Lowering the Error Floor of LDPC Codes Using Cyclic Liftings

Reza Asvadi, Amir H. Banihashemi, Senior Member, IEEE, and Mahmoud Ahmadian-Attari

Abstract—Cyclic liftings are proposed to lower the error floor of low-density parity-check (LDPC) codes. The liftings are designed to eliminate dominant trapping sets of the base code by removing the short cycles which form the trapping sets. We derive a necessary and sufficient condition for the cyclic permutations assigned to the edges of a cycle \(c\) of length \(\ell(c)\) in the base graph such that the inverse image of \(c\) in the lifted graph consists of only cycles of length strictly larger than \(\ell(c)\). The proposed method is universal in the sense that it can be applied to any LDPC code over any channel and for any iterative decoding algorithm. It also preserves important properties of the base code such as degree distributions. The proposed method is applied to both structured and random codes over the binary symmetric channel (BSC). The error floor improves consistently by increasing the lifting degree, and the results show significant improvements in the error floor compared to the base code, a random code of the same degree distribution and block length, and a random lifting of the same degree. Similar improvements are also observed when the codes designed for the BSC are applied to the additive white Gaussian noise (AWGN) channel.

I. INTRODUCTION

LOW-DENSITYITY parity-check (LDPC) codes have emerged as one of the top contenders for capacity approaching error correction over many important channels. A well-known construction of LDPC codes is based on protographs, also referred to as base graphs or projected graphs [7]. In such constructions, a bipartite base graph \(G\) is copied \(N\) times and for each edge \(e\) of \(G\), a permutation is applied to the \(N\) copies of \(e\) to interconnect the \(N\) copies of \(G\). The resulting graph, called the \(N\)-cover or the \(N\)-lifting of \(G\), is then used as the Tanner graph of the LDPC code. If the permutations are cyclic, the resulting LDPC code is called quasi-cyclic (QC). QC LDPC codes are attractive due to their simple implementation and analysis [7].

Iteratively decoded finite-length LDPC codes demonstrate an abrupt change in their error rate curves, referred to as error floor, in the high signal to noise ratio (SNR) region. The analysis of the error floor and techniques to improve the error floor performance of LDPC codes are still very active areas of research.

There is extensive literature on reducing the error floor of finite-length LDPC codes over different channels and for different iterative decoding algorithms. One category of such literature, focuses on modification of iterative decoding algorithms, see, e.g., [3], while another category is concerned with the code construction. In the second category, some researchers use indirect measures such as girth [8] or approximate cycle extrinsic message degree (ACE) [9], while others work with direct measures of error floor performance such as the distribution of stopping sets or trapping sets [10], [4]. In [10], edge swapping is proposed as a technique to increase the stopping distance of an LDPC code, and thus to improve its error floor performance over the binary erasure channel (BEC). Random cyclic liftings are also studied in [10] and shown to improve the average performance of the ensemble in the error floor region compared to the base code. Ivkovic et al. [4] apply the same technique of edge swapping between two copies of a base LDPC code to eliminate the dominant trapping sets of the base code over the binary symmetric channel (BSC).

In the proposed approach also, we focus on dominant trapping sets which are the main contributors to the error floor. We start from the code whose error floor is to be improved, as the base code. We then construct a new code by cyclically lifting the base code. The lifting is designed carefully to eliminate the dominant trapping sets of the base graph. This is achieved by removing the short cycles which form the dominant trapping sets. Our work has similarities to [4] and [10]. The similarity with both [4] and [10] is that we also use graph covers or liftings to improve the error floor performance of a base code. Our focus however, unlike [4], is restricted to cyclic liftings, as these are advantageous for implementation. Moreover, to eliminate the dominant trapping sets we use a different approach than the one in [4]. More specifically, our approach is based on the elimination of the short cycles involved in the trapping sets. To do so, we derive a necessary and sufficient condition for the problematic cycles of the base code such that they are mapped to strictly larger cycles in the lifted code. The difference with [10] is that while [10] is focused on the ensemble performance of random liftings, our work is concerned with the intentional design of a particular cyclic lifting.

Given a base code and its dominant trapping sets over a certain channel and under a specific iterative decoding algorithm, the proposed construction can lower the error floor by increasing the block length while preserving the important properties of the base code such as degree distributions, and the encoder and decoder structure. The code rate is also preserved or is decreased slightly depending on the rank deficiency of the parity-check matrix of the base code. Moreover, the cyclic nature of the lifting makes it implementation-friendly. We apply the proposed construction to a number of LDPC codes, to improve the error floor performance of
Gallager A/B algorithms over the BSC.\(^1\) Simulation results show a consistent improvement in the error floor performance by increasing the degree of liftings. The constructed codes are far superior to similar random codes or codes constructed by random liftings in the error floor region. We also examine the performance of the codes constructed for BSC/Gallager B algorithm, over the binary-input additive white Gaussian noise (BIAGWN) channel with min-sum decoding and observe similar improvements in the error floor performance. In the following, due to the limitation of space, the proof of Lemma 1 and Theorems 1 and 2 is omitted.

II. PRELIMINARIES: LDPC CODES, TANNER GRAPHS, GRAPH LIFTINGS AND TRAPPING SETS

A. LDPC Codes and Tanner Graphs

Consider a binary LDPC code \( \mathcal{C} \) represented by a Tanner graph \( G = (V_\text{b} \cup V_\text{c}, E) \), where \( V_\text{b} = \{b_1, \ldots , b_n\} \) and \( V_\text{c} = \{c_1, \ldots , c_m\} \) are the sets of variable nodes and check nodes, respectively, and \( E \) is the set of edges. Corresponding to \( G \), we have an \( m \times n \) parity-check matrix \( H = [h_{ij}] \sim \mathcal{C} \), where \( h_{ij} = 1 \) if and only if (i) the node \( c_i \in V_\text{c} \) is connected to the node \( b_j \in V_\text{b} \) in \( G \); or equivalently, iff \( \{b_j, c_i\} \in E \). If all the nodes in the set \( V_\text{b} \) have the same degree \( d_v \) and all the nodes in the set \( V_\text{c} \) have the same degree \( d_c \), the corresponding LDPC code is called a regular \( (d_v, d_c) \) code. Otherwise, it is called irregular.

A subgraph of \( G \) is a path of length \( k \) if it consists of a sequence of \( k + 1 \) nodes \( \{u_1, \ldots , u_{k+1}\} \) and \( k \) distinct edges \( \{\{u_i, u_{i+1}\} : i = 1, \ldots , k\} \). A path is a cycle if \( u_1 = u_{k+1} \), and all the other nodes are distinct. The length of the shortest cycle(s) in the graph is called girth.

B. Graph Liftings

Consider the set of all possible permutations \( S_N \) over the set of integer numbers \( Z_{1:N} \triangleq \{1, \ldots , N\} \). This set forms a group, known as the symmetric group, under composition. Each element \( \pi \in S_N \) can be represented by all the values \( \pi(i), i \in Z_{1:N} \). For the identity element \( \pi_0 \), we have \( \pi_0(i) = i, \forall i \), and the inverse of \( \pi \) is denoted by \( \pi^{-1} \) and defined as \( \pi^{-1}(\pi(i)) = i \). An alternate representation of permutations is to represent a permutation \( \pi \) with an \( N \times N \) matrix \( \Pi = [\pi_{ij}] \), whose elements are defined by \( \pi_{ij} = 1 \) if \( j = \pi(i) \), and \( \pi_{ij} = 0 \), otherwise. As a result, we have the isomorphic group of all \( N \times N \) permutation matrices with the group operation defined as matrix multiplication, and the identity element equal to the identity matrix \( I_N \).

Consider the cyclic subgroup \( \mathcal{C}_N \) of \( S_N \) consisting of the \( N \) circular permutations defined by \( \pi_{i,d}(i) = i + d \mod N + 1, d \in Z_{0:N-1} \). The permutation \( \pi_{d} \) corresponds to \( d \) cyclic shifts to the right. In the matrix representation, permutation \( \pi_{d} \) corresponds to a permutation matrix whose rows are obtained by cyclically shifting all the rows of the identity matrix \( I_N \) by \( d \) to the right. This matrix is denoted by \( I_N^{(d)} \). Note that \( I_N^{(0)} = I_N \). Clearly, \( \mathcal{C}_N \) can be generated by \( I_N^{(d)} \), where each element \( I_N^{(d)} \).\( \in Z_{0:N-1} \) of \( \mathcal{C}_N \) is the \( d \)-th power of \( I_N^{(1)} \). This defines a natural isomorphism between \( \mathcal{C}_N \) and the set of integers modulo \( N, Z_{0:N-1} \), under addition. The latter group is denoted by \( Z_N \).

Consider the following construction of a graph \( \tilde{G} = (\tilde{V}, \tilde{E}) \) from a graph \( G = (V,E) \): We first make \( N \) copies of \( G \) such that for each node \( v \in V \), we have \( N \) copies \( \tilde{v} \triangleq [v_1, \ldots , v_N] \) in \( \tilde{V} \). For each edge \( e = \{u,v\} \in E \), we assign a permutation \( \pi_e \in S_N \) to the \( N \) copies of \( e \) in \( \tilde{E} \) such that an edge \( \{u,v\} \) belongs to \( \tilde{E} \) if \( \pi_e(i) = j \). The set of these edges is denoted by \( e \). The graph \( \tilde{G} \) is called an \( N \)-cover or an \( N \)-lifting of \( G \), and \( G \) is referred to as the base graph, protograph or projected graph corresponding to \( \tilde{G} \). We also call the application of a permutation \( \pi_e \) to the \( N \) copies of \( e \), edge swapping, high lighting the fact that the permutation swaps edges among the \( N \) copies of the base graph.

In this work, \( G \) is a Tanner graph, and we define the edge permutations from the variable side to the check side, i.e., \( \{b,c\} \in E \) are defined by \( \{b_i,c_{i+1}\}, i \in Z_{1:N} \). Equivalently, \( \tilde{e} \) can be described by \( \{b_{(\pi_e^{-1})(i)},c_j\}, j \in Z_{1:N} \). Our focus in this paper is on cyclic liftings of \( G \), where the edge permutations are selected from \( \mathcal{C}_N \), or equivalently \( Z_N \). Thus the nomenclature cyclic edge swapping.

To the lifted graph \( \tilde{G} \), we associate an LDPC code \( \tilde{\mathcal{C}} \), referred to as the lifted code, such that the \( m \times n \) parity-check matrix \( \tilde{H} \) of \( \tilde{\mathcal{C}} \) is equal to the adjacency matrix of \( \tilde{G} \). More specifically, \( \tilde{H} \) consists of \( m \times n \) sub-matrices \( [\tilde{H}]_{ij}, 1 \leq i \leq m, 1 \leq j \leq n \), arranged in \( m \) rows and \( n \) columns. The sub-matrix \( [\tilde{H}]_{ij} \) in row \( i \) and column \( j \) is the permutation matrix from \( \mathcal{C}_N \) corresponding to the edge \( \{b_j,c_i\} \) where \( h_{ij} \neq 0 \); otherwise, \( [\tilde{H}]_{ij} \) is the all-zero matrix. Let the \( m \times n \) matrix \( D = [d_{ij}] \) be defined by \( [\tilde{H}]_{ij} = I_N^{(d_{ij})}, d_{ij} \in Z_N \) if \( h_{ij} \neq 0 \), and \( d_{ij} = +\infty \), otherwise. Matrix \( D \) called the matrix of edge permutation indices, fully describes \( \tilde{H} \) and thus the cyclically lifted code \( \tilde{\mathcal{C}} \).

C. Trapping Sets and Error Floor

It is well-known that trapping sets are the culprits in the error floor region of iterative decoding algorithms. An \((a,b)\) trapping set is defined as a set of \( a \) variable nodes which have \( b \) check nodes of odd degree in their induced subgraph. Among trapping sets, the most harmful ones are called dominant. Trapping sets depend not only on the Tanner graph of the code but also on the channel and the iterative decoding algorithm. In general, finding all the trapping sets is a hard problem, and one often needs to resort to efficient search techniques to obtain the dominant trapping sets, see, e.g., [6]. Trapping sets for Gallager A/B algorithms over the BSC are examined for a number of LDPC codes in [1], [12]. In this work, we assume that the dominant trapping sets are known and available.

In the context of symmetric decoders over the BSC, the error floor of frame error rate (FER) can be estimated by [11]

\[
\text{FER} \approx N_J e^{-f},
\] (1)

\(^1\)The choice of BSC/Gallager algorithms is for simplicity, and the proposed construction is applicable for any channel/decoding algorithm combination as long as the dominant trapping sets are known.
where $\epsilon$ is the channel crossover probability, and $J$ and $N_J$ are the size and the number of the smallest error patterns that the decoder fails to correct. From (1), it is clear that the dominant trapping sets over the BSC are those caused by the minimum number of initial errors. (In [1], [4], parameter $J$ in (1) is called minimum critical number.) In the double-logarithmic plane, one can see from (1) that $\log\text{FER}$ reduces linearly with $\log(\epsilon)$ and the slope of the line is determined by $J$.

III. DESIGN OF CYCLIC LIFTINGS TO ELIMINATE TRAPPING SETS

In this work, our focus is on the design of cyclic liftings of a given Tanner graph to eliminate its dominant trapping sets with respect to a given channel/decoding algorithm with the purpose of reducing the error floor. (This is equivalent to the design of non-infinity edge permutation indices of matrix $D$.) For example, equation (1) indicates that for improving the error floor on the BSC, one needs to increase $J$ and decrease $N_J$ corresponding to the dominant trapping sets. In particular, while increasing $J$ for dominant trapping sets increases the slope of $\log\text{FER}$ vs. $\log(\epsilon)$ at low channel crossover probabilities $\epsilon$, reducing $N_J$ amounts to a downward shift of the curve. So, the general idea is to primarily increase $J$ by eliminating the trapping sets with the smallest critical number, and the secondary goal is then to decrease $N_J$.

It is well-known that dominant trapping sets are composed of short cycles (see, e.g., [1], [12]). To eliminate the trapping sets, we thus aim at eliminating their constituent cycles in the lifted graph. In the following, we examine the inverse image of a (base) cycle in the cyclically lifted graph.

A. Cyclic Liftings of Cycles

**Lemma 1:** Consider a cyclic $N$-lifting $\tilde{G}$ of a Tanner graph $G$. Consider a path $\xi$ of length $\ell$ in $G$, which starts from a variable node $b$ and ends at a variable node $\tilde{b}'$ with the sequence of edges $e_1, \ldots, e_{\ell}$. Corresponding to the edges, we have the sequence of permutation matrices $I^{(d_1)}, \ldots, I^{(d_{\ell})}$. Then the permutation matrix that maps $b$ to $\tilde{b}'$ through the path $\xi$ is $I^{(d)}$, where

$$d = \sum_{i=0}^{\ell-1} (-1)^i d_{i+1} \mod N. \tag{2}$$

The value of $d$ given in (2) is called the permutation index of the path from $b$ to $\tilde{b}'$. Clearly, the permutation index of the path from $\tilde{b}'$ to $b$ is equal to $d' = N - d \mod N$. If $b = \tilde{b}'$ and all the other nodes are distinct, then the path will become a cycle and depending on the direction of the cycle, its permutation index will be equal to $d$ or $d'$.

**Theorem 1:** Consider the cyclic $N$-lifting $\tilde{G}$ of the Tanner graph $G$. Suppose that $c$ is a cycle of length $\ell$ in $G$. The inverse image of $c$ in $\tilde{G}$ is then the union of $N/k$ cycles, each of length $k\ell$, where $k$ is the order of the element(s) of $\mathbb{Z}_N$ corresponding to the permutation indices of $c$.

In what follows, we refer to the value $k$ in Theorem 1 as the order of cycle $c$, and use the notation $O(c)$ to denote it.

**Corollary 1:** Consider the cyclic $N$-lifting $\tilde{G}$ of the Tanner graph $G$. Suppose that $c$ is a cycle of length $\ell$ in $G$. The inverse image of $c$ in $\tilde{G}$ is the union of non-overlapping cycles, each strictly longer than $\ell$ iff $O(c) > 1$; or equivalently, iff the permutation index of $c$, given in (2), is nonzero.

B. Intentional Edge Swapping (IES) Algorithm

Suppose that $T$ is the set of all dominant trapping sets, and $C(T)$ is the set of all the cycles in $T$. We also use the notations $t$ and $C(t)$ for a trapping set and its constituent cycles, respectively. For an edge $e$, we use $C^e(t)$ to denote the set of cycles in the trapping set $t$ that include $e$. In the previous subsection, we proved that a cycle $c$ in the base Tanner graph $G$ is mapped to the union of larger cycles in the cyclically lifted graph $\tilde{G}$ iff $O(c) > 1$. To eliminate the dominant trapping sets, we are thus interested in assigning the edge permutation indices to the edges of $C(T)$ such that $O(c) > 1$ for every cycle $c \in C(T)$. To achieve this, we order the trapping sets according to increasing critical number. We still denote this ordered set by $T$ with a slight abuse of notation. Note that $T$ may now include trapping sets with critical numbers larger than the minimum one. We then go through the trapping sets in $T$ one at a time and identify and list all the cycles involved in each trapping set in $C(T)$, i.e., $C(T) = \{c \in C(t), \forall t \in T\}$. Note that $C(T)$ is also partially ordered based on the corresponding ordering of the trapping sets in $T$.

**Example 1:** Three typical trapping sets for Gallager A/B algorithms are shown in Fig. 1 [1]. The (4, 4) and (5, 3) trapping sets include one and three cycles of length 8, respectively, while the (4, 2) trapping set has 2 cycles of length 6 and one cycle of length 8.

Fig. 1. a) (5, 3) Trapping set b) (4, 2) Trapping set c) (4, 4) Trapping set. □ = Variable Node, □ = Even degree Check Node and □ = Odd degree Check Node.

The next step is to choose proper edges of each trapping set to be swapped, i.e., to choose the edges to which nonzero permutation indices are assigned. In general, the policy is to
select the minimum number of edges that can result in $O(c) > 1$ for every cycle $c$ in the trapping set $t$ under consideration.

**Example 2:** Going back to Fig. 1, for the $(4, 3)$ trapping set, it would be enough to just pick one of the edges of the cycle of length 8 to eliminate this cycle in the lifted graph. For the $(5, 3)$ and $(4, 2)$ trapping sets, however, at least two edges should be selected for the elimination of all the cycles. A proper choice would be to select one edge from the diagonal and the other edge from one of the sides.

Related to the edge selection, is the next step of permutation index assignment to the selected edges such that $O(c) > 1$ for every cycle $c \in C(t)$. In general, we would like to have larger orders for the cycles. This in turn would result in larger cycles in the lifted graph. To limit the complexity, however, we approach this problem in a greedy fashion and with the main goal of just eliminating all the cycles in $C(t)$, i.e., for each selected edge $e$, we choose the permutation index such that all the cycles $C' (t)$ have orders larger than one. This can be performed by sequentially testing the values in the set $Z_1 \ldots N^2$. As soon as such an index is found, we assign it to $e$ and move to the next selected edge and repeat the same process.

We call the proposed algorithm *intentional edge swapping* (IES) to distinguish it from “random edge swapping,” commonly used to construct lifted codes and graphs. The details and the pseudocode of the algorithm are omitted due to the limitation of space.

The process of permutation index assignment in the proposed algorithm is based on the satisfaction of inequalities $d \neq 0$ for certain cycles, where $d$ is given in (2). In general, this is easier to achieve if the variables involved in (2) are selected from a larger alphabet space. In fact, by increasing $N$, one can eliminate more trapping sets and achieve a better performance in the error floor region.

**C. Minimum Distance and Rate of Cyclic Liftings**

**Theorem 2:** If a code $C$ has minimum distance $d_{\text{min}}$ and rate $r$, then the minimum distance $d_{\text{min}}^{(N)}$ and rate $r^{(N)}$ of a cyclic $N$-lifting $C'$ of $C$ satisfy $d_{\text{min}} \leq d_{\text{min}}^{(N)} \leq N \cdot d_{\text{min}}$, and $r^{(N)} \leq r$, respectively, for $N$ an integer power of 2. In particular, $r^{(N)} = r$ if the parity-check matrix of $C$ has full rank.

**IV. NUMERICAL RESULTS**

We have applied the IES algorithm of Subsection III-B to a number of regular and irregular LDPC codes with promising results. In this section, due to the lack of space, we just present the results for the (155, 64) Tanner code [8], and a (504, 252) randomly constructed regular code [5].

**Example 3:** For the (155, 64) Tanner code under Gallager B algorithm, the most dominant trapping set is the (5, 3) trapping set, shown in Fig. 1, with critical number 3. We apply the IES algorithm to this code to design cyclic $N$-liftings for $N = 2, 3, 4,$ and 5. The FER curves of the designed codes are presented in Fig. 2 along with the FER of the base code.

A careful inspection of Fig 2 shows that using a 2-lifting, the slope of the curve changes from 3 to 4, an indication that all (5, 3) trapping sets are eliminated. In this case, $(4, 4)$ trapping sets play the dominant role. Further increase of $N$ to 3 and then 4, only causes a downward shift of the curve (with no change of slope), an indication that the minimal critical number remains at 4 for the 2 lifted codes and increasing the degree of lifting just reduces the number of $(4, 4)$ trapping sets. Increasing $N$ to 5 however, eliminates all the $(4, 4)$ trapping sets and the slope of the FER curve further increases to 5. The dominant trapping sets for the 5-lifting are $(5, 5)$ trapping sets.

It is important to note that for $N = 2$, the performance of the designed code is practically identical to that of the code designed in Example 3 of [4] based on a 2-cover of the Tanner code. There are however no results reported in [4] for covers of larger degree.

For comparison, we have also included in Fig 2, the FER of a random 5-lifting of the Tanner code. As can be seen, the error floor performance of this code is significantly worse than that of the designed 5-lifting. In particular, the slope of the random lifting is just 4 versus 5 for the designed lifting.

The code rates of the designed $N$-liftings are: 0.4065, 0.4043, 0.4032, and 0.4026, for $N = 2$ to 5, respectively. The small decrease in the code rate by increasing the degree of lifting is a consequence of the fact that the original parity-check matrix of the Tanner code is not of full rank. The rate of the Tanner code itself is 0.4129.

It is also worth noting that while the girth of the $N$-liftings, $N = 2, 3, 4$, remains the same as that of the Tanner code (i.e., $g = 8$), the girth is increased to 10 for the 5-lifting.

**Example 4:** In this example, we consider a regular (504, 252) code from [5] decoded by Gallager B algorithm. The dominant trapping sets in this case have critical number 3 and include (3, 3), (4, 2), and (5, 3) trapping sets among others. The IES algorithm is used to design cyclic $N$-liftings of this code for $N = 2$ to 6. The FER results of the liftings and the base code are reported in Fig. 3. Again, the performance of the 2-lifting is similar to that of the code designed in [4]. All (3, 3) trapping sets are eliminated in the 2-lifting, but the survival of other trapping sets with critical number 3 keeps the minimal critical number at 3, and thus no change of FER slope compared to the base code is attained. Increasing $N$ to 3, however, eliminates all the trapping sets with critical number 3 and changes the slope of the FER to 4. The dominant trapping sets in this case are $(4, 4)$ sets. Further increase of $N$ to 4 and 5 only reduces the number of $(4, 4)$ trapping sets and thus results in a downward shift of the FER curve. For $N = 6$, the IES algorithm can eliminate all the $(4, 4)$ trapping sets, and thus increases the slope of the FER curve to 5. The dominant trapping sets in this case are $(5, 5)$ sets.

For comparison, in Fig. 3, we have also shown the performance of a random 6-lifting of the (504, 252) code. As can be seen, the performance of this code in the error floor region is far poorer than that of the designed 6-lifting. In particular, the slope of the FER curve for this code is only 3 versus 5 for the designed code.

In this example, the parity-check matrix of the base code is full rank, and all the liftings have the same rate of 0.5 as the
random code (same block length and degree distributions) are also presented in Fig. 4. One can see that at high SNR values, the designed code performs far superior to the base code and the random code. In particular, the designed code shows no sign of error floor for FER values down to about $10^{-8}$, and its FER decreases at a much faster rate compared to the base code and the random code.

REFERENCES

[1] S. K. Chilappagari, S. Sankaranarayanan, and B. Vasic, “Error floors of LDPC codes on the binary symmetric channel,” Proc. Int. Conf. Commun. (ICC 2006), Istanbul, Turkey, Jun. 2006, pp. 1089 - 1094.

[2] M. Franceschini, G. Ferrari, and R. Raheli, “Does the performance of LDPC codes depend on the channel?” IEEE Trans. Commun., vol. 54, no. 12, pp. 2129-2132, Dec. 2006.

[3] Y. Han and W. E. Ryan, “Low-floor decoders for LDPC codes,” IEEE Trans. Comm., vol. 57, no. 6, pp. 1663 - 1673, June 2009.

[4] M. Ivkovic, S. K. Chilappagari, and B. Vasic, “Eliminating trapping sets in low-density parity-check codes by using Tanner graph covers,” IEEE Trans. Inform. Theory, vol. 54, no. 8, pp. 3763-3768, Aug. 2008.

[5] D. J. C. Mackay, Encyclopedia of Sparse Graph Codes [Online]. Available: http://www.soe.ucsc.edu/ tanner/isctaGrpStrLDPC.pdf.

[6] T. Richardson, “Error floors of LDPC codes,” Proc. 41st Annual Allerton Conf. Commun., Control and Computing, Monticello, IL, Oct. 2003, pp. 1426 - 1435.

[7] T. Richardson and R. Urbanke, Modern Coding Theory, Cambridge University Press, 2008.

[8] R. M. Tanner, D. Sridhara, and T. Fuji, “A class of group-structured LDPC codes,” Proc. ISCTA 2001, Ambleside, England, 2001. [Online]. Available: http://www.soe.ucsc.edu/ tanner/isctaGrpStrLDPD.pdf.

[9] D. Vukobratovic and V. Senk, “Evaluation and design of irregular LDPC codes using ACE spectrum,” IEEE Trans. Comm., vol. 57, no. 8, pp. 2272 - 2278, Aug. 2009.

[10] C.-C. Wang, “Code annealing and the suppressing effect of the cyclically lifted LDPC code ensembles,” IEEE Information Theory workshop 2006, Chengdu, China, Oct. 2006, pp. 86 - 90.

[11] H. Xiao and A. H. Banaihashemi, “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.

[12] H. Xiao and A. H. Banaihashemi, “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, June 2009.