Spatially Coupled Generalized LDPC Codes: Asymptotic Analysis and Finite Length Scaling

David G. M. Mitchell, *Senior Member, IEEE*, Pablo M. Olmos, Michael Lentmaier, *Senior Member, IEEE*, and Daniel J. Costello, Jr., *Life Fellow, IEEE*

***Abstract*— Generalized low-density parity-check (GLDPC) codes are a class of LDPC codes in which the standard single parity check (SPC) constraints are replaced by constraints defined by a linear block code. These stronger constraints typically result in improved error floor performance, due to better minimum distance and trapping set properties, at a cost of some increased decoding complexity. In this paper, we study spatially coupled generalized low-density parity-check (SC-GLDPC) codes and present a comprehensive analysis of these codes, including: (1) an iterative decoding threshold analysis of SC-GLDPC code ensembles demonstrating capacity approaching thresholds via the threshold saturation effect; (2) an asymptotic analysis of the minimum distance and free distance properties of SC-GLDPC code ensembles, demonstrating that the ensembles are asymptotically good; and (3) an analysis of the finite-length scaling behavior of both GLDPC block codes and SC-GLDPC codes based on a peeling decoder (PD) operating on a binary erasure channel (BEC). Results are compared to GLDPC block codes, and the advantages and disadvantages of SC-GLDPC codes are discussed.**

***Index Terms*— Generalized LDPC codes, spatially coupled codes, iterative decoding thresholds, minimum distance, finite length scaling.**

Manuscript received October 29, 2019; revised October 31, 2020; accepted April 2, 2021. Date of publication April 8, 2021; date of current version May 20, 2021. This work was supported in part by the National Science Foundation under Grant ECCS-1710920, Grant OIA-1757207, and Grant HRD-1914635; in part by the European Research Council (ERC) through the European Union's Horizon 2020 research and innovation program under Grant 714161; and in part by the Spanish Ministry of Science, Innovation and University under Grant TEC2016-78434-C3-3-R (AEI/FEDER, EU). This article was presented in part at the 2010 IEEE International Symposium on Information Theory, in part at the 2013 IEEE International Symposium on Information Theory, in part at the 2015 IEEE International Symposium on Information Theory, and in part at the 2018 IEEE International Symposium on Turbo Codes and Iterative Information Processing. *(Corresponding author: David G. M. Mitchell.)*

David G. M. Mitchell is with the Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM 88003 USA (e-mail: dgmm@nmsu.edu).

Pablo M. Olmos is with the Signal Theory and Communications Department, University of Carlos III in Madrid, 28911 Leganés, Spain (e-mail: olmos@tsc.uc3m.es).

Michael Lentmaier is with the Department of Electrical and Information Technology, Lund University, 22100 Lund, Sweden (e-mail: michael.lentmaier@eit.lth.se).

Daniel J. Costello, Jr. is with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556 USA (e-mail: costello.2@nd.edu).

Communicated by L. Dolecek, Associate Editor for Coding Techniques.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIT.2021.3071743.

Digital Object Identifier 10.1109/TIT.2021.3071743

I. Introduction

LOW-DENSITY parity-check (LDPC) block codes, with iterative message passing decoding, were introduced by Gallager in 1963 [1] as a class of codes whose decoder implementation complexity grows only linearly with block length, in contrast to maximum likelihood (ML) and maximum a posteriori (MAP) decoding methods whose complexity typically has exponential growth. As a result of the low-density constraint on the parity-check matrix $\mathbf{H}$, the minimum distance of LDPC block codes is sub-optimal. However, Gallager showed that regular constructions, where the variable and check node degrees of the Tanner graph representation of $\mathbf{H}$ are fixed, maintain linear minimum distance growth with block length, *i.e.*, they are *asymptotically good*, although their iterative decoding thresholds are bounded away from capacity. Irregular constructions, introduced by Luby et al. in 2001 [2], where the node degrees are not fixed and can be numerically optimized, have capacity-approaching thresholds, but typically involve a large fraction of low degree variable nodes that can preclude linear distance growth and result in problematic graphical objects causing failures in sub-optimal decoders. As a result, irregular codes perform best in the waterfall, or low signal-to-noise ratio (SNR), portion of the bit-error-rate (BER) performance curve, while regular codes perform better at high SNRs, *i.e.*, in the *error floor* region of the BER curve.

Generalized LDPC (GLDPC) block codes, first proposed by Tanner in 1981 [3], are constructed by replacing some/all of the single parity-check (SPC) constraint nodes in the Tanner graph of a conventional LDPC code by more powerful generalized constraint (GC) nodes corresponding to an (n, k) linear block code. The n variable nodes connected to a GC node in the Tanner graph of a GLDPC code are then considered as the code bits of the corresponding (n, k) code, and the sub-code associated with each GC node is referred to as a *constraint code*. In message passing decoding of GLDPC codes, the constraint codes are decoded using standard block code decoders which, in the case of simple constraint codes such as Hamming codes [4] or Hadamard codes [5], can be ML or MAP decoders. GLDPC codes have many potential advantages compared to conventional SPC/LDPC codes, such as large minimum distance [4], [6], low error floors [7], and fast iterative decoding convergence [8].

Spatially coupled LDPC (SC-LDPC) codes, also known as LDPC convolutional codes, were introduced by

Jimenez-Felstrom and Zigangirov in 1999 [9]. SC-LDPC codes can be viewed as a sequence of LDPC block codes whose graph representations are coupled together over time, resulting in a convolutional structure with block-to-block memory. A remarkable property of SC-LDPC codes, established numerically in [10] and analytically in [11], [12], is that asymptotically their iterative message passing decoding threshold is equal to the MAP decoding threshold of the underlying LDPC block code ensemble under certain conditions, a phenomenon known as *threshold saturation*. In other words, the (exponential complexity) MAP decoding performance of the underlying block code can be achieved by its coupled version with (linear complexity) message passing decoding.

Spatially coupled LDPC codes with generalized constraints, or *spatially coupled generalized LDPC (SC-GLDPC) codes*, and related constructions, including braided codes [13], [14], staircase codes [15], and product codes [16], have been investigated in the literature [17]–[24]. In particular, it has been shown that SC-GLDPC codes have good iterative decoding thresholds [17], [22], including excellent performance with hard decision iterative decoding [18], [21], [23], linear growth of minimum distance [19], and robust finite-length scaling performance [24]. We note that most of the existing work in this area can be considered as spatially coupled versions of product codes. Staircase codes, for example, can be seen as a variation of tightly braided block codes where the graphs are deterministic and less sparse. In particular, unlike the GLDPC code ensembles we consider in this paper, such codes get sparser only when the length of the component codes is increased.

Motivated to combine the threshold improvement of spatial coupling with the improved distance properties of generalized constraints, this paper investigates SC-GLDPC codes with linear block codes as generalized constraints and presents both asymptotic (threshold and distance) and finite-length analyses of SC-GLDPC code ensembles. A principle contribution of this paper is to extend the results of [17], [19], [20], [24] and present a unified treatment and analysis of SC-GLDPC codes. We first extend the threshold analysis of protograph-based SC-LDPC code ensembles in [10] to GLDPC code ensembles and use this to perform an iterative decoding threshold analysis of SC-GLDPC codes ensembles. This method is used to show numerically that threshold saturation is achieved for SC-GLDPC code ensembles, *i.e.*, their thresholds coincide with the maximum a-posteriori (MAP) decoding threshold of the underlying GLDPC block code ensemble.[1] This is followed by a minimum distance analysis of terminated and tail-biting SC-GLDPC code ensembles and a free distance analysis of unterminated ensembles, both of which demonstrate that the ensembles are asymptotically good and have large distance growth rates. In order to study the finite-length scaling properties of SC-GLDPC code ensembles, a method to analyze the finite-length scaling behavior of GLDPC block codes over the binary erasure channel (BEC) with peeling decoding (PD) is first introduced. We then extend this approach to study SC-GLDPC code ensembles and demonstrate robust finite-length scaling performance.

[1]In a recent paper [25], the authors found that, for a particular class of *doubly-generalized* LDPC codes introduced in [26], in which both variable and check nodes have generalized constraints, no threshold improvement from spatial coupling is observed. However, as we will demonstrate in this paper, threshold improvement is achieved with SC-GLDPC codes.

II. Protograph-Based SC-GLDPC Codes

A protograph [27] is a small bipartite graph that connects a set of n_v variable nodes $V = \{v_1, v_2, \ldots, v_{n_v}\}$ to a set of n_c constraint nodes $C = \{c_1, c_2, \ldots, c_{n_c}\}$ by a set of edges E. The edges connected to a variable node v_j of degree $\partial(v_j)$ or a constraint node c_i of degree $\partial(c_i)$ are labeled by $e^{\mathrm{v}}_{j,a}$ or $e^{\mathrm{c}}_{i,b}$, respectively, where $a \in \{1, \ldots, \partial(v_j)\}$ and $b \in \{1, \ldots, \partial(c_i)\}$. If the a^{th} edge associated with v_j is the b^{th} edge associated with c_i, then $e^{\mathrm{v}}_{j,a} = e^{\mathrm{c}}_{i,b}$.[2] In a protograph-based GLDPC code ensemble, each constraint node c_i can represent an arbitrary block constraint code $\mathcal{C}_i$ with parity-check matrix $\mathbf{H}_{c_i}$, length n^{c_i}, and m^{c_i} linearly independent parity-check equations where, throughout the manuscript, we use superscript labels on code parameters in order to distinguish between different constraint codes. The *design rate* of the GLDPC code ensemble is then given by

$$R = 1 - \frac{\sum_{i=1}^{n_c} m^{c_i}}{n_v}. \tag{1}$$

A protograph can be represented by means of an $n_c \times n_v$ bi-adjacency matrix $\mathbf{B}$, which is called the *base matrix* of the protograph. The nonnegative integer entry B_{ij} in row i and column j of $\mathbf{B}$ is equal to the number of edges that connect nodes c_i and v_j in the protograph. In order to construct ensembles of protograph-based GLDPC codes, a protograph can be interpreted as a template for the Tanner graph of a derived code, which is then obtained by a *copy-and-permute* or *graph lifting* operation [27]. In matrix form, the protograph is lifted by replacing each nonzero entry B_{ij} of $\mathbf{B}$ with a summation of B_{ij} non-overlapping permutation matrices of size $M \times M$, thereby creating an $Mn_c \times Mn_v$ constraint matrix $\mathbf{H}$ of a GLDPC code. Each row in the i^{th} set of M rows of $\mathbf{H}$ must satisfy the constraints $\mathbf{H}_{c_i}$ associated with constraint node c_i, where the length n^{c_i} of the i^{th} constraint code equals the sum of the entries in the i^{th} row of $\mathbf{B}$ and the constraint applies to the positions in a row of $\mathbf{H}$ with non-zero entries.[3] Allowing the permutations to vary over all $M!$ possible choices results in an ensemble of GLDPC block codes.

Example 1: Fig. 1 displays the protograph of an $(n_c, n_v) = (2, 7)$-regular GLDPC block code with base matrix

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{2}$$

If we suppose both the constraint nodes are $(7, 4)$ Hamming codes with parity-check matrix

$$\mathbf{H}_c = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}, \tag{3}$$

[2]This way of labeling will be useful for the density evolution equations described in Section III, since it takes into account the order of edges connected to a node and allows one to distinguish among multiple edges between a given pair of nodes.

[3]Strictly speaking, $\mathbf{H}$ is not a parity-check matrix since each row in the i^{th} set of M rows of $\mathbf{H}$ corresponds to m^{c_i} parity-checks. Consequently, we refer to $\mathbf{H}$ as a constraint matrix.

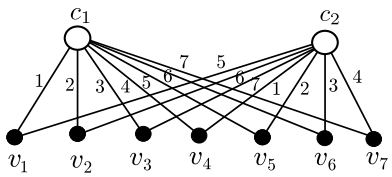

Fig. 1. Protograph of a $(2, 7)$-regular GLDPC block code. The white circles represent generalized constraint nodes and the black circles represent variable nodes. The labels on the edges indicate the corresponding columns of the parity-check matrix $\mathbf{H}_c$ of the generalized constraint code. In this case, both constraints are defined by the same $(7, 4)$ Hamming code, but with different orderings of columns.

where the constraint code length is $n^c = 7$ and the row rank of $\mathbf{H}_c$ is $m^c = 3$, then the resulting ensemble has design rate $R = 1/7$. Note that even though both constraints are defined by the same $(7, 4)$ Hamming code, a different ordering of columns can be used. In Fig. 1, the column of $\mathbf{H}_c$ that the variable node is connected to is shown on the edge. □

A. Unterminated SC-GLDPC Codes

An unterminated SC-GLDPC code can be described by a *convolutional protograph* [28] with base matrix

$$\mathbf{B}_{[0,\infty]} = \begin{bmatrix} \mathbf{B}_0 & & \\ \mathbf{B}_1 & \mathbf{B}_0 & \\ \vdots & \mathbf{B}_1 & \ddots \\ \mathbf{B}_w & \vdots & \ddots \\ & \mathbf{B}_w & \\ & & \ddots \end{bmatrix}, \tag{4}$$

where w denotes the *syndrome former memory* or *coupling width* of the code and the $b_c \times b_v$ *component base matrices* $\mathbf{B}_i$, $i \in \{0, 1, \ldots, w\}$, represent the edge connections from the b_v variable nodes at time t to the b_c (generalized) constraint nodes at time $t+i$.[4] An ensemble of (in general) time-varying SC-GLDPC codes can then be formed from $\mathbf{B}_{[0,\infty]}$ using the protograph construction method described above with lifting factor M.[5] The *decoding constraint length* of the resulting ensemble is given by $\nu_s = (w + 1)Mb_v$, the *design rate* is given by

$$R = 1 - \frac{\sum_{i=1}^{b_c} m^{c_i}}{b_v}, \tag{5}$$

and at each time instant t the encoder creates a block $\mathbf{v}_t$ of Mb_v symbols resulting in the unterminated code sequence $\mathbf{v} = [\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_t, \ldots]$.

Starting from a $b_c \times b_v$ base matrix $\mathbf{B}$ of a block code ensemble, one can construct SC-GLDPC code ensembles with the same variable and check node degrees as $\mathbf{B}$. This is achieved by an *edge spreading* procedure [28] that divides the edges connected to each variable node in the base matrix $\mathbf{B}$ among $w+1$ component base matrices $\mathbf{B}_i$, $i \in \{0, 1, \ldots, w\}$, such that the condition $\mathbf{B}_0 + \mathbf{B}_1 + \cdots + \mathbf{B}_w = \mathbf{B}$ is satisfied. We now give some examples of constructing SC-GLDPC code ensembles.

Example 2: For $w = 1$, we can apply the edge spreading technique to the $(b_c, b_v) = (2, 7)$-regular block code base matrix in (2) to obtain the following component base matrices

$$\mathbf{B}_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{6}$$

$$\mathbf{B}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{7}$$

The convolutional protograph associated with the resulting base matrix $\mathbf{B}_{[0,\infty]}$ defined in (4) is shown in Fig. 2, where time indices t are shown above the corresponding set (block) of variable nodes. We choose the upper and lower constraint nodes at each time instant to correspond to the $(7, 4)$ Hamming code with $n^c = 7$, $m^c = 3$, and parity-check matrix $\mathbf{H}_c$ from (3). (Note that the labels indicated on the edges correspond to columns of the component parity-check matrix $\mathbf{H}_c$ in (3) and that the constraint nodes c_1 and c_2 represent shortened codes.) After lifting, the constraint length of the resulting SC-GLDPC code ensemble is $\nu_s = 14M$ and the design rate is $R = 1/7$.

We will refer to this SC-GLDPC code ensemble as Ensemble A_7. An extension to the Ensemble A_{15} representing design rate $R = 7/15$ SC-GLDPC codes corresponding to the $(b_c, b_v) = (2, 15)$-regular all-ones matrix $\mathbf{B}$, $(15, 11)$ Hamming constraint codes with $n^c = 15$ and $m^c = 4$, and $w = 1$ edge-spreading based on (6) and (7), and to other values of $b_v = n^c$, m^c, and R, is straightforward. □

To illustrate the flexibility of SC-GLDPC code designs, multiple edges can also be introduced in the block protograph.

Example 3: Considering shortened $(14, 10)$ Hamming constraint codes with $n^c = 14$ and $m^c = 4$ as an example, where each variable node in a $b_c \times b_v = 1 \times 7$ protograph is connected with a double edge to a single check node. We split the corresponding multi-edge base matrix $\mathbf{B} = [2 \;\; 2 \;\; 2 \;\; 2 \;\; 2 \;\; 2 \;\; 2]$ into

$$\mathbf{B}_0 = \mathbf{B}_1 = [1 \;\; 1 \;\; 1 \;\; 1 \;\; 1 \;\; 1 \;\; 1] \tag{8}$$

and obtain the protograph of Ensemble B_{14} with $w = 1$, $b_c = 1$ check node, and $b_v = 7$ variable nodes at each time instant, a segment of which is illustrated in Fig. 3. From (5) we see that the design rate of Ensemble B_{14} is $R = 1 - 4/7 = 3/7$. Puncturing the first protograph variable node at each time instant t results in Ensemble $B_{14,P}$ with design rate $R = 0.5$. □

The protograph-based *braided block code* (BBC) ensembles considered in [29] are another example of SC-GLDPC code ensembles. These can be derived by using the Tanner graph of a *tightly* BBC [13] as a protograph. The component base matrices of such an SC-GLDPC code can be identified as

$$\mathbf{B}_0 = \begin{bmatrix} 1 & \mathbf{i} & \mathbf{0} \\ 1 & \mathbf{0} & \mathbf{i} \end{bmatrix}, \; \mathbf{B}_i = \begin{bmatrix} 0 & \mathbf{0} & \mathbf{e}_i \\ 0 & \mathbf{e}_i & \mathbf{0} \end{bmatrix}, \tag{9}$$

where $i = 1, \ldots, w$, $\mathbf{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ is the length w vector with a one at the i^{th} position and zeros

[4] We note that in a series of papers by Kudekar et al. [11], [12], the authors refer to a "smoothing parameter" w which is equal to the syndrome former memory plus one.

[5] In this paper, we restrict our attention to time-invariant protographs. Consequently, the choice of permutations in the graph lifting stage will determine if the lifted graph is time-invariant or time-varying. A random lifting will typically result in a time-varying graph.

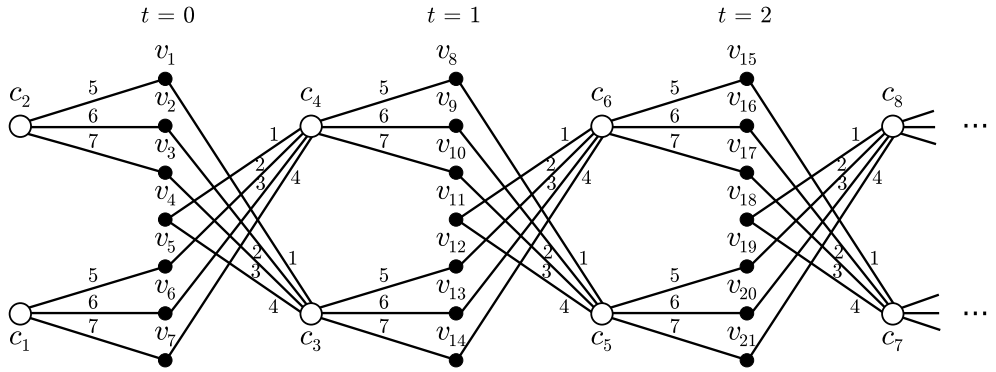

Fig. 2. Convolutional protograph of the $(2, 7)$-regular SC-GLDPC code ensemble A_7. The white circles represent generalized constraint nodes and the black circles represent variable nodes.

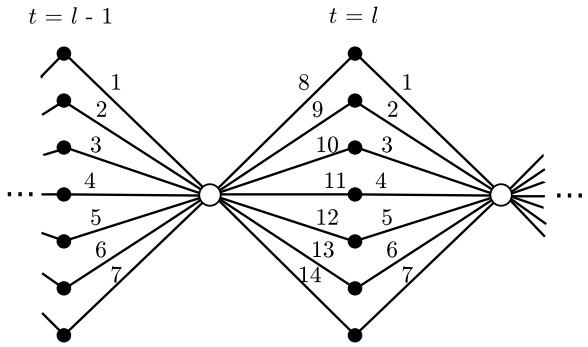

Fig. 3. Segment of the $w = 1$ convolutional protograph defining Ensemble B_{14}.

elsewhere, $\mathbf{0}$ is the all-zero vector, and $\mathbf{i}$ the all-one vector, of length w. We will use the term Ensemble C_{n^c} when referring to such SC-GLDPC code ensembles based on constraint codes of length $n^c = 2w + 1$.

Example 4: For the $(b_c, b_v) = (2, 7)$-regular GLDPC base matrix with $(7, 4)$ Hamming constraint codes and design rate $R = 1/7$ from Example 1, the convolutional protograph resulting from the tightly BBC construction of (9) with $w = 3$, $n^c = 7$, and $m^c = 3$, corresponding to Ensemble C_7, is illustrated in Fig. 4, where the upper constraint nodes correspond to the "horizontal constraints" and the lower constraint nodes correspond to the "vertical constraints" of the braided construction. Its girth is equal to eight, which follows from the structure of the array and is true for any SC-GLDPC code resulting from a tightly BBC protograph.

Observe that the sum of the component base matrices in (9) is equal to the base matrix $\mathbf{B}$ in (2) of the corresponding GLDPC code, *i.e.*, the all-one matrix of dimension $b_c \times b_v = 2 \times 7$. This reflects the fact that the graph in Fig. 4 can be obtained by repeating the GLDPC graph in Fig. 1 and spreading the edges over $w = 3$ adjacent time instants. An extension to the Ensemble C_{15}, representing SC-GLDPC codes corresponding to the $(b_c, b_v) = (2, 15)$-regular all-one base matrix $\mathbf{B}$, $(15, 11)$ Hamming constraint codes, and $w = 7$ edge-spreading based on (9) with $n^c = 15$, $m^c = 4$, and design rate $R = 7/15$, as well as to other values of $b_v = n^c$, m^c, and R, is straightforward. □

B. Terminated and Tail-Biting SC-GLDPC Codes

Suppose that we start the SC-GLDPC code with convolutional base matrix defined in (4) at time $t = 0$ and terminate it after L time instants. The resulting finite-length base matrix is then given by

$$\mathbf{B}_{[0,L-1]} = \begin{bmatrix} \mathbf{B}_0 & & & \\ \mathbf{B}_1 & \mathbf{B}_0 & & \\ \vdots & \mathbf{B}_1 & \ddots & \\ \mathbf{B}_w & \vdots & \ddots & \mathbf{B}_0 \\ & \mathbf{B}_w & & \mathbf{B}_1 \\ & & \ddots & \vdots \\ & & & \mathbf{B}_w \end{bmatrix}_{(L+w)b_c \times Lb_v}, \quad (10)$$

where L is called the *coupling length*. The matrix $\mathbf{B}_{[0,L-1]}$ is then the base matrix of a *terminated SC-GLDPC code*. The corresponding terminated convolutional protograph is slightly irregular, with lower constraint node degrees at both ends. This is illustrated for the A_7 ensemble of Example 2 in Fig. 5. The reduced degree constraint nodes at each end of the convolutional protograph are associated with shortened constraint codes, in which the symbols corresponding to the missing edges are removed. For decoding purposes, such a code shortening is equivalent to fixing these removed symbols and assigning an infinite reliability to them. Note that the variable node degrees are not affected by termination.

The constraint matrix $\mathbf{H}_{[0,L-1]}$ of the terminated SC-GLDPC code derived from $\mathbf{B}_{[0,L-1]}$ by lifting with some factor M has Mb_vL columns and $(L + w)Mb_c$ rows. It follows that the rate of the terminated SC-GLDPC code is equal to

$$R_L = 1 - \frac{(L + w)b_c m^c - \Delta}{Lb_v}, \quad (11)$$

where m^c denotes the (constant) number of independent parity checks associated with each constraint code and $\Delta \geq 0$ accounts for a possible rate increase due to the shortened

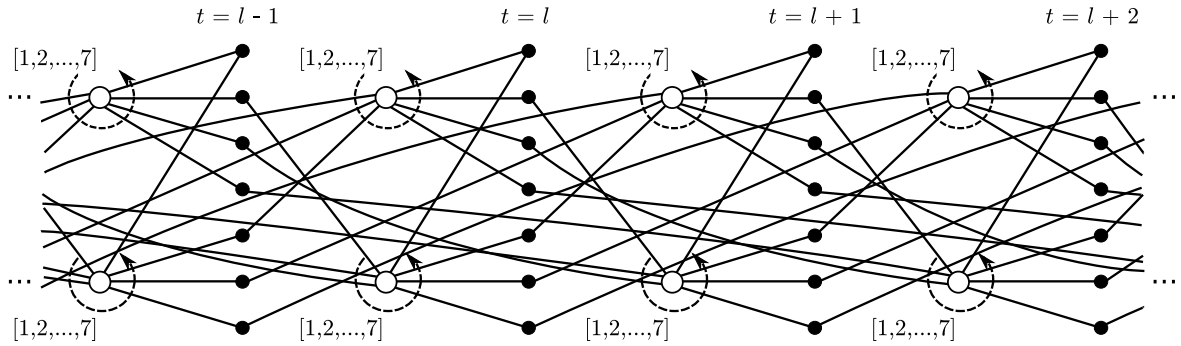

Fig. 4. Convolutional protograph of a SC-GLDPC code with $(7, 4)$ Hamming constraint codes, defining Ensemble C_7. The nodes are grouped according to the time instant t at which the code symbols, designated by the filled circles, are generated. Edge labels $1, 2, \ldots, 7$, corresponding to the columns of the component matrix $\mathbf{H}_c$, are ordered anti-clockwise from the upper left of each constraint node.

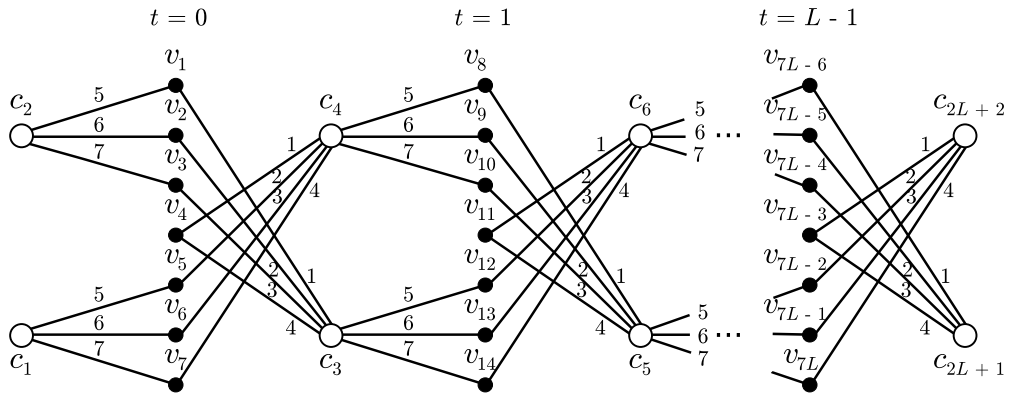

Fig. 5. Protograph of the $(2, 7)$-regular terminated SC-GLDPC code ensemble A_7 with coupling length L.

constraint codes.[6] If $\mathbf{H}_{[0,L-1]}$ has full rank, the rate increase parameter is $\Delta = 0$. However, the shortened constraint codes at the ends of the graph can cause a reduced rank for $\mathbf{H}_{[0,L-1]}$, which slightly increases R_L. In this case, $\Delta > 0$ and depends on both the particular constraint code chosen and the assignment of edges to the columns of its parity-check matrix $\mathbf{H}_c$. As $L \to \infty$, the rate R_L of the terminated SC-GLDPC code converges to the design rate $R = 1 - b_c m^c / b_v$ of the underlying GLDPC block code with base matrix $\mathbf{B}$.[7]

The generalized convolutional base matrix $\mathbf{B}_{[0,\infty]}$ can also be terminated using *tail-biting* [30], [31], resulting in the base matrix of a tail-biting generalized LDPC (TB-GLDPC) code ensemble. Here, for any $\lambda \geq w$, the last $b_c w$ rows of the terminated matrix $\mathbf{B}_{[0,\lambda-1]}$ are removed and added to the first $b_c w$ rows to form the $\lambda b_c \times \lambda b_v$ tail-biting matrix $\mathbf{B}_{tb}^{(\lambda)}$ with tail-biting coupling length λ

$$\mathbf{B}_{tb}^{(\lambda)} = \begin{bmatrix} \mathbf{B}_0 & & & & & \mathbf{B}_w & \cdots & \mathbf{B}_1 \\ \mathbf{B}_1 & \mathbf{B}_0 & & & & & \ddots & \vdots \\ \vdots & \vdots & & & & & & \mathbf{B}_w \\ \mathbf{B}_w & \mathbf{B}_{w-1} & & & & & & \\ & \mathbf{B}_w & \ddots & & & & & \\ & & & \mathbf{B}_0 & & & & \\ & & \ddots & \vdots & \mathbf{B}_0 & & & \\ & & \ddots & \mathbf{B}_{w-1} & \vdots & \ddots & & \\ & & & \mathbf{B}_w & \mathbf{B}_{w-1} & \cdots & \mathbf{B}_0 \end{bmatrix}. \tag{12}$$

Note that, if $w = 1$ and $\lambda = 1$, the tail-biting base matrix is simply the original block code base matrix, *i.e.*, $\mathbf{B}_{tb}^{(1)} = \mathbf{B}$. Terminating $\mathbf{B}_{[0,\infty]}$ in such a way preserves the design rate of the ensemble, *i.e.*, $R_\lambda = 1 - \lambda b_c m^c / \lambda b_v = 1 - b_c m^c / b_v = R$, and we see that $\mathbf{B}_{tb}^{(\lambda)}$ has exactly the same degree distribution as the original block code base matrix $\mathbf{B}$.

[6] We assume here the simplified case where each generalized constraint code is described by a parity-check matrix $\mathbf{H}_c$ with m^c independent parity checks. Under this assumption, the rate formula in (1) for GLDPC block codes becomes $R = 1 - b_c m^c / b_v$.

[7] We note here that the $(L + w)Mb_c$ rows of $\mathbf{H}_{[0,L-1]}$ should be viewed as $(L+w)b_c$ groups of rows, with M entries in each group, that are decoded according to the same constraint code with m^c rows.

III. Threshold Analysis of SC-GLDPC Codes

We assume belief propagation (BP) decoding, with log-likelihood ratios (LLRs) acting as messages. At every iteration ℓ, first all constraint nodes and then all variable nodes are updated. The outgoing messages computed at a constraint node c_i toward its b^{th} neighboring variable node at iteration ℓ are then equal to

$$L_{\text{c}}^{(\ell)}(e_{i,b}^{\text{c}}) = \log \sum_{\mathbf{x}\in\mathcal{C}_i^{b,0}} \prod_{b'\neq b} \exp\left(L_{\text{v}}^{(\ell-1)}(e_{i,b'}^{\text{c}})(1/2 - x_{b'})\right) - \log \sum_{\mathbf{x}\in\mathcal{C}_i^{b,1}} \prod_{b'\neq b} \exp\left(L_{\text{v}}^{(\ell-1)}(e_{i,b'}^{\text{c}})(1/2 - x_{b'})\right), \quad (13)$$

where $b, b' \in \{1, \ldots, \partial(c_i)\}$, $L_{\text{v}}^{(\ell-1)}(e_{i,b'}^{\text{c}})$ is the LLR received at constraint node c_i from the variable node connected to $e_{i,b'}^{\text{c}}$ at iteration ℓ, and we have partitioned $\mathcal{C}_i$ into the sets $\mathcal{C}_i^{b,0}$ and $\mathcal{C}_i^{b,1}$, corresponding to codewords $\mathbf{x} = [\, x_1 \;\; x_2 \;\; \cdots \;\; x_{\partial(c_i)} \,] \in \mathcal{C}_i$ for which $x_b = 0$ and $x_b = 1$, respectively. The message $L_{\text{c}}^{(\ell)}(e_{i,b}^{\text{c}})$ corresponds to the b^{th} extrinsic output generated by an optimal *a posteriori probability* (APP) decoder for component code $\mathcal{C}_i$, which is computed from the incoming messages $L_{\text{v}}^{(\ell-1)}(e_{i,b'}^{\text{c}})$, $b' \neq b$, to constraint node c_i. The incoming messages of the first iteration are initialized by the channel LLRs $L_{\text{ch}}(v_j)$ of the neighboring variable nodes, *i.e.*, $L_{\text{v}}^{(0)}(e_{i,b'}^{\text{c}}) = L_{\text{ch}}(v_j)$, where v_j is the variable node connected to $e_{i,b'}^{\text{c}}$. The outgoing messages computed at a variable node v_j at iteration ℓ are equal to

$$L_{\text{v}}^{(\ell)}(e_{j,a}^{\text{v}}) = L_{\text{ch}}(v_j) + \sum_{a'\neq a} L_{\text{c}}^{(\ell)}(e_{j,a'}^{\text{v}}), \quad (14)$$

where $a, a' \in \{1, \ldots, \partial(v_j)\}$.

A. Density Evolution for GLDPC Code Ensembles

For transmission over a binary erasure channel (BEC), the messages that are passed between the nodes represent either an erasure or the correct symbol values 0 or 1. In this case, the BP decoder is particularly simple and exact density evolution can be described explicitly. Let $q^{(\ell)}(e_{i,b}^{\text{c}})$ denote the probability that the check to variable node message sent along edge $e_{i,b}^{\text{c}}$ in decoding iteration ℓ is an erasure. Assuming a conventional LDPC code ensemble, where c_i corresponds to an SPC code, this is the case if at least one of the incoming messages from the other neighboring variable nodes is erased, *i.e.*,

$$q^{(\ell)}(e_{i,b}^{\text{c}}) = 1 - \prod_{b'\neq b} \left(1 - p^{(\ell-1)}(e_{i,b'}^{\text{c}})\right), \quad (15)$$

where the $p^{(\ell-1)}(e_{i,b'}^{\text{c}})$, $b, b' \in \{1, \ldots, \partial(c_i)\}$, denote the probabilities that the incoming messages to c_i computed in the previous iteration are erasures. For a GLDPC code ensemble, where c_i corresponds to an arbitrary block code, (15) can be replaced by the general expression

$$q^{(\ell)}(e_{i,b}^{\text{c}}) = f_b^{\mathcal{C}_i}\left(p^{(\ell-1)}(e_{i,b'}^{\text{c}}), b' \neq b\right), \quad (16)$$

where $f_b^{\mathcal{C}_i}$ is a multi-dimensional input/output transfer function that characterizes the APP decoder that computes the messages $L_{\text{c}}^{(\ell)}(e_{i,b}^{\text{c}})$ corresponding to (13). Note that, for generalized codes, $f_b^{\mathcal{C}_i}$ can be different for each $b \in \{1, \ldots, \partial(c_i)\}$, which implies that the ordering of edges can affect the performance of the ensemble. A method for computing explicit expressions for the APP decoder output distributions that can be used in (16) was presented in [32]. This method is based on a Markov chain analysis of the decoder metrics using a trellis representation of the block code $\mathcal{C}_i$.

The variable to check node message sent along edge $e_{j,a}^{\text{v}}$ is an erasure if all incoming messages from the channel and from the other neighboring check nodes are erasures. Thus we have

$$p^{(\ell)}(e_{j,a}^{\text{v}}) = \epsilon \prod_{a'\neq a} q^{(\ell)}(e_{j,a'}^{\text{v}}), \quad (17)$$

where $a, a' \in \{1, \ldots, \partial(v_j)\}$ and ϵ is the erasure probability of the BEC. The largest channel value ϵ for which (16) and (17) converge, denoted ϵ_{BP}, is the *threshold* of the BP decoder for the GLDPC code ensemble.

B. Bounding MAP Thresholds With BP Extrinsic Information Transfer Functions

The extrinsic probability $p_{\text{BP,extr}}(v_j, \epsilon)$ that a symbol associated with variable node v_j remains erased after ℓ iterations of BP decoding can be expressed as

$$p_{\text{BP,extr}}(v_j, \epsilon) = \prod_{a} q^{(\ell)}(e_{j,a}^{\text{v}}). \quad (18)$$

Note that here the product is over *all* incoming messages to v_j and the channel erasure probability ϵ does not appear in the expression but implicitly involved in the calculation of $q^{(\ell)}(e_{j,a}^{\text{v}})$. The BP extrinsic information transfer (EXIT) function $h_{\text{BP}}(\epsilon)$ [33] is given by the average of $p_{\text{BP,extr}}(v_j, \epsilon)$ over all transmitted variable nodes $v_j \in V$, *i.e.*, the average is computed excluding all of the punctured variable nodes.

Example 5: Consider the $(2, 7)$-regular protograph-based GLDPC block code ensemble with Hamming component codes of length $n^c = 7$ from Example 1. The BP EXIT function $h_{\text{BP}}(\epsilon)$ of this ensemble is shown in Fig. 6. The vertical line indicates the channel value at which the grey area below the curve is equal to the rate of the ensemble, which forms an upper bound $\epsilon_{\text{MAP}} = 0.856$ on the threshold of an optimal MAP decoder. This follows from the area theorem [34] and the fact that $h_{\text{BP}}(\epsilon)$ can never be below the EXIT function of the MAP decoder. In this case, the calculated BP threshold is given by $\epsilon_{\text{BP}} = 0.756$, and we see that there exists a large gap between the BP and the MAP thresholds, which indicates the suboptimality associated with BP decoding.[8] □

C. Threshold Saturation of Terminated SC-GLDPC Code Ensembles

Assume now that we start encoding at time $t = 0$ and terminate after L time instants. As a result we obtain the terminated base matrix $\mathbf{B}_{[0,L-1]}$ from (10). These terminated

[8] A detailed analysis of unstructured irregular ensembles, including results on the tightness of this bound, can be found in [33]. For structured protograph ensembles, this technique has been applied in [35].

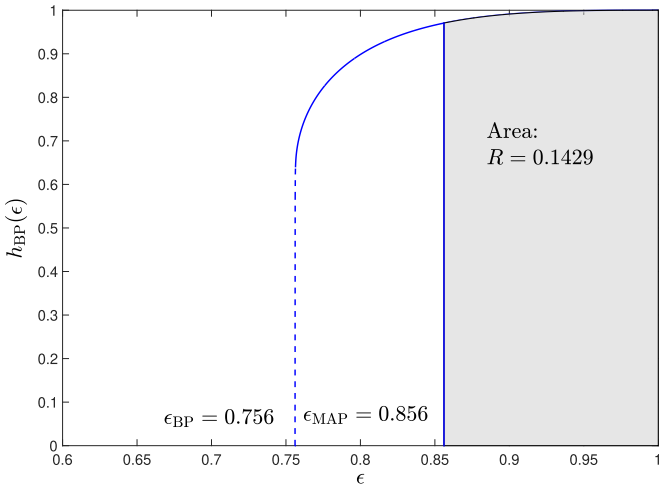

Fig. 6. BP EXIT function of a $(2, 7)$-regular protograph-based GLDPC block code ensemble with $(7, 4)$ Hamming component codes.

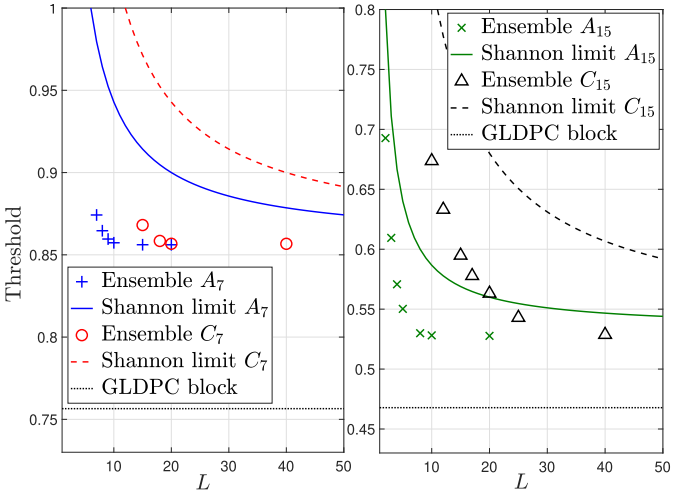

Fig. 7. BP decoding thresholds as functions of the coupling length L. Ensembles A_7 and A_{15} are shown in comparison to ensembles C_7 and C_{15}, respectively. The dotted line indicates the thresholds of the underlying GLDPC block code ensembles.

SC-GLDPC codes can be interpreted as GLDPC block codes that inherit the structure of convolutional codes. The length of these codes depends not only on the lifting factor M but also on the coupling length L. For a fixed L, the BEC density evolution thresholds ϵ_{BP} corresponding to codes with base matrix $\mathbf{B}_{[0,L-1]}$ can be calculated using the method described in Section III-A. In Fig. 7, the obtained thresholds for the $w = 1$ ensembles A_7 and A_{15} are compared with the BBC-based ensembles C_7 and C_{15} for different coupling lengths L. (The larger thresholds and Shannon limits of ensembles C_7 and C_{15} compared to A_7 and A_{15} for small to moderate L is due to the fact that the larger w BBC ensembles C_7 and C_{15} have a larger rate increase parameter Δ than the $w = 1$ ensembles A_7 and A_{15}. This difference vanishes as $L \to \infty$.) The thresholds of all the $w = 1$ ensembles versus code rate are shown in Fig. 8. Analogously to SC-LDPC codes (see [28]) with SPC constraints, it can be observed that, as $L \to \infty$, the BP thresholds numerically coincide with the upper bounds on the MAP decoding thresholds of the underlying block code ensembles, thus exhibiting the *threshold saturation* phenomenon (see [10], [11]).

The BP EXIT functions of the terminated codes from ensemble A_7 are shown in Fig. 9, where it can be seen that, with increasing L, the BP and the MAP thresholds of the terminated SC-GLDPC code ensembles are converging. Moreover, the MAP thresholds (and hence the BP thresholds) of the terminated SC-GLDPC code ensemble can also be observed to converge to the MAP threshold $\epsilon_{\mathrm{MAP}} = 0.856$ of the underlying GLDPC block code ensemble with increasing L, demonstrating threshold saturation for the ensemble A_7. We note that the converging BP and MAP thresholds of the terminated SC-GLDPC code ensembles implies that, asymptotically, BP decoding of SC-GLDPC codes provides optimal (MAP) decoding performance.

Further, we note that large values of L are realistic in conjunction with sliding window decoders, like those suggested in [36], where decoding delay and storage requirements depend on the window size W, which is independent of the

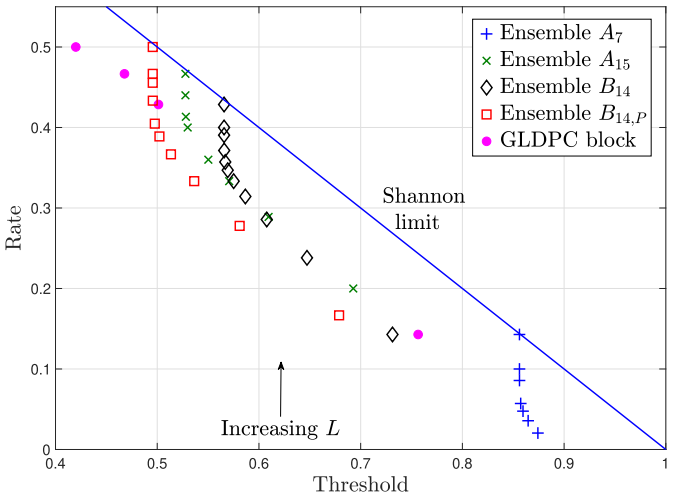

Fig. 8. BP decoding thresholds versus code rate for different $w = 1$ SC-GLDPC code ensembles and coupling lengths L.

coupling length L (typically $W \ll L$) of the transmitted code sequences. For shorter values of L, which induce rate loss, BP decoding of terminated SC-GLDPC codes is suboptimal but still provides a flexible adjustment between code rate and threshold (see Fig. 8).

IV. Distance Analysis of SC-GLDPC Codes

In this section, we first perform an asymptotic minimum distance analysis of terminated and tail-biting SC-GLDPC code ensembles. We then present an approach to bound the free distance growth rate of the unterminated periodically time-varying SC-GLDPC code ensemble from above and below using the minimum distance growth rates of the terminated and tail-biting SC-GLDPC code ensembles, respectively, and provide numerical results. The ensemble free distance growth rate provides a measure of the strength of SC-GLDPC codes and an assessment of their ML decoding performance.

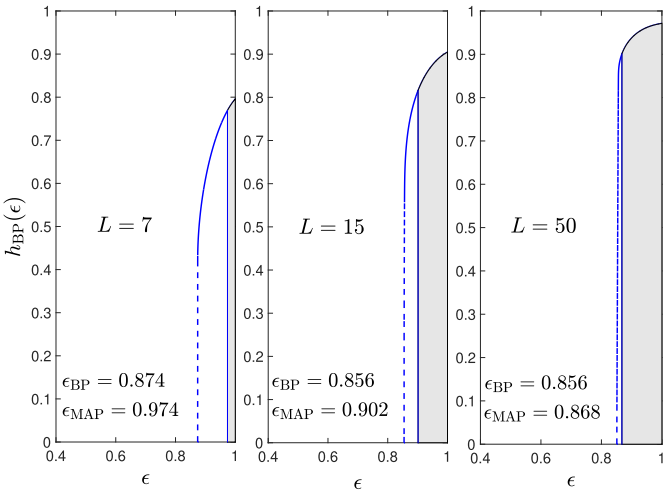

Fig. 9. BP EXIT functions of terminated SC-GLDPC codes from Ensemble A_7 for different coupling lengths L.

A. Minimum Distance Analysis of Terminated SC-GLDPC Codes

From the convolutional protograph with base matrix $\mathbf{B}_{[0,\infty]}$ in (4), we can form a periodically time-varying M-fold graph cover with period T by applying the graph lifting operation described in Section II to the $b_c \times b_v$ submatrices $\mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_w$ in the first T columns of $\mathbf{B}_{[0,\infty]}$ to form $Mb_c \times Mb_v$ submatrices $\mathbf{H}_0(t), \mathbf{H}_1(t+1), \ldots, \mathbf{H}_w(t+w)$, respectively, for $t \in \{0, 1, \ldots, T-1\}$. These submatrices can then be repeated periodically (and indefinitely) to form a convolutional constraint matrix $\mathbf{H}_{[0,\infty]}$ such that $\mathbf{H}_i(t+T) = \mathbf{H}_i(t)$, $\forall i, t$. An ensemble of periodically time-varying SC-GLDPC codes with period T, design rate $R = 1 - Mm^c b_c/Mb_v = 1 - m^c b_c/b_v$,[9] and decoding constraint length $\nu_s = (w+1)Mb_v$ can then be derived by letting the permutation matrices used to form $\mathbf{H}_0(t), \mathbf{H}_1(t+1), \ldots, \mathbf{H}_w(t+w)$, for $t \in \{0, 1, \ldots, T-1\}$, vary over all $M!$ choices of an $M \times M$ permutation matrix.

In [37], Abu-Surra, Divsalar, and Ryan presented a technique to calculate the average weight enumerator and asymptotic spectral shape function $r(\delta)$ for protograph-based GLDPC block code ensembles. The spectral shape function can be used to test if an ensemble is *asymptotically good*, *i.e.*, if the minimum distance typical of most members of the ensemble is at least as large as $\delta_{\min} n$, where $\delta_{\min}$ is the *minimum distance growth rate* of the ensemble and n is the block length.[10]

Example 6: Consider the $(2, 7)$-regular GLDPC block code protograph with base matrix $\mathbf{B}$ from (2) and the generalized constraint nodes shown in Fig. 1. If we suppose the constraint codes to be $(7, 4)$ Hamming codes with parity-check matrix $\mathbf{H}_c$ from (3), then the resulting ensemble has design rate $R = 1/7$, is asymptotically good, and has growth rate $\delta_{\min} = 0.186$ [37]. □

[9]For simplicity, we again assume here the case that each generalized constraint node c_i has m^c independent parity checks.

[10]Suppose that the first positive zero crossing of $r(\delta)$ occurs at $\delta = \delta_{\min}$. If $r(\delta)$ is negative in the range $0 < \delta < \delta_{\min}$, then $\delta_{\min}$ is called the minimum distance growth rate of the code ensemble, see [37].

We now consider the associated $(2, 7)$-regular terminated SC-GLDPC code ensembles A_7 discussed above in Sections II and III, whose protograph is shown in Fig. 5, with constraints corresponding to the $(7, 4)$ Hamming code with parity-check matrix $\mathbf{H}_c$. After termination, the design rate of the ensemble is given by

$$R_L = 1 - \frac{6(L+1) - 2}{7L}, \tag{19}$$

where $\Delta = 2$ in this case because the two leftmost (shortened) constraint nodes in Fig. 2 correspond to shortened codes with rate $1/3$, *i.e.*, the number of parity checks in these two constraint nodes is 2, while all the other constraint nodes have $m^c = 3$ parity-checks. These ensembles were shown to have thresholds numerically indistinguishable from the MAP threshold of the underlying GLDPC block code ensemble as $L \to \infty$ in Section III.

The asymptotic weight enumerator corresponding to a terminated or tail-biting convolutional protograph can be determined by applying the general approach of [37]; however, we note that a useful conjecture regarding simplification of the numerical evaluation proposed in [37] cannot be applied to SC ensembles. This conjecture relies on grouping together nodes of the same type and optimizing them together, but in the SC-GLDPC case, nodes from different time instants must be optimized separately, even if they are of the same type.

Fig. 10 shows the asymptotic spectral shape functions for the SC-GLDPC code ensembles A_7 with coupling lengths $L \in \{7, 8, 10, 12, 14, 16, 18, 20\}$. Also shown are the asymptotic spectral shape functions for random codes with the corresponding rates R_L calculated using (see [1])

$$r(\delta) = H(\delta) - (1 - R_L)\ln(2), \tag{20}$$

where $H(\delta) = -(1-\delta)\ln(1-\delta) - \delta\ln(\delta)$ is the binary entropy function. We observe that the SC-GLDPC code ensembles A_7 are asymptotically good and have relatively large minimum distance growth rates, ranging from about 25% to 65% of the (optimal) random coding growth rates. This indicates that long codes chosen from these ensembles have, with probability near one, a large minimum distance. As L increases, the design rate R_L approaches $R = 1/7$ and the minimum distance growth rate decreases, as was also observed in the case of SC-LDPC codes with SPC constraints (see [28]).

B. Free Distance Analysis of SC-GLDPC Code Ensembles

In Fig. 10 we saw that the minimum distance growth rates of terminated SC-GLDPC codes decrease as the coupling length L increases. However, since SC-GLDPC codes can be decoded as unterminated (no preset coupling length) convolutional codes by employing a sliding window decoder, a more appropriate distance measure for assessing their ML decoding performance is the *free distance* growth rate of the SC-GLDPC code ensemble. In this section, we first calculate the minimum distance growth rates of TB-GLDPC code ensembles and show that, for sufficiently large coupling lengths, the growth rates coincide with those calculated for the terminated SC-GLDPC code ensembles in Section IV-A. We then show that the growth rates of the TB-GLDPC code ensembles and the terminated

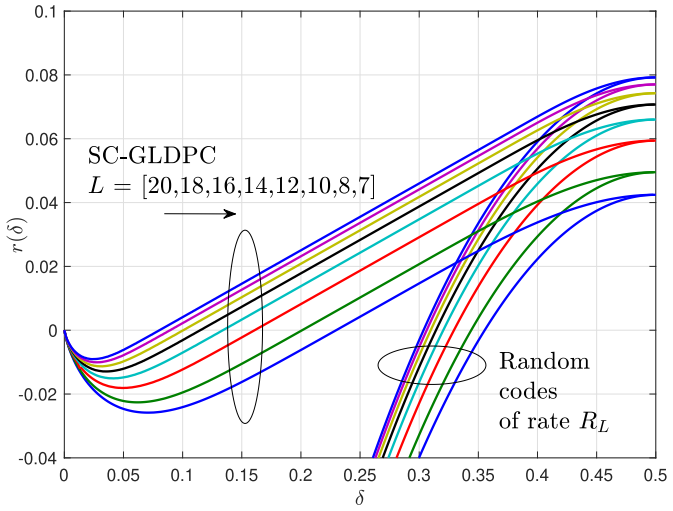

Fig. 10. Spectral shape functions of design rate R_L SC-GLDPC code ensembles A_7 and random linear codes of the corresponding rate. Curves from left to right in both groups correspond to $L = 20, 18, 16, 14, 12, 10, 8, 7$, respectively.

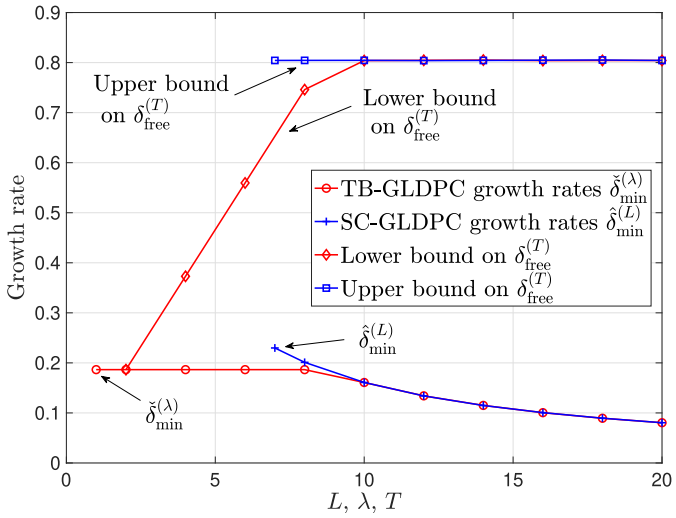

Fig. 11. Minimum distance growth rates of terminated SC-GLDPC code ensembles and TB-GLDPC code ensembles and calculated upper and lower bounds on the free distance growth rates of the associated periodically time-varying unterminated SC-GLDPC code ensembles.

SC-GLDPC code ensembles can be used to obtain lower and upper bounds on the free distance growth rate of the unterminated SC-GLDPC code ensemble, respectively.

1) Minimum Distance Analysis of TB-GLDPC Code Ensembles: We now consider terminating the protograph in Fig. 2 as a TB-GLDPC code with coupling length λ. Unlike the previous termination technique, this results in a $(2, 7)$-regular protograph with design rate $R_\lambda = 1/7$ for all λ. The minimum distance growth rates $\check{\delta}_{\min}^{(\lambda)}$ of the TB-GLDPC code ensembles are presented in Fig. 11 alongside those corresponding to the terminated SC-GLDPC code ensembles $\hat{\delta}_{\min}^{(L)}$. We observe that the TB-GLDPC growth rates remain constant at $\check{\delta}_{\min}^{(\lambda)} = 0.186$ (the growth rate of the original GLDPC block code ensemble $\delta_{\min}$) for $\lambda = 1, 2, \ldots, 8$, and then begin to decay to zero as $\lambda \to \infty$. Also, as a result of the convolutional structure, we observe that the TB-GLDPC and SC-GLDPC growth rates coincide for $L, \lambda \geq 10$. This is the same behavior observed for TB-LDPC and SC-LDPC codes with SPC constraints [38].

2) Free Distance Bounds for SC-GLDPC Code Ensembles: Now consider an ensemble of periodically time-varying unterminated SC-GLDPC codes with rate $R = 1 - b_c m^c / b_v$ and period T constructed from a convolutional protograph with base matrix $\mathbf{B}_{[0,\infty]}$ (see (4)) as described in Section II-A. Using a modification of the proof techniques in [38], [39], we can show that the average free distance of this ensemble is bounded below by the average minimum distance of an ensemble of TB-GLDPC codes derived from the base matrix $\mathbf{B}_{tb}^{(\lambda)}$ (see (12)) with coupling length $\lambda = T$. Here, we show that the average free distance of the unterminated SC-GLDPC ensemble can also be bounded above by the average minimum distance of the ensemble of terminated SC-GLDPC codes derived from the base matrix $\mathbf{B}_{[0,L-1]}$ (see (10)) with coupling length $L = T$.

Theorem 1: Consider a rate $R = 1 - b_c m^c / b_v$ unterminated, periodically time-varying SC-GLDPC code ensemble with syndrome former memory w, decoding constraint length $\nu_s = M(w + 1)b_v$, and period T derived from $\mathbf{B}_{[0,\infty]}$. Let $\overline{d}_{\min}^{(L)}$ be the average minimum distance of the terminated SC-GLDPC code ensemble with block length $n = Mb_vL$ and coupling length L. Then the ensemble average free distance $\overline{d}_{\text{free}}^{(T)}$ of the unterminated SC-GLDPC code ensemble is bounded above by $\overline{d}_{\min}^{(L)}$ for termination factor $L = T$, *i.e.*,

$$\overline{d}_{\text{free}}^{(T)} \leq \overline{d}_{\min}^{(T)}. \tag{21}$$

Proof: There is a one-to-one relationship between members of the periodically time-varying unterminated SC-GLDPC code ensemble and members of the corresponding terminated SC-GLDPC code ensemble with coupling length $L = T$. For any such pair of codes, every codeword $\mathbf{x} = [\, x_0 \;\; x_1 \;\; \cdots \;\; x_{Mb_vL-1} \,]$ in the terminated SC-GLDPC code can also be viewed as a codeword $\mathbf{x}_{[0,\infty]} = [\, x_0 \;\; x_1 \;\; \cdots \;\; x_{Mb_vL-1} \;\; 0 \;\; \cdots \,]$ in the unterminated code. It follows that the free distance $d_{\text{free}}^{(T)}$ of the unterminated code cannot be larger than the minimum distance $d_{\min}^{(T)}$ of the terminated code. The ensemble average result $\overline{d}_{\text{free}}^{(T)} \leq \overline{d}_{\min}^{(T)}$ then follows directly. □

Since there is no danger of ambiguity, we will henceforth drop the overline notation when discussing ensemble average distance measures.

3) Free Distance Growth Rates of SC-GLDPC Code Ensembles: For unterminated SC-GLDPC codes, it is natural to define the free distance growth rate with respect to the decoding constraint length ν_s, *i.e.*, as the ratio of the free distance d_{free} to ν_s. By bounding $d_{\text{free}}^{(T)}$ using (21), we obtain an upper bound on the free distance growth rate as

$$\delta_{\text{free}}^{(T)} = \frac{d_{\text{free}}^{(T)}}{\nu_s} \leq \frac{\hat{\delta}_{\min}^{(T)} T}{(w + 1)}, \tag{22}$$

where $\hat{\delta}_{\min}^{(T)} = d_{\min}^{(T)}/n = d_{\min}^{(T)}/(Mb_vT)$ is the minimum distance growth rate of the terminated SC-GLDPC code ensemble

with coupling length $L = T$ and base matrix $\mathbf{B}_{[0,T-1]}$. Further, using a similar argument to that presented in [38], we have

$$\delta_{\text{free}}^{(T)} \geq \frac{\check{\delta}_{\min}^{(T)} T}{(w+1)}, \tag{23}$$

where $\check{\delta}_{\min}^{(T)}$ is the minimum distance growth rate of the TB-GLDPC code ensemble with tail-biting coupling length $\lambda = T$ and base matrix $\mathbf{B}_{tb}^{(\lambda)}$.

The free distance growth rate $\delta_{\text{free}}^{(T)}$ that we bound from above using (22) is, by definition, an existence-type lower bound on the free distance of most members of the ensemble, *i.e.*, with high probability a randomly chosen code from the ensemble has minimum free distance at least as large as $\delta_{\text{free}}^{(T)}\nu_s$ as $\nu_s \to \infty$.

4) Numerical Results: As an example, we consider once more the $(2, 7)$-regular SC-GLDPC code ensemble A_7 with memory $w = 1$ and design rate $R = 1/7$ depicted in Fig. 2. For this case, we calculate the upper bound on the free distance growth rate of the periodically time-varying unterminated SC-GLDPC code ensemble as $\delta_{\text{free}}^{(T)} \leq \hat{\delta}_{\min}^{(T)} T/2$ using (22) for coupling lengths $L = T \geq 7$. Fig. 11 displays the minimum distance growth rates $\hat{\delta}_{\min}^{(L)}$ of the terminated SC-GLDPC code ensembles defined by $\mathbf{B}_{[0,L-1]}$ for $L \in \{7, 8, 10, 12, \ldots, 20\}$ that were calculated using the technique proposed in [37] and the associated upper bounds on the unterminated SC-GLDPC code ensemble growth rates $\delta_{\text{free}}^{(T)} \leq \hat{\delta}_{\min}^{(T)} T/2$ for $L = T$. Also shown are the minimum distance growth rates $\check{\delta}_{\min}^{(\lambda)}$ of the TB-GLDPC code ensembles defined by base matrix $\mathbf{B}_{tb}^{(\lambda)}$ for $\lambda \in \{1, 2, 4, \ldots, 20\}$ and the associated lower bounds on the unterminated SC-GLDPC code ensemble growth rates $\delta_{\text{free}}^{(T)} \geq \check{\delta}_{\min}^{(T)} T/2$ for $\lambda = T$ calculated using (23).

We observe that the calculated TB-GLDPC code ensemble minimum distance growth rates $\check{\delta}_{\min}^{(\lambda)}$ remain constant for $\lambda = 1, \ldots, 8$ and then start to decrease as the coupling length λ grows, tending to zero as λ tends to infinity. Correspondingly, as λ exceeds 8, the lower bound calculated for $\delta_{\text{free}}^{(T)}$ levels off at $\delta_{\text{free}}^{(T)} \geq 0.805$. The calculated terminated SC-GLDPC code ensemble minimum distance growth rates $\hat{\delta}_{\min}^{(L)}$ are larger for small values of L (where the rate loss is larger) and decrease monotonically to zero as $L \to \infty$. Using (22) to obtain an upper bound on $\delta_{\text{free}}^{(T)}$ we observe that, for $T \geq 10$, the upper and lower bounds coincide, indicating that, for these values of the period T, $\delta_{\text{free}}^{(T)} = 0.805$, significantly larger than the minimum distance growth rate $\delta_{\min} = 0.186$ of the underlying GLDPC block code ensemble.[11] In addition, we note that, at the point where the upper and lower bounds on $\delta_{\text{free}}^{(T)}$ coincide, the minimum distance growth rates for both termination methods also coincide. Recall that the TB-GLDPC code ensembles all have rate $1/7$, wheras the rate of the SC-GLDPC code ensembles is a function of the coupling length L given by (19). Although we have demonstrated the approach only for ensemble A_7, the general technique can be used to bound the free distance growth rate above and below for any regular or irregular periodically time-varying protograph-based unterminated SC-GLDPC code ensemble, thus allowing for the evaluation and comparison of different SC-GLDPC code designs from the perspective of minimum distance.

While large free distance growth rates are indicative of good ML decoding performance, when predicting the iterative decoding performance of a code ensemble in the high SNR region other graphical objects such as trapping sets, pseudocodewords, absorbing sets, etc., come into effect. Based on results from the SPC case [38], we would expect SC-GLDPC codes with large minimum/free distance growth rates to also have large trapping set growth rates, indicating good iterative decoding performance in the high SNR region.

[11] Note that the free distance growth rate may also be calculated with respect to the encoding constraint length ν_e, which corresponds to the maximum number of transmitted symbols that can be affected by a single nonzero block of information digits. As a result of normalizing by the decoding constraint length, it is possible to have free distance growth rates larger than 0.5. For further details, see [38].

V. Finite-Length Analysis of GLDPC Block and SC-GLDPC Codes

To analyze the finite-length performance of LDPC codes over the BEC, a decoding method called peeling decoding (PD) can be employed [40]. PD is a simple algorithm that is initialized by removing all of the correctly received variable nodes, as well as their attached edges, from the Tanner graph of $\mathbf{H}$ after BEC transmission. The algorithm then iteratively removes degree-one check nodes from the graph, along with their attached variable nodes and edges. We now describe an extension of PD to GLDPC block and terminated SC-GLDPC codes, referred to as generalized peeling decoding (GPD).

A. Type Vectors and Initialization of GPD

Recall that

$$\begin{aligned} E &= \{e_{j,a}^{\text{v}} \,|\, j \in \{1, 2, \ldots, n_v\}, a \ \in \{1, 2, \ldots, \partial(v_j)\}\} \\ &= \{e_{i,b}^{\text{c}} \,|\, i \in \{1, 2, \ldots, n_c\}, b \in \{1, 2, \ldots, \partial(c_i)\}\} \end{aligned}$$

represents the set of edges in a given protograph. In the Tanner graph of a resulting lifted constraint matrix $\mathbf{H}$ (resp. $\mathbf{H}_{[0,L-1]}$), we say that a particular edge is of *type* $e_{j,a}^{\text{v}}$ if it connects a variable node and a constraint node that are copies of the two nodes that edge $e_{j,a}^{\text{v}}$ connects in the protograph. For a variable node v in the Tanner graph of $\mathbf{H}$ (resp. $\mathbf{H}_{[0,L-1]}$), we also define the *variable node type* by a binary $|E|$-dimensional vector $\mathbf{t}_v$, where each entry is a "1" iff a particular edge type is connected to variable node v. Similarly, for a constraint node c, we define its type by a binary $|E|$-dimensional vector $\mathbf{t}_c$. We denote the set of variable and constraint node types in the Tanner graph of $\mathbf{H}$ (resp. $\mathbf{H}_{[0,L-1]}$) by $\mathcal{F}_{\mathbf{t}_v}$ and $\mathcal{F}_{\mathbf{t}_c}$, respectively. Note that the sets $\mathcal{F}_{\mathbf{t}_v}$ and $\mathcal{F}_{\mathbf{t}_c}$ are determined from the connectivity of the protograph, as will be illustrated in Example 7 below. Finally, we let $L_{\mathbf{t}_v}$ and $R_{\mathbf{t}_c}$ represent the number of variable and constraint nodes of type $\mathbf{t}_v$ and $\mathbf{t}_c$ in $\mathbf{H}$ (resp. $\mathbf{H}_{[0,L-1]}$), respectively.

The details of the GPD algorithm are presented in Section V-B. We first discuss GPD initialization, which is identical to PD initialization: the correctly received variable nodes of $\mathbf{H}$ (resp. $\mathbf{H}_{[0,L-1]}$) and their attached edges are removed from the graph. After initialization, the residual graph contains constraint nodes with types that are not included in

$\mathcal{F}_{\mathbf{t}_c}$, the set of constraint node types in the original graph, but the set of variable node types $\mathcal{F}_{\mathbf{t}_v}$ remains the same. We now define $\mathcal{D}(\mathbf{t}_c)$ as the set of constraint node types that can appear in the graph of $\mathbf{H}$ (resp. $\mathbf{H}_{[0,L-1]}$) after GPD initalization when a constraint node of type $\mathbf{t}_c \in \mathcal{F}_{\mathbf{t}_c}$ loses one or more edges. The extended set of all possible constraint node types which are present in the residual graph after GPD initialization is then given by $\overline{\mathcal{F}}_{\mathbf{t}_c} = \bigcup_{\mathbf{t}_c \in \mathcal{F}_{\mathbf{t}_c}} \mathcal{D}(\mathbf{t}_c)$.

Example 7: To illustrate the type vectors, consider the $(2, 7)$-regular GLDPC block code protograph of Example 1. In this case, if we order the protograph edges as

$$(e^{\mathrm{c}}_{1,1}, e^{\mathrm{c}}_{1,2}, \ldots, e^{\mathrm{c}}_{1,7}, e^{\mathrm{c}}_{2,1}, e^{\mathrm{c}}_{2,2}, \ldots, e^{\mathrm{c}}_{2,7}),$$

then the type vector for c_1 is $\mathbf{t}_{c_1} = (1, 1, \ldots, 1, 0, 0, \ldots, 0) \in \mathcal{F}_{\mathbf{t}_c}$ and the type vector for c_2 is $\mathbf{t}_{c_2} = (0, 0, \ldots, 0, 1, 1, \ldots, 1) \in \mathcal{F}_{\mathbf{t}_c}$.[12] Both vectors are of length $|E| = 14$, the number of edges in the protograph, and have weight $n^c = 7$, the length of the constraint code. In any lifted graph with lifting factor M, there are precisely M copies of each edge, variable node, or constraint node of a given type, where the types are defined from the protograph as described above. Corresponding to each of the constraint node types in this example, $2^7 = 128$ residual types can appear in the graph when edges are removed. Thus, in total, $\overline{\mathcal{F}}_{\mathbf{t}_c}$ contains 256 constraint node types. □

According to the above definitions, the expected degree distribution (DD) of the residual graph after initialization can be expressed as follows:

$$L_{\mathbf{t}_v}(0) = \epsilon L_{\mathbf{t}_v}, \tag{24}$$

$$R_{\mathbf{t}'_c}(0) = \sum_{\substack{\mathbf{t}_c \in \mathcal{F}_{\mathbf{t}_c} \\ \mathbf{t}'_c \in \mathcal{D}(\mathbf{t}_c)}} R_{\mathbf{t}_c} \epsilon^{|\mathbf{t}'_c|} (1-\epsilon)^{|\mathbf{t}_c| - |\mathbf{t}'_c|}, \tag{25}$$

for $\mathbf{t}_v \in \mathcal{F}_{\mathbf{t}_v}$ and $\mathbf{t}'_c \in \overline{\mathcal{F}}_{\mathbf{t}_c}$, where $L_{\mathbf{t}_v}(0)$ (resp. $R_{\mathbf{t}'_c}(0)$) represents the number of variable (resp. constraint) nodes of type $\mathbf{t}_v$ (resp. $\mathbf{t}'_c$) after GPD initialization and $|\mathbf{t}'_c|$ (resp. $|\mathbf{t}_c|$) is the weight of the vector $\mathbf{t}'_c$ (resp. $\mathbf{t}_c$).

B. Decodable Constraint Nodes and the GPD

In general, each constraint node type in the protograph $\mathbf{B}$ (resp. $\mathbf{B}_{[0,L-1]}$) of a GLDPC code can be associated with a different constraint code. Let $\mathcal{C}_{\mathbf{t}_c}$ be the constraint code associated with the constraint nodes in the base matrix $\mathbf{B}$ (resp. $\mathbf{B}_{[0,L-1]}$) of type $\mathbf{t}_c \in \mathcal{F}_{\mathbf{t}_c}$. By extension, each constraint node in the graph of $\mathbf{H}$ (resp. $\mathbf{H}_{[0,L-1]}$) is associated with a constraint code according to its type. After GPD initialization, the type of a given constraint node can be modified from $\mathbf{t}_c$ to $\mathbf{t}'_c$, where $|\mathbf{t}'_c| < |\mathbf{t}_c|$, and we say that $\mathbf{t}'_c$ is the input erasure pattern seen by the constraint code $\mathcal{C}_{\mathbf{t}_c}$. The question now is if, by decoding the constraint code $\mathcal{C}_{\mathbf{t}_c}$ associated with a constraint node of type $\mathbf{t}'_c$ in the residual graph using a given decoding algorithm, for instance ML decoding or some suboptimal algorithm, we are able to recover the $|\mathbf{t}'_c|$ variables still connected to the constraint node. In general, for each constraint code $\mathcal{C}_{\mathbf{t}_c}$, only a subset of input erasure patterns can be decoded. This subset is denoted by $\mathcal{A}(\mathbf{t}_c) \subset \mathcal{D}(\mathbf{t}_c)$. If a constraint node in the residual graph is of type $\mathbf{t}'_c \in \mathcal{A}(\mathbf{t}_c)$, then we say it is a *decodable* constraint node and $\mathbf{t}'_c$ a *decodable* constraint node type. For example, if all the constraint codes are SPCs, then only constraint node types with input erasure patterns containing exactly one erasure are decodable. However, if the constraint code is a $(7, 4)$ Hamming code with ML decoding, then all input erasure patterns with one and two erasures and some input erasure patterns of weight three are decodable.

The set of all decodable constraint node types is defined as $\mathcal{A} \doteq \cup_{\mathbf{t}_c \in \mathcal{F}_{\mathbf{t}_c}} \mathcal{A}(\mathbf{t}_c) \subset \overline{\mathcal{F}}_{\mathbf{t}_c}$. Given the discussion above, the GPD algorithm can now be seen as a straightforward extension of PD for LDPC codes to GLDPC codes. After the graph is initialized, GPD chooses one constraint node at random from the graph that is decodable. This constraint node, all connected variable nodes, and all attached edges are then removed from the graph. GPD continues in this way until there are no further constraint nodes that can be removed from the graph, which corresponds to a *decoding failure*, or until there are no variable nodes left in the graph, which corresponds to *successful decoding*.

C. Expected Graph Evolution

We now define the normalized DD at time τ as

$$\tau \doteq \frac{\ell}{n}, \qquad r_{\mathbf{t}_c}(\tau) \doteq \frac{R_{\mathbf{t}_c}(\tau)}{n}, \qquad l_{\mathbf{t}_v}(\tau) \doteq \frac{L_{\mathbf{t}_v}(\tau)}{n}, \tag{26}$$

where ℓ is the GPD iteration index, $R_{\mathbf{t}_c}(\tau)$ (resp. $L_{\mathbf{t}_v}(\tau)$) is the number of constraint (resp. variable) nodes in the graph of type $\mathbf{t}_c$ (resp. $\mathbf{t}_v$) at time τ, and $n = Mb_v$ (resp. Mb_vL) is the block (resp. termination) length.

In [41], it is shown that if we apply PD to elements of an LDPC code ensemble, then the expected DD of the sequence of residual graphs can be described as the solution of a set of differential equations. This analysis is based on a result on the evolution of Markov processes due to Wormald [42]. Furthermore, the deviation of the process w.r.t. the expected evolution decreases exponentially fast with the LDPC code block length, and this result was used in [40] to analyze the finite-length BEC performance of LDPC block codes. This methodology has been extended to unstructured GLDPC codes in [43] and also to spatially coupled codes with split-component codes [23], which can be consider as a particular sub-class of SC-GLDPC codes. In a similar way, we can investigate the finite-length BEC performance of GLDPC codes constructed from protographs by analyzing the statistical evolution of the normalized DD in (26) during the decoding process. As shown in [40], the expected value of $r_{\mathbf{t}_c}(\tau)$ and $l_{\mathbf{t}_v}(\tau)$, denoted by $\hat{r}_{\mathbf{t}_c}(\tau)$ and $\hat{l}_{\mathbf{t}_v}(\tau)$, respectively, can be computed as the solution to the following system of differential equations

$$\frac{\partial \hat{l}_{\mathbf{t}_v}(\tau)}{\partial \tau} = \mathbb{E}\left[L_{\mathbf{t}_v}\left(\tau + \frac{1}{n}\right) - L_{\mathbf{t}_v}(\tau) \middle| \{\hat{l}_{\mathbf{t}_v}(\tau), \hat{r}_{\mathbf{t}_c}(\tau)\}_{\mathbf{t}_v \in \mathcal{F}_{\mathbf{t}_v}, \mathbf{t}_c \in \overline{\mathcal{F}}_{\mathbf{t}_c}} \right],$$

[12] The ordering of the protograph edges (which can affect the entries of the type vector) does not matter provided that we use a consistent ordering for all node types.

(27)

$$\frac{\partial \hat{r}_{\mathbf{t}_c}(\tau)}{\partial \tau} = \mathbb{E}\left[R_{\mathbf{t}_c}\left(\tau+\frac{1}{n}\right) - R_{\mathbf{t}_c}(\tau)\middle|\{\hat{l}_{\mathbf{t}_v}(\tau), \hat{r}_{\mathbf{t}_c}(\tau)\}_{\mathbf{t}_v\in\mathcal{F}_{\mathbf{t}_v},\mathbf{t}_c\in\overline{\mathcal{F}}_{\mathbf{t}_c}}\right], \tag{28}$$

i.e., the derivative of $\hat{r}_{\mathbf{t}_c}(\tau)$ w.r.t. τ in (28) can be evaluated by computing the variation in the number of constraint nodes of type $\mathbf{t}_c$ with GPD iteration given that the normalized DD at time τ is at its mean $\{\hat{l}_{\mathbf{t}_v}(\tau), \hat{r}_{\mathbf{t}_c}(\tau)\}_{\mathbf{t}_v\in\mathcal{F}_{\mathbf{t}_v},\mathbf{t}_c\in\overline{\mathcal{F}}_{\mathbf{t}_c}}$.[13] A similar interpretation holds for (27). Further, the solution to (27) and (28) is unique and, with probability $1-\mathcal{O}(\mathrm{e}^{-\sqrt{n}})$, any particular realization of the normalized DD in (26) deviates from its mean by a factor of less than $n^{-1/6}$ for the initial conditions

$$\hat{r}_{\mathbf{t}_c}(0) = \mathbb{E}[r_{\mathbf{t}_c}(\ell=0)] = \mathbb{E}[R_{\mathbf{t}_c}(\ell=0)]/n, \tag{29}$$

$$\hat{l}_{\mathbf{t}_v}(0) = \mathbb{E}[l_{\mathbf{t}_v}(\ell=0)] = \mathbb{E}[L_{\mathbf{t}_v}(\ell=0)]/n, \tag{30}$$

which can be computed from (24) and (25) [40]. The actual computation of the expectations in (27) and (28) is described in [20]. The GPD threshold is defined as the maximum value of ϵ for which the expected fraction of decodable constraint nodes

$$\hat{a}(\tau) \doteq \sum_{\mathbf{t}_c\in\mathcal{A}} \hat{r}_{\mathbf{t}_c}(\tau) \tag{31}$$

is positive for any $\tau \in [0,\epsilon)$, where $\hat{a}(\tau)$ is the mean of the random process

$$a(\tau) \doteq \sum_{\mathbf{t}_c\in\mathcal{A}} r_{\mathbf{t}_c}(\tau). \tag{32}$$

Finally, we can compute the expected fraction of variable nodes in the graph at any time τ, denoted by $\hat{v}(\tau)$, as

$$\hat{v}(\tau) \doteq \sum_{\mathbf{t}_v\in\mathcal{F}_{\mathbf{t}_v}} \hat{l}_{\mathbf{t}_v}(\tau). \tag{33}$$

In addition to characterizing the asymptotic behavior, *i.e.*, to computing the GPD threshold ensemble, the solution to the system of equations given by (27) and (28) can be used to determine the quantities needed to assess the finite-length performance of GLDPC block and terminated SC-GLDPC codes. We refer to *critical points* as the points in time for which $\hat{a}(\tau)$ has a local minima. As shown in [40], the average (over the ensemble of codes) error probability is dominated by the probability that the process $a(\tau)$ *survives*, *i.e.*, does not go to zero around the critical points. Therefore, characterizing the critical points and the expected fraction of decodable constraint nodes in the graph at those points in time are the parameters needed to determine the GLDPC block or terminated SC-GLDPC code finite-length performance, and they can be computed from (27) and (28).

[13]This result, based on Wormald's theorem [42], requires certain conditions on the random sequence of GLDPC Tanner graphs during GPD to be met. We do not include the proof that these conditions are met, but an equivalent proof for unstructured GLDPC codes can be found in [43].

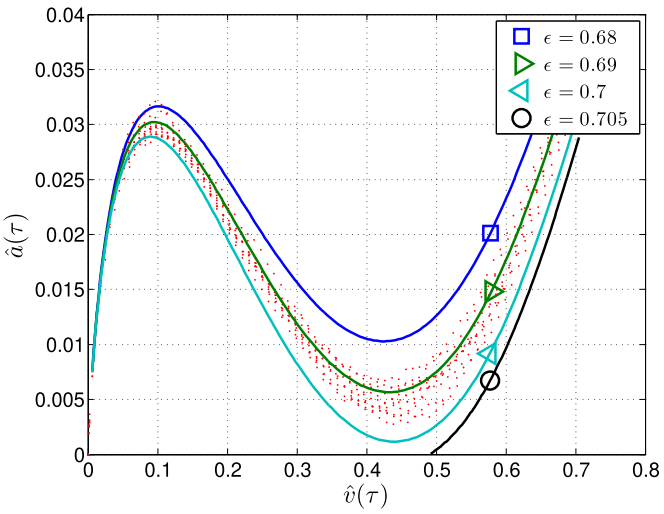

Fig. 12. Evolution of the expected fraction of decodable constraint nodes $\hat{a}(\tau)$ in the residual graph during iterations of the GPD for the $(2,7)$-regular GLDPC block code ensemble with $(7,4)$ Hamming constraint codes and an ML decoder. The dotted curves represent simulated trajectories computed for $\epsilon = 0.69$ with lifting factor $M = 4000$.

D. Numerical Results: GLDPC Block Codes

With the tools described above, we can now investigate the asymptotic and finite-length performance of GLDPC code ensembles. We start by considering the uncoupled $(2,7)$-regular GLDPC block code ensemble from Example 1.

Example 8: Consider the $(2,7)$-regular GLDPC block code ensemble of Example 1. Assume that a $(7,4)$ Hamming code is associated with each of the two constraint nodes and that the constraint codes are decoded using ML decoding. The design rate of this ensemble is $R = 1/7$. All constraint node types with one or two erasures can be decoded, as well as some constraint node types with three erasures. Fig. 12 shows the evolution of the expected fraction of decodable constraint nodes $\hat{a}(\tau)$ versus the expected fraction of variable nodes $\hat{v}(\tau)$ in the graph for different ϵ values.[14] We also include a set of 10 simulated trajectories of $a(\tau)$ for $\epsilon = 0.69$ to demonstrate that they concentrate around the predicted mean. Note first that $\hat{a}(\tau)$ has a single critical point at $\hat{v}(\tau^*) \approx 0.43$. Indeed, we can compute the threshold ϵ^* as the maximum value of ϵ for which the minimum is exactly zero, and in this case we obtain $\epsilon^* \approx 0.7025$.

The finite-length error probability is dominated by the statistics of $a(\tau)$ around τ^*. Following [40], for each n and ϵ pair, we can estimate the finite-length error probability as

$$P_{\text{Block}} = \Pr(a(\tau)=0, v(\tau)>0) \approx \mathcal{Q}\left(\frac{\hat{a}(\tau^*)}{\sqrt{\text{Var}[a(\tau^*)]}}\right), \tag{34}$$

where $\hat{a}(\tau^*)$ is the expected value of $a(\tau)$ at τ^* and $\text{Var}[a(\tau^*)]$ represents its variance. $\hat{a}(\tau^*)$ was computed using numerical

[14]Note that the time variable τ runs backwards in this figure (right to left), in the sense that small values of τ correspond to $\hat{v}(\tau)$ on the right, where the graph still contains a relatively large fraction of variable nodes, whereas large values of τ correspond to small values of $\hat{v}(\tau)$ on the left.

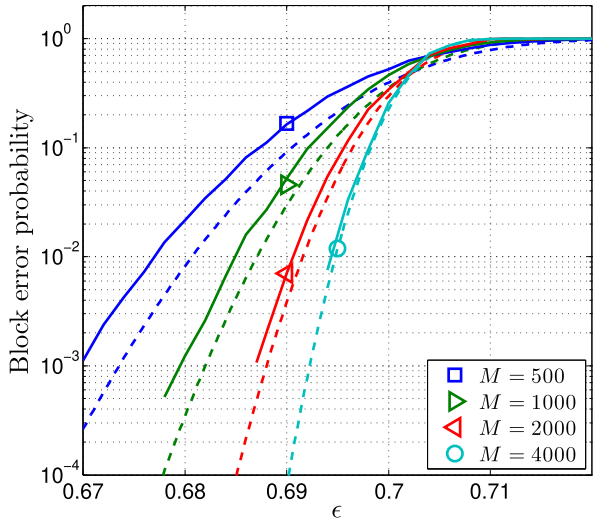

Fig. 13. Simulated performance (solid lines) and estimated performance using (35) (dashed lines) for the $(2, 7)$-regular GLDPC block code ensemble with $(7, 4)$ Hamming constraint codes and an ML decoder.

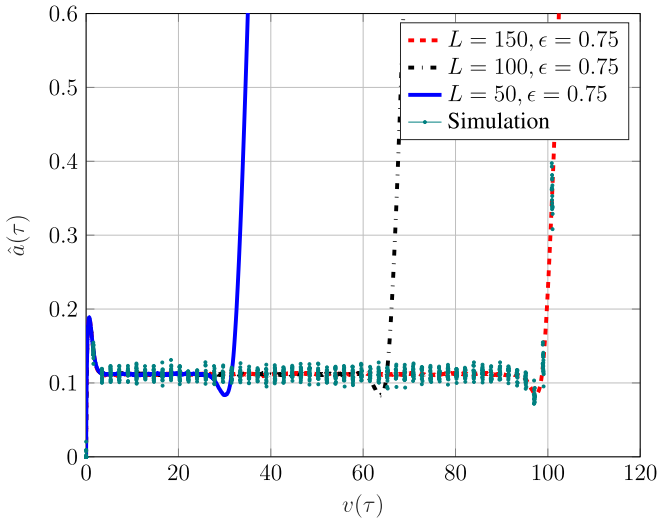

Fig. 14. Evolution of the expected fraction of decodable constraint nodes $\hat{a}(\tau)$ in the residual graph during iterations of the GPD for the terminated $(2, 7)$-regular A_7 SC-GLDPC code ensemble with coupling lengths $L = 50$, 100, and 150, $(7, 4)$ Hamming constraint codes, and an ML decoder. The dotted curves represent simulated trajectories for $L = 150$ and lifting factor $M = 1000$.

integration of the system of differential equations in (27)-(28), and $\mathrm{Var}[a(\tau^*)]$ was estimated using Monte Carlo simulation.[15] Also in [40], the authors showed that the ratio of the expected number of degree-one constraint nodes to the standard deviation at the critical point approximately scales as $\alpha\sqrt{n}(\epsilon^* - \epsilon)$, where α is a *scaling parameter* that only depends on the DD. In the GLDPC case, simulated trajectories for $a(\tau)$ suggest that the same scaling holds and that the performance for any pair (n, ϵ) can be estimated as

$$P_{\text{Block}} \approx \mathcal{Q}\left(\alpha\sqrt{n}(\epsilon^* - \epsilon)\right). \tag{35}$$

After computing $\hat{a}(\tau^*)/\sqrt{\mathrm{Var}[a(\tau^*)]}$ for a given (n, ϵ) pair, we estimate α by equating the arguments in (34) and (35), so that

$$\alpha = \frac{1}{\sqrt{n}(\epsilon^* - \epsilon)} \frac{\hat{a}(\tau^*)}{\sqrt{\mathrm{Var}[a(\tau^*)]}}. \tag{36}$$

For the $(2, 7)$-regular GLDPC block code ensemble with $(7, 4)$ Hamming constraint codes and an ML decoder, we thus obtain $\alpha = 1.8024$. In Fig. 13, we plot the simulated performance versus (35), where we observe that the estimate is very accurate for a sufficiently large lifting factor. □

E. Numerical Results: SC-GLDPC Codes

We now investigate the asymptotic and finite-length performance of a terminated version of the coupled A_7 SC-GLDPC code ensemble from Example 2.

Example 9: Following a similar procedure as in Example 8, we now analyze the finite-length behavior of terminated SC-GLDPC codes. In Fig. 14, we show the evolution of the expected fraction $\hat{a}(\tau)$ of decodable check nodes during iterations of the GPD for a terminated version of the A_7 SC-GLDPC code ensemble of Example 2 (corresponding to the GLDPC block code ensemble of Example 1) with coupling lengths $L = 50$, 100, and 150, lifting factor $M = 1000$, and a channel parameter $\epsilon = 0.75$. Also included is a set of simulated decoding trajectories of $a(\tau)$, computed for $L = 150$. Unlike the GLDPC block code, the expected evolution $\hat{a}(\tau)$ displays a constant *critical phase* that corresponds to a decoding wave traveling towards the central positions of the graph. Further, the critical value $\hat{a}(\tau^*)$ during such a phase does not depend on L, and the length of the critical phase is roughly proportional to L. The threshold ϵ^* is given by the maximum value of ϵ for which the critical value is exactly zero, and in this case we obtain $\epsilon^* = 0.8$.[16] Similar effects were first described in [44] for (non-generalized) terminated SC-LDPC codes.

As also suggested in [44], using simulated decoding trajectories we should observe that $\mathrm{Var}[a(\tau)]$ is fairly flat during the critical phase and that the covariance $\mathrm{CoVar}[a(\tau), a(\xi)]$ decays exponentially fast with $|\tau - \xi|$, with a rate of decay that we denote by θ. Fig. 15 demonstrates that this is indeed the case. In Fig. 15(a), we show the empirical variance of the process $a(\tau)$ computed from 500 simulated trajectories with $L = 100$ and $M = 2000$. In Fig. 15(b), we show the empirical covariance $\mathrm{CoVar}[a(\tau), a(\xi)]$ of the process obtained from the same set of simulations, where the covariance is normalized by $\mathrm{Var}[a(\xi)]$ so that the maximum value is equal to one. Observe that an exponentially decaying function provides an accurate estimate of the normalized covariance, in which the parameter $\theta = 0.87$ was obtained by a least squares fit. Based on this

[15] $\mathrm{Var}[a(\tau^*)]$ can be obtained from the solution of the *covariance evolution* system of differential equations, first presented in [40] for LDPC code ensembles

[16] The threshold $\epsilon^* = 0.8$ computed here for GPD differs from the value $\epsilon^* \approx 0.85$ obtained earlier for the terminated A_7 ensemble with $L = 150$ (see Fig. 8) due to a difference in the method of decoding the constraint codes (ML decoding in this example vs. BCJR decoding in Fig. 8). If we change the component decoder from blockwise ML to bitwise MAP (BCJR), the scaling law in (37) that predicts the finite-length performance of the SC-GLDPC code will be the same, but with different scaling parameters.

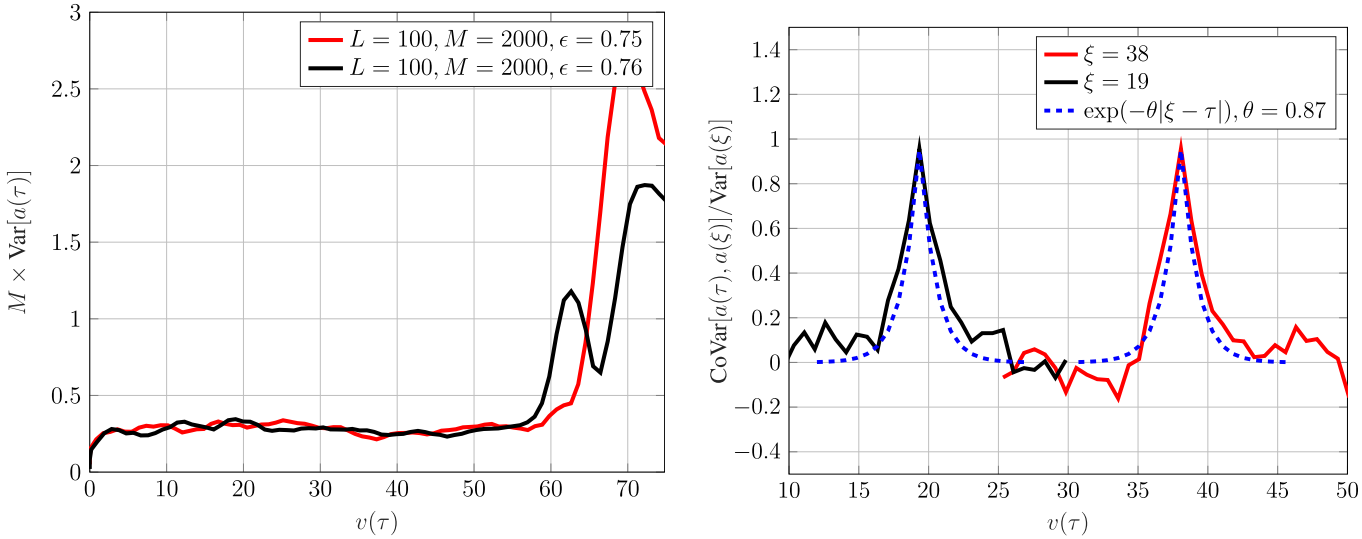

Fig. 15. Simulated trajectories of the terminated $(2, 7)$-regular A_7 SC-GLDPC code ensemble: (a) empirical variance of the process $a(\tau)$ computed from 500 simulated trajectories with $L = 100$ and $M = 2000$, and (b) empirical covariance $\mathrm{CoVar}[a(\tau), a(\xi)]$ of the process $a(\tau)$ obtained from the same set of simulations, where the covariance is normalized by $\mathrm{Var}[a(\xi)]$ so that the maximum value is equal to one.

evidence, the survival probability of the $a(\tau)$ process during the critical phase follows a scaling law of the same form as the one proposed in [44], and thus the block error probability P_{Block} can be estimated as

$$P_{\text{Block}} \approx 1 - \exp\left(- \frac{\epsilon L}{\frac{2\pi}{\theta} \int_0^{\alpha\sqrt{M}(\epsilon^* - \epsilon)} \Phi(z) \mathrm{e}^{\frac{1}{2}z^2} \mathrm{d}z} \right), \quad (37)$$

where $\Phi(z)$ is the c.d.f. of the standard Gaussian distribution, $\mathcal{N}(0, 1)$, ϵL is the length of the critical phase, and, as in the uncoupled case, $\alpha\sqrt{M}(\epsilon^* - \epsilon)$ corresponds to the ratio of the expected number of decodable constraint nodes during the critical phase to the standard deviation of $a(\tau)$. Both θ and α are parameters that depend on the underlying GLDPC block code and the edge spreading. Given the results in Fig. 14 and Fig. 15, we estimate that $\alpha \approx 5.66$.

Fig. 16 shows a comparison between the simulated performance (solid lines) and estimated error probability using (37) (dashed lines) for the terminated A_7 SC-GLDPC code ensemble with $L = 50$ and $(7, 4)$ Hamming constraint codes decoded with ML decoding. We again note that, as the lifting factor M increases, the performance estimate becomes very accurate.[17] We also show the corresponding results for the uncoupled $(2, 7)$-regular GLDPC block code ensemble of Fig. 1 with ML-decoded $(7, 4)$ Hamming constraint codes and comparable lifting factors. Besides the advantage enjoyed by the SC-GLDPC codes in decoding threshold, we see that they also exhibit better finite-length scaling behavior than GLDPC block codes, in the sense that their performance converges more quickly to the threshold. □

[17] An improved scaling law for terminated SC-LDPC codes over the BEC was recently presented in [45], where the decoding process is modeled as two independent Ornstein-Uhlenbeck processes. Such an approach should also improve the performance estimate for terminated SC-GLDPC codes.

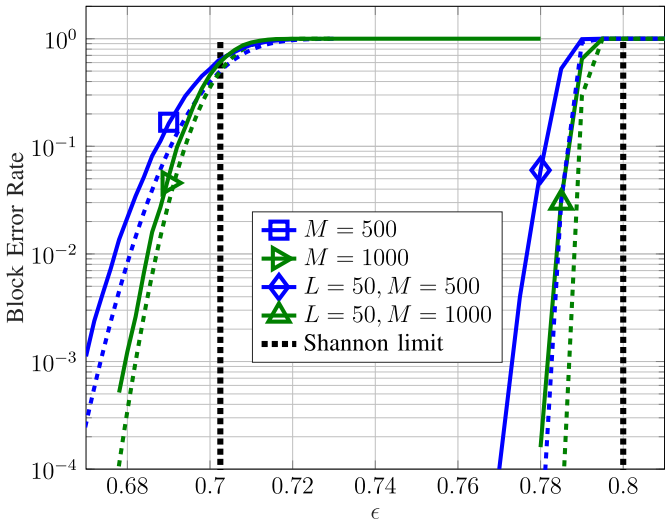

Fig. 16. Simulated performance (solid lines) and estimated performance (dashed lines) for the $(2, 7)$-regular GLDPC block code and the terminated $(2, 7)$-regular A_7 SC-GLDPC code ensembles with $M = 500$ and $M = 1000$.

VI. Concluding Remarks

Generalized LDPC (GLDPC) codes can offer significant performance improvements when compared to LDPC codes with SPC constraints at the expense of an increase in decoding complexity (depending on the particular constraint codes and decoders chosen), albeit with the advantage of a typically smaller number of message passing iterations. In this paper, we presented a comprehensive study of spatially coupled generalized LDPC (SC-GLDPC) codes, including both asymptotic

and finite-length analyses. Specifically, terminated SC-GLDPC code ensembles were shown to numerically achieve threshold saturation with near-capacity iterative decoding thresholds, thus assuring SC-GLDPC codes of having better waterfall performance than their underlying GLDPC block codes. They were also shown to be asymptotically good and to possess large minimum distance growth rates, thus assuring them of also having excellent error floor performance. Finally, terminated SC-GLDPC codes were shown to outperform their GLDPC block code counterparts in the finite length regime. Based on these results, we believe SC-GLDPC codes are an attractive choice for applications requiring excellent performance throughout the entire range of decoded error rates with a limited number of decoding iterations.

REFERENCES

[1] R. G. Gallager, "Low-density parity-check codes," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, MA, USA, 1963.

[2] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 585–598, Feb. 2001.

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

[4] M. Lentmaier and K. S. Zigangirov, "On generalized low-density parity-check codes based on Hamming component codes," *IEEE Commun. Lett.*, vol. 3, no. 8, pp. 248–250, Aug. 1999.

[5] G. Yue, L. Ping, and X. Wang, "Generalized low-density parity-check codes based on Hadamard constraints," *IEEE Trans. Inf. Theory*, vol. 53, no. 3, pp. 1058–1079, Mar. 2007.

[6] J. J. Boutros, O. Pothier, and G. Zémor, "Generalized low density Tanner codes," in *Proc. IEEE Int. Conf. Commun.*, Vancouver, BC, Canada, Jun. 1999, pp. 441–445.

[7] G. Liva, W. E. Ryan, and M. Chiani, "Quasi-cyclic generalized LDPC codes with low error floors," *IEEE Trans. Commun.*, vol. 56, no. 1, pp. 49–57, Jan. 2008.

[8] I. P. Mulholland, E. Paolini, and M. F. Flanagan, "Design of LDPC code ensembles with fast convergence properties," in *Proc. IEEE Int. Black Sea Conf. Commun. Netw. (BlackSeaCom)*, May 2015, pp. 53–57.

[9] A. J. Felström and K. S. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 2181–2191, Sep. 1999.

[10] M. Lentmaier, A. Sridharan, D. J. Costello, and K. S. Zigangirov, "Iterative decoding threshold analysis for LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 5274–5289, Oct. 2010.

[11] S. Kudekar, T. J. Richardson, and R. L. 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.

[12] S. Kudekar, T. Richardson, and R. L. Urbanke, "Spatially coupled ensembles universally achieve capacity under belief propagation," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 7761–7813, Dec. 2013.

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

[14] W. Zhang, M. Lentmaier, K. S. Zigangirov, and D. J. Costello, "Braided convolutional codes: A new class of turbo-like codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 316–331, Jan. 2010.

[15] 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. 1, 2012.

[16] 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 Global Commun. Conf. (GLOBECOM)*, Atlanta, GA, USA, Dec. 2013, pp. 2376–2381.

[17] M. Lentmaier and G. P. Fettweis, "On the thresholds of generalized LDPC convolutional codes based on protographs," in *Proc. IEEE Int. Symp. Inf. Theory*, Austin, TX, USA, Jun. 2010, pp. 709–713.

[18] Y.-Y. Jian, "On the analysis of spatially-coupled GLDPC codes and the weighted min-sum algorithm," Ph.D. dissertation, Texas A&M Univ., College Station, TX, USA, 2013.

[19] D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, "On the minimum distance of generalized spatially coupled LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Istanbul, Turkey, Jul. 2013, pp. 1874–1878.

[20] P. M. Olmos, D. G. M. Mitchell, and D. J. Costello, "Analyzing the finite-length performance of generalized LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 2683–2687.

[21] Y.-Y. Jian, H. D. Pfister, and K. R. Narayanan, "Approaching capacity at high rates with iterative hard-decision decoding," *IEEE Trans. Inf. Theory*, vol. 63, no. 9, pp. 5752–5773, Sep. 2017.

[22] C. Hager, H. D. Pfister, A. G. I. Amat, and F. Brannstrom, "Density evolution for deterministic generalized product codes on the binary erasure channel at high rates," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4357–4378, Jul. 2017.

[23] L. M. Zhang, D. Truhachev, and F. R. Kschischang, "Spatially coupled split-component codes with iterative algebraic decoding," *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 205–224, Jan. 2018.

[24] D. J. Costello, D. G. M. Mitchell, P. M. Olmos, and M. Lentmaier, "Spatially coupled generalized LDPC codes: Introduction and overview," in *Proc. IEEE 10th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Hong Kong, Dec. 2018, pp. 1–6.

[25] A. D. Yardi, I. Andriyanova, and C. Poulliat, "EBP-GEXIT charts over the binary-input AWGN channel for generalized and doubly-generalized LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 496–500.

[26] Y. Wang and M. Fossorier, "Doubly generalized LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2006, pp. 669–673.

[27] J. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," Jet Propulsion Lab., Pasadena, CA, USA, INP Prog. Rep. 42-154, Aug. 2003.

[28] D. G. M. Mitchell, M. Lentmaier, and D. J. Costello, "Spatially coupled LDPC codes constructed from protographs," *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 4866–4889, Sep. 2015.

[29] M. Lentmaier, B. Nöthen, and G. P. Fettweis, "Density evolution analysis of protograph-based braided block codes on the erasure channel," in *Proc. Int. ITG Conf. Source Channel Coding*, Siegen, Germany, Jan. 2010, pp. 1–6.

[30] G. Solomon and H. C. A. Tilborg, "A connection between block and convolutional codes," *SIAM J. Appl. Math.*, vol. 37, no. 2, pp. 358–369, Oct. 1979.

[31] H. Ma and J. Wolf, "On tail biting convolutional codes," *IEEE Trans. Commun.*, vol. COM-34, no. 2, pp. 104–111, Feb. 1986.

[32] M. Lentmaier, M. B. S. Tavares, and G. P. Fettweis, "Exact erasure channel density evolution for protograph-based generalized LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Seoul, South Korea, Jun. 2009, pp. 566–570.

[33] C. Méasson, A. Montanari, and R. L. Urbanke, "Maxwell construction: The hidden bridge between iterative and maximum a posteriori decoding," *IEEE Trans. Inf. Theory*, vol. 54, no. 12, pp. 5277–5307, Dec. 2008.

[34] A. Ashikhmin, G. Kramer, and S. T. Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2657–2673, Nov. 2004.

[35] E. Paolini, M. Varrella, M. Chiani, B. Matuz, and G. Liva, "Low-complexity LDPC codes with near-optimum performance over the BEC," in *Proc. 4th Adv. Satell. Mobile Syst.*, Aug. 2008, pp. 274–282.

[36] A. R. Iyengar, M. Papaleo, P. H. Siegel, J. K. Wolf, A. Vanelli-Coralli, and G. E. Corazza, "Windowed decoding of protograph-based LDPC convolutional codes over erasure channels," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2303–2320, Apr. 2012.

[37] S. Abu-Surra, D. Divsalar, and W. E. Ryan, "Enumerators for protograph-based ensembles of LDPC and generalized LDPC codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 858–886, Feb. 2011.

[38] D. G. M. Mitchell, A. E. Pusane, and D. J. Costello, "Minimum distance and trapping set analysis of protograph-based LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 1, pp. 254–281, Jan. 2013.

[39] D. Truhachev, K. S. Zigangirov, and D. J. Costello, "Distance bounds for periodically time-varying and tail-biting LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4301–4308, Sep. 2010.

[40] A. Amraoui, A. Montanari, T. Richardson, and R. Urbanke, "Finite-length scaling for iteratively decoded LDPC ensembles," *IEEE Trans. Inf. Theory*, vol. 55, no. 2, pp. 473–498, Feb. 2009.

[41] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 569–584, Feb. 2001.

[42] N. C. Wormald, "Differential equations for random processes and random graphs," *Ann. Appl. Probab.*, vol. 5, no. 4, pp. 1217–1235, Nov. 1995.

[43] Y. Liu, P. M. Olmos, and T. Koch, "A probabilistic peeling decoder to efficiently analyze generalized LDPC codes over the BEC," *IEEE Trans. Inf. Theory*, vol. 65, no. 8, pp. 4831–4853, Aug. 2019.
[44] P. M. Olmos and R. L. Urbanke, "A scaling law to predict the finite-length performance of spatially-coupled LDPC codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 6, pp. 3164–3184, Jun. 2015.
[45] R. Sokolovskii, F. Brännström, and A. I. G. Amat, "A refined scaling law for spatially coupled LDPC codes over the binary erasure channel," in *Proc. IEEE Inf. Theory Workshop*, Visby, Sweden, Aug. 2019, pp. 1–5.

David G. M. Mitchell (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from The University of Edinburgh, U.K., in 2009. From 2009 to 2015, he was a Post-Doctoral Research Associate and a Visiting Assistant Professor with the Department of Electrical Engineering, University of Notre Dame, USA. Since 2015, he has been an Assistant Professor with the Klipsch School of Electrical and Computer Engineering, New Mexico State University, USA. His research interests lie in the area of digital communications, with emphasis on error control coding and information theory. He currently serves as an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY.

Pablo M. Olmos was born in Granada, Spain, in 1984. He received the B.Sc., M.Sc., and Ph.D. degrees in telecommunication engineering from the University of Sevilla, in 2008 and 2011, respectively. He has held appointments as a Visiting Researcher with Princeton University, École Polytechnique Fédérale de Lausanne (EPFL), Notre Dame University, École Nationale Supérieure de l'Électronique et de ses Applications (ENSEA), and Nokia-Bell Labs. He is currently an Associate Professor with University Carlos III of Madrid. His research interests include approximate inference methods for Bayesian machine learning to information theory and digital communications. A detailed CV and list of publications can be accessed at http://www.tsc.uc3m.es/~olmos.

Michael Lentmaier (Senior Member, IEEE) received the Dipl.-Ing. degree in electrical engineering from the University of Ulm, Germany, in 1998, and the Ph.D. degree in telecommunication theory from Lund University, Sweden, in 2003. He then worked as a Post-Doctoral Research Associate with the University of Notre Dame, USA, and the University of Ulm. From 2005 to 2007, he was with the German Aerospace Center (DLR), Institute of Communications and Navigation, Oberpfaffenhofen, where he worked on signal processing techniques in satellite navigation receivers. From 2008 to 2012, he was a Senior Researcher and a Lecturer with the Vodafone Chair Mobile Communications Systems, TU Dresden, where he was heading the Algorithms and Coding Research Group. He is currently an Associate Professor with the Department of Electrical and Information Technology, Lund University, which he joined in January 2013. His research interests include design and analysis of coding systems, graph-based iterative algorithms, and Bayesian methods applied to decoding, detection, and estimation in communication systems. He was awarded the Communications Society and Information Theory Society Joint Paper Award in 2012 for his paper "Iterative Decoding Threshold Analysis for LDPC Convolutional Codes." He served as an Editor for IEEE COMMUNICATIONS LETTERS from 2010 to 2013, IEEE TRANSACTIONS ON COMMUNICATIONS from 2014 to 2017, and IEEE TRANSACTIONS ON INFORMATION THEORY from 2017 to 2020.

Daniel J. Costello, Jr. (Life Fellow, IEEE) received the B.S.E.E. degree from Seattle University, Seattle, WA, USA, in 1964, and the M.S. and Ph.D. degrees in electrical engineering from the University of Notre Dame, Notre Dame, IN, USA, in 1966 and 1969, respectively.

He joined the Illinois Institute of Technology as a Faculty Member in 1969. In 1985, he became a Professor of electrical engineering at the University of Notre Dame and from 1989 to 1998, he served as the Department Chair. He has more than 400 technical publications in his field, and in 1983 he co-authored a popular textbook entitled *Error Control Coding: Fundamentals and Applications*, the 2nd edition of which was published in 2004. His research interest includes digital communications, with special emphasis on error control coding and information theory.

Dr. Costello served 18 years as a member of the IEEE Information Theory Society Board of Governors. In 1986, he was the President of the IT Society BOG. He received the Humboldt Research Prize from the German Government in 1999, the Third-Millennium Medal from the IT Society in 2000, the Aaron D. Wyner Distinguished Service Award from the IT Society in 2013, and the IEEE Leon K. Kirchmayer Graduate Teaching Award in 2015. In 2000, he was named the Leonard Bettex Professor of electrical engineering at Notre Dame. He was a co-recipient of the Donald G. Fink Prize Paper Award in 2009 and the joint IT Society/COM Society Prize Paper Award in 2012.