A Sub-Graph Expansion-Contraction Method for Error Floor Computation

Nithin Raveendran, *Graduate Student Member, IEEE*, David Declercq, *Senior Member, IEEE*, and Bane Vasić, *Fellow, IEEE*

***Abstract*— In this paper, we present a computationally efficient method for estimating error floors of low-density parity-check (LDPC) codes over the binary symmetric channel (BSC) without any prior knowledge of its trapping sets (TSs). Given the Tanner graph G of a code, and the decoding algorithm $\mathcal{D}$, the method starts from a list of short cycles in G, and expands each cycle by including its sufficiently large neighborhood in G. Variable nodes of the expanded sub-graphs $\mathcal{G}_{\text{EXP}}$ are then corrupted exhaustively by all possible error patterns, and decoded by $\mathcal{D}$ operating on $\mathcal{G}_{\text{EXP}}$. Union of support of the error patterns for which $\mathcal{D}$ fails on each $\mathcal{G}_{\text{EXP}}$ defines a subset of variable nodes that is a TS. The knowledge of the minimal error patterns and their strengths in each TSs is used to compute an estimation of the frame error rate. This estimation represents the contribution of error events localized on TSs, and therefore serves as an accurate estimation of the error floor performance of $\mathcal{D}$ at low BSC cross-over probabilities. We also discuss trade-offs between accuracy and computational complexity. Our analysis shows that in some cases the proposed method provides a million-fold improvement in computational complexity over standard Monte-Carlo simulation.**

***Index Terms*— Iterative decoding, LDPC codes, Trapping set, Iterative decoding failures, Error floor computation.**

I. Introduction

CHARACTERIZING the error performance of low-density parity check (LDPC) codes [2], [3] under iterative decoding algorithms has been of active research focus in the error correction coding community for almost twenty years (see [4] for a survey). In the asymptotic limit of the code length, the error performance over various channels is well studied, and methods for its characterization exists for a large class of iterative decoders. For example, belief propagation (BP) and similar message passing algorithms are analyzed using density evolution [3], [5], bounds on error correction capability of bit flipping (BF) and message passing algorithms under adversarial model can be obtained using expander-based arguments [6], [7], and the performance of linear programming (LP) decoder [8], [9] depends on the low-weight pseudo-codewords. It is also known that when used on finite length LDPC codes, iterative decoding algorithms fail on specific sub-graphs of a Tanner graph, generically known as trapping sets (TSs) [10], [11]. Trapping sets give rise to an *error floor* (EF) which is seen a degradation in the error performance of the decoder at low channel noise levels. The experimental evidence of error floors of LDPC decoders was shown in a pioneering work [10], wherein importance sampling method was used to estimate EF attributed to TSs. Since then, numerous importance sampling variants have been proposed for estimating error floors of LDPC codes for additive white Gaussian noise (AWGN) channel (see [12] for a detailed survey). A notable analytical method in [13] is based on linear state-space model of decoder dynamics on elementary trapping sets. A prior knowledge of all harmful TSs is key to the correctness of these importance sampling methods, which depends on both the LDPC code as well as the iterative decoder used. However, well designed long codes (e.g., [14], [15]) decoded by powerful iterative decoders (e.g., [16]) have error floors that occurs at very low Frame-Error-Rates (FERs), and are not reachable by Monte-Carlo (MC) simulations or importance sampling techniques that require all harmful configurations a-priori. Hence, analytical or semi-analytical methods without prior assumptions of harmful TSs are necessary to estimate the FER in the EF regime correctly. This is crucial for practical high-throughput, power-efficient applications such as optical communication and flash memory data storage, where the requirements on data reliability are often under $\text{FER} = 10^{-12}$.

The decoders used for these applications are low-precision message passing decoders, and their channel is often modeled as the *binary symmetric channel* (BSC), which is the primary focus in this paper. To characterize error floors of LDPC codes decoded by hard-decision decoding algorithms over the BSC, the method in [17] attempts direct enumeration of all smallest-weight uncorrectable error patterns after performing iterative decoding using the entire Tanner graph, followed by an estimate identifying contribution of error patterns of larger weights to the FER. Although this approach does not assume any harmful configurations a-priori unlike methods discussed in [10], [18], [19], the computational complexity is not amenable with long codes combined with better decoders. In [20], the complexity is significantly reduced by enumerating

Manuscript received November 19, 2019; revised February 29, 2020; accepted April 9, 2020. Date of publication April 20, 2020; date of current version July 15, 2020. D. Declercq and N. Raveendran are supported in part by the NSF under SBIR Phase II Grant 1534760. B. Vasić acknowledges the support of the NSF under grants SaTC- 1813401 and CCF-1855879. This article was presented at the 2020 IEEE Information Theory and Applications (ITA) Workshop. The associate editor coordinating the review of this article and approving it for publication was Q. Huang. *(Corresponding author: Nithin Raveendran.)*

Nithin Raveendran is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 USA (e-mail: nithin@email.arizona.edu).

David Declercq is with Codelucida. Inc., Tucson, AZ 85701 USA (e-mail: declercq@codelucida.com).

Bane Vasić is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 USA, and also with Codelucida, Inc., Tucson, AZ 85701 USA (e-mail: vasic@ece.arizona.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCOMM.2020.2988676

and decoding short cycles only, relying on the assumption that low-weight error patterns are subsets of short cycles for hard-decision iterative decoders. However, extension of this method to (quantized) soft-decision decoders over AWGN channel [21] is not easily generalizable and particularly not effective for long LDPC codes with variable degree greater than 3. In comparison to these approaches, our generic method computes EFs of LDPC codes without any prior assumption of harmful TSs with significantly lower computational complexity compared to MC simulation and direct enumeration techniques and is applicable for arbitrary iterative decoding algorithms.

Analytical estimation of EF exists for simpler algorithms such as the binary message passing (Gallager-B decoder) and bit-flipping algorithms, as well as regular LDPC codes of variable nodes degree equal to three for which there exists a theoretical characterization of uncorrectable error patterns and thus EF [22]. However, for stronger decoders such as min-sum algorithm or finite alphabet iterative decoding (FAID) algorithms [16], and codes with variable degree equal to four and irregular variable degrees, estimating EF by importance sampling is not the best approach. It is because the importance sampling method [10] to estimate FER relies on the existence of a database of trapping sets for the decoding algorithm of interest. Such an enumeration of all possible trapping sets as combinatorial objects may not a feasible task, for example, in the case of codes with variable degree equal to four and higher. Even if it was feasible, running a decoding algorithm (a step in importance sampling) for random error patterns corrupting all trapping sets on entire Tanner graph is computationally prohibitive for long codes. On the other hand, analyzing decoding errors based on an isolated trapping set, while accurate for Gallager B and bit flipping algorithms, is inaccurate for stronger decoders as it ignores the effect of messages from the trapping set neighborhood passed towards the trapping set. As we will show, the messages coming from outside the TS, called external messages, play an important role in the characterization of the TS harmfulness [1]. Indeed, the values of the external messages strongly depend on the location of the TS in the Tanner graph. The exact location which is also critical for the accuracy of the EF prediction is correctly captured in our method described next.

In this paper, we introduce a general procedure for TS characterization, which starts with a list of cycles of length g and $g+2$, where g is the girth of the Tanner graph G, together with their locations in the Tanner graph, and then finds their sufficiently large neighborhoods to ensure that the messages in the interior of such sub-graph accurately represent actual messages in a decoding algorithm operating on an entire graph. An *exhaustive decoding* of error patterns on this *expanded* sub-graph results in a list of all error patterns up to some weight, located on variable nodes of this sub-graph, which lead to a decoding failure. The union of the support of these error patterns gives the location of the variable nodes in the Tanner graph G that is exactly a trapping set of the decoder. The size of the induced graph of the TS, referred to as the *contracted* sub-graph, compared to that of the expanded sub-graph is a measure of a message approximation accuracy and thus the accuracy of contribution of this trapping set to the overall FER. We refer to this procedure as an *expansion-contraction*, and by repeating it on each and every short cycle in the initial list allows us to find contributions of all error patterns up to given weight to the FER in the EF. Computational saving from *expansion-contraction* procedure compared to FER estimation using MC simulation as well as enumeration techniques [20] which run on the entire graph G is huge and comes from the fact that the expanded sub-graph has much less nodes and edges than G. Also, the contraction step is generic and applicable to any iterative decoding algorithm as long as it can be described by local message update rules. Moreover, unlike importance sampling approaches, in the contraction step, we perform exhaustive decoding of error patterns on the expanded sub-graph to find harmful error patterns and obtain a trapping set's true critical number and multiplicity.

Also, note that unlike previous approaches, our method does not make any a-priori assumptions about what graph topologies makes a trapping set dominant or harmful. On the contrary, these topologies are identified by the procedure. Moreover, some of these topologies may be harmful only in a given neighborhood, while some of them are universally harmful, i.e., harmful irrespective of their location in the Tanner graph of a code. As an important side benefit, the expansion-contraction procedure produces a list of dominant trapping sets, i.e., trapping sets that are most harmful to a decoder in the EF.

The remainder of the paper is organized as follows. In Section II, the preliminaries on LDPC codes, followed by basic notations and definitions are discussed. In Section III, we develop the proposed expansion-contraction procedure. We semi-analytically estimate the FER in EF regime for different decoders making use of the expansion-contraction procedure, and compare with Monte-Carlo simulations and direct enumeration of lowest-weight error patterns in Section IV. In Section V, we conclude the paper along with future research directions.

II. Preliminaries

Consider an LDPC code with code rate R defined by a sparse parity check matrix H of dimension $M \times N$, $N \geq M$. The parity check matrix can be represented graphically by a bipartite graph called Tanner graph $G = (V \cup C, E)$, where $V = \{v_1, \ldots, v_N\}$ is the set of N variable nodes (VNs) corresponding to the N columns of H, $C = \{c_1, \ldots, c_M\}$ is the set of M check nodes (CNs) corresponding to the M rows of H, and E is the set of edges: $\{(v_j, c_i) : v_j \in V,\ c_i \in C\}$. A VN $v_j, 1 \leq j \leq N$ is connected to a CN $c_i, 1 \leq i \leq M$ by an edge in the Tanner graph if the entry $H(i,j) = 1$ in the corresponding parity check matrix. The nodes v_j and c_i are then referred to as neighbors. Let us denote the set of CNs connected to a VN v_j by $\mathcal{N}(v_j)$, and $|\mathcal{N}(v_j)|$, where $|\cdot|$ denotes cardinality, is referred to as the degree of VN v_j. Similarly, we can define the neighbor set and the degree of a CN c_i as $\mathcal{N}(c_i)$ and $|\mathcal{N}(c_i)|$, respectively. For a subset of variable nodes, say $K \subseteq V$, $\mathcal{N}(K)$ denotes the set of CN neighbors. The induced sub-graph $\mathcal{G}(K)$ is the

graph containing the nodes $K \cup \mathcal{N}(K)$ along with the edges $\{(x, y) \in E : x \in K, y \in \mathcal{N}(K)\}$. For a regular LDPC code, $|\mathcal{N}(v_j)| = d_v$ and $|\mathcal{N}(c_i)| = d_c$, $\forall i, j$. The girth, g, of the Tanner graph G is the length of the shortest cycle in G. If G has χ_g, $\chi_{g+2}, \ldots$ cycles of length $g, g+2, \ldots$, then the cycle enumerator series $\mathrm{CYC}(x) = \sum_{r \geq 0} \chi_r x^r$ defines the cycle profile of G. We use the cycle profile and location of short cycles to determine the initial list of expansion in the expansion-contraction procedure.

Let $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ denote a codeword of length N such that $H\mathbf{x}^T = \mathbf{0}$. When $\mathbf{x}$ is transmitted over the BSC, an error vector $\mathbf{e} = (e_1, e_2, \ldots, e_N)$ is superimposed to the codeword, and the received word $\mathbf{r} = (r_1, r_2, \ldots, r_N)$ is $\mathbf{x} \oplus \mathbf{e}$, where $\oplus$ is component-wise XOR. Each bit $x_j \in \mathbf{x}$ is flipped with a probability α, called the BSC cross-over probability. For linear codes transmitted over output symmetric channel (such as the BSC) and decoded using symmetric decoding algorithms [5], we typically assume, without loss of generality, that the all-zero-codeword is transmitted. An iterative message passing decoder $\mathcal{D}$ attempts to recover the transmitted codeword by passing messages over the edges of the Tanner graph iteratively and in practice, the maximum number of iterations is pre-determined, denoted by ℓ_{max}. The output of the decoder at ℓ-th iteration is denoted by $\hat{\mathbf{x}}^\ell = (\hat{x}_1^\ell, \hat{x}_2^\ell, \ldots, \hat{x}_N^\ell)$, wherein the superscript indicates the iteration. The decoder is said to converge correctly to the codeword if $\hat{\mathbf{x}}^\ell = \mathbf{x}$ for any $\ell \leq \ell_{max}$ and fail to converge correctly otherwise.[1] A VN v_j is eventually correct if there exists a positive integer l such that for all iterations $\ell \geq l$, $\hat{x}_j^\ell = x_j$. Then, trapping sets are defined as follows:

Definition 1 ([4]): A trapping set $\mathcal{T}$ for an iterative decoder $\mathcal{D}$ is a non-empty set of variable nodes in a Tanner graph G that are not eventually correct. If the sub-graph $\mathcal{G}(\mathcal{T})$ induced by such a set of variable nodes has a VNs and b odd degree CNs, then the trapping set $\mathcal{T}$ is conventionally labeled as an (a, b) trapping set.

When the b odd degree CNs have all degree-1 in $\mathcal{G}(\mathcal{T})$, and the remaining even degree CNs all have degree-2, then the TS is called elementary. For an elementary TS, if each and every VN has more even degree CNs than odd degree CNs, it is called an elementary absorbing set. Note that Definition 1 rely on the iterative decoder $\mathcal{D}$ and the Tanner graph G, and does not specify which property of a set of variable nodes results in a decoding failure.

A. FER Estimation

Suppose a decoder $\mathcal{D}$ is able to correct all error patterns of Hamming weight t for a code of length N. Then

$$\mathrm{FER} = \sum_{i=t+1}^{N} n_i \alpha^i (1-\alpha)^{N-i}, \tag{1}$$

where n_i represents number of error patterns of Hamming weight i that are uncorrectable, and α is the BSC cross-over probability. Obtaining values for all n_i in Eq. (1) is computationally infeasible for large N. However, to estimate the EF part of the FER curve, we need to find all the uncorrectable error patterns only up to some weight. Hence, determining the weight and the number of smallest uncorrectable error patterns are the key steps for estimating the EF. This observation brings us to the definition of critical number μ and strength s of a trapping set.

Definition 2: Critical number μ of a trapping set $\mathcal{T}$ is the minimal number of variable nodes that have to be initially in error for the decoder to fail to converge.

Definition 3: Failure inducing set is a set of variable nodes that have to be initially in error for the decoder to fail to converge.

Definition 4: Strength s of a trapping set $\mathcal{T}$ is the number of failure inducing sets of cardinality μ.

In other words, the cardinality of the minimal failure inducing set for a TS gives its critical number and the number of weight-μ error patterns gives its strength. If $\Gamma_{\mu,\mathrm{s}}$ is the number of all TSs with the critical number μ and strength s across all trapping sets, then for low values of α,

$$\mathrm{FER} = \sum_{\mu} \sum_{\mathrm{s}} \mathrm{s} \Gamma_{\mu,\mathrm{s}} \alpha^\mu (1-\alpha)^{(N-\mu)}. \tag{2}$$

The behavior of the FER curve for low values of α [4] is dominated by $\log(\mathrm{FER}) \approx \mu_{\min} \log(\alpha) + \log\left(\sum_{\mathrm{s}} \Gamma_{\mu_{\min},\mathrm{s}}\right)$. The $\log(\mathrm{FER})$ vs $\log(\alpha)$ graph is close to a straight line with slope equal to the minimum critical number, $\mu_{\min}$. Eq. (2) shows how the critical number μ and strength s of TSs determine the FER in EF regime. We refer to the TSs with the smallest critical number that determines the FER slope as the dominant TSs. Even though Eq. (2) is straightforward to evaluate, finding the exact critical number and strength for all trapping sets, specific to a given decoder and a given Tanner graph, is non trivial.

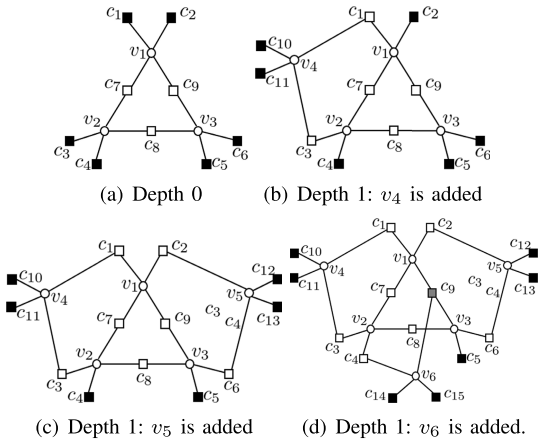
III. Expansion-Contraction Procedure

In this section, we describe our method to identify the dominant trapping sets together with their critical numbers and strengths for any given decoder and Tanner graph. The general theme observed in trapping set research is that dominant trapping sets are composed of short cycles [11]. Therefore, each short cycle and its neighboring cycles is a potential failure inducing subset. Note that not all these subsets would cause convergence failure of a decoder. Decoding failure depends on the sparseness, the neighborhood, and the exact location of the induced sub-graph in the Tanner graph, in addition to the decoding rule itself. Nevertheless, we include all the short cycles of length g and $g+2$ present in the Tanner graph to obtain an initial list. Such short cycles can be located efficiently [20], and from this initial list of cycles, we first *expand* to bigger and denser neighboring sub-graphs present in the Tanner graph. From each of the resulting expanded sub-graphs, we *contract* to a failure inducing sub-graph (if any) and obtain the list of dominant TSs of a decoder.

A. Expansion Procedure

From the initial list of short cycles, say of length $2k$, let us consider the expansion from a single cycle. We label the

[1] We mean by converge, to 'correctly' converge to the transmitted codeword, throughout this paper.



(a) Depth 0 (b) Depth 1: v_4 is added

(c) Depth 1: v_5 is added (d) Depth 1: v_6 is added.

Fig. 1. An illustration of expansion steps from an initial short cycle shown in Fig. 1(a) to depth 1 sub-graph as shown in Fig. 1(d). In Fig. 1(a), we have $\mathcal{V}^{(0)} = \{v_1, v_2, v_3\}$ and $\mathcal{C}_1^{(0)} = \{c_1, \ldots c_6\}$. From the neighbors of c_1, we selected v_4 for expansion as it has another CN neighbor $c_3 \in \mathcal{C}^{(0)}$ as shown in Fig. 1(b). Similarly, we added v_5 in Fig. 1(c) and v_6 in Fig. 1(d) in $\mathcal{V}^{(1)}$. The set of VNs $\mathcal{V}^{(1)} = \{v_1, v_2, \ldots, v_6\}$ and the induced graph $\mathcal{G}(\mathcal{V}^{(1)})$ as shown in Fig. 1(d) is obtained when there are no more variable nodes that satisfies the condition for expansion in depth 1.

k variable nodes present in the cycle as $v_1, \ldots, v_k$ with slight abuse of notation from Section II. Denote this initial set of variable nodes $\{v_1, \ldots, v_k\}$ as $\mathcal{V}^{(0)}$ and its induced sub-graph as $\mathcal{G}(\mathcal{V}^{(0)})$. An example of a cycle of length six is shown in Fig. 1(a). An external VN with respect to $\mathcal{V}^{(0)}$ refers to any VN in V excluding the set $\mathcal{V}^{(0)}$, denoted by $v \in V \setminus \mathcal{V}^{(0)}$. Let $\mathcal{C}^{(0)}$, the set of check nodes in $\mathcal{G}(\mathcal{V}^{(0)})$, be partitioned into the disjoint union: $\mathcal{C}^{(0)} = \mathcal{C}_1^{(0)} \cup \mathcal{C}_2^{(0)} \cup \ldots \mathcal{C}_{d_c}^{(0)}$ based on their CN degrees, where $\mathcal{C}_x$ represents the set of degree-x CNs. For example, $\mathcal{C}_1^{(0)}$ is a set of all degree-1 CNs in the induced sub-graph. The superscript zero in the notation denotes the initial list for expansion, and in general shows the depth of expansion in the recursive step as follows.

We begin by initializing $\mathcal{V}^{(1)}$ with the initial set of variable nodes, i.e., $\mathcal{V}^{(1)} = \mathcal{V}^{(0)}$. An external VN w that is a neighbor of a degree-1 CN i.e., $w \in \mathcal{N}(c)$, where $c \in \mathcal{C}_1^{(0)}$, is selected for expansion if w has a CN neighbor in the set $\mathcal{C}^{(0)} \setminus c$. We repeat this selection step for the VN neighbors of all the CNs in $\mathcal{C}_1^{(0)}$. The resulting selected VNs are added into the set $\mathcal{V}^{(1)}$. Note that $\mathcal{V}^{(1)} \supseteq \mathcal{V}^{(0)}$, where equality holds when no new VNs are added in the expansion step. Fig. 1 gives an illustration of expansion from depth 0 to 1. Similar to the expansion step shown in Fig. 1, we recursively expand to depth 2 and higher. After δ recursions, we have the sub-graph $\mathcal{G}(\mathcal{V}^{(\delta)})$, induced by the set $\mathcal{V}^{(\delta)}$ defined as

$$\mathcal{V}^{(\delta)} = W \cup \mathcal{V}^{(\delta-1)}, \tag{3}$$

where W is the set of VNs selected such that every $w \in W$ is a neighbor of $c \in \mathcal{C}_1^{(\delta-1)}$ and $\mathcal{N}(w) \setminus c \in \mathcal{C}^{(\delta-1)}$. In other words, the expansion condition above is met if the intersection of $\mathcal{N}(w) \setminus c$ and $\mathcal{C}^{(\delta-1)}$ is non-empty. Fig. 2 shows the expansion procedure to depth 2 starting from a short cycle. We stop the expansion process if no external VN is added at expansion depth δ i.e., $|\mathcal{V}^{(\delta)}| = |\mathcal{V}^{(\delta-1)}|$, or if the cardinality of the expanded set of VNs $|\mathcal{V}^{(\delta)}|$ exceeds a threshold, κ. The threshold κ is chosen to stop the recursive expansion process,

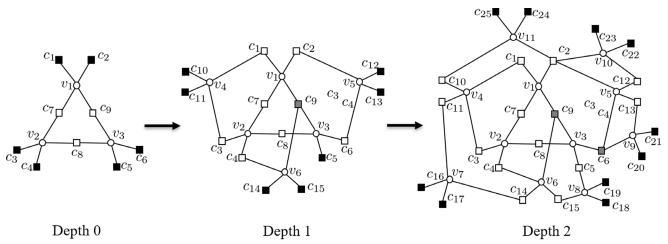


Fig. 2. Expansion procedure steps from a short cycle of length six up to depth 2. The resulting expanded graph is used in the contraction step.

limiting the number of VNs in the expanded sub-graph and thereby, the complexity in the next step of contraction. The size of the resulting expanded graph could be either close to κ or even two or three times larger than κ depending on the neighborhood of the initial cycle. Note that in the expansion step, we do not make any assumptions or restrictions on the topology of a potential TS. In other words, the procedure will give both elementary and non-elementary trapping sets. Expansion procedure starting from all degree-1 CNs is a computationally effective and sufficient strategy, however this is not comprehensive compared to exhaustively expanding from all the CNs in every step of the expansion. We note that since the expansion procedure and the initial list are chosen based on finding the densest sub-graphs, some sparser sub-graph configurations might be missed. For completeness, we will briefly discuss on some of those sub-graphs and on trade-offs between accuracy and computational complexity later in the Section IV-B.

The expansion procedure is formally described in Algorithm 1. After expansion step, the graph $\mathcal{G}(\mathcal{V}^{(\delta)})$ is the input to the contraction step explained in the next subsection whose goal is to find the dominant TSs of the decoder as well as to compute their critical number and strength.

B. Contraction Procedure

As explained in Section III-A, when applied to a given cycle in the initial list, the expansion procedure produces a set of variable nodes $\mathcal{V}_{\text{EXP}} = \mathcal{V}^{(\delta)}$, and its induced graph $\mathcal{G}_{\text{EXP}} = \mathcal{G}(\mathcal{V}_{\text{EXP}})$. Since $\mathcal{G}_{\text{EXP}}$ is by construction composed of many short cycles, $\mathcal{V}_{\text{EXP}}$ is potentially a trapping set for a given decoder. The purpose of the contraction step in a nutshell is to identify failure inducing sets in $\mathcal{V}_{\text{EXP}}$ and to reduce $\mathcal{V}_{\text{EXP}}$ to a TS for a given decoder $\mathcal{D}$. This is achieved by exhaustively decoding error patterns running $\mathcal{D}$ on $\mathcal{G}_{\text{EXP}}$ ensuring that the messages accurately represent the actual messages operating on the entire Tanner graph G. After obtaining a list of all minimum-weight failure inducing error patterns, $\mathcal{V}_{\text{EXP}}$ is contracted to the union of support of such error patterns under the assumption that if a variable node is not in any of the failure inducing sets obtained through exhaustive decoding process, it could be removed from the TS.

Before we explain how this is precisely done, we note that the contraction step is also generic and is applicable to any iterative decoding algorithm as long as it can be described by local message update rules. Consider such an iterative message passing decoder $\mathcal{D}$, defined as a 6-tuple $\mathcal{D} = (\mathcal{M}, \mathcal{Y}, \Phi, \Psi, \hat{\Phi}, \hat{\Psi})$, where $\mathcal{Y}$ and $\mathcal{M}$ are the channel

Algorithm 1 Expansion of Short Cycles to Larger Sub-Graphs

Inputs: Tanner graph G, Expansion threshold κ
Compute cycle profile of G, and obtain list $\mathcal{L}_{\text{INI}}$, where elements of $\mathcal{L}_{\text{INI}}$ are variable node sets inducing all g- and $(g+2)$-cycles in G.
Initialization: $\mathcal{L}_{\text{EXP}} \leftarrow \emptyset$
Repeat
for all elements $\in \mathcal{L}_{in}$ **do**
  Initial set of variable nodes $= \mathcal{V}^{(0)}$.
  Expansion depth $\delta = 0$.
  do
  Initialization: $W \leftarrow \emptyset$
  Induced sub-graph: $\mathcal{G}(\mathcal{V}^{(\delta)})$ and set of check nodes: $\mathcal{C}^{(\delta)}$
  for all $c \in \mathcal{C}_1^{(\delta)}$ **do**
    for all $w \in \mathcal{N}(c)$, $w \notin \mathcal{V}^{(\delta)}$ **do**
      if $\mathcal{N}(w) \setminus c \in \mathcal{C}^{(\delta)}$ **then**
        $W \leftarrow w$
      end if
    end for
  end for
  $\delta = \delta + 1$
  $\mathcal{V}^{(\delta)} = W \cup \mathcal{V}^{(\delta-1)}$
  while $|\mathcal{V}^{(\delta)}| > |\mathcal{V}^{(\delta-1)}|$ and $|\mathcal{V}^{(\delta)}| < \kappa$
  Expanded set of variable nodes $= \mathcal{V}^{(\delta)}$
  $\mathcal{L}_{\text{EXP}} \leftarrow \mathcal{V}^{(\delta)}$
end for
Output: $\mathcal{L}_{\text{EXP}}$ = List of expanded variable node sets in the Tanner graph G.

(output) and message alphabets, respectively, Φ and Ψ are the update functions used in variable and check nodes, respectively, and $\hat{\Phi}$ is the decision function, and $\hat{\Psi}$ is the check estimate function. The function $\Psi : \{\mathcal{M}\}^{d_c-1} \rightarrow \mathcal{M}$ updates the outgoing message η of a CN c with degree d_c computed with $d_c - 1$ extrinsic incoming messages, and the function $\Phi : \mathcal{Y} \times \{\mathcal{M}\}^{d_v-1} \rightarrow \mathcal{M}$ is used for updating outgoing message ν of a VN v with degree d_v. At iteration ℓ, we have $\nu_{v\rightarrow c}^{(\ell)} = \Phi(y_v, \{\eta_{c'\rightarrow v}^{(\ell)}\}^{d_v-1})$ and $\eta_{c\rightarrow v}^{(\ell)} = \Psi(\{\nu_{v'\rightarrow c}^{(\ell-1)}\}^{d_c-1})$, where ν and η represents the VN and CN update messages, respectively. For completeness of our discussion, we describe a generic iterative decoder in Appendix A (See [16] and references within for more details).

We ensure that the messages in the interior of the sub-graph $\mathcal{G}_{\text{EXP}}$ accurately represent actual messages in a decoding algorithm operating on an entire graph by extending $\mathcal{G}_{\text{EXP}}$ to external variable nodes as shown in Fig. 3 and then, carefully modeling the behavior of correct messages coming from these external variable nodes to $\mathcal{C}_1^{\text{EXP}}$, the degree-1 check nodes in $\mathcal{G}_{\text{EXP}}$ as we explain next.

a) Modeling Outside Messages: In trapping set analysis of iterative decoders that propagate binary messages such as Gallager-B and BF, the messages from external variables are modeled as "correct", i.e., they are set to zero in every iteration. For larger cardinalities of message alphabets $\mathcal{M}$, it would be inaccurate to simply saturate all such "outside" messages to $\max(\mathcal{M})$ as the magnitudes of messages sent

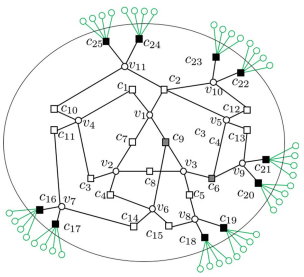


Fig. 3. The extended graph of the depth 2 sub-graph shown in Fig. 1(d), is the sub-graph $\mathcal{G}_{\text{EXP}}$ along with all the external variable nodes connected to the degree-1 check nodes $\mathcal{C}_1^{\text{EXP}}$. The enclosed sub-graph is $\mathcal{G}_{\text{EXP}}$ with ■ representing the degree-1 check nodes. The external variable nodes connected to all the check nodes $c \in \mathcal{C}_1^{\text{EXP}}$ are also shown.

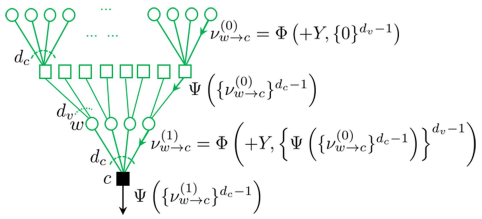


Fig. 4. Figure shows how messages are updated on a computation tree $\mathcal{J}(c)$ towards the degree-1 CN c. All external variable nodes in $\mathcal{J}(c)$ are correct and send message updates according to the decoding rule $\mathcal{D}$. Here we show the messages in the first two iterations.

from external variable nodes towards their neighbors in $\mathcal{C}_1^{\text{EXP}}$ affect the correctability of the error patterns that corrupt the variables in $\mathcal{V}_{\text{EXP}}$.

Note that the check nodes in $\mathcal{G}_{\text{EXP}}$ of degree greater than one also receive correct messages from outside the sub-graph. However, these correct external VN messages of large magnitude will not affect the magnitude of CN update passed within the sub-graph $\mathcal{V}_{\text{EXP}}$. For example, in the standard min-sum decoder on regular LDPC codes, wherein a CN update computes the minimum among extrinsic messages, for a CN of degree two or more in $\mathcal{G}_{\text{EXP}}$, the minimum magnitude among its incoming messages will always be from the erroneous variable nodes within the sub-graph. Hence, such messages do not affect the correctability of the error patterns. Thus, we only focus on modeling the behavior of correct messages coming from external variable nodes to $\mathcal{C}_1^{\text{EXP}}$, the degree-1 check nodes in $\mathcal{G}_{\text{EXP}}$. Fig. 3 shows the extended graph for the example in Fig. 2 with external variable nodes added to all check nodes $c \in \mathcal{C}_1^{\text{EXP}}$. We assume that no two external variable nodes share a common neighbor (isolation assumption beyond the expanded sub-graph [16]). To model the messages passed in decoding iterations, we further assume that each CN $c \in \mathcal{C}_1^{\text{EXP}}$ is a root of an external computation tree $\mathcal{J}(c)$ in which all variable nodes are correct. An example of a computation tree is shown in Fig. 4 along with the message updates.

While the above assumption oversimplifies the graph topology beyond the expanded sub-graph, we are able to model that in the beginning of decoding, lower degree external variable nodes send smaller messages towards $\mathcal{C}_1^{\text{EXP}}$ and thus, contribute less to correctability of the error pattern. To simplify the message analysis, let us assume that in $\mathcal{J}(w)$, all the variable nodes have the same degree $d_v = |\mathcal{N}(w)|$. Since,

all external variable nodes are correct, the messages sent from each VN in the tree towards the CNs at a particular depth from the root of the tree are equal. Also, under the all-zero codeword assumption, the channel value of all external variable nodes is $y_w = +Y$, for every $w \in \mathcal{J}(c)$ and $c \in \mathcal{C}_1^{\text{EXP}}$. The correct VN message passed from $w \in \mathcal{J}(c)$ to a CN $c \in \mathcal{C}_1^{\text{EXP}}$ in ℓ-th iteration satisfy the following recursion:

$$\nu_{w\to c}^{(\ell)} = \Phi\left(+Y, \left\{\Psi\left(\{\nu_{w\to c}^{(\ell-1)}\}^{d_c-1}\right)\right\}^{d_v-1}\right), \qquad (4)$$

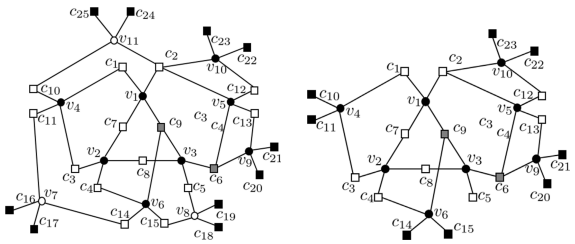
with initialization as $\nu_{w\to c}^{(0)} = \Phi\left(+Y, \{0\}^{d_v-1}\right)$, where $+Y$ is the channel value corresponding to a correct bit and $Y \in \mathcal{Y}$. In Fig. 4, we show how the messages are recursively updated in a computation tree $\mathcal{J}(c)$ directed towards the degree-1 CN c. Based on Eq. (4), we have the following proposition.

Proposition 1: Under the isolation assumption for the external variable nodes connected to a degree-1 CN c, the messages sent to c from a variable $w \in \mathcal{J}(c)$ with degree $|\mathcal{N}(w)| = d_w$, would approximately increase at the rate $d_w - 1$.

Let us denote these messages from external correct variable nodes to c by θ_c, $c \in \mathcal{C}_1^{\text{EXP}}$. The vector of $\boldsymbol{\theta} = (\theta_c)$, $\forall c \in \mathcal{C}_1^{\text{EXP}}$ is referred to as an external message vector, computed according to Eq. (4). $\boldsymbol{\theta}$ models the external messages for the decoder $\mathcal{D}$ in the contraction step to find the trapping set and uncorrectable error patterns as described next.

b) Finding Trapping Sets: As discussed in subsection III-A, since $\mathcal{G}_{\text{EXP}}$ is dense, it is potentially a trapping set for a given decoder $\mathcal{D}$. To determine failure inducing sets located within $\mathcal{V}_{\text{EXP}}$, we corrupt variable nodes in $\mathcal{G}_{\text{EXP}}$ by all possible error patterns, and for each error pattern run the decoding algorithm while assuming that all external variable nodes are not corrupted by errors. We exhaustively attempt decoding of all error patterns starting from weight 1 to $|\mathcal{V}_{\text{EXP}}|$. For every weight $\mu \in [1, |\mathcal{V}_{\text{EXP}}|]$, we have $\binom{|\mathcal{V}_{\text{EXP}}|}{\mu}$ error patterns in the set $\mathbb{E}^{(\mu)}$. If all error patterns up to weight $|\mathcal{V}_{\text{EXP}}|$ are successfully decoded to an all-zero vector, we label that particular sub-graph as harmless for the decoding algorithm. Instead, if any error pattern of weight μ fails to be decoded correctly, we record that error pattern as $\mathbf{e}_1$. We continue the decoding attempt on all of the remaining error patterns in the set $\mathbb{E}^{(\mu)}$ and record those for which the decoder $\mathcal{D}$ fails. Let $\{\mathbf{e}_1, \ldots, \mathbf{e}_{\text{s}}\}$ be the set of uncorrectable error patterns of weight μ. Define $\mathcal{V}_{\text{CONTR}}$ as the union of the support of all error patterns: $\mathcal{V}_{\text{CONTR}}(\mathbf{e}_1, \ldots, \mathbf{e}_{\text{s}}) := \bigcup_{k=1}^{\text{s}} \text{supp}(\mathbf{e}_k)$. $\mathcal{V}_{\text{CONTR}}$ is composed of all variable nodes in $\mathcal{G}_{\text{EXP}}$ which participate in *at least* one decoding failure, i.e., the failure inducing set. As we know from Definition 2 and Definition 4, we have now obtained the critical number and strength of the trapping set $\mathcal{V}_{\text{CONTR}}$ as μ and s, respectively. Let us also define $\mathcal{E}_{\text{CONTR}}$ as a collection of the support sets of all error patterns: $\mathcal{E}_{\text{CONTR}}(\mathbf{e}_1, \ldots, \mathbf{e}_{\text{s}}) := \{\text{supp}(\mathbf{e}_1), \ldots, \text{supp}(\mathbf{e}_{\text{s}})\}$.

Fig. 5 shows an example of contraction step to find the trapping set from the expanded graph. $\mathcal{G}_{\text{EXP}}$ is shown in Fig. 5 (a). Let us suppose that the supports of error patterns are $\text{supp}(\mathbf{e}_1)=\{v_1, v_2, v_3, v_4, v_5\}$, $\text{supp}(\mathbf{e}_2)=\{v_1, v_2, v_3, v_4, v_6\}$, $\text{supp}(\mathbf{e}_3)=\{v_2, v_3, v_4, v_6, v_9\}$, $\text{supp}(\mathbf{e}_4)=\{v_2, v_3, v_4, v_9, v_{10}\}$. Then $\mathcal{V}_{\text{CONTR}}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4) = \{v_1, v_2, v_3, v_4, v_5, v_6, v_9, v_{10}\}$.



(a) ● denotes variables appearing in at least one failure inducing set

(b) Contracted graph - the dominant trapping set

Fig. 5. From the expanded graph shown in Fig. 1(d), we perform the contraction step to find all minimal failure inducing sets. In Fig. 5(a), • denotes variable node that belongs to at least one failure inducing set. Dominant trapping set contains only those variable nodes and its induced sub-graph is the contracted graph shown in Fig. 5(b).

Critical number of $\mathcal{V}_{\text{CONTR}}$ is 5 and strength is 4. $\mathcal{E}_{\text{CONTR}} = \{\text{supp}(\mathbf{e}_1), \ldots, \text{supp}(\mathbf{e}_4)\}$. The graph induced by $\mathcal{V}_{\text{CONTR}}$ is $\mathcal{G}(\mathcal{V}_{\text{CONTR}})$ in the Fig. 5 (b).

We obtain $\mathcal{V}_{\text{CONTR}}$, μ, s, and $\mathcal{E}_{\text{CONTR}}$ after the contraction step from $\mathcal{V}_{\text{EXP}}$ if there exists at least one uncorrectable error pattern of weight μ. The contraction step is now applied to each and every expanded variable node sets in our expanded list $\mathcal{L}_{\text{EXP}}$. From $\mathcal{V}_{\text{CONTR}}$, μ, and s obtained from the contraction steps, we create a list $\mathcal{L}_{\text{TS}}$ which contains all topologically unique harmful TSs present in the code along with their corresponding critical number and strength. We refer to $\mathcal{L}_{\text{TS}}$ as the *trapping set profile* for the given LDPC code and decoder $\mathcal{D}$. The number of TSs present in the list $\mathcal{L}_{\text{TS}}$ is typically smaller than the number of cycles present in $\mathcal{L}_{\text{INI}}$ as some expanded sub-graphs are harmless and also, two different expanded sub-graphs can contract to the same TS. From $\mathcal{E}_{\text{CONTR}}$ obtained from the contraction steps, we have $\mathcal{L}_{\text{ERRORS}}$, a list of unique uncorrectable error patterns of the given LDPC code. Suppose we have n_i uncorrectable error patterns of weight i in the list $\mathcal{L}_{\text{ERRORS}}$. We estimate FER at low values of α using Eq. (1). For an accurate EF estimate, it is important to ensure that the uncorrectable error patterns are not double-counted. Only unique error patterns are included in $\mathcal{L}_{\text{ERRORS}}$ and contribute to FER in Eq. (1). As a simple example, suppose there are two distinct TSs: $\mathcal{V}_{\text{CONTR}}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$ and $\mathcal{V}_{\text{CONTR}}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_4, \mathbf{e}_5)$, where $\mathbf{e}_j, j \in [1, 5]$ is an uncorrectable error pattern of some weight i. Then, we only count $n_i = 5$ for FER calculation in Eq. (1).

As a summary of this section, using the expansion-contraction procedure, we obtained a list of harmful low-weight error patterns, $\mathcal{L}_{\text{ERRORS}}$, and the trapping set profile, $\mathcal{L}_{\text{TS}}$, for a given Tanner graph G and decoder $\mathcal{D}$. The EF FER estimate is computed using Eq. (1). Now, we present simulation results obtained using the procedure in Section IV.

IV. Simulation Results and Discussion

In this section, we present the results of our sub-graph expansion-contraction method for EF computation applied to different codes and decoders, focusing mainly on quasi-cyclic LDPC (QC-LDPC) codes and quantized-message passing iterative decoding algorithms (e.g., quantized min-sum (MS), quantized offset min-sum (OMS)), which are widely used in practical applications. Two QC-LDPC codes of regular

TABLE I

LDPC CODE PARAMETERS

Code	N	M	Z	g	χ_g	χ_{g+2}
QC 1KB	9216	896	32	6	5257 x 32	893535 x 32
QC 2KB	18432	1792	32	6	3093 x 32	899974 x 32

TABLE II

NUMBER OF WEIGHT-k UNCORRECTABLE ERROR PATTERNS FOR DIFFERENT OMS (Y, γ) DECODERS PLOTTED IN FIG. 6

OMS / k	4	5	6	7	8
$10, 1$	19232	15712	12554400	32256	32
$7, 1$	0	23360	169472	7997280	148384
$5, 1$	0	864	20832	979136	1606880

VN degree $d_v = 4$, code rate $R = 0.903$ and circulant size $Z = 32$ of lengths $N = 9216$ (referred as QC 1KB) and $N = 18432$ (referred as QC 2KB) are chosen. Note that both codes have girth $g = 6$ as given in Table I. The number of g- and $(g+2)$-cycles that are used to initialize the expansion-contraction procedure are also given in Table I.

For all the MS and OMS decoders considered for simulation, we select a quantization of 4-bits (preferred over 3-bits and lower which suffer from message saturation and precision issues resulting in a high EF, and over 5-bits and higher to limit the decoder complexity) and use a flooding schedule with maximum number of decoding iterations l_{max} set to 100. More details on the decoder are described in the Appendix A. We choose OMS decoders with different channel values that have good and similar waterfall performances to predict their EF locations.

A. Error Floor FER Results

Consider first, the QC 2KB code. Quasi-cyclic property of the chosen code can be made use of to reduce the complexity of the expansion-contraction procedure considerably. This reduction is proportional to the circulant size, Z by enumerating only the non-isomorphic 6-cycles and 8-cycles present in the QC 2KB code, and performing the expansion-contraction procedure as discussed in Section III. Quantized OMS decoders with different channel values $Y = 5$, $Y = 7$, $Y = 10$, with an offset $\gamma = 1$ are used for the MC simulation and the contraction step. Fig. 6 shows FER curves plotted using MC simulation and the FER estimate in the EF regime computed using Eq. (1). The EF estimate from the expansion-contraction method accurately matches the corresponding MC simulation FER curve of OMS decoders with channel value $Y = 7$ and $Y = 10$ at low values of α. For the OMS decoder with $Y = 5, \gamma = 1$, the EF is estimated to be below 10^{-9} by extrapolating from the MC simulation curve. The OMS decoders chosen for simulation with different Y values, even though they exhibit similar waterfall curves, exhibit very different error floors. Fig. 6 highlights that the expansion-contraction procedure is able to efficiently quantify the difference in error floors of various decoders. This difference in the slope and EF location can be attributed to the weight and number of low-weight dominant error patterns given in Table. II.

High EF estimation of the OMS $Y = 10, \gamma = 1$ decoder results from the 19232 uncorrectable weight-4 error patterns

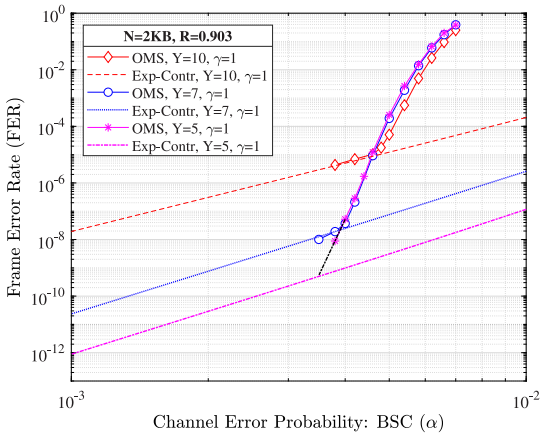


Fig. 6. FER plots comparing Monte-Carlo (MC) simulation and the expansion-contraction method for QC 2KB code with different offset min-sum (OMS) decoders with channel value $Y = 5, 7$ and 10 and offset $\gamma = 1$. In the expansion-contraction method, we expand the 6- and 8-cycles present in the Tanner graph with expansion parameter $\kappa = 10$. The expansion-contraction procedure computes a tight lower bound to the EF curves obtained from MC simulation. For the OMS decoder with $Y = 5, \gamma = 1$, we obtain the EF below the FER of 10^{-9} by extrapolating from the MC simulation curve to the FER estimate.

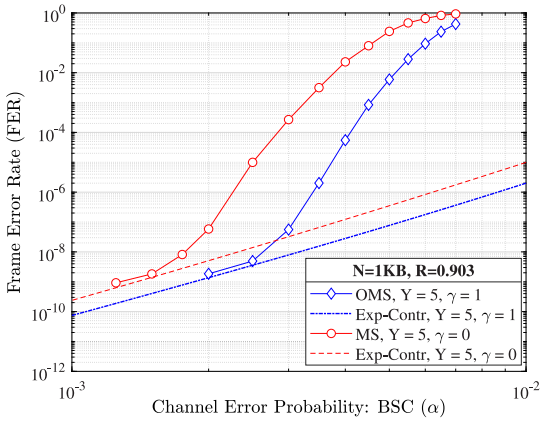
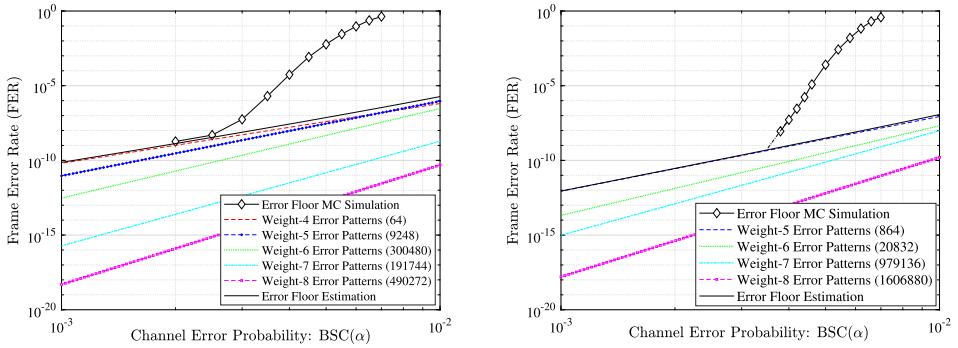


Fig. 7. Figure shows that the FERs estimated for the QC 1KB code, $\kappa = 10$, for the OMS decoder with $Y = 5$, $\gamma = 1$ and the MS decoder with $Y = 5$ closely trace the corresponding MC simulation curves in the EF.

$(\mu_{\min} = 4)$ compared to the OMS decoders $Y = 7, \gamma = 1$ and $Y = 5, \gamma = 1$, which fail only for weight 5- and higher error patterns $(\mu_{\min} = 5)$.

For the QC 1KB code in Table. I, we obtained the error floor FER estimate for an OMS decoder with $Y = 5, \gamma = 1$ and a MS decoder with $Y = 5$. Fig. 7 shows that the estimated FER curves using the expansion-contraction method match the EF curves obtained from MC simulations.

In addition to the FER computation, the procedure can also identify the contribution of error patterns to the FER slope. In Fig. 8, we characterize the contribution of error patterns of different weights to the FER estimate for the OMS decoder with $Y = 5, \gamma = 1$. FER estimates from error patterns obtained from the expansion-contraction procedure starting from 8-cycles in the QC 1KB and QC 2KB code are plotted in Fig. 8 (a) and Fig. 8 (b), respectively. The FER obtained in Eq. (1) is computed by summing the error probabilities over error patterns of different weight. Likewise, each of the "Weight-k Error Patterns" lines in Fig. 8 computes



(a) QC 1KB code, OMS $Y = 5, \gamma = 1$ (b) QC 2KB code, OMS $Y = 5, \gamma = 1$

Fig. 8. Contribution of error patterns of different weights to the error floor FER estimation of OMS decoder $Y = 5, \gamma = 1$. Number of uncorrectable weight-i error patterns (n_i) is given in parenthesis. Identifying low-weight error patterns gives an exact estimation of FER in EF region. (a) The weight-4 error patterns (dashed line) and weight-5 error patterns (dashed line with marker) determine the EF estimation. (b) The weight-5 error patterns contribute the most to the EF estimation.

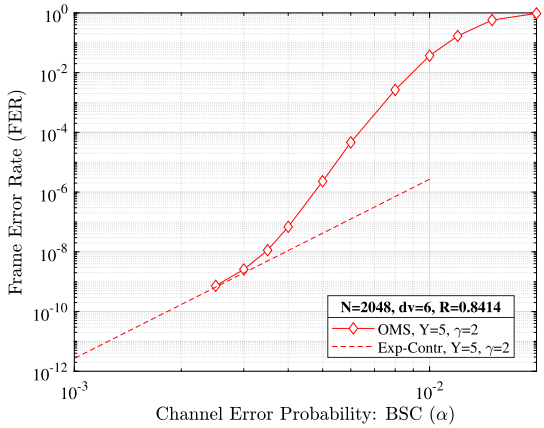


Fig. 9. Figure shows the FER estimated for the IEEE 802.3an code along with the MC simulated curve. The expansion procedure expands from the (8,8) graphical structures present and contracts using the OMS decoder with $Y = 5$, $\gamma = 2$. The EF estimate closely traces the corresponding MC simulation curve.

the FER contribution of weight-k error patterns obtained from the expansion-contraction procedure and we add up all the curves to obtain the FER "Error Floor Estimation" curve. In Fig. 8 (a), the contribution from the weight-4 error patterns (minimum critical number), shown as a dashed line, dominates the error floor FER estimation curve, shown as a solid line, as noted in [4]. Observe that the contribution from the weight-5 error patterns, shown as a dashed line with marker, to the FER is also not negligible as there are 9248 uncorrectable weight-5 error patterns compared to only 64 uncorrectable weight-4 error patterns as given in the Fig. 8 (a) legend. This reinforces the importance of identifying all such error patterns to get an accurate estimation of FER in the EF region.

In Fig. 9, the EF of the OMS $(Y = 5, \gamma = 2)$ decoder is estimated accurately for the well-studied IEEE 802.3an $(2048, 1723)$ regular LDPC code [13], [23] using the expansion-contraction method. This code has a larger column weight, $d_v = 6$ and a shorter code length in comparison with the QC codes considered before. The $(8, 8)$ TS present in this code is well studied in the literature. This knowledge about previously studied/standard LDPC codes can be made use of as the starting structure for the expansion-contraction procedure. The neighborhood of such structures and the decoder is taken into account during the expansion-contraction procedure to give the exact critical number and the multiplicity of the uncorrectable error patterns. The OMS $(Y = 5, \gamma = 2)$ decoder fails to converge for error patterns of weight 6, not just in the $(8, 8)$ TS, but also in its neighborhood. More importantly, the algorithm identifies non-elementary TSs with critical number 6 which affect the true EF estimation. These newly identified non-elementary TSs require in-depth analysis beyond the focus and scope of this paper.

B. Trade-Offs Between Error Floor Estimate Accuracy and Computational Savings

Accuracy of FER estimation in the EF regime depends on identifying low-weight error patterns resulting in decoding failure as shown in Fig. 8. The expansion-contraction procedure is shown to accurately estimate the error floor FER in Fig. 6 and Fig. 7. Now, we discuss various approximate graph expansion methods employed to facilitate the trade-off between the accuracy and computational saving obtained from the method.

1) Expansion Details: In the expansion process given in Section III-A, we expand a given sub-graph by finding a single external variable node that adds a short cycle in the neighborhood of the sub-graph. Extending this search to find two or more external variable nodes in one recursion of the expansion procedure significantly increases the computational complexity. Moreover, expanding with many external variable nodes results in much larger expanded sub-graphs. We choose the expansion threshold parameter κ to limit the expanded sub-graph size (in terms of number of variable nodes) in order to limit the number of error patterns used in our exhaustive contraction step. To illustrate this effect, in Fig. 10, we show how the number of variable nodes increases in the expanded sub-graphs with respect to the expansion depth in the QC 1KB code starting from all the 6-cycles in the Tanner graph with κ set to 10. All the sub-graphs at expansion depth 0 have 3 variable nodes and the lines emerging from the marker on the Y axis shows their expansion to sub-graphs with the number of variable nodes ranging from 3 to 9 in depth 1 and to subsequent depths. The resulting expanded list, $\mathcal{L}_{\text{EXP}}$ contains sub-graphs with number of VNs ranging from 3 (isolated 6-cycles

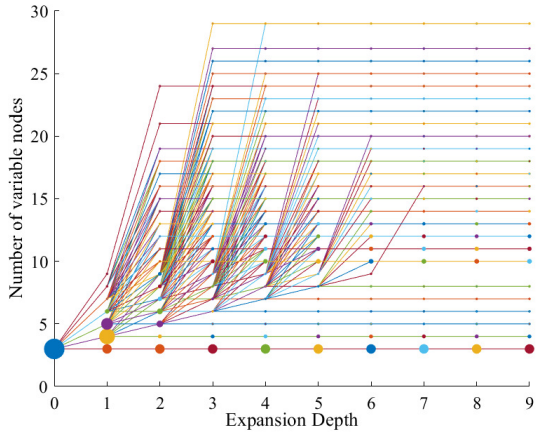


Fig. 10. Number of variable nodes in sub-graphs in each step of expansion process starting from all 6-cycles in QC 1KB code. The marker size at each expansion depth is proportional to the fraction of the total expanded graphs. All the sub-graphs at expansion depth 0 have 3 variable nodes as we expanded from 6-cycles, represented by the marker on Y-axis. Lines emerging from this point show that in expansion depth 1, we have sub-graphs with number of variable nodes ranging from 3 to 9, and their marker sizes show their contribution in the expanded list. All sub-graphs complete the expansion process by expansion depth 8 to form the final expanded list $\mathcal{L}_{\text{EXP}}$.

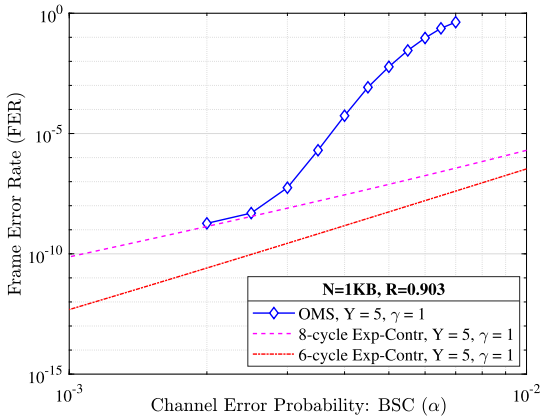


Fig. 11. Comparing the accuracy of FER estimate from expansion-contraction procedure starting separately from two different initial list: 6-cycles and 8-cycles list. For the QC 1KB code, and an OMS decoder with channel value $Y = 5$ and offset $\gamma = 1$, we show that the procedure starting from a 6-cycle initial list gives only a lower bound on the FER estimate.

without short cycles in their neighborhood) to 29 (sub-graphs with very dense neighborhood). The horizontal lines below the number of variable nodes of 10 (as $\kappa = 10$) represent the sub-graphs that already completed their maximal expansion and those above represent the sub-graphs that expanded beyond κ and are not expanded in subsequent steps. This ensures that the expansion process obtains a sufficiently large enough neighborhood of all the cycles taking into account the actual location in the Tanner graph.

The initial list for expansion, $\mathcal{L}_{\text{INI}}$ in Algorithm 1, includes all short cycles of length g and $g + 2$ present in the Tanner graph, where g is the girth. The cycle profiles of the QC LDPC codes used for simulation show that $\chi_{g+2} \gg \chi_g$ (See Table I). This observation also holds true for most of the practically used LDPC codes. Therefore, from a computational standpoint, restricting $\mathcal{L}_{\text{INI}}$ to only the g-cycles makes the expansion-contraction process efficient. However, this approach may adversely affect the accuracy of EF estimation. For example, suppose the input to the expansion in Algorithm 1 is a Tanner graph with girth $g = 6$ and $\mathcal{L}_{\text{INI}}$ containing only the 6-cycles. The algorithm output $\mathcal{L}_{\text{EXP}}$ will not contain any isolated 8-cycles, or sub-graphs composed of only 8-cycles. These missed sub-graphs may contain TSs with the minimum critical number. Consequently, the contraction step will not be able to identify any failure-inducing sets (if present) in such sub-graphs and we will estimate only a lower bound to the FER. Fig. 11 depicts a lower bound estimate on the error floor FER computed from the expansion-contraction procedure from an initial list that includes only 6-cycles for the OMS $Y = 5, \gamma = 1$. The gap to the accurate estimation of EF depends exactly on the contribution to the FER by those failure inducing sets missed in the expansion-contraction process.

2) Contraction Details: The contraction step described in Section III-B is the computationally intensive step in the expansion-contraction procedure. The creation of the initial list followed by the expansion procedure is not factored into the computational savings calculation presented next as they represent a negligible portion of the overall computational complexity. Also, for the task of comparing the EF locations of several decoders for the same LDPC code, the cycle counting and the expansion steps need to be performed only once, while the contraction step needs to be reproduced for every decoder.

Let us first describe the computational complexity of a classical MC simulation reaching an EF, which will serve as reference. In order to obtain statistically confident data in the EF regime with FER $= 10^{-10}$ and record at least 20 errors at this FER, we need to decode at least 20×10^{10} erroneous codewords of length N. Each decoding requires approximately $\log_2 N$ iterations on average to converge. Each decoding iteration performs VN update and CN update along all edges and has a complexity in the order of $(N \times d_v) + (M \times 2d_c)$ elementary operations, where $N \times d_v = M \times d_c$ is the number of edges in the Tanner graph. Therefore, FER estimation in the EF at a single crossover probability requires roughly: $20 \times 10^{10} \times \log_2 N \times \Big((N \times d_v) + (M \times 2d_c) \Big)$ operations.

Compared to the MC simulation of the decoder $\mathcal{D}$ on the entire Tanner graph G, the contraction step performs exhaustive MC simulations only on comparatively very small sub-graphs $\mathcal{G}_{\text{EXP}}$ present in the expansion output list $\mathcal{L}_{\text{EXP}}$. In order to evaluate the computational complexity on a fair scale, we take into account the contribution of error patterns to the FER value we are interested in. To estimate FER at 10^{-10}, we need only error patterns up to weight-7 (see Fig. 8). Checking all error patterns up to weight-7 is sufficient to estimate the EF at 10^{-10} or above. In $\mathcal{L}_{\text{EXP}}$, let us denote the number of sub-graphs with $|\mathcal{V}_{\text{EXP}}| = p$ variable nodes by A_p. Total number of error patterns used in the contraction step for a sub-graph with p variable nodes is $B_p = \sum_{i=1}^{\min(p,7)} \binom{p}{i}$. Each decoding attempt requires approximately $\log_2 p$ iterations on average to converge. Each decoding iteration has a reduced complexity in the order of $(p \times d_v) + 2q$ elementary operations, where q denotes the number of check nodes in the expanded sub-graph. The reduction in the computations

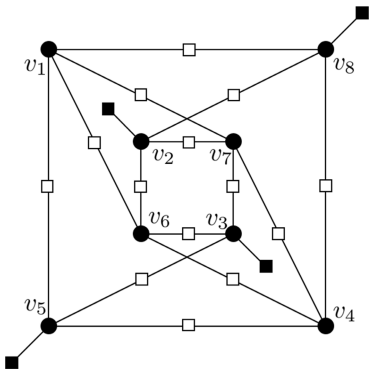


Fig. 12. Minimal critical number of OMS $(Y = 5, \gamma = 1)$ decoder is 4 due to this particular failure configuration present in the QC 1KB code. Two 4-error patterns $\{v_1, v_2, v_3, v_4\}$ and $\{v_5, v_6, v_7, v_8\}$ in the TS fail and contribute the most to the EF.

involved at the check nodes, that results from modeling the external "correct" messages as discussed in Section III-B, contributes to computational savings especially for high rate codes with large CN degrees. For densely connected sub-graphs, q is approximately equal to p. Therefore, the total computational complexity required for contraction step is calculated as $\sum_p A_p \times B_p \times \log_2 p \times \big(p \times d_v + 2p\big)$.

For the QC 2KB code with a code length of 18432, $d_v = 4$ and average $d_c = 41$, MC simulation requires in the order of 6×10^{17} operations to obtain FER $= 10^{-10}$ at a single cross over probability. Considering all expanded cycles from initial list of 8-cycles of the same code, the total number of computations required for the contraction step is only in the order of 8×10^{11}. Compared to the MC simulation, our method is nearly 1,000,000 times faster in computing the EF. As expected, in comparison to the highly complex method of direct enumeration and testing of minimal weight error patterns [17] which requires $\Big(\sum_{i=1}^{5} \binom{N}{i}\Big) \times \log_2 N \times \Big((N \times d_v) + (M \times 2d_c)\Big)$ computations, our method is nearly 10^{14} times faster in computing the EF. Note that the minimal-weight of error patterns obtained for the 2KB code is 5. In [20], wherein cycle enumeration based EF estimate is obtained for hard-decision decoders, to obtain just the critical weight-5 error pattern, we need to at least enumerate all cycles of length 6, 8 and 10 and test for the decodability of all error patterns after decoding over the complete graph. This complexity can be approximated as $\big(\sum_{i=g/2}^{5} \chi_{2i}(2^i - 1)\big) \times \log_2 N \times \Big((N \times d_v) + (M \times 2d_c)\Big)$, where χ_{2i} denotes the number of cycles of length $2i$. Using the number of cycles in the 2KB code as given in Table I (also, $\chi_{10} = 2,540,767,232$), total computations required is in the order of 2×10^{17}. Thus, expansion-contraction method is 3×10^5 faster, in addition to being more accurate especially for stronger decoders whose harmful TSs are much larger and may not be obtained by enumeration of small cycles alone. Using different graph expansion techniques mentioned here, varying expansion parameter κ and choosing the initial list $|\mathcal{L}_{\text{INI}}|$ carefully, we can efficiently and accurately estimate FER in the EF regime with huge computational savings.

C. Identifying Dominant Trapping Sets

In addition to estimating the error floor FER efficiently, the expansion-contraction method produces a list of dominant TSs for the decoder as an important side benefit. Insights obtained on dominant TSs are relevant and useful for improved code design and decoder-specific improvements in future research. Some of these topologies may be harmful only in a given neighborhood while some of them are universally harmful, i.e., harmful irrespective of their location in the Tanner graph of a code. As a trivial example, for the OMS decoder with $Y = 10, \gamma = 1$, we observe that any error pattern in a single isolated eight cycle is always corrected: $(8, 4)$ TS is not a harmful configuration for the decoder. Also, a non-elementary $(5, 9)$ TS, which has a six cycle and 2 eight cycles, is also not a harmful configuration. However, this decoder fails for all $(4, 6)$ TS structures present, regardless of neighborhood, and these failure significantly contributes to its EF slope. Fig. 12 shows a TS present in the QC 1KB code for the offset min-sum decoder with $Y = 5$, $\gamma = 1$. Two 4-error patterns: $\mathbf{e}_1 = \{v_1, v_2, v_3, v_4\}$ and $\mathbf{e}_2 = \{v_5, v_6, v_7, v_8\}$ in the TS fail for the decoder. We also obtain that the critical number, $\mu = 4$ and strength, $\mathrm{s} = 2$ for this TS. Being the lowest weight failure inducing error patterns, they contribute the most to the EF as shown in Fig. 8 (a). Interestingly, the two error patterns in the TS do not overlap. The dense graph structure having numerous overlapping 8 cycles and their symmetry contributes to the decoder failure. Another interesting observation was finding dominant non-elementary TSs in the IEEE 802.3an standard code, for the OMS $(Y = 5, \gamma = 2)$ decoder. These harmful structures can be attributed to their neighborhood (8,8) TSs along with the observation [24] that the BSC channel errors with high error magnitudes can result in dominant TSs with odd degree > 1 CNs.

V. Conclusion and Future Work

We devised a computationally efficient method for estimating error floors of LDPC codes over the BSC channel with or without a-priori knowledge of harmful TSs. Short cycles in the Tanner graph are expanded and exhaustively contracted to obtain a list of harmful trapping sets. TSs are identified from sufficiently large neighborhood of short cycles in an actual Tanner graph, i.e. not isolated from their neighborhood. Also, TSs are identified by decoding error patterns using a particular message passing decoder, i.e., the list of harmful trapping sets differs from one decoder to another. The method is applicable to regular and irregular codes and can be readily extended to a more general binary-input, finite-level output symmetric memoryless channels with arbitrary message update rule with an expense of increased computational complexity. Expansion-contraction method can be fine-tuned further for improving the speed and accuracy of obtaining the EF. As future extension, we would extend the method for EF computation for the binary input-additive white Gaussian noise (AWGN) channel [12] with quantized outputs. We plan to utilize the trapping set profile to develop a new harmfulness characterization that can help design LDPC codes with very low EF.

APPENDIX A
ITERATIVE DECODING ALGORITHM

An iterative decoder $\mathcal{D}$, is a 6-tuple $\mathcal{D} = (\mathcal{M}, \mathcal{Y}, \Phi, \Psi, \hat{\Phi}, \hat{\Psi})$, where $\mathcal{Y}$ and $\mathcal{M}$ are the channel (output) and message alphabets, respectively, Φ, Ψ are the update functions used in variable nodes (VNs) and check nodes (CNs), and $\hat{\Phi}$ is the decision function, and $\hat{\Psi}$ is the check estimate function. Let the messages passed over the edges of the Tanner graph be denoted as follows: $\mathbf{m}^{(\ell)} = \eta^{(\ell)}_{\mathcal{N}(v)\setminus c \to v}$ denote all incoming messages to the VN v except a message from the CN c at ℓ-th iteration. Similarly, $\mathbf{n}^{(\ell)} = \nu^{(\ell)}_{\mathcal{N}(c)\setminus v \to c}$ denote all incoming messages to the CN c except from the VN v at ℓ-th iteration. We denote by $\mathbf{l}^{(\ell)} = \eta^{(\ell)}_{\mathcal{N}(v)\to v}$, messages incoming to v, and by $\mathbf{i}^{(\ell)} = \eta^{(\ell)}_{\mathcal{N}(c)\to c}$ messages incoming to c in the ℓ-th iteration.

For a quantized decoder with K levels, the message alphabet $\mathcal{M}$ consists of $K = 2k + 1$ levels to which the message values are confined to. The message alphabet is defined as follows: $\mathcal{M} = \{-L_k, \ldots, -L_1, 0, L_1, \ldots, L_k\}$, where $L_i \in \mathbb{Z}+$ (positive integers) and $L_i > L_j$ for any $i > j$. The sign of a message $m \in \mathcal{M}$ can be interpreted as the estimate of the bit associated with the VN for which m is being passed to or from (positive for zero and negative for one), and the magnitude of a message as a measure of how reliable this value is. The set $\mathcal{Y}$ which denotes the set of possible channel values is defined for the case of BSC as $\mathcal{Y} = \{\pm Y\}$, and the channel value $y_i \in \mathcal{Y}$ corresponding to node v_i is determined by $y_i = (-1)^{r_i} Y$, i.e., we use the mapping $0 \to Y$ and $1 \to -Y$. The message from VN v_i is initialized to $\Phi(y_i, \mathbf{0})$, and in each iteration updated according to the rules Φ and Ψ. The function $\Psi : \{\mathcal{M}\}^{d_c - 1} \to \mathcal{M}$ is used for update at a CN c with degree d_c, so that $\eta^{(\ell)}_{c\to v} = \Psi(\mathbf{n}^{(\ell-1)})$. Note that Ψ is a symmetric function, i.e., any permutation of the function variables leaves the function unchanged. The function $\Phi : \mathcal{Y} \times \{\mathcal{M}\}^{d_v - 1} \to \mathcal{M}$ is used for updating outgoing message of a VN v with degree d_v. $\nu^{(\ell)}_{v\to c} = \Phi(y_v, \mathbf{m}^{(\ell)})$. Note that Φ is partially symmetric in the variables $\mathbf{m}$. The decision function computes $\lambda^{(\ell)}_v = \hat{\Phi}(\mathbf{l}^{(\ell)}, y_v)$, which is used to decide the value $\hat{x}_v$. The decided bit value in ℓ-th iteration is calculated based on the sign as $\hat{x}^{(\ell)}_v = \mathbb{1}_{\lambda^{(\ell)}_v < 0}$. An estimate of the CN value c in the ℓ-th iteration is $\sigma^{(\ell)}_c = \hat{\Psi}\left(\mathbf{i}^{(\ell)}\right) = \text{sgn}\left(\prod \mathbf{i}^{(\ell)}\right)$. A CN is said to be satisfied if $\sigma_c = 1$, unsatisfied if $\sigma_c = -1$, and undecided if $\sigma_c = 0$. A syndrome checking verifies that all the CN estimates $\hat{\sigma}^{(\ell)}_c = \hat{\Psi}\left(\hat{\mathbf{x}}^{(\ell)}_{\mathcal{N}_c}\right) = \text{sgn}\left(\prod \hat{\mathbf{x}}^{(\ell)}_{\mathcal{N}_c}\right)$ based on the decoded word $\hat{\mathbf{x}}^{(\ell)}$ are satisfied. In this case, we say that a codeword is found.

As examples of iterative decoder $\mathcal{D}$, consider the standard min-sum (MS) decoder and the offset min-sum (OMS) decoder. The CN update function of a quantized MS decoder $\Psi : \{\mathcal{M}\}^{d_c - 1} \to \mathcal{M}$ is the minimum of the extrinsic incoming messages to the CN. i.e., $\Psi(m_1, \ldots, m_{d_c - 1}) = \left(\prod_{j=1}^{d_c - 1} \text{sgn}(m_j)\right) \min_{j \in [1, d_c - 1]} (|m_j|)$, where sgn(.) denotes the sign function, and $[x, y]$ denotes the set of integers no smaller than x and not larger than y. The VN update function $\Phi : \mathcal{Y} \times \{\mathcal{M}\}^{d_v - 1} \to \mathcal{M}$ is the sum of the extrinsic incoming CN messages and the channel value: $\Phi(y_i, m_1, \ldots, m_{d_v - 1}) = Q\left(y_i + \sum_{j=1}^{d_v - 1} m_j\right)$, where the function $Q(.)$ is defined [16] as follows based on a threshold set $\{t_1, \ldots, t_{k+1}\}$ such that $t_i \in \mathbb{R}^+$ and $t_i > t_j$ if $i > j$, and $t_{k+1} = \infty$. $Q(x) = \begin{cases} \text{sgn}(x) L_i, & \text{if } t_i \le |x| < t_{i+1} \\ 0, & \text{otherwise} \end{cases}$. For the offset min-sum (OMS) decoder, an offset factor γ is introduced in the CN update function as $\Psi'(m_1, \ldots, m_{d_c - 1}) = \left(\prod_{j=1}^{d_c - 1} \text{sgn}(m_j)\right) \max\left(\min_{j \in \{[1, d_c - 1]\}} (|m_j|) - \gamma, 0\right)$. An equivalent representation obtained by introducing the offset factor at the VN keeps the CN update the same for both the decoders.

REFERENCES

[1] N. Raveendran, D. Declercq, and B. Vasić, "Accurate and efficient error floor computation of LDPC codes using expansion-contraction method," in *Proc. Inf. Theory Appl. Workshop*, San Diego, CA, USA, Feb. 2020, pp. 1–7.

[2] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA, USA: MIT Press, 1963.

[3] L. Bazzi, T. J. Richardson, and R. L. Urbanke, "Exact thresholds and optimal codes for the binary-symmetric channel and Gallager's decoding algorithm A," *IEEE Trans. Inf. Theory*, vol. 50, no. 9, pp. 2010–2021, Sep. 2004.

[4] B. Vasić, D. Nguyen, and S. K. Chilappagari, "Failures and error floors of iterative decoders," in *Channel Coding: Theory, Algorithms, and Applications*. Oxford, U.K.: Academic, 2014, pp. 299–341.

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

[6] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 1710–1722, Jun. 1996.

[7] D. Burshtein, "On the error correction of regular LDPC codes using the flipping algorithm," *IEEE Trans. Inf. Theory*, vol. 54, no. 2, pp. 517–530, Feb. 2008.

[8] J. Feldman, T. Malkin, R. A. Servedio, C. Stein, and M. J. Wainwright, "LP decoding corrects a constant fraction of errors," *IEEE Trans. Inf. Theory*, vol. 53, no. 1, pp. 82–89, Jan. 2007.

[9] C. Daskalakis, A. G. Dimakis, R. M. Karp, and M. J. Wainwright, "Probabilistic analysis of linear programming decoding," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3565–3578, Aug. 2008.

[10] T. Richardson, "Error floors of LDPC codes," in *Proc. 41st Annu. Allerton Conf. Commun., Control, Comput.*, Monticello, IL, USA, Oct. 2003, pp. 1426–1435.

[11] B. Vasić, S. K. Chilappagari, D. V. Nguyen, and S. K. Planjery, "Trapping set ontology," in *Proc. 47th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Monticello, IL, USA, Sep./Oct. 2009, pp. 1–7.

[12] B. K. Butler and P. H. Siegel, "Error floor approximation for LDPC codes in the AWGN channel," *IEEE Trans. Inf. Theory*, vol. 60, no. 12, pp. 7416–7441, Dec. 2014.

[13] C. Schlegel and S. Zhang, "On the dynamics of the error floor behavior in (regular) LDPC codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3248–3264, Jul. 2010.

[14] D. V. Nguyen, S. K. Chilappagari, M. W. Marcellin, and B. Vasić, "On the construction of structured LDPC codes free of small trapping sets," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2280–2302, Apr. 2012.

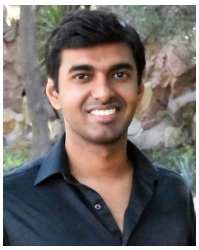
[15] Q. Huang, Q. Diao, S. Lin, and K. Abdel-Ghaffar, "Cyclic and quasi-cyclic LDPC codes on constrained parity-check matrices and their trapping sets," *IEEE Trans. Inf. Theory*, vol. 58, no. 5, pp. 2648–2671, May 2012.

[16] S. K. Planjery, D. Declercq, L. Danjean, and B. Vasić, "Finite alphabet iterative decoders—Part I: Decoding beyond belief propagation on the binary symmetric channel," *IEEE Trans. Commun.*, vol. 61, no. 10, pp. 4033–4045, Oct. 2013.

[17] H. Xiao and A. H. Banihashemi, "Estimation of bit and frame error rates of finite-length low-density parity-check codes on binary symmetric channels," *IEEE Trans. Commun.*, vol. 55, no. 12, pp. 2234–2239, Dec. 2007.

[18] S. Chilappagari, S. Sankaranarayanan, and B. Vasić, "Error floors of LDPC codes on the binary symmetric channel," in *Proc. IEEE Int. Conf. Commun.*, vol. 3, Istanbul, Turkey, Jun. 2006, pp. 1089–1094.

[19] L. Dolecek, P. Lee, Z. Zhang, V. Anantharam, B. Nikolic, and M. Wainwright, "Predicting error floors of structured LDPC codes: Deterministic bounds and estimates," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 6, pp. 908–917, Aug. 2009.
[20] H. Xiao and A. Banihashemi, "Error rate estimation of low-density parity-check codes on binary symmetric channels using cycle enumeration," *IEEE Trans. Commun.*, vol. 57, no. 6, pp. 1550–1555, Jun. 2009.
[21] H. Xiao, A. H. Banihashemi, and M. Karimi, "Error rate estimation of low-density parity-check codes decoded by quantized soft-decision iterative algorithms," *IEEE Trans. Commun.*, vol. 61, no. 2, pp. 474–484, Feb. 2013.
[22] S. K. Chilappagari and B. Vasić, "Error-correction capability of column-weight-three LDPC codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2055–2061, May 2009.
[23] I. Djurdjevic, J. Xu, K. Abdel-Ghaffar, and S. Lin, "A class of low-density parity-check codes constructed based on Reed–Solomon codes with two information symbols," *IEEE Commun. Lett.*, vol. 7, no. 7, pp. 317–319, Jul. 2003.
[24] A. Hareedy, C. Lanka, and L. Dolecek, "A general non-binary LDPC code optimization framework suitable for dense flash memory and magnetic storage," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2402–2415, Sep. 2016.



Nithin Raveendran (Graduate Student Member, IEEE) received the B.E. (Hons) degree in electrical engineering from BITS Pilani, India, in 2011, and the M.Sc. Engg. degree from the Indian Institute of Science, Bangalore, in 2015. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Arizona, Tucson, AZ, USA. His research interests include classical and quantum error correction, iterative decoding techniques, signal processing, coding, and information theory.

David Declercq (Senior Member, IEEE) was born in June 1971. He received the Ph.D. degree in statistical signal processing from the University of Cergy-Pontoise, France, in 1998. He has held the junior position at the Institut Universitaire de France from 2009 to 2014. He is the Co-Founder and a CTO of Codelucida Inc. His research topics lie in digital communications and error-correction coding theory. He worked several years on LDPC codes from the code and decoder design aspects. Since 2003, he developed a strong expertise on non-binary LDPC codes and decoders in high-order Galois fields GF(q). A large part of his research projects are related to non-binary LDPC codes. More recently, he also contributed to the efficient hardware implementation of LDPC decoders, and proposed several low-complexity and low-power solutions for message passing LDPC decoders as well as bit- and symbol-flipping decoders. He published more than 55 articles in major journals IEEE TRANSACTIONS COMMUNICATION, IEEE TRANSACTIONS ON INFORMATION THEORY, *Communication Letters*, IEEE TRANSACTIONS CIRCUITS AND SYSTEMS, and so on and more than 150 articles in major conferences in *Information Theory and Signal Processing*.

Bane Vasić (Fellow, IEEE) is currently a Professor of electrical and computer engineering and mathematics with the University of Arizona and the Director of the Error Correction Laboratory. He is an inventor of the soft error-event decoding algorithm, and the key architect of a detector/decoder for Bell Labs data storage read channel chips which were regarded as the best in industry. His pioneering work on structured low-density parity check (LDPC) error correcting codes and invention of codes has enabled low-complexity iterative decoder implementations. Structured LDPC codes are today adopted in a number of communications standards and data storage systems. He is known for his theoretical work in error correction coding theory and codes on graphs which has led to characterization of the hard decision iterative decoders of LDPC codes and design of decoders with best error-floor performance known today. He is a Co-Founder of Codelucida, a startup company developing advanced error correction solutions for communications and data storage. He is a Fulbright Scholar, da Vinci Fellow, and a past Chair of IEEE Data Storage Technical Committee.