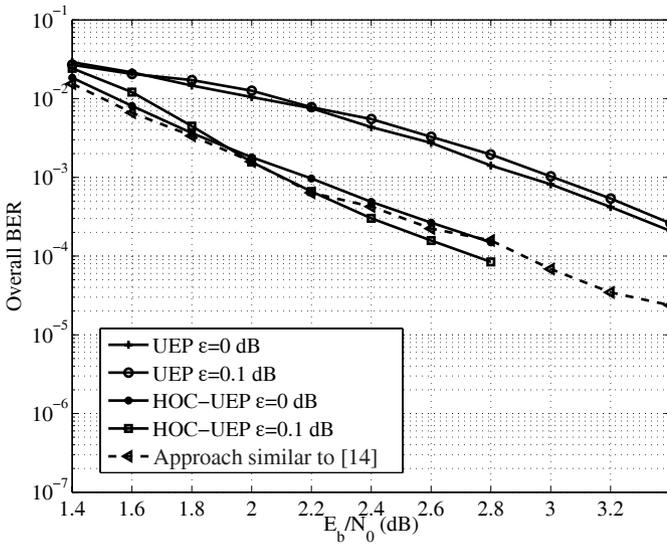
Fig. 4. Overall bit-error rate performance of the UEP scheme and the HOC-UEP scheme for $\epsilon = 0$ dB and $\epsilon = 0.1$ dB.
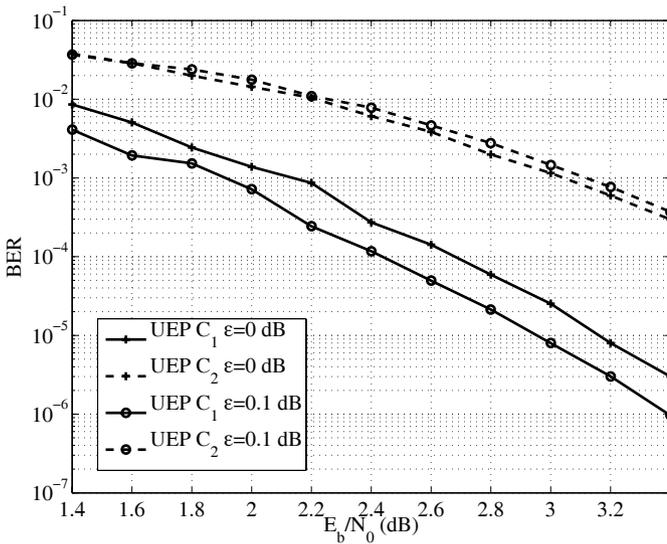

Fig. 6. Bit-error rate performance of protection class C_1 and C_2 for the HOC-UEP scheme and 8-PSK.

Fig. 5. Bit-error rate performance of protection class C_1 and C_2 for the UEP scheme and 8-PSK.

The BER performance of the individual protection classes C_1 and C_2 for the UEP scheme are shown in Fig. 5. Note that we do not show protection class C_3 because it contains parity bits only and is of little interest for possible applications. As expected, the UEP capability is increased with increasing ϵ.

Fig. 6 shows the BER performance of protection classes C_1 and C_2 for the HOC-UEP scheme. The results show that the HOC-UEP $\epsilon = 0.1$ dB code has lower BER for both information classes than the HOC-UEP $\epsilon = 0$ dB code. Note that the differences in BER are partly balanced by protection class C_3 which is not shown here. By comparing Fig. 5 and Fig. 6, we see that the BERs of the HOC-UEP scheme are much lower than those of the UEP scheme at high E_b/N_0. The gain of the HOC-UEP scheme compared to the UEP scheme is 0.8 dB for protection classes C_1 and C_2, at a BER of 10^{-6} and 10^{-4}, respectively. We also see that the difference in performance between the protection classes is higher for the HOC-UEP scheme than for the UEP scheme, especially
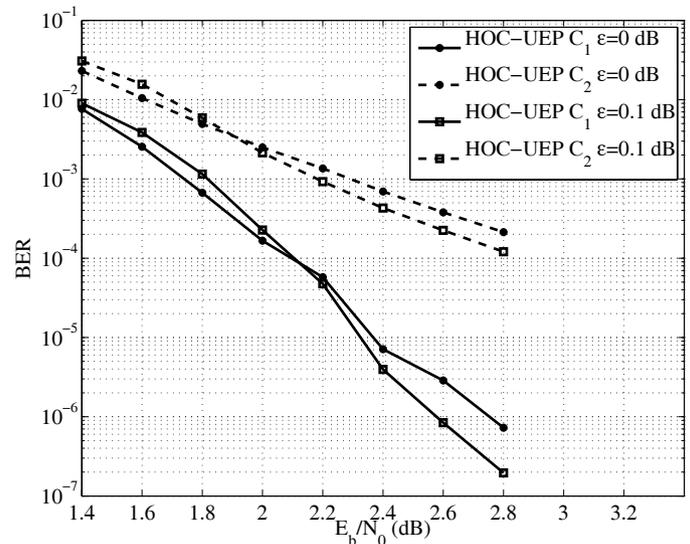
at high E_b/N_0. Because of the large difference in BER of the protection classes and also the knowledge of the proportion of bits in the classes, we know that it is mainly C_2 that governs the performance in terms of the overall BER. By comparing also with Fig. 4, it is seen that the overall BER of the approach similar to [14] without UEP is almost equal to the overall BER of the HOC-UEP scheme with $\epsilon = 0$ dB and worse than the HOC-UEP scheme with $\epsilon = 0.1$ dB. When comparing these schemes, remember also that the HOC-UEP scheme provides a BER significantly lower than the overall BER for C_1.

Fig. 7 shows the effect of different threshold offsets in the HOC-UEP scheme, with BER as a function of ϵ for $E_b/N_0 = 2.4$ dB. It shows that $\epsilon = 0.1$ dB is a good choice for our example, since the BERs for both classes are lower than for code design without increased UEP capability ($\epsilon = 0$ dB). For higher values of ϵ, the first class is assigned even more edges and the second and third classes have concentrated low degrees. With increasing ϵ, which implies increasing the code threshold, the global convergence of the code becomes worse and this affects the performance of all classes. Please note that for short-length codes, the degree distribution with the lowest threshold may not yield the best performance [7].

Fig. 8 shows the BER of the HOC-UEP scheme as a function of the number of decoder iterations for $E_b/N_0 = 2.4$ dB. The UEP capability is not affected much with increasing number of decoder iterations, at least up to a maximum of 200 iterations.

B. Results for 64-QAM

In this section, we give an example of a UEP-LDPC code designed for and used with 64-QAM. We design a code with the parameters n, k, R, N_c, $\boldsymbol{\alpha}$, $\boldsymbol{\rho}$, and $d_{v_{max}}$ as in the previous section. 64-QAM yields three modulation classes of equal size, and thus, $N_s = 3$ and $\boldsymbol{\beta} = [1/3, \ 1/3, \ 1/3]$. Fig. 9 shows the BER performance of protection classes C_1 and C_2 of the 64-QAM HOC-UEP code compared to a UEP code designed for BPSK, both with $\epsilon = 0.1$ dB and transmitted over the 64-QAM modulated channel. The plot shows that for very
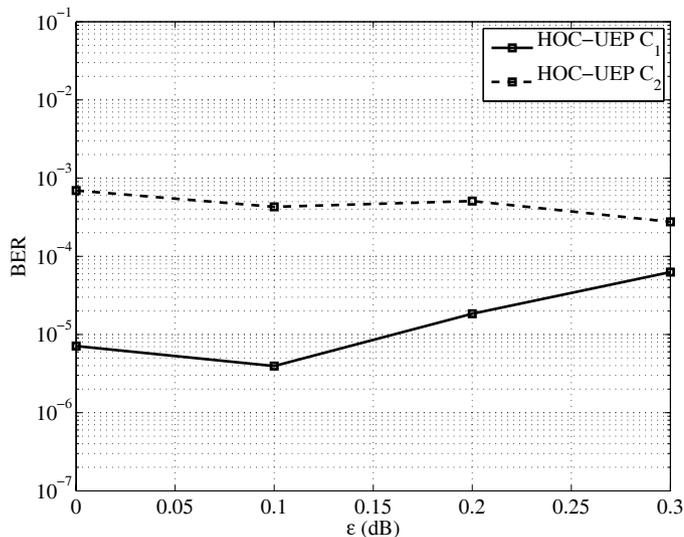
Fig. 7. Performance of the HOC-UEP scheme for $E_b/N_0 = 2.4$ dB for various values of ϵ and 8-PSK.
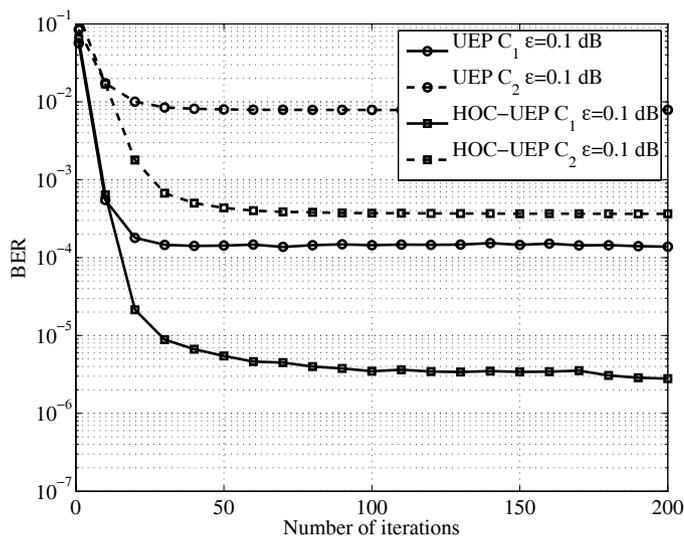

Fig. 8. Bit-error rate performance as a function of the number of decoder iterations for $E_b/N_0 = 2.4$ dB and 8-PSK.

low SNR, the BERs of the HOC-UEP scheme are slightly worse but outperform the UEP scheme for an SNR higher than approximately 1.7 dB. For a BER of $2 \cdot 10^{-5}$, HOC-UEP C_1 has an E_b/N_0 gain of 1.2 dB compared to UEP C_1. This result confirms the assumption that a higher modulation index yields greater performance gain for the HOC-UEP scheme compared to the UEP scheme.

V. CONCLUSIONS

We present a flexible design method for UEP-LDPC codes with higher order constellations. The design algorithm is applicable to arbitrary signal constellations, an arbitrary number of protection classes and arbitrary protection class sizes. For an example with 8-PSK, it is shown that the overall BER is reduced by the proposed method and there is a gain of 0.8 dB at BER 10^{-3} compared to using the UEP-LDPC codes designed for BPSK. An example with 64-QAM shows an even higher gain. The results also show that the UEP capability is increased
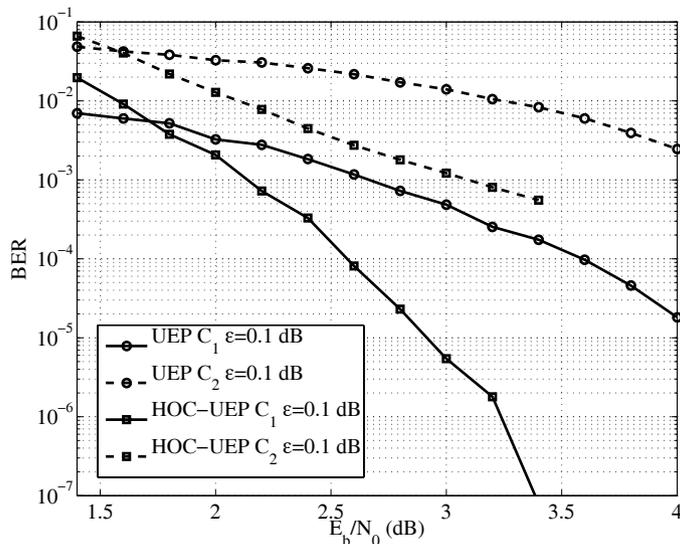

Fig. 9. Bit-error rate performance of the UEP and HOC-UEP schemes for 64-QAM modulation.

if codes are properly designed for the modulation scheme. The proposed code design is compared to a design method for higher order constellations without UEP and we show that the two methods have similar overall BER even though our scheme provides a protection class with BER significantly lower than the overall BER. The achieved UEP capability may be of major interest for a number of applications, e.g., when header information must be transmitted.

REFERENCES

[1] G. Ungerböck, "Channel coding with multilevel/phase signals," *IEEE Trans. Inf. Theory*, vol. 28, pp. 55-67, Jan. 1982.
[2] H. Imai and S. Hirakawa, "A new multilevel coding method using error correcting codes," *IEEE Trans. Inf. Theory*, vol. 23, pp. 371-377, May 1977.
[3] G. Ungerböck, "Trellis-coded modulation with redundant signal sets—part I: introduction," *IEEE Commun. Mag.*, vol. 25, pp. 5-11, Feb. 1987.
[4] G. Ungerböck, "Trellis-coded modulation with redundant signal sets—part II: state of the art," *IEEE Commun. Mag.*, vol. 25, pp. 12-21, Feb. 1987.
[5] U. Wachsmann, R. Fischer, and J. Huber, "Multilevel codes: theoretical concepts and practical design rules," *IEEE Trans. Inf. Theory*, vol. 45, pp. 1361-1391, July 1999.
[6] R. Gallager, "Low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 8, pp. 21-28, Jan. 1962.
[7] T. Richardson, M. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 619-637, Feb. 2001.
[8] M. Aydinlik and M. Salehi, "Turbo coded modulation for unequal error protection," *IEEE Trans. Commun.*, vol. 56, pp. 555-564, Apr. 2008.
[9] M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. 27, pp. 533-547, Sept. 1981.
[10] F. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, pp. 498-519, Feb. 2001.
[11] S.-Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inf. Theory*, vol. 47, pp. 657-670, Feb. 2001.
[12] J. Hou, P. Siegel, L. Milstein, and H. Pfister, "Capacity-approaching bandwidth-efficient coded modulation schemes based on low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 49, pp. 2141-2155, Sept. 2003.
[13] G. Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," *IEEE Trans. Inf. Theory*, vol. 44, pp. 927-946, May 1998.
[14] H. Sankar, N. Sindhushayana, and K. Narayanan, "Design of low-density parity-check (LDPC) codes for high order constellations," in *Proc. Globecom 2004*, pp. 3113-3117, Nov. 2004.

[15] A. Sanaei and M. Ardakani, "LDPC code design for nonuniform power-line channels," *EURASIP J. Advances Signal Process.*, vol. 2007, pp. Article ID 76146, 9 pages, 2007. doi:10.1155/2007/76146.

[16] Y. Li and W. Ryan, "Bit-reliability mapping in LDPC-coded modulation systems," *IEEE Commun. Lett.*, vol. 9, pp. 1-3, Jan. 2005.

[17] R. Maddock and A. Banihashemi, "Reliability-based coded modulation with low-density parity-check codes," *IEEE Trans. Commun.*, vol. 54, pp. 403-406, Mar. 2006.

[18] V. Kumar and O. Milenkovic, "On unequal error protection LDPC codes based on Plotkin-type constructions," *IEEE Trans. Commun.*, vol. 54, pp. 994-1005, June 2006.

[19] K. Kasai, T. Shibuya, and K. Sakaniwa, "Detailed representation of irregular LDPC code ensembles and density evolution," in *Proc. ISIT 2003*, p. 121, June 2003.

[20] C. Poulliat, D. Declercq, and I. Fijalkow, "Optimization of LDPC codes for UEP channels," in *Proc. ISIT 2004*, p. 451, June 2004.

[21] L. Sassatelli, W. Henkel, and D. Declercq, "Check-irregular LDPC codes for unequal error protection under iterative decoding," in *4th International Symp. Turbo-codes Related Topics* (D. D. L. Sassatelli, W. Henkel, ed.), 2006.

[22] C. Poulliat, D. Declercq, and I. Fijalkow, "Enhancement of unequal error protection properties of LDPC codes," *EURASIP J. Wireless Commun. Netw.*, vol. 2007, pp. Article ID 92659, 9 pages, 2007, doi:10.1155/2007/92659.

[23] N. Rahnavard, H. Pishro-Nik, and F. Fekri, "Unequal error protection using partially regular LDPC codes," *IEEE Trans. Commun.*, vol. 55, pp. 387-391, Mar. 2007.

[24] H. Pishro-Nik, N. Rahnavard, and F. Fekri, "Nonuniform error correction using low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 51, pp. 2702-2714, July 2005.

[25] K. Fazel and M. Ruf, "Combined multilevel coding and multiresolution modulation," in *Proc. ICC 1993*, pp. 1081-1085, May 1993.

[26] Y. Liu and C. Heneghan, "Optimizing scalable media content delivery using hierarchical modulation techniques," in *European Symp. on Mobile Media Delivery (EuMob)*, Sept. 2006.

[27] A. Sezgin, D. Wübben, R. Böhnke, and V. Kühn, "On EXIT-charts for space-time block codes," in *Proc. ISIT 2003*, p. 64, June 2003.

[28] Y. Nana, E. Sharon, and S. Lytsin, "Improved decoding of LDPC coded modulation," *IEEE Commun. Lett.*, vol. 10, pp. 375-377, May 2006.

[29] J. Hou, P. Siegel, L. Milstein, and D. Pfister, "Multilevel coding with low-density parity-check component codes," in *Proc. Globecom '01*, vol. 2, pp. 1016-1020, Nov. 2001.

[30] J. Proakis, *Digital Communications*. McGraw-Hill, 2001.

[31] N. von Deetzen and S. Sandberg, "On the UEP capabilities of several LDPC construction algorithms,"' to appear in *IEEE Trans. Commun.* [Online]. Available: http://www.ltu.se/staff/s/saras.

[32] T. Tian, C. Jones, D. Villasenor, and R. Wesel, "Selective avoidance of cycles in irregular LDPC code construction," *IEEE Trans. Commun.*, vol. 52, pp. 1242-1247, Aug. 2004.

[33] D. Vukobratovic and V. Senk, "Generalized ACE constrained progressive edge-growth LDPC code design," *IEEE Commun. Lett.*, vol. 12, pp. 32-34, Jan. 2008.

Sara Sandberg was born in Skellefteå, Sweden, in 1978. She received the M.S. degree in engineering physics in 2002 from Luleå University of Technology (LTU), Sweden and the Ph.D. degree in electrical engineering, also from LTU, in March 2009. Since April 2009, she has been with LTU as a researcher. Her research interests include coding theory and wireless communications.

Neele von Deetzen was born in Oldenburg, Germany, in 1980. She received her Diploma (M.Sc.) in electrical engineering from the University of Bremen, Germany, in February 2005, and her Ph.D. degree from Jacobs University Bremen, Germany, in January 2009. Since March 2009, she has been with Silver Atena Electronic Systems Engineering GmbH.