A Deterministic Construction and Density Evolution Analysis for Generalized Product Codes

Christian Häger[†], Henry D. Pfister[‡], Alexandre Graell i Amat[†], Fredrik Brännström[†], and Erik Agrell[†]
[†]Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden
[‡]Department of Electrical and Computer Engineering, Duke University, Durham, North Carolina
{christian.haeger, alexandre.graell, fredrik.brannstrom, agrell}@chalmers.se, henry.pfister@duke.edu

***Abstract*—Generalized product codes (GPCs) are extensions of product codes (PCs) where code symbols are protected by two component codes but not necessarily arranged in a rectangular array. In this tutorial paper, we review a deterministic construction for GPCs that has been previously proposed by the authors together with an accompanying density evolution (DE) analysis. The DE analysis characterizes the asymptotic performance of the resulting GPCs under iterative bounded-distance decoding of the component codes over the binary erasure channel. As an application, we discuss the analysis and design of three different classes of GPCs: spatially-coupled PCs, symmetric GPCs, and GPCs based on component code mixtures.**

I. Introduction

Several authors have proposed modifications of the classical product code (PC) construction by Elias [1], typically by considering non-rectangular code arrays. These modifications can be regarded as generalized low-density parity-check (GLDPC) codes [2]. In particular, they are GLDPC codes where the underlying Tanner graph consists exclusively of degree-2 variable nodes (VNs) (i.e., each bit is protected by two component codes). We refer to such codes as generalized PCs (GPCs).

In practice, the component codes of a GPC are typically Bose–Chaudhuri–Hocquenghem or Reed–Solomon codes, which can be efficiently decoded via algebraic bounded-distance decoding (BDD). The overall GPC can then be suboptimally decoded using iterative hard-decision decoding, i.e., by iteratively performing BDD of all component codes. This makes GPCs particularly suited for high-speed applications due to their significantly reduced decoding complexity compared to "soft" message-passing decoding of low-density parity-check (LDPC) codes [3]. For example, GPCs have been investigated by many authors as practical solutions for high-speed fiber-optical communications [3]–[7].

A standard tool to analyze the performance of iteratively decoded codes is density evolution (DE) [8], [9], which is based on an ensemble argument. That is, rather than analyzing a particular code directly, one considers a set of codes defined via suitable randomized edge connections in the Tanner graph. While this approach can be applied to GPCs, many classes of GPCs have a very regular Tanner graph structure. Therefore, the performance of such codes is not necessarily well predicted by using an ensemble analysis. In general, it would be desirable to make precise statements about the performance of sequences of deterministic codes without resorting to an ensemble argument.

This work was partially funded by the Swedish Research Council under grant #2011-5961.

In this tutorial paper, we discuss some recent results about the performance of deterministically constructed GPCs over the binary erasure channel (BEC) presented in [10]–[12]. We start in Section II by reviewing the deterministic construction for GPCs proposed in [10]. The resulting GPCs are defined by Tanner graphs that consist of a fixed arrangement of (degree-2) VNs and constraint nodes (CNs). In Section III, it is shown that the asymptotic performance of these GPCs is rigorously characterized by a recursive DE equation. Finally, in Section IV, we present a high-level overview of different results presented in [10]–[12]. In particular, we discuss the analysis and design of spatially-coupled PCs, symmetric GPCs, and GPCs based on component code mixtures.

Notation. We use boldface to denote column vectors and matrices (e.g., $\boldsymbol{x}$ and $\boldsymbol{A}$). The symbols $\mathbf{0}_m$ and $\mathbf{1}_m$ denote the all-zero and all-one vectors of length m, respectively, where the subscript may be omitted. The tail-probability of a Poisson random variable is defined as $\Psi_{\geq t}(x) \triangleq 1 - \sum_{i=0}^{t-1} \frac{x^i}{i!} e^{-x}$. We use boldface to denote the element-wise application of a scalar-valued function to a vector. For example, if $\boldsymbol{x}$ is a vector, then $\boldsymbol{\Psi}_{\geq t}(\boldsymbol{x})$ applies the function to each element. For vectors $\boldsymbol{x} = (x_1, \dots, x_m)^\intercal$ and $\boldsymbol{y} = (y_1, \dots, y_m)^\intercal$, we use $\boldsymbol{x} \succeq \boldsymbol{y}$ if $x_i \geq y_i$ for all i. We also define $[m] \triangleq \{1, 2, \dots, m\}$. Lastly, the indicator function is denoted by $\mathbb{1}\{\cdot\}$.

II. A Deterministic Construction for Generalized Product Codes

A. Motivation

Recall that a PC is defined as the set of $n \times n$ arrays such that every row and every column is a codeword in some binary linear component code $\mathcal{B}$ of length n. The corresponding Tanner graph has a fixed deterministic structure that resembles a complete bipartite graph: There exists two types of CNs (n CNs corresponding to "row codes" and n CNs corresponding to "column codes") and each CN of one type is connected to all CNs of the other type through a VN. This gives rise to exactly n^2 VNs, where each VN corresponds to one element in the array. An illustration is shown for example in [2, Fig. 3].

Consider now the code arrays shown in Fig. 1. We will discuss these arrays (and the resulting GPCs) in more detail in the next section. For now, we note that one can almost apply

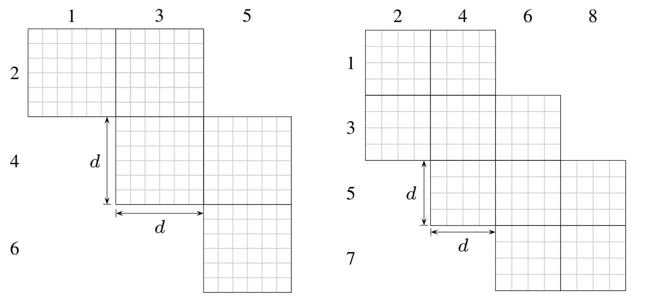


Fig. 1. Code arrays for $\mathcal{C}_{12}(\boldsymbol{\eta})$, where in (a) $\gamma = 1/2$ and in (b) $\gamma = 1/3$. Numbers indicate the position indices in the code construction.

the exact definition of a PC to these arrays. In particular, fill the array with bits such that every row and every column is a codeword in some component code. The underlying Tanner graph that results from this definition is again easily seen to be very structured. We essentially seek a general and flexible way to directly construct the Tanner graphs corresponding to the GPCs defined by these arrays.

B. Code Construction

We denote a GPC by $\mathcal{C}_n(\boldsymbol{\eta})$, where n is proportional to the number of CNs in the underlying Tanner graph and $\boldsymbol{\eta}$ is a binary, symmetric $L \times L$ matrix that defines the graph connectivity. Due to the natural representation of GPCs in terms of two-dimensional code arrays, one may alternatively think about $\boldsymbol{\eta}$ as specifying the array shape. We will see in the following that different choices for $\boldsymbol{\eta}$ recover well-known code classes.

Let $\gamma > 0$ be some fixed and arbitrary constant such that $d \triangleq \gamma n$ is an integer. To construct the Tanner graph that defines $\mathcal{C}_n(\boldsymbol{\eta})$, assume that there are L classes of CNs, here called "positions". Then, place d CNs at each position and connect each CN at position i to each CN at position j through a VN if and only if $\eta_{i,j} = 1$.

Example 1. A PC is obtained for $L = 2$ and $\boldsymbol{\eta} = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$. The two positions correspond to "row codes" and "column codes". If we choose $\gamma = 1$, then the code array is of size $n \times n$. △

Example 2. For $L \geq 2$, the matrix $\boldsymbol{\eta}$ describing a staircase code [3] has entries $\eta_{i,i+1} = \eta_{i+1,i} = 1$ for $i \in [L-1]$ and zeros elsewhere. For example, for $L = 6$, we have

$$\boldsymbol{\eta} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \tag{1}$$

The corresponding code array is exactly the one shown in Fig. 1(a), where $n = 12$ and $\gamma = 1/2$. △

Example 3. For even $L \geq 4$, the matrix $\boldsymbol{\eta}$ for a particular instance of a block-wise braided code [13] has entries $\eta_{i,i+1} = \eta_{i+1,i} = 1$ for $i \in [L-1]$, $\eta_{2i-1,2i+2} = \eta_{2i+2,2i-1} = 1$ for $i \in [L/2-1]$, and zeros elsewhere. For example, we have

$$\boldsymbol{\eta} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \tag{2}$$

for $L = 8$. The corresponding code array is the one shown in Fig. 1(b), where $n = 12$ and $\gamma = 1/3$. △

For a fixed n, the constant γ scales the number of CNs in the graph. This is inconsequential for the asymptotic analysis (where we assume that $n \to \infty$) and γ manifests itself in the DE equations merely as a scaling parameter. The scaling parameter γ in the previous two examples is chosen such that the component codes have length n in both cases, except at the array boundaries, see Fig. 1.

From the code construction, it follows that the total number of VNs (i.e., the length of the code $\mathcal{C}_n(\boldsymbol{\eta})$) is given by

$$m = \binom{d}{2} \sum_{i=1}^{L} \eta_{i,i} + d^2 \sum_{1 \leq i < j \leq L} \eta_{i,j}. \tag{3}$$

By construction, all of these VNs have degree two. Moreover, the total number of CNs is dL. In general, CNs at position i have degree $d \sum_{j \neq i} \eta_{i,j} + \eta_{i,i}(d-1)$, where the second term arises from the fact that we cannot connect a CN to itself if $\eta_{i,i} = 1$. The CN degree specifies the length of the component code associated with the CN. We assume in the following that each CN corresponds to a t-erasure correcting component code. This assumption is relaxed in Section IV-C.

III. Density Evolution Analysis

A. Iterative Decoding

Suppose that a codeword of $\mathcal{C}_n(\boldsymbol{\eta})$ is transmitted over the BEC with erasure probability p. The decoding is performed iteratively assuming ℓ iterations of BDD for the component codes associated with all CNs. This means that in each iteration, if the weight of an erasure pattern associated with a CN is less than or equal to t, the pattern is corrected. If the weight exceeds t, we say that the component code declares a decoding failure in that iteration.

The decoding can be represented by applying the following peeling procedure to the so-called residual graph [4], [14]. The residual graph is obtained by deleting known VNs and their adjacent edges. Furthermore, erased VNs are collapsed into edges between CNs. In each iteration, determine all vertices that have degree at most t and remove them, together with all adjacent edges. The decoding is successful if the resulting graph is empty after (at most) ℓ iterations.

B. Density Evolution

We wish to characterize the decoding performance in the limit as $n \to \infty$. The first important observation is that with the assumptions given so far (in particular the finite erasure-correcting capability of the component codes), this problem is ill-posed for a fixed erasure probability p. The reason is that for $n \to \infty$, with high probability there will be a large number of erasures associated with each component code. Even if we choose p very small, eventually, the number of erasures will exceed the (assumed) finite erasure-correcting capability of each component code. In other words, for any fixed p and $n \to \infty$, the decoding will fail with high probability.

In order to allow for a meaningful analysis, the natural choice is to let the erasure probability decay slowly as $p = c/n$ for some $c > 0$. Since now $p \to 0$ as $n \to \infty$, one may (falsely) conclude that the decoding will always be successful in the limit. As we will see, however, the answer depends crucially on the choice of c, which may thus be interpreted as the effective channel quality in this regime. Its operational meaning (assuming an appropriate choice for γ, see [10, Sec. VI-A]) is given in terms of the expected number of initial erasures per component code.

Now, assume that we compute

$$\boldsymbol{z}^{(\ell)} = \boldsymbol{\Psi}_{\geq t+1}(c\boldsymbol{B}\boldsymbol{x}^{(\ell-1)}), \text{with } \boldsymbol{x}^{(\ell)} = \boldsymbol{\Psi}_{\geq t}(c\boldsymbol{B}\boldsymbol{x}^{(\ell-1)}), \quad (4)$$

where $\boldsymbol{x}^{(0)} = \mathbf{1}_L$ and $\boldsymbol{B} \triangleq \gamma\boldsymbol{\eta}$. The main technical result is that the fraction of component codes that declare decoding failures in iteration ℓ converges almost surely to $\frac{1}{L}\sum_{i=1}^{L} z_i^{(\ell)}$ as $n \to \infty$. In other words, the code performance concentrates around a deterministic value computed by the recursion (4) for sufficiently large n. This result is analogous to the DE analysis for LDPC codes [9, Th. 2]. The proof exploits the above peeling representation of the decoding and is based on a convergence result for so-called inhomogeneous random graphs in [15], see [10] for details.

For notational convenience, we define $h(x) \triangleq \Psi_{\geq t}(cx)$, so that the recursion in (4) can be succinctly written as

$$\boldsymbol{x}^{(\ell)} = \boldsymbol{h}(\boldsymbol{B}\boldsymbol{x}^{(\ell-1)}). \quad (5)$$

Furthermore, the decoding threshold is defined in terms of the effective channel quality as

$$\bar{c} \triangleq \sup\{c \geq 0 \,|\, \boldsymbol{x}^{(\infty)} = \mathbf{0}_L\}. \quad (6)$$

Remark 1. For component codes with fixed erasure-correcting capabilities, one can show that the code rate of $\mathcal{C}_n(\boldsymbol{\eta})$ approaches 1 as $n \to \infty$. The studied setup is sometimes also referred to as the high-rate regime or high-rate scaling limit [16]. It turns out that the regime that can be analyzed is also the regime that is relevant in practice: It is at high rates where GPCs are competitive in terms of performance and complexity compared to other code families, e.g., LDPC codes [3]–[5].

IV. Applications

In this section, we discuss the analysis and design of three different classes of GPCs: spatially-coupled PCs, symmetric GPCs, and GPCs based on component code mixtures. This section is based on results presented in [10]–[12], [17].

A. Spatially-Coupled Product Codes

Of particular interest are cases where the matrix $\boldsymbol{\eta}$ has a band-diagonal "convolutional-like" structure. The associated GPC can then be classified as a spatially-coupled PC. For example, the GPCs discussed in Examples 2 and 3, i.e., staircase and braided codes, are particular instances of spatially-coupled PCs. The matrix $\boldsymbol{B}$ is referred to as an averaging matrix in this case. Spatially-coupled codes have attracted a lot of attention in the literature due to their outstanding performance under iterative decoding [18], [19].

Spatially-coupled PCs have been previously analyzed using ensemble-based methods in [6], [16]. In [12], we compare the obtained DE recursion in (5) for deterministic spatially-coupled PCs to the DE recursion for the spatially-coupled PC *ensemble* in [16]. Without going into the details, the ensemble performance is described by the recursion (see [16, eq. (9)] and [12, Sec. III])

$$\boldsymbol{x}^{(\ell)} = \boldsymbol{h}(\tilde{\boldsymbol{B}}\boldsymbol{x}^{(\ell-1)}), \quad (7)$$

where $\boldsymbol{x}^{(0)} = \mathbf{1}_L$, $\tilde{\boldsymbol{B}} \triangleq \boldsymbol{A}^\intercal \boldsymbol{A}$, and $\mathbf{A}$ is an $L - w + 1 \times L$ matrix with entries $A_{i,j} = w^{-1}\mathbb{1}\{1 \leq j - i + 1 \leq w\}$ for $i \in [L - w + 1]$ and $j \in [L]$. The parameter w is referred to as the coupling width. For example, for $L = 6$, the matrix $\tilde{\boldsymbol{B}}$ for $w = 2$ and $w = 3$ is given by

$$\frac{1}{4}\begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}, \quad \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 2 & 1 & 0 & 0 \\ 1 & 2 & 3 & 2 & 1 & 0 \\ 0 & 1 & 2 & 3 & 2 & 1 \\ 0 & 0 & 1 & 2 & 2 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}, \quad (8)$$

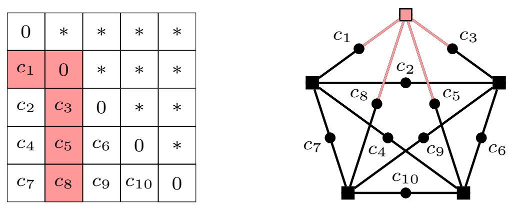
respectively. The ensemble DE recursion (7) has evidently the same form as (5). The difference lies in the averaging due to the matrix $\tilde{\boldsymbol{B}}$. This is illustrated in the following example.

Example 4. For the braided codes in Example 3, one can simplify (5) by exploiting the inherent symmetry in the code construction which implies $x_i^{(\ell)} = x_{i+1}^{(\ell)}$ for odd i and any ℓ. It is then sufficient to retain odd (or even) positions in (5). With this simplification, the effective averaging matrix is given by

$$\boldsymbol{B}' = \frac{1}{3}\begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad (9)$$

for $L = 12$. The matrix $\boldsymbol{B}'$ may be used to replace $\boldsymbol{B}$ in (5). Moreover, $\boldsymbol{B}'$ differs from both matrices $\tilde{\boldsymbol{B}}$ in (8). In general, the effective averaging matrices for the randomized and deterministic constructions are not the same. △

It is shown in [12] that there exists a different but related family of (deterministic) braided codes that has the same effective averaging matrix as the spatially-coupled PC ensemble,



(a) 5×5 array (b) Tanner graph

Fig. 2. Illustrations for an HPC with $n = 5$. In the array, "*" means "equal to the transposed element". The highlighted array elements illustrate one particular code constraint, which is also highlighted in the Tanner graph.

i.e., we have $\boldsymbol{B}' = \tilde{\boldsymbol{B}}$. This implies that the resulting DE recursions are identical and certain ensemble-properties proved in [16] (in particular lower bounds on the decoding threshold) also apply to certain deterministically constructed spatially-coupled PCs.

B. Symmetric Generalized Product Codes

All examples for $\mathcal{C}_n(\boldsymbol{\eta})$ discussed so far share the property that the corresponding matrix $\boldsymbol{\eta}$ does not contain any ones on the diagonal, i.e., $\eta_{i,i} = 0$ for all $i \in [L]$. In this section, we discuss the implications of choosing $\eta_{i,i} = 1$. In other words, we discuss the implications of connecting CNs to other CNs *at the same position* in the deterministic GPC construction.

The simplest case is obtained when there is only one position (i.e., $L = 1$) and we have $\boldsymbol{\eta} = 1$ with $\gamma = 1$. The resulting Tanner graph can be described as a "complete Tanner graph": There exist n CNs in total and each CNs is connected to all other CNs through a VN. All CNs have degree $n - 1$ and the total number of VNs, i.e., the length of the resulting code $\mathcal{C}_n(\boldsymbol{\eta})$, is given by $m = \binom{n}{2}$. Tanner already used such a construction as one of the first examples in [2, Fig. 6].

While the graph structure appears to be appealing due to its simplicity, it is not immediately clear if $\mathcal{C}_n(\boldsymbol{\eta})$ has a corresponding interpretation in terms of a code array. Such an interpretation was later provided by Justesen in [4, Sec. III-B]. In particular, assume that we start with a conventional (square) PC based on a component code with length n. Then, form a subcode of this PC by retaining only symmetric codeword arrays (i.e., arrays that are equal to their transpose) with a zero diagonal. After puncturing the diagonal and the upper (or lower) triangular part of the array, one obtains a code of length $m = \binom{n}{2}$. Justesen termed the resulting codes half-product codes (HPCs), emphasizing the fact that they have roughly half the length of the PCs from which they are derived.

Example 5. Figs. 2(a) and (b) show the code array and Tanner graph of an HPC for $n = 5$ and $m = 10$. The highlighted array elements show the code symbols participating in the second row constraint, which, due to the enforced symmetry, is also the second column constraint. Effectively, each component code acts on an L-shape in the array, i.e., both a partial row and column, which includes one diagonal element. The degree of each CN is $n-1 = 4$, due to the zeros on the diagonal. △

The definition of an HPC as a (punctured) symmetric subcode of a conventional PC extends without much difficulty to other GPCs. This leads to the class of symmetric GPCs which can be seen as a subclass of GPCs [17]. In general, symmetric GPCs use symmetry to reduce the block length of a GPC while employing the same component code [17].

Example 6. Consider again the code array in Fig. 1(b) corresponding to the braided code in Example 3. Similar to an HPC, we can form a half-braided code by enforcing the additional constraint that the array should be equal to its transpose and the array diagonal is zero (see [11, Fig. 1] for an illustration). After puncturing, we find that this GPC is defined by a matrix $\boldsymbol{\eta}$ where $\eta_{i,j} = 1$ if and only if $|i - j| < 3$. For example, if we start with a braided code where $L = 12$, then the matrix $\boldsymbol{\eta}$ for the corresponding half-braided code is given by $\boldsymbol{\eta} = 3\boldsymbol{B}'$, where $\boldsymbol{B}'$ is given in (9). △

An interesting question is how symmetric PCs perform when compared to their nonsymmetric counterparts. Partial answers to this question are given in [17] and [11]. For example, it is shown in [17] that, depending on the parameters, HPCs can have a larger normalized minimum distance than the PC from which they are derived. For the half-braided codes discussed in Example 5, a comparison with staircase codes and regular braided codes can be found in [11]. The comparison is based on the derived DE equations and supplemented with an error floor analysis. It is shown that half-braided codes can outperform both staircase codes and regular braided codes in the waterfall region, at a lower error floor and decoding delay. In general, symmetric PCs appear to be interesting candidates for further theoretical investigation and also implementation in practical communication systems.

C. Component Code Mixtures

In the construction of $\mathcal{C}_n(\boldsymbol{\eta})$, it is assumed that each CN corresponds to a t-erasure correcting component code. More generally, one may wish to assign different erasure-correcting capabilities to the component codes associated with the CNs. One example is given by a PC where the row and column codes can correct a different number of erasures. If the erasure-correcting capabilities also vary across the row (or column) codes, one obtains a so-called irregular PC [20], [21].

In order to formalize this concept in the context of the deterministic GPC construction, assume that $\boldsymbol{\tau} = (\tau_1, \dots, \tau_{t_{\max}})^\intercal$ is a probability vector (i.e., $\mathbf{1}^\intercal \boldsymbol{\tau} = 1$ and $\boldsymbol{\tau} \succeq 0$). We let τ_t be the fraction of CNs at each position that can correct t erasures, where $t_{\max}$ is the maximum erasure-correcting capability. We further define the average erasure-correcting capability as $\bar{t} \triangleq \sum_{t=1}^{t_{\max}} t\tau_t$. The assignment of the erasure-correcting capabilities to the component codes can be done in different ways. For example, we can do the assignment deterministically if $\tau_t d$ is an integer for all t, or independently at random according to the distribution $\boldsymbol{\tau}$. In both cases, $\boldsymbol{\tau}$ manifests itself in the DE equation (5) by changing the function h to $h(x) = \sum_{t=1}^{t_{\max}} \tau_t \Psi_{\geq t}(cx)$, see [10] for details.

The resulting GPCs now depend on $\boldsymbol{\tau}$ and this change is reflected in our notation by writing $\mathcal{C}_n(\boldsymbol{\eta}, \boldsymbol{\tau})$. For a fixed $\boldsymbol{\eta}$, we

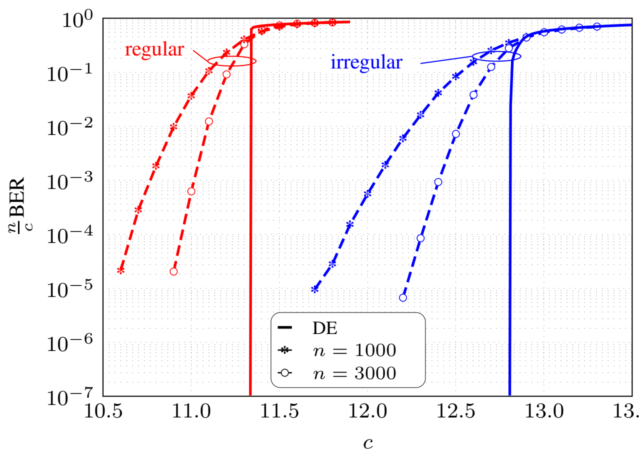


Fig. 3. Simulation results (dashed) for regular and optimized irregular HPCs for two values of n and $\ell = 100$. DE results (solid) are shown for $\ell = 100$.

are interested in finding "good" distributions $\boldsymbol{\tau}$, in the sense that they lead to large decoding thresholds for $\mathcal{C}_n(\boldsymbol{\eta}, \boldsymbol{\tau})$.

Example 7. For $L = 1$, $\boldsymbol{\eta} = 1$, and $\gamma = 1$, we refer to the resulting code $\mathcal{C}_n(\boldsymbol{\eta}, \boldsymbol{\tau})$ as an irregular HPC. This case is considered in detail in [10]. It is shown that the performance of HPCs can be improved by employing component codes with different strengths. Using an approach based on linear programming and fixing the average erasure-correcting capability to be $\bar{t} = 7$, we obtain the optimized distribution

$$\tau_4 = 0.495, \quad \tau_9 = 0.029, \quad \tau_{10} = 0.476. \tag{10}$$

The decoding threshold is given by $\bar{c} \approx 12.88$ compared to $\bar{c} \approx 11.34$ for a regular HPC with $\tau_7 = 1$. Fig. 3 shows simulation results for $n = 1000$ and $n = 3000$ together with the DE prediction, where we used $\ell = 100$. The performance gain predicted by DE is similar to what is achieved for finite lengths. Note that the figure shows a scaled bit error rate (BER) plotted against the effective channel quality c in order to better illustrate the convergence of the simulation results towards the asymptotic DE curve for increasing n, see [10, Sec. II-D] and [10, Sec. VII-E] for details. $\triangle$

Example 8. For spatially-coupled PCs, one may use the approach described in [19] to study iterative decoding thresholds. In particular, the decoding thresholds for the braided code family mentioned in the last paragraph of Section IV-A coincides with the so-called potential threshold defined in [19], provided that the coupling width is sufficiently large. This result is useful since it is typically easier to characterize the potential threshold (both numerically and theoretically) than the actual decoding threshold. Now, assume that we employ different component codes according to $\boldsymbol{\tau}$. In this case, the potential threshold depends on $\boldsymbol{\tau}$. In [12, Th. 2], it is proved that for a fixed $\bar{t} \in \{2, 3, \dots\}$, the potential threshold is maximized by a regular distribution where $\tau_{\bar{t}} = 1$. From this, we can conclude that employing component code mixtures for spatially-coupled PCs is not beneficial from an asymptotic point of view. $\triangle$

V. Conclusion

A deterministic construction of GPCs is reviewed along with a DE analysis for code sequences. As an application, these results are used to design and analyze spatially-coupled PCs, symmetric GPCs, and GPCs with component code mixtures.

References

[1] P. Elias, "Error-free coding," *IRE Trans. Inf. Theory*, vol. 4, no. 4, pp. 29–37, Apr. 1954.

[2] R. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. 27, no. 5, pp. 533–547, Sep. 1981.

[3] B. P. Smith, A. Farhood, A. Hunt, F. R. Kschischang, and J. Lodge, "Staircase codes: FEC for 100 Gb/s OTN," *J. Lightw. Technol.*, vol. 30, no. 1, pp. 110–117, Jan. 2012.

[4] J. Justesen, "Performance of product codes and related structures with iterated decoding," *IEEE Trans. Commun.*, vol. 59, no. 2, pp. 407–415, Feb. 2011.

[5] Y.-Y. Jian, H. D. Pfister, K. R. Narayanan, R. Rao, and R. Mazahreh, "Iterative hard-decision decoding of braided BCH codes for high-speed optical communication," in *Proc. IEEE Glob. Communication Conf. (GLOBECOM)*, Atlanta, GA, 2014.

[6] L. M. Zhang and F. R. Kschischang, "Staircase codes with 6% to 33% overhead," *J. Lightw. Technol.*, vol. 32, no. 10, pp. 1999–2002, May 2014.

[7] C. Häger, A. Graell i Amat, H. D. Pfister, A. Alvarado, F. Brännström, and E. Agrell, "On parameter optimization for staircase codes," in *Proc. Optical Fiber Communication Conf. (OFC)*, Los Angeles, CA, 2015.

[8] M. G. Luby, M. Mitzenmacher, and M. A. Shokrollahi, "Analysis of random processes via and-or tree evaluation," in *Proc. 9th Annual ACM-SIAM Symp. Discrete Algorithms*, San Franscisco, CA, 1998.

[9] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[10] C. Häger, H. D. Pfister, A. Graell i Amat, and F. Brännström, "Density evolution for deterministic generalized product codes on the binary erasure channel," *submitted to IEEE Trans. Inf. Theory*, 2015. [Online]. Available: http://arxiv.org/pdf/1512.00433.pdf

[11] ——, "Density evolution and error floor analysis of staircase and braided codes," in *Proc. Optical Fiber Communication Conf. (OFC)*, Anaheim, CA, 2016.

[12] ——, "Deterministic and ensemble-based spatially-coupled product codes," 2016. [Online]. Available: http://arxiv.org/pdf/1512.09180.pdf

[13] A. J. Feltström, D. Truhachev, M. Lentmaier, and K. S. Zigangirov, "Braided block codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 6, pp. 2640–2658, Jul. 2009.

[14] J. Justesen and T. Høholdt, "Analysis of iterated hard decision decoding of product codes with Reed-Solomon component codes," in *Proc. IEEE Information Theory Workshop (ITW)*, Tahoe City, CA, 2007.

[15] B. Bollobás, S. Janson, and O. Riordan, "The phase transition in inhomogeneous random graphs," *Random Structures and Algorithms*, vol. 31, no. 1, pp. 3–122, Aug. 2007.

[16] Y.-Y. Jian, H. D. Pfister, and K. R. Narayanan, "Approaching capacity at high rates with iterative hard-decision decoding," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Cambridge, MA, 2012.

[17] H. D. Pfister, S. K. Emmadi, and K. Narayanan, "Symmetric product codes," in *Proc. Information Theory and Applications Workshop (ITA)*, San Diego, CA, 2015.

[18] S. Kudekar, T. Richardson, and R. Urbanke, "Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 803–834, Feb. 2011.

[19] A. Yedla, Y.-Y. Jian, P. S. Nguyen, and H. D. Pfister, "A simple proof of Maxwell saturation for coupled scalar recursions," *IEEE Trans. Inf. Theory*, vol. 60, no. 11, pp. 6943–6965, Nov. 2014.

[20] S. Hirasawa, M. Kasahara, Y. Sugiyama, and T. Namekawa, "Modified product codes," *IEEE Trans. Inf. Theory*, vol. 30, no. 2, pp. 299–306, Mar. 1984.

[21] M. Alipour, O. Etesami, G. Maatouk, and A. Shokrollahi, "Irregular product codes," in *Proc. IEEE Information Theory Workshop (ITW)*, Lausanne, Switzerland, 2012.