Analysis of Generalized LDPC Codes with Random Component Codes for the Binary Erasure Channel

Enrico PAOLINI†, Marc FOSSORIER‡ and Marco CHIANI†

† D.E.I.S., WiLAB
University of Bologna
via Venezia 52, 47023 Cesena, Italy
E-mail: {epaolini,mchiani}@deis.unibo.it

‡ Department of Electrical Engineering
University of Hawaii at Manoa
2540 Dole Street, Honolulu, HI 96822
E-mail: marc@spectra.eng.hawaii.edu

Abstract

In this paper, a method for the asymptotic analysis of generalized low-density parity-check (GLDPC) codes on the binary erasure channel (BEC) is proposed. The considered GLDPC codes have block linear codes as check nodes. Instead of considering specific check component codes, like Hamming or BCH codes, random codes are considered, and a technique is developed for obtaining the expected check EXIT function for the overall GLDPC code. Each check component code is supposed to belong to an expurgated ensemble. Some GLDPC thresholds obtained by this technique are compared with those of GLDPC codes, with the same distribution and component codes lengths, using specific codes. Results obtained by combining our analysis with differential evolution tool are also presented.

1. INTRODUCTION

EXIT charts [1] represent a powerful approach for the approximate evaluation of iterative decoders asymptotic threshold. The applicability of EXIT charts includes LDPC codes [2] and GLDPC codes [1, 3, 4, 5].

It has been shown in [1] that the EXIT function of a linear code without idle components can be expressed in terms of information functions [6] or split information functions, when both the communication and the extrinsic channels are BEC’s. This is the case of a check component code for a GLDPC code used to communicate on a BEC. The afore mentioned relationship between EXIT function and information functions is then very useful for the asymptotic analysis on the BEC of GLDPC codes built up with component codes whose information functions are known. However, for a wide range of linear codes including binary double error-correcting and more powerful BCH codes, the information functions are still unknown.

In this paper, we develop the following idea. Instead of considering specific check component codes, we consider random codes, and develop a method to exactly compute the expected EXIT function of each component code. We focus on GLDPC codes with repetition codes as variable nodes, random binary linear block codes as check codes, and random bipartite graph. Specifically, each component code is assumed to belong to an expurgated ensemble, and the expectation is on this expurgated ensemble. The expected EXIT function computation for the generic component code is based on the exact evaluation of its expected information functions. The expected EXIT functions for the component codes are then used to evaluate the expected check EXIT function for the overall GLDPC code.

2. BOUNDED DISTANCE EXIT FUNCTIONS AND EXPURGATED ENSEMBLE DEFINITION

The EXIT function of a \((n, k)\) linear code of length \(n\) and dimension \(k\) without idle components, assuming that no communication channel is present and that the extrinsic channel is a BEC with erasure probability \(p\), has been shown in [1, eq. 40] to be expressed by

\[
I_E(p) = 1 - \frac{1}{n} \sum_{g=1}^{n} (1 - p)^{n-g} p^{n-g} \cdot [g \bar{e}_g - (n - g + 1) \bar{e}_{g-1}].
\]

In (1), \(\bar{e}_g\) is the \(g\)-th un-normalized information function, defined as the summation of the dimensions of all the possible codes obtained considering just \(g\) positions in the code block of length \(n\). We consider in this work the following expression of \(I_E(\cdot)\), equivalent to (1):

\[
I_E(p) = 1 - \frac{1}{n} \sum_{t=0}^{n-1} (1 - p)^{n-t} p^t \cdot [(n-t) \bar{e}_{n-t} - (t+1) \bar{e}_{n-t-1}].
\]
From the definition of EXIT function, it is readily shown that \[ H(V_i | W[\tau] = W[i]) \] where \( H(\cdot) \) is the entropy function, \( V_i \) is the \( i \)-th encoded bit, \( W[\tau] \) is the codeword except bit \( i \), with bits erased independently with probability \( p \), and \( W[i] \) is a specific realization of \( W[\tau] \) with \( t \) erasures.

Note that, for a specific \((n, k)\) code with minimum distance \( d_{\text{min}} \), the summation in \( t \) in (2) can always start from \( t = d_{\text{min}} - 1 \). In fact, \( H(V_i | W[\tau] = W[i]) = 0 \) for each \( i \) if \( t < d_{\text{min}} - 1 \), i.e. decoding is always possible if \( t < d_{\text{min}} - 1 \). Decoding is sometimes possible if \( d_{\text{min}} - 1 \leq t < n - k \), and it is never possible for \( t \geq n - k \). However, even in the latter case, some bits may be correctable.

Relationships (1) and (2) assume MAP decoding. Consider now the following decoding strategy: “if the number of received erasures from the extrinsic channel is less than or equal to \( d \), execute MAP decoding, otherwise declare a decoding failure”. This is equivalent to assume is equal to 1 the entropy of each encoded bit \( V_i \) given any realization of \( W[\tau] \) with \( t \geq d \) erasures. Then, the EXIT function is given by:

\[
I_E(p) = 1 - \frac{1}{n} \sum_{t=0}^{d-1} (1 - p)^{n-t-1} p^t [(n - t) \bar{e}_{n-t} - (t + 1) \bar{e}_{n-t-1}] - \sum_{t=d}^{n-1} (1 - p)^{n-t-1} p^t \binom{n-1}{t}. \tag{3}
\]

For example, the thresholds evaluated in [8, Tab. 2] through a generalization of density evolution, can also be obtained through an EXIT charts approach exploiting (3) for the BCH codes, with \( d = d_{\text{min}} - 1 \).

Consider a GLDPC code with \( T \) different types of check codes, the \( i \)-th one with EXIT function \( I_E^{(i)}(p) \). Variable and check nodes are assumed randomly connected. Each \((n_i, k_i)\) check code is assumed random, from an expurgated ensemble \( C_i(n_i, k_i) \). We define the expurgated ensemble in order to satisfy the following constraint: the expectation of the overall GLDPC check EXIT function must tend to 0 as \( p \) tends to 1, and must tend to 1 as \( p \) tends to 0. This is necessary for a correct application of the EXIT chart analysis. Denoting by \( E[I_E(p)] \) the expected GLDPC check EXIT function and by \( \rho_i \), the fraction of edges towards check nodes of type \( i \), for random bipartite graph it is

\[
E[I_E(p)] = \frac{1}{n} \sum_{i=1}^{T} \rho_i I_E^{(i)}(p) = \sum_{i=1}^{T} \rho_i E_{C_i(n_i, k_i)}[I_E^{(i)}(p)]. \tag{4}
\]

Note that if the \( i \)-th check code type is a single parity check (SPC) code, no expectation is needed. In order to make the constraint satisfied for \( E[I_E(p)] \), it is sufficient that it holds for each \( I_E^{(i)}(p) \). Exploiting (2), we have \( \lim_{n \to \infty} I_E^{(i)}(p) = 1 - \bar{e}_{n_i}^{(i)} / n_i \) and \( \lim_{p \to 0} I_E^{(i)}(p) = 1 + \bar{e}_{n_i}^{(i)} / n_i - \bar{e}_{n_i}^{(i)} \). Hence, we need \( e_{n_i}^{(i)} = n_i \) and \( \bar{e}_{n_i}^{(i)} = n_i \bar{e}_{n_i}^{(i)} \) for each check code.

The first condition simply means that the component code should have no idle components, i.e. the generator matrix \( G^{(i)} \) should have no zero columns. The second condition is satisfied if the generator matrix is full rank (\( \text{rank}(G^{(i)}) = k_i \)), and removing any single column from \( G^{(i)} \) does not reduce the rank.

In summary, \( C_i(n_i, k_i) \) is defined as the ensemble of all the \((k_i \times n_i)\) binary matrices \( G^{(i)} \) without zero columns, such that \( \text{rank}(G^{(i)}) = k_i \), and such that the elimination of any single column does not reduce the rank. It is worth mentioning that this implies \( d_{\text{min}} \geq 2 \) for all the codes of the ensemble.

The computation of \( E[I_E(p)] \) can be performed according to (4). It follows from (3) that the problem can be completely solved by evaluating the expected values of the information functions for each component code \( i \), over \( C_i(n_i, k_i) \). A solution to this problem is proposed in next the section.

3. EXPECTED INFORMATION FUNCTIONS COMPUTATION

In this section we present an approach to compute the expected values of the information functions for any linear \((n, k)\) code, where the expectation is intended over the expurgated ensemble described in Section 2. The method is based on some recursive formulas that permit to compute the exact number of \((m \times n)\) binary matrices with specific properties. We always suppose \( m \leq n \). Furthermore, when discussing the initial conditions for the recursions, if not specified otherwise, we always implicitly assume is equal to zero both the number of binary matrices with \( m = 0 \) or \( n = 0 \) and the number of matrices without zero columns and rank \( 0 \).

Let \( S_g \) be a submatrix of \( G \) obtained selecting \( g \) columns. The expectation of \( \bar{e}_{g} \) can be developed as:

\[
E_{C_i(n_i, k_i)}[\bar{e}_{g}] = E_{C_i(n_i, k_i)}[\sum_{S_g} \text{rank}(S_g)]
= \sum_{S_g} E_{C_i(n_i, k_i)}[\text{rank}(S_g)]
= \binom{n}{g} E_{C_i(n_i, k_i)}[\text{rank}(S_g)], \tag{5}
\]

where the last equality is due to the fact that for the afore defined expurgated ensemble the expectation of
the rank when selecting $g$ columns is independent of the specific selected columns. In the following, we suppose that $S_g$ in (5) is the submatrix composed of the first $g$ columns of $G$. Without loss of generality, the expectation of rank($S_g$) in (5) can be further developed in the following way:

$$E_{c_{g \times k}}[\text{rank}(S_g)] = \sum_{u=1}^{\min(k,g)} u \Pr\{\text{rank}(S_g) = u\}$$

$$= \sum_{u=1}^{\min(k,g)} K(k,n,u,u) J(k,n,k).$$

(6)

In (6), $J(m,n,r)$ is the number of rank-$r$ $(m \times n)$ binary matrices without zero columns, and such that removing any column does not reduce the rank. Thus, $J(k,n,k)$ is the total number of generator matrices in the expurgated ensemble. The function $K(m,n,g,u,r)$ represents the number of rank-$r$ $(m \times n)$ binary matrices without zero columns, such that removing any column does not affect the rank, and such that the first $g$ columns have rank $u$. Hence, for any $1 \leq g \leq n$, we have

$$\sum_{u=1}^{\min(k,g)} K(m,n,g,u) J(m,n,r) = J(m,n,r).$$

In the following we develop recursive formulas for functions $J(\cdot)$ and $K(\cdot)$. Even if $J(\cdot)$ can be expressed in terms of $K(\cdot)$ according to previous relationship, an independent recursive formula for $J(\cdot)$ is presented.

3.1. Computation of $J(m,n,r)$

For any binary matrix, we define an independent column any column linearly independent of all the other columns. Removing an independent column from a rank-$r$ matrix leads to a new matrix with rank $r-1$. Let $F(m,n,r)$ denote the number of rank-$r$ $(m \times n)$ binary matrices without zero columns. $J(m,n,r)$ can be computed as the difference between $F(m,n,r)$ and the number of rank-$r$ binary $(m \times n)$ matrices without zero columns and with at least one independent column.

The function $F(\cdot)$ can be easily expressed in a recursive way: $F(m,n,r)$ is equal to the total number of rank-$r$ $(m \times n)$ binary matrices minus the number of rank-$r$ $(m \times n)$ binary matrices with at least one zero column. The total number of rank-$r$ $(m \times n)$ binary matrices is given by $\prod_{j=1}^{r-1} (2^m - 2^j) / (2^r - 2^j)$. Denoting by $z$ the number of zero columns, we obtain

$$F(m,n,r) = \prod_{j=0}^{r-1} \frac{(2^m - 2^j)(2^n - 2^j)}{(2^r - 2^j)}$$

$$- \sum_{z=0}^{n-r} \binom{n}{z} F(m,n-z,r),$$

(7)

with $F(m,n,1) = 2^m - 1$ as initial condition. In the following, we denote by $T(m,n)$ the total number of $(m \times n)$ binary matrices without zero columns, i.e. $T(m,n) = \sum_{r=1}^{n} F(m,n,r)$. Next, we express the number $J^{(j)}(m,n,r)$ of rank-$r$ $(m \times n)$ binary matrices without zero columns and with $j$ independent columns. First, note that there are $\binom{n}{j}$ possible positions for the $j$ independent columns, and that the number of possible $j$ independent columns is $\prod_{i=0}^{j-1} (2^m - 2^i)$. Hence we have

$$J^{(j)}(m,n,r) = \binom{n}{j} \left( \prod_{i=0}^{j-1} (2^m - 2^i) \right) D^{(j)}(m,n-j,r-j),$$

where $D^{(j)}(m,n-j,r-j)$ is the number of rank-$r-j$ $(m \times (n-j))$ binary matrices without zero columns, without independent columns, and whose columns belong to a subspace of dimension $r-j$. Since $D^{(j)}(m,n-j,r-j)$ is independent of the specific choice of the $j$ independent columns, we can reason on the specific matrix shown in Fig. 1. The rank of such a matrix is always equal to $j + \text{rank}(M_1)$. Thus $M_1$ must have rank $r-j$, and it must have no independent columns (since the total number of independent columns must be $j$). Moreover, since each of the last $n-j$ columns must be independent of each of the first $j$ columns, each column of $M_1$ must have at least one 1. Hence, the number of $M_1$ matrices is equal to $J(m-j,n-j,r-j)$. Since the last $n-j$ columns must have rank $r-j$, any row in $M_2$ must be a linear combination of rows in $M_1$. The total number of such combinations is $2^{(r-j)}$. Hence it results

$$D^{(j)}(m,n-j,r-j) = J(m-j,n-j,r-j) 2^{(r-j)},$$

and we finally obtain the recursion:

$$J(m,n,r) = F(m,n,r) - \sum_{j=1}^{r-1} \binom{n}{j} \left( \prod_{i=0}^{j-1} (2^m - 2^i) \right)$$

$$2^{(r-j)} J(m-j,n-j,r-j).$$

(8)

The initial condition is again $J(m,n,1) = 2^m - 1$. The summation over $j$ is up to $r-1$ since $J(m-r,n-r,0) = 0$. In the special case $m = n = r$ we have

$$J(m,m,m) = \prod_{i=1}^{m-1} (2^m - 2^i).$$

3.2. Computation of $K(m,n,g,u,r)$

In order to compute the number of rank-$r$ binary $(m \times n)$ matrices without zero columns, without independent columns, and such that the first $g$ columns have rank equal to $u$, we use a method analog to that one used for function $J(\cdot)$. Let $M(m,n,g,u,r)$ be the number of rank-$r$ binary $(m \times n)$ matrices without zero columns and such that the first $g$ columns have rank equal to $u$. Then $K(m,n,g,u,r)$ is equal to the difference between $M(m,n,g,u,r)$ and the number of such matrices with at least one independent column.

The function $M(m,n,g,u,r)$ can be expressed as $F(m,g,u)$ times the number of $(m \times (n-g))$ binary matrices without zero columns and such that the overall rank is $r$. Since this number is independent of
the specific choice of the first $g$ columns, we can reason on the specific matrix of Fig. 2. In order to have an overall rank $r$, we must have \( \text{rank}(M_1) = r - u \). Denoting by $z$ the number of zero columns in $M_1$, the number of $M_1$ matrices can be expressed as \( (n-g) F(m-u, n-g-z, r-u) \). Since no zero columns must be present in the matrix, the only constraint on $M_1$ is that at least one must be present in each column of $M_2$ corresponding to a zero column of $M_1$. Thus the number of $M_2$ matrices corresponding to a $M_1$ matrix with $z$ zero columns is $T(u, z) 2^{u(n-g-z)}$, where $T(\cdot)$ is defined in previous subsection. Letting $s = (n-g) - (r-u)$ we finally obtain

\[
M(m, n, g, u, r) = F(m, g, u) \sum_{z=0}^{s} \binom{n-g}{z} T(u, z) 2^{u(n-g-z)} \cdot 2^{u(n-g-z)} F(m-u, n-g-z, r-u). \tag{9}
\]

In (9), the constraint $T(u, 0) = 1$ must be imposed.

Next, consider the problem to evaluate the number $K^{(j)}(m, n, g, u, r)$ of rank-$r$ \((m \times n)\) binary matrices without zero columns, with the first $g$ columns of rank $u$, and with $j$ independent columns. Let the number of independent columns among the first $g$ columns be $l \leq \min\{u, j\}$. The number of independent columns among the last $n-g$ columns is $j-l$. The number of possible positions of the independent columns is $\binom{g}{l} \binom{n-g}{j-l}$, while the number of choices of the $j$ independent columns is $\prod_{i=0}^{j-1} (2^m - 2^i)$. We can again reason on a specific position and choice of the independent columns. The specific choice is depicted in Fig. 3, where matrices $A$ and $B$ are defined. We prove next that the number of $[M_1|M_2]$ possible matrices is $K(m-j, n-j, g-l, u-l, r-j)$ and, for each choice of $[M_1|M_2]$, the number of $B|M_1|M_2$ matrices is $2^{j(r-j)}$.

The rank of the overall matrix is equal to $j + \text{rank}(M_2)$. Consequently, $\text{rank}(M_2|M_3) = r - j$. Furthermore, $[M_1|M_2]$ must have no independent columns, and must have at least one for each column due to the linear independence between the $j$ independent columns and all the columns of $B$. Finally, $\text{rank}(M_1) = u - l$. This condition can be obtained in the following way. Since $\text{rank}(B) = r - j$, each row in $B|\{M_1|M_2\}$ must be a linear combination of rows in $M_1$ and $M_2$. This implies in particular that each row in $M_3$ must be a linear combination of rows in $M_1$, i.e. $\text{rank}(A) = \text{rank}(M_1)$. The rank of the first $g$ columns is equal to $l + \text{rank}(A)$. Since this rank must be equal to $u$, it follows $\text{rank}(A) = u - l$, i.e. $\text{rank}(M_1) = u - l$.

Since each row of $B|\{M_1|M_2\}$ is a linear combination of rows of $[M_1|M_2]$, and since $\text{rank}(M_1|M_2) = r - j$, then for each choice of $[M_1|M_2]$ there are $2^{j(r-j)}$ $B|\{M_1|M_2\}$ matrices allowed.

Hence, we finally obtain the following recursion:

\[
K(m, n, g, u, r) = M(m, n, g, u, r) - \sum_{j=1}^{r-1} \sum_{l=0}^{\min\{u, j\}} \binom{g}{l} \binom{n-g}{j-l} \prod_{i=0}^{j-1} (2^m - 2^i) 2^{j(r-j)} \cdot K(m-j, n-j, g-l, u-l, r-j), \tag{10}
\]

where $M(\cdot)$ is given in (9), and with initial conditions $K(m, n, u, u, u) = J(m, n, u)$, $K(m, n, 0, 0, r) = J(m, n, r)$, $K(m, n, g, 1, 1) = 2^{m-1}$, $K(m, g, g, m) = F(m, m, m)$. In summary, for some $k$ and $n$, $E_{c_{[K]}^{u,v}}[\tilde{e}_g]$ can be computed from (6), where $J(\cdot)$ is recursively given in (8), and $K(\cdot)$ is given in (10).

4. RESULTS

4.1. GLDPC Codes with Uniform Check Nodes

In [8] some thresholds on the BEC have been found through density evolution for GLDPC codes with degree-2 variable nodes and Reed-Solomon and BCH check codes, under bounded distance decoding.

Consider a GLDPC code belonging to this class, with \((31, 21)\) BCH codes as check nodes, and assume MAP decoding up to $d$ erasures for the BCH codes. The GLDPC code threshold can be evaluated with an EXIT chart approach based on (3), by numerically evaluating the \((31, 21)\) BCH code information functions. The EXIT functions for some values of $d$ are plotted in Fig. 4 (solid lines) as a function of the erasure probability $p$, while the GLDPC thresholds are given in Table 1.

Assume that the check nodes are random \((31, 21)\) block linear codes from the expurgated ensemble discussed in Section 2. The corresponding expected EXIT functions are plotted in Fig. 4 (dotted lines), and the GLDPC thresholds evaluated considering the expected check EXIT function are given in Table 1. This suggests a number of considerations. First, for all values of $d$ there is good match between the BCH EXIT function and the expected EXIT function. Second, when the maximum number of erasures is bounded, it can be convenient to use a component code with good minimum distance properties, like the BCH code. Third, our results clearly show that, if no bound is imposed on the maximum number of erasures ($d = 31$), codes exist within the expurgated ensemble that guarantee a better GLDPC threshold than the BCH code. For this specific example, the crossover point between the BCH code and the ensemble average is at $d = 12$.

We were able to generate a \((31, 21)\) linear code which guarantees a GLDPC threshold better than the ensemble average, under unconstrained MAP decoding. This code is characterized by $d_{\text{min}} = 2$, with a
Table 1: Thresholds of GLPDC codes with (31, 21) BCH check codes and thresholds evaluated with the expected (31, 21) EXIT function.

<table>
<thead>
<tr>
<th>d</th>
<th>BCH</th>
<th>expectation</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0.21915</td>
<td>0.21879</td>
</tr>
<tr>
<td>7</td>
<td>0.35596</td>
<td>0.35407</td>
</tr>
<tr>
<td>10</td>
<td>0.46256</td>
<td>0.45929</td>
</tr>
<tr>
<td>31</td>
<td>0.50187</td>
<td>0.51426</td>
</tr>
</tbody>
</table>

Table 2: Thresholds for capacity approaching LDPC and GLPDC codes with code rate 1/2.

<table>
<thead>
<tr>
<th>LDPC</th>
<th>0.49611</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLDPC, d_{min} = 2</td>
<td>0.49627</td>
</tr>
<tr>
<td>GLDPC, expectation</td>
<td>0.49639</td>
</tr>
<tr>
<td>GLDPC, d_{min} = 3</td>
<td>0.49648</td>
</tr>
<tr>
<td>GLDPC, BCH</td>
<td>0.49671</td>
</tr>
</tbody>
</table>

carefully chosen weight distribution for the low weight codewords. The GLDPC threshold corresponding to this code is 0.51920. We also generated a (31, 21) linear code characterized by d_{min} = 3, and we found a threshold equal to 0.51310 for the corresponding GLDPC code. This value is intermediate with respect to the thresholds corresponding to the BCH code and to the d_{min} = 2 code. A plot detail of the EXIT functions for these three codes and for the ensemble average is presented in Fig. 5 as a function of I_A = 1 - p. Denoting by T(C) the GLDPC code threshold corresponding to the choice of code C for the check nodes, we have T(C^{(31,21)}_{BCH}) < T(C^{(31,21)}_{d_{min}=2}) < T(E[C^{(31,21)}]) < T(C^{(31,21)}_{d_{min}=3}). This reveals that using somewhat weak codes as check component codes for GLDPC codes with uniform check structure can be more efficient than using more powerful codes, like the BCH codes. This conclusion is no longer valid when considering hybrid check structures, as explained in the next subsection.

4.2. GLDPC Codes with Hybrid Check Nodes

Consider at first the optimization problem on the BEC for an LDPC distribution with the following constraints: variable nodes degree from 2 to 30, check nodes degree from 3 to 14, code rate R = 1/2. We solved the problem using the differential evolution (DE) algorithm [7], for different starting distribution populations. The threshold of the best found distribution was 0.49611, quite close to the channel capacity 1 - R = 0.5.

Next, we solved the same optimization problem for a GLDPC code with a hybrid check nodes structure, composed of SPC codes and (31, 21) linear block codes. We solved again the optimization problem through DE algorithm, assuming the same degree constraints for the variable nodes and for the SPC codes, and again R = 1/2. More specifically, we separately solved the problem in the cases where the (31, 21) check nodes are represented by the BCH code, or by the d_{min} = i code (i = 2, 3 ) considered in previous subsection. We also solved the problem using the expected EXIT function for the (31, 21) expurgated ensemble.

In all cases, the edges for the optimized distribution are mostly connected to the SPC nodes with degree 9 and to the (31, 21) nodes. Moreover, the optimized distributions (both variable and check) are very similar in all cases. The fraction of edges connected to the (31, 21) codes ranges from about 5.73% for the d_{min} = 2 code to about 8.72% for the BCH code. The corresponding GLDPC thresholds are presented in Table 2. The threshold for the best GLDPC distribution is always higher than the LDPC code threshold.

This result reveals that it is possible to improve the threshold of an LDPC code by introducing check component codes different from the SPC codes, and modifying accordingly the edge distribution. The presented example is even more meaningful, since the threshold for the starting LDPC distribution is already close to capacity. Furthermore, when considering hybrid check nodes structures instead of uniform ones, using more powerful codes like the BCH codes leads to better thresholds. For the hybrid case it results T(C_{d_{min}=2}^{(31,21)}) < T(E[C^{(31,21)}]) < T(C_{d_{min}=3}^{(31,21)}) < T(C^{(31,21)}_{BCH}), which is the opposite of what found for a uniform check structure.

The reason is that the role of weak codes (necessary for obtaining good thresholds) is now played by the SPC codes. Finally, we observe that the developed technique for evaluating the expected EXIT function for linear block codes closely matches the results obtained for the BCH codes, in both the analysis and the generation of optimal GLDPC distributions. Hence, it can be confidently used for longer component codes for which the information functions remain unknown.

5. CONCLUSIONS

The asymptotic analysis on the BEC has been investigated, for GLDPC codes with random block linear component codes as check nodes. The method is based on the computation of the expected check EXIT function for the GLDPC code, which is accomplished by evaluating the expected information functions on an expurgated ensemble for each of the component codes. Our results indicate that the expected EXIT function can be confidently used instead of the exact EXIT func-
tion, for component codes for which the information functions are unknown, in both the analysis and the generation of optimized GLDPC distributions. They also indicate that hybrid GLDPC distributions can out-perform LDPC ones in terms of asymptotic threshold.

References


