Multi-Edge Type Unequal Error Protecting Low-Density Parity-Check Codes

H. V. Beltrão Neto, W. Henkel
Jacobs University Bremen
Campus Ring 1
D-28759 Bremen, Germany
Email: {h.beltrao, w.henkel}@jacobs-university.de

V. C. da Rocha Jr.
Department of Electronics and Systems
Federal University of Pernambuco
P.O. Box 7800 Recife, Brazil, 50711-970
Email: vcr@ufpe.br

Abstract—**Irregular low-density parity-check (LDPC) codes are particularly well-suited for transmission schemes that require unequal error protection (UEP) of the transmitted data due to the different connection degrees of its variable nodes. However, this UEP capability is strongly dependent on the connection profile among the protection classes. This paper applies a multi-edge type analysis of LDPC codes for optimizing such a connection profile according to the performance requirements of each protection class. This allows the construction of UEP-LDPC codes where the difference between the performance of the protection classes can be adjusted and with an UEP capability that does not vanish as the number of decoding iterations grows.**

I. INTRODUCTION

In communication systems where source bits with different sensitivities to errors are being transmitted, it is often wasteful or even infeasible to provide uniform protection for the whole data stream. In this scenario, the common strategy is the use of schemes with unequal error protection (UEP) capabilities. There are mainly three strategies to achieve UEP on transmission systems: bit loading, multilevel coded modulation, and channel coding [1]. In this paper, we will focus on the latter, more specifically on low-density parity-check (LDPC) codes that provide inherent unequal error protection within a codeword as explored in [2] and [3], for example.

Irregular LDPC codes [4] can inherently provide unequal error protection due to the different connection degrees of the coded bits. The connection degrees of the variable and check nodes of such codes are defined by the polynomials $\lambda(x) = \sum_{i=2}^{d_{v_{max}}} \lambda_i x^{i-1}$ and $\rho(x) = \sum_{i=2}^{d_{c_{max}}} \rho_i x^{i-1}$, where $d_{v_{max}}$ and $d_{c_{max}}$ are the maximum variable and check node degrees of the code. From now on, we will refer to irregular LDPC codes where the variable nodes are divided into disjoint sets called protection classes as unequal error protecting LDPC codes (UEP-LDPC). A flexible optimization of the irregularity profile of irregular LDPC codes based on a hierarchical optimization of the variable node degree distribution was proposed in [3], where the authors interpret the UEP properties of an LDPC code as different local convergence speeds, i.e., the most protected bits are assigned to the bits in the codeword which converge to their right value in the smallest number of iterations. This assumption is made in order to cope with the observation that the UEP gradation vanishes as the number of iterations grow, a fact also observed in [5]. In [6], the

authors observed that this vanishing UEP gradation of an iteratively decoded LDPC code is dependent on the algorithm used to construct the parity check matrix, and suggested that the connectivity between the classes is the key factor to be observed if the UEP capabilities should be held as the number of iterations grows.

Herein, we propose an optimization algorithm for the connectivity profile between the different protection classes of LDPC codes in order to not only keep the UEP capability of a code for a moderate to large number of decoding iterations, but also to adjust the performance of the protection classes as required for different applications. This is achieved by means of a multi-edge type (MET) analysis [7], [8] of the LDPC codes. The multi-edge analysis enables us to distinguish between the messages exchanged during the iterative decoding among the different protection classes within one codeword. Thus, we can control the amount of information that the most protected classes receive from the less protected ones and vice versa. If the most protected classes receive a lot of information from the less protected ones, its performance will be decreased while the one of the less protected classes will be enhanced. Our main goal is to show how this exchange of performance among the protection classes can be controlled and optimized.

This paper is organized as follows. In Section II, we describe the multi-edge type analysis of UEP-LDPC codes. Section III discusses the asymptotic analysis of multi-edge type UEP-LDPC codes and the optimization algorithm used to optimize the connection profile between the protection classes. In Section IV, we show the results of the developed optimization method for a chosen example. Finally, some concluding remarks are drawn in Section V.

II. MULTI-EDGE TYPE UNEQUAL ERROR PROTECTING LDPC CODES

A. Multi-edge LDPC codes

Multi-edge type LDPC codes [7] are a generalization of irregular and regular LDPC codes. Diverting from standard LDPC ensembles where the graph connectivity is constrained only by the node degrees, in the multi-edge setting, several edge classes can be defined and every node is characterized by the number of connections to edges of each class. Within this framework, the code ensemble can be specified through
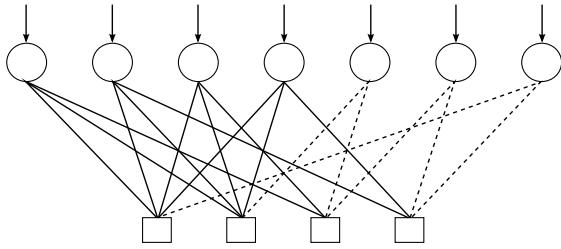
Fig. 1. Multi-edge graph with two different edge types and one received distribution.

two multinomials associated to variable and check nodes. The two multinomials are defined by [8]

$$L(\mathbf{r}, \mathbf{x}) = \sum L_{\mathbf{b}, \mathbf{d}} \mathbf{r}^{\mathbf{b}} \mathbf{x}^{\mathbf{d}} \quad \text{and} \quad R(\mathbf{x}) = \sum R_{\mathbf{d}} \mathbf{x}^{\mathbf{d}}, \quad (1)$$

where $\mathbf{b}$, $\mathbf{d}$, $\mathbf{r}$, and $\mathbf{x}$ are vectors which are explained as follows. First, let m_e denote the number of edge types used to represent the graph ensemble and m_r the number of different received distributions. The number m_r represents the fact that the different bits can go through different channels and thus, have different received distributions. Each node in the ensemble graph has associated to it a vector $\mathbf{x} = (x_1, ..., x_{m_e})$ that indicates the different types of edges connected to it, and a vector $\mathbf{d} = (d_1, ..., d_{m_e})$ referred to as *edge degree vector* which denotes the number of connections of a node to edges of type i, where $i \in (1, \ldots, m_e)$.

For the variable nodes, there is additionally the vector $\mathbf{r} = (r_1, ..., r_{m_r})$ which represents the different received distributions[1], and the vector $\mathbf{b} = (b_0, ..., b_{m_r})$ that indicates the number of connections to the different received distributions (b_0 is used to indicate the puncturing of a variable node). In the sequel, we assume that $\mathbf{b}$ has exactly one entry set to 1 and the rest set to zero. This simply indicates that each variable node has access to only one channel observation at a time. We use $\mathbf{x}^{\mathbf{d}}$ to denote $\prod_{i=1}^{m_e} x_i^{d_i}$ and $\mathbf{r}^{\mathbf{b}}$ to denote $\prod_{i=0}^{m_r} r_i^{b_i}$. Finally, the coefficients $L_{\mathbf{b}, \mathbf{d}}$ and $R_{\mathbf{d}}$ are non-negative reals such that if n is the total number of variable nodes, $L_{\mathbf{b}, \mathbf{d}} n$ and $R_{\mathbf{d}} n$ represent the number of variable nodes of type ($\mathbf{b}, \mathbf{d}$) and check nodes of type[2] $\mathbf{d}$, respectively. Furthermore, we have the additional notations defined in [8]

$$L_{x_j}(\mathbf{r}, \mathbf{x}) = \frac{\mathrm{d}L(\mathbf{r}, \mathbf{x})}{\mathrm{d}x_j} \quad \text{and} \quad R_{x_j}(\mathbf{x}) = \frac{\mathrm{d}R(\mathbf{x})}{\mathrm{d}x_j}. \quad (2)$$

Unequal error protecting LDPC codes can be included in a multi-edge framework in a straightforward way. This can be done by distinguishing between the edges connected to different protection classes within a codeword. According to this strategy, the edges connected to variable nodes within a

[1]In the multi-edge framework, one can consider that the different variable node types may have different received distributions, i.e., the associated bits may be transmitted through different channels. In this work, we consider that the variable nodes have access solely to one observation and that the transmission is made through an AWGN channel.

[2]We will frequently refer to nodes with edge degree vector $\mathbf{d}$ as "type $\mathbf{d}$" nodes.

protection class are all of the same type. For example, in Fig. 1 the first 4 variable nodes can be seen as one protection class since they are connected only to type 1 edges (depicted by solid lines), and the last 3 variable nodes compound another protection class, since they are only connected to type 2 edges (depicted by the dashed lines). The variable and check node multinomials for this example are given by

$$L(\mathbf{r}, \mathbf{x}) = \frac{4}{7} r_1 x_1^3 + \frac{3}{7} r_1 x_2^2, \qquad R(\mathbf{x}) = \frac{2}{7} x_1^4 x_2 + \frac{2}{7} x_1^2 x_2^2 .$$

It is worth noting that as opposed to the variable nodes, the check nodes admit connections with edges of different types simultaneously as can be inferred from Fig.1. In the following, we will divide the variable nodes into m_e protection classes $(C_1, C_2, \ldots, C_{m_e})$ with decreasing levels of protection.

B. Edge perspective notation

The connection between the protection classes occurs through the check nodes since they can have different types of edges attached to them. Consider irregular LDPC codes with node perspective variable and check node multi-edge multinomials $L(\mathbf{r}, \mathbf{x}) = \sum L_{\mathbf{b}, \mathbf{d}} \mathbf{r}^{\mathbf{b}} \mathbf{x}^{\mathbf{d}}$ and $R(\mathbf{x}) = \sum R_{\mathbf{d}} \mathbf{x}^{\mathbf{d}}$, respectively. In this paper, we consider unpunctured codes and that the variable nodes have access to only one observation, i.e., $\mathbf{b} = (0, 1)$. Also, variable node within C_j are only connected to edges of type j.

In order to implement the optimization algorithm, it will be more convenient to work with the edge, instead of the node perspective. We now define the following edge perspective multi-edge multinomials

$$\lambda^{(j)}(\mathbf{r}, \mathbf{x}) = \frac{L_{x_j}(\mathbf{r}, \mathbf{x})}{L_{x_j}(\mathbf{1}, \mathbf{1})} = r_1 \sum_i \lambda_i^{(j)} x_j^{i-1}, \quad (3)$$

$$\rho^{(j)}(\mathbf{x}) = \frac{R_{x_j}(\mathbf{r}, \mathbf{x})}{R_{x_j}(\mathbf{1}, \mathbf{1})} = \sum_{\mathbf{d}} \rho_{\mathbf{d}}^{(j)} \mathbf{x}^{\mathbf{d}'} x_j^{d_j - 1}, \quad (4)$$

where $\lambda_i^{(j)}$ denote the fraction of type j edges connected to variable nodes of degree i, $\rho_{\mathbf{d}}^{(j)}$ denote the fraction of type j edges connected to check nodes with edge degree vector $\mathbf{d}$, $\mathbf{x}^{\mathbf{d}'} = \prod_{i=1}^{m_e} x_i^{d_i}$ with $d_j = 0$, and $\mathbf{1}$ denotes a vector with all entries equal to 1 with length being clear from the context.

In the next section, we will use Eqs. (3) and (4) in the derivation of the optimization algorithm for the connection profile among the protection classes of an UEP-LDPC code.

III. CHECK NODE PROFILE OPTIMIZATION

A. Asymptotic Analysis

Our main objective is, given the overall variable ($\lambda(x)$) and check node ($\rho(x)$) degree distributions of an UEP-LDPC code, to optimize the connection profiles between the different protection classes in order to control the amount of protection of each class while preserving the UEP capability of the code after a moderate to high number of decoding iterations. The described optimization algorithm we derive here can be applied for any irregular pair of degree distributions. However, in order to reduce the search space of the optimization algorithm,

we suppose from now on that the LDPC code to be optimized is check-regular, i.e., all the check nodes have the same degree.

Despite of having the same degree, each check node may have a different number of edges belonging to each one of the m_e classes. Consider for example a check node with an associated edge degree vector $\mathbf{d} = (d_1, d_2, \ldots, d_{m_e})$, where d_i is the number of connections to the protection class i and $\sum_{i=1}^{m_e} d_i = d_{c_{max}}$. If we then consider a code with $m_e = 3$ protection classes, each check node may be connected to d_1 edges of class 1, d_2 edges of class 2, and d_3 edges of class 3. This posed, one can compute the evolution of the iterative decoding by means of density evolution. We assume here standard belief propagation decoding of LDPC codes where the messages exchanged between the variable and check nodes are independent log-likelihood ratios having a symmetric Gaussian distribution (variance equals twice the mean).

Let $I_{v,l}^{(j)}$ and $I_{c,l}^{(j)}$ denote the mutual information between the messages sent through edges of class j at the output of variable and at the output of check nodes at iteration l and the associated codeword bit, respectively. Assuming Gaussian approximation [9] and noting that, for optimizing the connection profile between the protection classes, we need to consider the case where check nodes with different edge degree vector $\mathbf{d}$ are allowed we have

$$I_{v,l}^{(j)} = \sum_{i=2}^{d_{v_{max}}} \lambda_i^{(j)} J\left(\sqrt{4/\sigma^2 + (i-1)[J^{-1}(I_{c,l-1}^{(j)})]^2}\right), \quad (5)$$

$$I_{c,l}^{(j)} = 1 - \sum_{i=1}^{d_{c_{max}}} \sum_{\mathbf{d}:d_j=i} \rho_{\mathbf{d}}^{(j)}$$
$$\times J\left(\sqrt{(d_j-1)J^{-1}(1-I_{v,l}^{(j)})^2 + \sum_{s\neq j} d_s J^{-1}(1-I_{v,l}^{(s)})^2}\right), \quad (6)$$

with the $J(.)$ function defined as in [10]

$$J(\sigma) = 1 - \int_{-\infty}^{\infty} \frac{e^{\frac{(\xi-\sigma^2/2)^2}{2\sigma^2}}}{\sqrt{2\pi}\sigma} \times \log_2[1 + e^{-\xi}]d\xi . \quad (7)$$

Combining Eqs. (5) and (6), one can summarize the density evolution as a function of the mutual information of the previous iteration, the mutual information contribution from the other classes, noise variance, and degree distributions, i.e.,

$$I_{v,l}^{(j)} = F(\boldsymbol{\lambda}^{(j)}, \boldsymbol{\rho}_{\mathbf{d}}^{(j)}, \sigma^2, I_{v,l-1}^{(j)}, \boldsymbol{I}_{v,l-1}), \quad (8)$$

where the bold symbols represent sequences of values defined as $\boldsymbol{\lambda}^{(j)} = \{\lambda_i^{(j)}\}_{i=2}^{d_{v_{max}}}$, $\boldsymbol{\rho}_{\mathbf{d}}^{(j)} = \{\rho_{\mathbf{d}:d_j=i}^{(j)}\}_{i=1}^{d_{c_{max}}}$ and, $\boldsymbol{I}_{v,l-1} = \{I_{v,l-1}^{(s)}\}_{s=1}^{m_e}$ with $s \neq j$. By means of Eq. (8), we can predict the convergence behavior of the decoding and than optimize the degree distribution $\rho^{(j)}(\mathbf{x})$ under the constraint that the mutual information shall be increasing as the number of iterations grow, i.e.,

$$F(\boldsymbol{\lambda}^{(j)}, \boldsymbol{\rho}_{\mathbf{d}}^{(j)}, \sigma^2, I^{(j)}, \boldsymbol{I}) > I^{(j)}. \quad (9)$$

At this point, it is worth noting that, as pointed out in [5] and [6], the UEP capabilities of a code depend on the amount of connection among the protection classes, i.e., if the most protected class is well connected to the least protected one, the performance of the former will decrease while the performance of the latter will be improved. For example, suppose a code with 2 protection classes and $d_{c_{max}} = 4$. The possible values for $\mathbf{d} = (d_1, d_2)$ are (0,4), (1,3), (2,2), (3,1), and (4,0). On the one hand, if a code has a majority of check nodes with $\mathbf{d} = (4, 0)$, the first protection class will be very isolated from class 2 which will lead to an enhanced performance difference between the two classes. On the other hand, if a large amount of the check nodes are of type $\mathbf{d} = (2, 2)$, one can expect the protection classes to be very connected, which favors the overall performance but mitigates the UEP capability of a code at a moderate to large number of decoding iterations. This indicates that for controlling the UEP capability of an LDPC code and to prevent this characteristic from vanishing as the number of decoding iterations grows, one has to control the amount of check nodes of each type, i.e., optimize $\rho^{(j)}(\mathbf{x})$.

These observations about the influence of the connection between the protection classes and its UEP characteristics can be further analyzed by means of a detailed computation of the mutual information which may be performed by considering the edge-based mutual information messages traversing the graph instead of node-based averages. Such an analysis has been done for protographs in [11] and applied for the analysis of UEP-LDPC codes in [6].

B. Optimization Algorithm

The algorithm described here aims at optimizing the connection profile between the various protection classes present on a UEP-LDPC code, i.e., $\rho^{(j)}(\mathbf{x})$. Initially, the algorithm computes the variable node degree distribution of each class $\lambda^{(j)}(x)$ based on $\lambda(x)$, $d_{c_{max}}$, and the number of edges on each class. The algorithm then proceeds sequentially optimizing the connection profile to the check nodes of one class at a time, proceeding from the less protected class to the most protected one. This scheduling is done in order to control the amount of messages coming up from the less protected classes that are forwarded to the more protected ones.

Since we are using linear programming (LP) with only a single objective function, we chose it to be the minimization of the average check node degree within the class being optimized, i.e., it minimizes the average number of edges of such a class connected to the check nodes. This minimization aims at diminishing the amount of unreliable messages (i.e., the ones coming up from the less protected variable nodes) that flows through a check node. In addition to it, we control the proportion of check nodes of type $\mathbf{d}$ introducing the parameter $max\rho_{\mathbf{d}}^{(j)}$ which is an upper bound on $\sum_{\mathbf{d}:d_j=s} \rho_{\mathbf{d}}^{(j)}$, i.e., it limits the proportion of check nodes of type $\mathbf{d} : d_j = s$ for $s = 0, \ldots, d_{c_{max}}$, thus regulating the degree of connection among the protection classes.

The optimization is then performed for each class C_j by minimizing its average check node degree for a decreasing

$d_{min}^{(j)}$ from $d_{c_{max}}$ to 1, where $d_{min}^{(j)}$ is the minimum number of edges of class j connected to a check node. At this point, one can argue that since our goal is to minimize the average connection degree within a protection class, we should thus set $d_{min}^{(j)} = 1$. The problem with this strategy is that it would shorten the degree of freedom for the optimization of the next class, e.g., suppose the optimization of a two-class code with $d_{min}^{(2)} = 3$ and $d_{c_{max}} = 5$. Once we proceed to the optimization of class two, the coefficients $\rho_{(2,3)}^{(2)}$, $\rho_{(1,4)}^{(2)}$, and $\rho_{(0,5)}^{(2)}$ are determined and consequently fixed for the next optimization step, i.e., the optimization of class 1, we will have as variables only the coefficients $\rho_{(3,2)}^{(1)}$, $\rho_{(4,1)}^{(1)}$, and $\rho_{(5,0)}^{(1)}$. Note that in this case, if we had set $d_{min}^{(2)} = 1$, there will be no degree of freedom for optimizing class 1 since the only non-optimized $\mathbf{d}$ would be $\mathbf{d} = (5,0)$ which would be determined by $\sum_{s=d_{min}^{(j)}}^{d_{c_{max}}} \sum_{\mathbf{d}:d_j=s} \rho_{\mathbf{d}}^{(j)} = 1$, i.e., the sum of all fraction of edges must be equal to one.

The iterative procedure is successful, when a solution $\rho_{\mathbf{d}}^{(j)}$ is found which converges for the given σ^2 and $d_{min}^{(j)} > 0$. We assume that the optimizations for classes $\{C_{j'}, j' < j\}$ have already been performed and the results of these optimizations are used as constraints in the current optimization process. The optimization algorithm can be written, for given $\lambda(x)$, σ^2, m_e, $d_{c_{max}}$, and $max\boldsymbol{\rho}_{\mathbf{d}}^{(j)}$ for $j = 1, \ldots, m_e$ as shown in Fig. 2.

---

1) Compute $\lambda^{(j)}(x)$
2) Initialization $d_{min}^{(j)} = d_{c_{max}}$
3) While optimization failure
   a) minimize the average check node degree
     $\sum_{s=d_{min}^{(j)}}^{d_{c_{max}}} s \times \sum_{\mathbf{d}:d_j=s} \rho_{\mathbf{d}}^{(j)}$ under the following constraints,

    $\mathcal{C}_1 : \sum_{s=d_{min}^{(j)}}^{d_{c_{max}}} \sum_{\mathbf{d}:d_j=s} \rho_{\mathbf{d}}^{(j)} = 1,$

    $\mathcal{C}_2 : \sum_{\mathbf{d}:d_j=s} \rho_{\mathbf{d}}^{(j)} \in [0, max\boldsymbol{\rho}_{\mathbf{d}}^{(j)}],$

    $\mathcal{C}_3 : F(\boldsymbol{\lambda}^{(j)}, \boldsymbol{\rho}_{\mathbf{d}}^{(j)}, \sigma^2, x, \boldsymbol{I}) > x,$
      $\forall x \in [0,1),$

    $\mathcal{C}_4 : \forall j' > j$ and $\mathbf{d} : d_{min}^{(j')} \leq d_{j'} \leq d_{c_{max}},$
      $\rho_{\mathbf{d}}^{(j)}$ is fixed.

   b) $d_{min}^{(j)} = d_{min}^{(j)} - 1$
End (While)

Fig. 2. Check node profile optimization algorithm.

Note that the optimization can be solved by linear programming since the cost function and the constraints $(\mathcal{C}_1)$, $(\mathcal{C}_2)$, and $(\mathcal{C}_3)$ are linear in the parameters $\rho_{\mathbf{d}}^{(j)}$. The constraint $(\mathcal{C}_4)$ is the previous optimization constraint. Once we have optimized the check node profile, the code can be realized through the construction of a parity check matrix following the desired profile.

C_1	C_2	C_3
$\lambda_4^{(1)} = 0.00197$	$\lambda_3^{(2)} = 0.23982$	$\lambda_2^{(3)} = 0.93901$
$\lambda_{18}^{(1)} = 0.57263$	$\lambda_4^{(2)} = 0.76018$	$\lambda_3^{(3)} = 0.06099$
$\lambda_{19}^{(1)} = 0.21085$		
$\lambda_{30}^{(1)} = 0.21455$		

	C_1	C_2	C_3
$max\boldsymbol{\rho}_{\mathbf{d}}^{(2)} = 0.35$	$\rho_2^{(1)} = 0.08977$	$\rho_2^{(2)} = 0.35$	$\rho_1^{(3)} = 0.00024$
	$\rho_3^{(1)} = 0.19637$	$\rho_3^{(2)} = 0.35$	$\rho_2^{(3)} = 0.93877$
	$\rho_4^{(1)} = 0.34911$	$\rho_4^{(2)} = 0.30$	$\rho_3^{(3)} = 0.06099$
	$\rho_7^{(1)} = 0.36475$		
$max\boldsymbol{\rho}_{\mathbf{d}}^{(2)} = 0.55$	$\rho_3^{(1)} = 0.25248$	$\rho_3^{(2)} = 0.45$	$\rho_1^{(3)} = 0.00024$
	$\rho_4^{(1)} = 0.54860$	$\rho_4^{(2)} = 0.55$	$\rho_2^{(3)} = 0.93877$
	$\rho_7^{(1)} = 0.19892$		$\rho_3^{(3)} = 0.06099$

IV. SIMULATION RESULTS

In this section, simulation results for multi-edge type UEP-LDPC codes with optimized check node connection profile are presented. We designed UEP-LDPC codes of length $n = 4096$ with $m_e = 3$ protection classes, rate 1/2, and $d_{v_{max}} = 30$ following the algorithm of [3]. The proportions of the classes are chosen such that C_1 contains 20% of the information bits and C_2 contains 80%. The third protection class C_3 contains all parity bits. Therefore, we are mainly interested in the performances of classes C_1 and C_2. The optimized variable and check node degree distribution for the UEP-LDPC code are given by $\lambda(x) = 0.2130x + 0.0927x^2 + 0.2511x^3 + 0.2521x^{17} + 0.0965x^{18} + 0.0946x^{29}$ and $\rho(x) = x^8$, respectively.

In order to have a low-complexity systematic encoder, we construct parity check matrices in lower triangular form [12]. This approach also leads to a simplification in our optimization procedure, i.e., given that the parity bits are in the less protected class C_3 and that they should be organized in a lower triangular form, we start the optimization from the less protected information bits class C_2, since all the connections between the variable nodes of class C_3 and the check nodes are completely determined by the lower triangular form construction algorithm. Table I summarizes the classes' variable degree distributions $\lambda^{(j)}(x)$. We applied the optimization algorithm for different values of $max\boldsymbol{\rho}_{\mathbf{d}}^{(j)}$ to enable the observation of the varying UEP capabilities of the codes. The resulting distributions are summarized in Table II. All the simulations were done for a total of 50 decoding iterations and the constructed codes were all realized through a modification of progressive edge-growth (PEG) [13] algorithm done in order to ensure that the optimized check node degree is realized.

Figure 3 shows that the difference between the performances

of the protection classes is reduced as we increase the value of $max\boldsymbol{\rho_d^{(2)}}$. This is an expected effect, since the greater $max\boldsymbol{\rho_d^{(2)}}$, the greater is the amount of information that C_2 exchanges with C_1. Obviously, this is expected to enhance the performance of C_2 while lowering the one of C_1. As a benchmark, Fig. 3 also shows the performance of an LDPC code (referred to as non-UEP) with the same degree distributions of the UEP-LDPC ($\lambda(x) = 0.2130x + 0.0927x^2 + 0.2511x^3 + 0.2521x^{17} + 0.0965x^{18} + 0.0946x^{29}$ and $\rho(x) = x^8$) constructed without optimizing the connections between the classes.

Furthermore, Fig. 4 shows the BER as a function of the number of decoder iterations at $E_b/N_0 = 1.25$ dB for the UEP code optimized with $max\boldsymbol{\rho_d^{(2)}} = 0.35$ and the non-UEP code. Note that for a high number of iterations, the benchmark code has almost no UEP while the code with an optimized distribution has a constant UEP capability, something regarded as infeasible in [3] and not observed in [6] for codes realized by means of PEG.
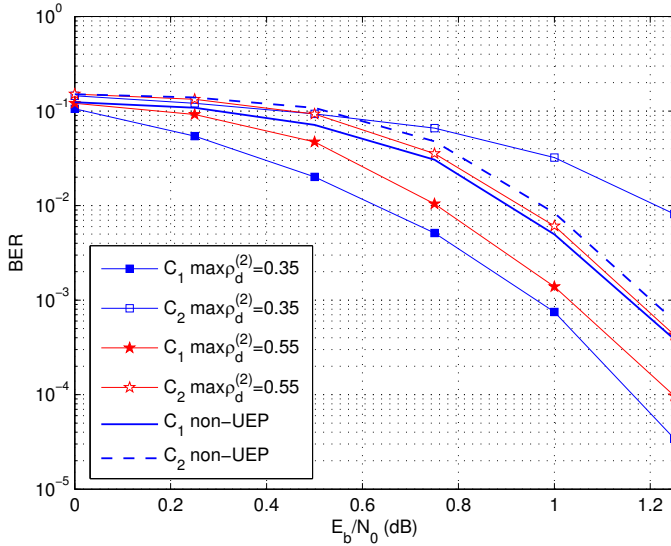

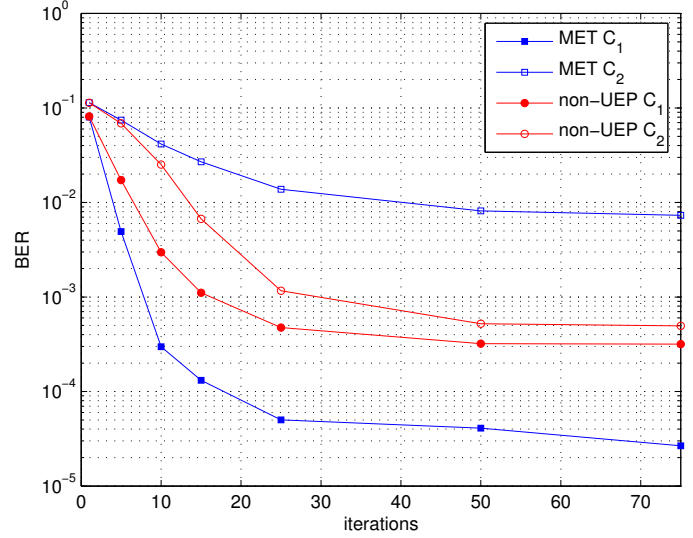Fig. 4. BER as a function of the number of decoder iterations for the multi-edge UEP LDPC code at E_b/N_0=1.25 dB.

Fig. 3. Classes bit error rate of the optimized multi-edge unequal error protecting LDPC codes for different values of $max\boldsymbol{\rho_d^{(2)}}$.

V. CONCLUDING REMARKS

In this paper, we introduced a multi-edge type analysis of unequal error protecting LDPC codes. By means of such an analysis, we derived an optimization algorithm that aims at optimizing the connection profile between the protection classes within a codeword. This optimization allowed us not only to control the differences in the performances of the protection classes by means of a single parameter, but also to prevent the UEP capability of an LDPC code to vanish after a moderate to large number of decoding iterations.

ACKNOWLEDGMENT

REFERENCES

[1] W. Henkel, K. Hassan, N. von Deetzen, S. Sandberg, L. Sassatelli, and D. Declercq, "UEP concepts in modulation and coding," *Hindawi, Advances in Multimedia*, Vol. 2010, Article ID 416797, 14 pages, 2010. doi:10.1155/2010/416797.

[2] H. Pishro-Nik, N. Rahnavard, and F. Fekri, "Non-uniform error correction unsing low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 51, no. 7, pp. 2702–2714, 2005.

[3] C. Poulliat, D. Declercq, and I. Fijalkow, "Enhancement of unequal error protection properties of LDPC codes," *EURASIP Journal on Wireless Communications and Networking*, vol. 2007, Article ID 92659, 9 pages, doi:10.115/2007/92659.

[4] T. Richardson, M. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[5] V. Kumar and O. Milenkovic, "On unequal error protection LDPC codes based on Plotkin-type constructions," *IEEE Transactions on Communications*, vol. 54, no. 6, pp. 994–1005, 2006.

[6] N. von Deetzen and S. Sandberg, "On the UEP capabilities of several LDPC construction algorithms," *IEEE Transactions on Communications*, vol. 58, no. 11, pp. 3041–3046, November 2010.

[7] T. Richardson and R. Urbanke, "Multi-Edge Type LDPC Codes," Tech. Rep., 2004, submitted to IEEE Transaction on Information Theory.

[8] ——, *Modern Coding Theory*. Cambridge University Press, 2008.

[9] S. Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 657–670, Feb. 2001.

[10] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Transactions on Communications*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.

[11] G. Liva and M. Chiani, "Protograph LDPC codes design based on EXIT analysis," in *Proceedings of the 50th Annual IEEE Global Telecommunications Conference (GLOBECOM 07)*, November 2007, p. 32503254.

[12] T. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 638–656, Feb. 2001.

[13] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 386–398, January 2005.