Protograph-Based Raptor-Like LDPC Codes

Tsung-Yi Chen, *Member, IEEE,* Kasra Vakilinia, *Student Member, IEEE,* Dariush Divsalar, *Fellow, IEEE,* and Richard D. Wesel, *Senior Member, IEEE*
tsungyi.chen@engineering.ucla.edu, vakiliniak@ucla.edu, Dariush.Divsalar@jpl.nasa.gov, wesel@ee.ucla.edu

***Abstract*—This paper proposes protograph-based Raptor-like (PBRL) codes as a class of rate-compatible (RC) LDPC codes for binary-input AWGN channels. As with the Raptor codes, exclusive-OR operations on precoded bits produce additional parity bits providing extensive rate compatibility. Unlike Raptor codes, each additional parity bit in the protograph is explicitly designed to optimize the density evolution threshold. During the lifting process, ACE and CPEG constraints are used to avoid undesirable graphical structures. Some density-evolution performance is sacrificed to obtain lower error floors, especially at short blocklengths.**

Simulation results are shown for information block sizes of $k = 1032$ and 16384. For a target frame error rate of 10^{-5}, at each rate the $k = 1032$ and 16384 code families perform within 1 dB and 0.4 dB of both the Gallager bound and the normal approximation, respectively. The 16384 code family outperforms the best known standardized code family, the AR4JA codes. The PBRL codes also outperform DVB-S2 codes that have the advantages of longer blocklengths and outer BCH codes. Performance is similar to RC code families designed by Nguyen et al. that do not constrain codes to have the PBRL structure and involve simulation in the optimization process at each rate.

***Index Terms*—Channel coding, Low-Density Parity-Check Codes.**

I. Introduction

THIS paper provides a general technique for constructing families of rate-compatible (RC) low-density parity-check (LDPC) codes and provides numerical results showing excellent performance.

A. Rate-Compatible Channel Codes

RC punctured convolutional (RCPC) codes and RC punctured turbo (RCPT) codes are among the most popular RC channel codes used in incremental redundancy (IR) systems [1]. For both RCPC [2] and RCPT [3] codes, a collection of RC puncturing patterns are often carefully designed to ensure good error rate performance across the family of rates despite the RC constraint. Liu and Soljanin showed that RCPT code performance degrades significantly when the punctured code rate is above a threshold [4].

R. D. Wesel, K. Vakilinia and D. Divsalar are with the Department of Electrical Engineering, University of California, Los Angeles, Los Angeles, CA 90095, USA. T.-Y. Chen is with SpiderCloud Wireless.

This material is based upon work supported by the National Science Foundation under Grant Number 1162501. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. This research was carried out in part at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with NASA. Parts of this work were presented at the Global Communications Conference 2011 and the International Conference on Communications 2012.

Low-Density Parity-Check (LDPC) codes were first introduced by Gallager in his dissertation in 1963 [5]. Tanner [6] introduced the representation of LDPC codes as bipartite graphs. MacKay [7] showed that LDPC codes provide capacity-approaching performance similar to turbo codes [8] when decoded by a message-passing algorithm with soft information.

Irregular LDPC codes have parity-check matrices that have a variety of column weights and row weights. By optimizing the variable-node and check-node degree distributions, Luby et al. [9] showed that properly constructed irregular LDPC codes can achieve rates even closer to capacity than regular codes, which have a single column weight and a single row weight. Richardson, Shokrollahi and Urbanke [10] created a systematic method called density evolution to design and analyze the optimal degree distribution of LDPC codes based on the assumption that the blocklength can be infinitely long.

Inspired by the capacity-approaching performance of LDPC at individual rates, approaches including [11]–[25] construct RC LDPC code families.

Any RC code may be considered as a low-rate code that is punctured to produce higher rates, but design approaches can be distinguished as to whether the design begins with the lowest rate or the highest rate. In [11]–[16], the lowest rate "mother code" is designed first and then puncturing is designed to obtain the higher rates. However, as observed in [17] and elsewhere, finite-length LDPC RC code families obtained in this way suffer from a larger performance degradation than punctured turbo codes at high rates.

To avoid this problem at high rates, the method of *extension* designs the high-rate code first and then the lower rate codes are designed on top of that foundation. This approach has been explored in articles including [17]–[25]. Our paper uses the extension approach. Also, as in [12], [17], [19] our design approach focuses on maximizing the density-evolution/EXIT threshold as a design objective.

As in [18], [24], our RC code family is designed by extending a protograph. Thorpe [26] introduced protograph-based LDPC codes, or protograph codes. These codes were studied extensively by Divsalar et al. [27]. The design of protograph codes begins with the construction of a relatively small bipartite graph called the protograph. After using density evolution to properly design the protograph, a copy-and-permute operation, often referred to as "lifting", is applied to the protograph to obtain larger graphs of various sizes, resulting in longer-blocklength LDPC codes. As part of lifting, the variable-node connections of the edges of the same type are permuted among the protograph replicas. Even when the protograph has parallel edges, lifting can ensure that the final

code does not have parallel edges. A detailed discussion on parallel edges in protographs can be found in [27], which also discusses how protograph codes facilitate efficient decoder implementation in hardware.

In contrast to [18] and [24], this paper and its precursors [22], [23], restrict the code familiy to have the basic structure of Raptor codes [28]. Constraining the design in this way makes the construction and optimization manageable while still providing outstanding performance and extensive rate-compatibility. Introduced by Luby [29] and Shokrollahi [28] respectively, LT codes and Raptor codes share many similarities with LDPC codes and are shown to achieve the capacity of binary erasure channel (BEC) universally. Etesami et al. [30] explored the application of Raptor codes to binary memoryless symmetric channels and derive various results, including the fact that Raptor codes are not universal except for the BEC. Note that results on Raptor codes such as [28] and [30] rely heavily on the assumption of large information blocks.

Following the Raptor-like structure proposed in [22], Nitzold et al. applied spatial coupling [31] to improve the threshold, These spatially coupled codes can be viewed as Raptor-like LDPC convolutional codes [32]. Nitzold et al. focused on the analysis of the asymptotic decoding threshold where the rate loss due to the time-spreading number L is negligible.

Recent work by Nguyen and Nosratinia [25] considered a general structure for extending RC protograph codes but ends up proposing an example protograph that has the Raptor-like structure as first proposed in [22] and [23].

B. Organization and Main Contributions

This paper proposes a class of RC LDPC codes called protograph-based raptor-like (PBRL) LDPC codes. The construction and optimization of PBRL codes are discussed and simulation results are presented. Comparing to existing codes in the literature (e.g. AR4JA codes in [33], DVB-S2 codes in [34] and protograph codes in [24]), PBRL codes show outstanding performance while providing extensive rate-compatibility.

This paper is based on the precursor conference papers [22], [23]. The PBRL approach was introduced in [22] and used to design $k = 192$ RC code families. In [22] a PBRL code family with multiple parallel edges to the punctured node produced thresholds within 0.34 dB of capacity at every rate, but the best simulated performance was achieved by a code family with higher thresholds that had only one pair of parallel edges connected to the punctured node. This is an example of the well-known inability of threshold alone to guide short-blocklength design.

In [23] the PBRL approach is applied to the design of long-blocklength code families, providing a $k = 16368$ LDPC code family as an example. Also, reciprocal channel approximation (RCA) replaces the standard density evolution of [22] to provide a fast and accurate approximation of the density evolution threshold to speed up the optimization process. In [23] the punctured node is connected to every check node in the IR part at least once to provide low thresholds. As in [22], parallel edges to the punctured node from the IR check nodes are kept to a minimum, in this case two check nodes had a pair of parallel edges to the punctured node. In both [22] and [23] the circulant progressive edge growth (CPEG) algorithm [35] was used for lifting.

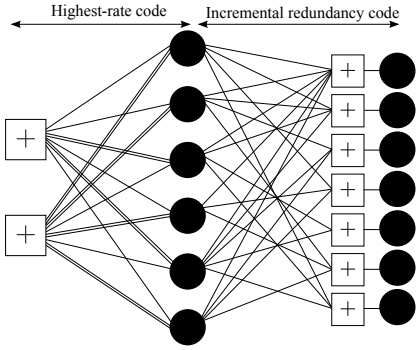

Fig. 1. Protograph for a PBRL code with a highest-rate code (HRC) with rate $2/3$ followed by an incremental redundancy code (IRC) that uses only degree-one variable nodes. The IRC provides lower rates as more of its variable nodes are included, starting from the top.

This paper unifies and extends the material in the precursor conference papers [22] and [23]. The contributions beyond the precursor conference papers include the description of a modified RCA algorithm for use with single-check variable nodes (which are essential to PBRL codes), the design of new code families for $k = 1032$ and $k = 16384$ that use lifting that incorporates both CPEG and Approximate Cycle Extrinsic Message Degree (ACE) constraints, comparison with finite-length performance bounds and approximations, and a careful discussion of how constraints on the connections can be used to sacrifice some threshold performance to avoid problematic error floors. These constraints become less stringent for longer blocklengths.

The rest of the paper is organized as follows: Sec. II presents the PBRL code structure. Sec. III provides the design procedure to construct PBRL codes. Sec. IV constructs example PBRL code families and presents analysis and simulation results. Finally, Sec. V concludes the paper.

II. Protograph-Based Raptor-Like LDPC Code

This section introduces the structure, encoding, and decoding of PBRL codes.

A. The Structure of PBRL Codes

Fig. 1 shows the protograph structure of a PBRL code. This protograph consists of two parts: (1) a highest-rate code (HRC) protograph and (2) an incremental redundancy code (IRC) protograph. The IRC provides lower rates as more of its variable nodes are transmitted, starting from the top.

The bipartite graph of a PBRL protograph can be described by a protomatrix, which is the parity-check matrix of a protograph. Let $\mathbf{0}$ be the all-zeros matrix and $\mathbf{I}$ be the identity matrix with the appropriate dimensions. The protomatrix of the protograph shown in Fig. 1 is given as

$$H = \begin{bmatrix} H_{\text{HRC}} & \mathbf{0} \\ H_{\text{IRC}} & \mathbf{I} \end{bmatrix} \tag{1}$$

where the HRC parity-check matrix H_{HRC} and IRC parity-check matrix H_{IRC} are given as

$$H_{\text{HRC}} = \begin{bmatrix} 1 & 1 & 2 & 1 & 2 & 1 \\ 2 & 2 & 1 & 2 & 1 & 2 \end{bmatrix} \tag{2}$$

and

$$H_{\text{IRC}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}. \tag{3}$$

Hence to fully express the protomatrix of a PBRL protograph it is enough to specify H_{HRC} and H_{IRC}.

The overall protograph determined by HRC and IRC protographs is lifted to produced the final code. We use circulant matrices in the lifting process. After lifting, the HRC code is structurally identical to the precode in a Raptor code. Similarly. the degree-one variable nodes of the second part can be efficiently encoded as modulo-2 sums of the precode symbols in a manner similar to the LT code in a Raptor code.

The PBRL structure resembles a Raptor code, but with some important differences: 1) the variable nodes of the HRC are transmitted for a PBRL code but the precode variable nodes are not transmitted for a Raptor code 2) the connections that create a variable node in the LT part are random and potentially infinite for a Raptor code, but the connections that create a variable node in the IRC are deterministic, finite in number, and carefully designed for a PBRL code, and 3) the decoding of PBRL codes is different from that of Raptor codes as we discuss in the next subsection.

The overall PBRL protograph may be considered as the concatenation of the highest-rate LDPC code having parity-check matrix H_{HRC} and the low-density generator matrix (LDGM) code [36] having the systematic generator matrix

$$G = \begin{bmatrix} I & H_{\text{IRC}}^T \end{bmatrix}. \tag{4}$$

LDGM codes by themselves are known generally to have high error floors and poor asymptotic minimum distance [7], [36], [37]. However, as observed in [36], [37], concatenating an LDGM code with an outer code (the HRC in our case) mitigates the high error floor problem.

The IRC is created by the addition of variable nodes that are exclusively degree-one, which correspond to the identity matrix in (1). Such degree-one nodes are generally associated with poor error-floor performance. However, rate compatibility forces the sub matrix that is an identity matrix in (1) to be at least lower triangular. Any full-rank choice of the overall matrix in (1) that incorporates this lower triangular component will be range-equivalent to a matrix for which the lower triangular part is the identity matrix. Thus, the possible codes (i.e. the possible sets of valid codewords) is not affected by restricting the variable nodes in the IRC have degree one.

The remaining concern is that these degree-one variable nodes might introduce bad performance (i.e. a high error floor) under iterative decoding. As observed in [36], [37] the error floor problem can be resolved by concatenation with another code, as we have done by concatenating with the HRC. Moreover, increasing the degree of these variable nodes can introduce small cycles which must be addressed during lifting.

With the PBRL approach, each additional degree-one variable node boosts the reliability of the variable nodes with which it connects through its only neighboring check node (as described by the corresponding row of H_{IRC}). This is similar to the active feedback incremental redundancy approach used in [38], [39]. A PBRL code family is thus a well-designed highest-rate code combined with additional variable nodes that provide efficient incremental redundancy to that highest-rate code such that each additional variable node in the protograph lowers as much as possible the SNR at which decoding succeeds.

B. Decoding and Encoding of PBRL Codes

A traditional Raptor code encodes the information bits to produce the precoded symbols and then encodes the precoded symbols with an LT code. In the case of an LDPC precode used with an LT code, the decoding often proceeds as follows: the decoder first performs BP decoding on the LT code and then performs BP decoding on the precode using the results of the completed LT decoding.

In [40], the authors note that because of the two-stage decoding, the complexity of Raptor codes is higher than that of RC LDPC codes. The PBRL code family always transmits the output symbols of the precode and has deterministic connections in the LT code. These two properties facilitate joint decoding of the HRC code part and the IRC code part, an idea that first appeared in [41] for Raptor codes.

For traditional Raptor codes that do not transmit the precode symbols and use randomized encoding, the initial transmission of the LT symbols may not contain enough information for BP decoding to succeed even in a noiseless setting. Always transmitting the precode symbols allows PBRL codes to have the potential for successful decoding at the initial transmission.

For high-rate PBRL codes, the decoder can deactivate those check nodes in the IRC part for which the neighboring degree-one variable node is not transmitted, offering significant complexity reduction.

Encoding of PBRL codes is as efficient as of Raptor codes: after encoding the precode, the encoding of the IRC part only involves exclusive-or operations on the precode output symbols. For efficient encoding of the precode, see the discussion in [27] on efficient encoding of protograph codes.

The Raptor-like structure is very restrictive. One might expect the structural constraints to limit performance as compared to less-restrictive structures for extending LDPC codes. One of the main conclusions of our paper is that despite these constraints, we obtain Raptor-like protographs with very low iterative decoding thresholds. By careful design of the protograph and the lifting process, the resulting finite-length codes can outperform existing RC LDPC codes that have been designed without the constraint of a Raptor-like structure.

III. Optimization of PBRL LDPC Codes

This section presents optimization procedures for finding the HRC and IRC components that comprise a good PBRL code

family, considering both short and long blocklengths. Belief propagation (BP) decoding is assumed. The optimization criteria for both long and short-blocklength codes are primarily based on minimizing the iterative decoding threshold at each rate while enforcing constraints on the connections to avoid problematic error floors. These constraints are more stringent for short blocklength designs than for long blocklength designs.

To simplify the threshold computations, we use a modified version of the reciprocal channel approximation (RCA) algorithm. After presenting the modified RCA in subsection III-A, subsection III-B describes the design of the HRC and subsection III-C describes the design of the IRC.

A. Density Evolution with Reciprocal Channel Approximation

The asymptotic *iterative decoding threshold* [42] characterizes the performance of the ensemble of LDPC codes that share a specified protograph. This threshold indicates the minimum SNR required to transmit reliably, meaning that the expected bit error rate goes to zero with that ensemble of codes as the blocklength grows to infinity. Note that this may not coincide with the block error rate going to zero.

Computing the exact iterative decoding threshold for BI-AWGN requires significant computation. The RCA [43] [27] provides a fast and accurate approximation to the density evolution algorithm originally proposed by Richardson et al. [42] [10]. Experimental results [27], [43] show that the deviation of RCA from the exact density evolution threshold is less than 0.01 dB.

The RCA for BI-AWGN channel uses a single real-valued parameter s, the SNR, to approximate the density. Define the reciprocal SNR as $r \in \mathbb{R}$ such that $C(s) + C(r) = 1$ where $C(s)$ is the capacity of the BI-AWGN channel with SNR s. For computational simplicity and numerical precision we prefer to express $C(s)$ as

$$C(s) = 1 - \int_{-\infty}^{\infty} \log_2\left(1 + e^{-(2\sqrt{2s}u + 2s)}\right) \frac{e^{-u^2}}{\sqrt{\pi}} du, \quad (5)$$

which is obtained from [44, (15)] through a change of variables. The self-inverting reciprocal energy function

$$R(s) = C^{-1}\left(1 - C(s)\right) \quad (6)$$

in [43] transforms between s and r: $r = R(s)$ and $s = R(r)$.

Let s_{chl} be the channel SNR, s_e be the message passed along an edge e from a variable node to a check node and r_e be the message passed along an edge e from a check node to a variable node. Let E_c be the set of edges that connect to a check node c and E_v be the set of edges that connect to a variable node v.

RCA first initializes the message s_e to 0 if the edge e is connected to a punctured variable node and to s_{chl} otherwise. For all edges e in the graph, RCA then computes a sequence of messages $(s_e^{(n)}, r_e^{(n)})$, $n = 0, \ldots, N$ where N is the maximum number of iterations.

The original density evolution [42] determines the threshold based on the pdfs of all outgoing messages from variable nodes. Aiming to approximate this density evolution, RCA [43] determines the decoding threshold s_{th} as the minimum s_{chl} such that $s_e^{(N)} > T$ for all edges e in the graph, where T is a stopping threshold.

Note that for the edge connecting to a degree-one variable node, $s_e = s_{\text{chl}}$ regardless of the number of iterations. For this reason, the original RCA does not work if the graph contains degree-one variable nodes. We use a slightly modified version of the RCA that focuses on the overall reliability of each variable node S_v, rather than the reliability of every edge s_e. This modification allows computation of a meaningful decoding threshold for protographs with degree-one variable nodes. Letting N be the maximum number of iterations and $T > 0$ be the stopping threshold, the modified RCA is summarized as follows:

Algorithm 1 (Modified Reciprocal Channel Approximation): Let $f_{\text{RCA}}(s_{\text{chl}}) : \mathbb{R} \mapsto \{0, 1\}$ be a binary-valued function that returns 1 if s_{chl} is higher than s_{th} and 0 otherwise. To determine its output, the modified RCA computes the sequence $\left(s_e^{(n)}, r_e^{(n)}\right), n = 0, \ldots, N$, for all edges e in the graph. The computation of the sequence is given as follows:

0) For edges e connected to punctured variable nodes, set $s_e^{(0)} = 0$. For all other edges set $s_e^{(0)} = s_{\text{chl}}$.
1) For $n = 1, \ldots, N$, generate $(s_e^{(n)}, r_e^{(n)})$ as follows:

$$r_e^{(n+1)} = \sum_{i \in E_c \setminus e} R\left(s_i^{(n)}\right), \quad (7)$$

$$s_e^{(n+1)} = s_e^{(0)} + \sum_{i \in E_v \setminus e} R\left(r_i^{(n)}\right). \quad (8)$$

2) At each iteration compute $S_v^{(n)}$ as

$$S_v^{(n)} = S_v^{(0)} + \sum_{e \in E_v} s_e^{(n)}, \quad (9)$$

for all variable nodes v in the graph, where $S_v^{(0)}$ is

$$S_v^{(0)} = \begin{cases} 0 & \text{if } v \text{ is punctured} \\ s_{\text{chl}} & \text{otherwise} \end{cases}. \quad (10)$$

3) Let $S^*(n) = \min\left\{S_v^{(n)} : \forall v \text{ in the graph}\right\}$. At each iteration, compute $S^*(n)$. If $S^*(n) > T$, set $f_{\text{RCA}}(s_{\text{chl}}) = 1$ and stop; otherwise, set $f_{\text{RCA}}(s_{\text{chl}}) = 0$ and continue to the next iteration, stopping if $n = N$.

By monotonicity of the threshold [45] we can perform bisection search at the desired level of precision using the function f_{RCA}. To increase computation speed, a lookup table and linear interpolation are used to compute $R(s)$.

B. Optimizing the Highest-Rate Code of a PBRL Family

Primarily, the HRC photograph is simply the protograph of a good code at the desired rate. Our design follows the work of Divsalar et al. [27, Sec.III], with some additional optimization through RCA analysis and LDPC code simulation.

One main conclusion of [27] as also observed in [46]–[49] is that protograph ensembles with a minimum variable-node degree of 3 or higher are guaranteed to have linear minimum distance growth with the blocklength. This observation for irregular LDPC codes was also recognized in [50] and [51].

Divsalar et al. [27] also noted that judiciously adding a few variable nodes with degree 2 or even degree 1 improves the protograph's threshold. A small number of variable nodes with degree less than 3 can be present in the protograph while still retaining the linear minimum distance growth property as long as there is no loop comprised entirely of degree-2 nodes. The HRC protographs designed in Secs. IV-A and IV-B ensure the linear minimum distance growth property.

As explained in [27], puncturing a node in the HRC can improve the threshold performance. A variant of PBRL codes for which the protograph of the HRC has at least one punctured (untransmitted) node is referred to as the Punctured-Node PBRL (PN-PBRL) codes in [22]. In the current paper, all of our examples belong to the class of PN-PBRL codes due to their superior performance. We will refer to these codes simply as "PBRL codes", but they are also PN-PBRL codes since they have a punctured node.

We identify the HRC for a family of PBRL codes by a triplet of parameters (n_v, n_c, n_p): number of variable nodes n_v, number of check nodes n_c and number of punctured nodes n_p. For the code families designed in Secs. IV-A and IV-B, $n_p = 1$. Assuming that the matrix is full rank, the set of possible rates of this family of RC codes is given by $(n_v - n_c)/(n_v - n_p + i)$ for all integers $i > 0$.

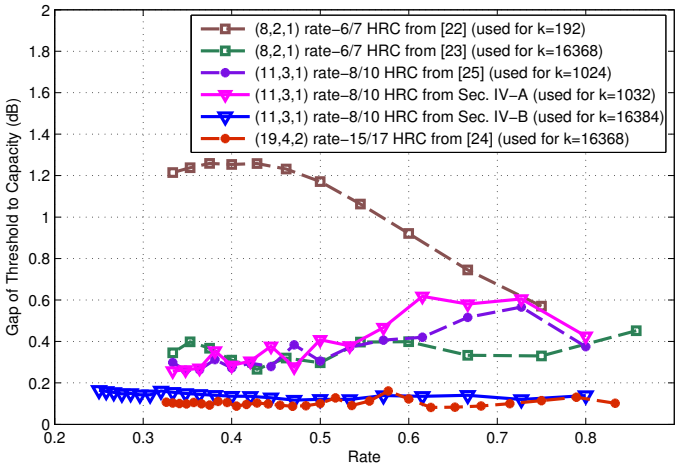

Fig. 2. Gap of protograph threshold to binary AWGN capacity for each rate in the PBRL code family for four different PBRL protographs. These four protographs correspond to the following codes: the $k = 192$ PBRL family featured in [22], the $k = 16368$ PBRL family featured in [23], the $k = 16384$ PBRL family whose protograph is designed in Sec. IV-B and the protograph for $k = 1032$ PBRL families designed in IV-A. These thresholds are computed with modified RCA algorithm presented in Sec. III-A. As a benchmark for rate-$1/2$ codes, recall that the corresponding gap in threshold for a $(3, 6)$ regular LDPC code is 0.913 dB from the BI-AWGN capacity of 0.187 dB.

C. Optimizing the IRC Protograph of a PBRL Family

Optimization of the IRC part includes consideration of linear minimum distance growth, the threshold values obtained at each rate, and the ability to realize the performance those threshold values predict by avoiding high error floors.

At the heart of IRC optimization is a greedy algorithm to optimize the threshold of the protograph of each rate given the protograph of the previous (higher) rate. This optimization algorithm is summarized as follows:

Algorithm 2: Given an HRC protograph, the protograph for the IRC of a PBRL code family is obtained by the following steps:

1) Add a new check node and a new degree-one variable node connected to the new check node.
2) Use the modified RCA to find the connections between the new check node and the variable nodes of the HRC that give the lowest threshold under specified constraints on the connections. These constraints can vary with the intended blocklength but always include the constraint that each new row of H_{IRC} has at least two edges.
3) Stop when the lowest rate desired has been obtained. Otherwise, go to step 1.

PBRL code families retain linear minimum distance growth by requiring that each new row in H_{IRC} has at least two edges. As long as the HRC has linear minimum distance growth, ensuring non-trivial connections for each new check node in the IRC in this way preserves linear minimum distance growth for all rates. This follows from the analysis of linear minimum distance growth in the context of LDPC codes concatenated with LDGM codes in [36], [37], [52], [53].

There are two key features that dramatically improve protograph thresholds at low rates. The first is that parallel edges improve the threshold and the second is that the punctured node of the precode should connect to all (or almost all) of check nodes in the IRC part with at least a single edge. Fig. 2 illustrates the importance of these features by comparing two PBRL code families that both use $(8, 2, 1)$ HRC protographs and eleven degree-one variable nodes in the IRC protograph. The protograph family from [22] connects the punctured node to seven of the eleven check nodes connected to degree-one IRC variable nodes. Only the first such check node is connected to the punctured node with a parallel edge. In contrast, the protograph family from [23] connects the punctured node to all eleven check nodes connected to degree-one IRC variable nodes, Also, for this family the first two such check nodes are connected to the punctured node with parallel edges. Fig. 2 shows the dramatic threshold improvement of the protograph family of [23] over the one of [22].

Increased connectivity to the punctured node through single and parallel edges and the use of parallel edges elsewhere in the IRC lowers the thresholds significantly for the lower rates. The threshold improvement, however, diminishes as the number of parallel edges increases. Moreover, even at long blocklengths lifting cannot overcome the resulting damage to the graphical structure of the code if the parallel edges are added too aggressively. Thus, an important aspect of IRC design is controlling the number of parallel edges to give the best possible thresholds for which lifting can successfully translate the excellent thresholds into excellent frame error rate (FER) and bit error rate (BER) performance. We will explore this trade-off in Secs. IV-A and IV-B to see how differently it plays out in the context of short-blocklength and long-blocklength codes.

IV. Design Examples

This section presents design examples. Subsection IV-A presents a short-block-length design example. Subsection IV-B presents a long-block-length design example. Subsection IV-C provides the gaps of SNR to the Shannon limit of SNR for reliable communication for FER 10^{-5} for PBRL families designed in this paper and previously in [22], [23]. The description of the codes discussed here can be found on the UCLA CSL website[1].

A. A Short-Blocklength PBRL Design Example

This subsection provides an example PBRL protograph family designed for information blocklength $k = 1032$ and compares it to a comparable code from [25]. The HRC uses the $(10, 2, 1)$ protograph shown below:

$$H_{\text{HRC}}^{(1032)} = \begin{bmatrix} 3 & 1 & 3 & 1 & 3 & 1 & 3 & 1 & 2 & 1 \\ 1 & 3 & 1 & 3 & 1 & 3 & 1 & 2 & 1 & 2 \end{bmatrix}. \quad (11)$$

The IRC adds 15 degree-one check nodes at the lowest rate. The first variable node in the HRC is always punctured, and the first degree-one variable node in the IRC is always transmitted. Thus the protograph of our highest rate code is effectively (11,3,1) giving a range of rates of the form $8/(9+i)$ from $8/10$ to $8/24$. The H_{IRC} obtained from Algorithm 2 (with constraints we discuss below) is

$$H_{\text{IRC}}^{(1032)} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 2 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 2 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 2 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \quad (12)$$

For this code design, we restrict our attention to potential HRC protographs with variable nodes having either degree 3 or degree 4. This restriction ensures linear minimum distance growth according to [27].

For this short-blocklength HRC, we address the trade-off between threshold and error floor by varying the number of degree-4 nodes in the HRC photograph. Using all degree-3 variable nodes provides the best threshold. Increasing the number of degree-4 variable nodes in the HRC lowers the error floor but increases the threshold as shown in Fig. 3. With fewer than 6 degree-4 variable nodes, the ACE-and-CPEG lifted codes for $k = 1032$ result in high error floors. As shown in (11), our H_{HRC} has seven degree-4 variable nodes, the smallest number of degree-4 variable nodes that yielded good error-floor performance.

For the IRC design, we require every check node in the IRC to connect to the punctured variable node of the HRC. These connections alternate between a single edge and a parallel edge (alternating 1s and 2s in the first column of (12)). In [23] two of the eleven check nodes in the IRC (the two associated with highest two rates) connected to the punctured node with parallel edges. As discussed in Sec. III-C, both [22], [23] show that further threshold improvement can be obtained with more parallel edges. However, use of parallel edges is limited in [22], [23] because the CPEG-lifted codes displayed high error floors when more parallel edges were used.

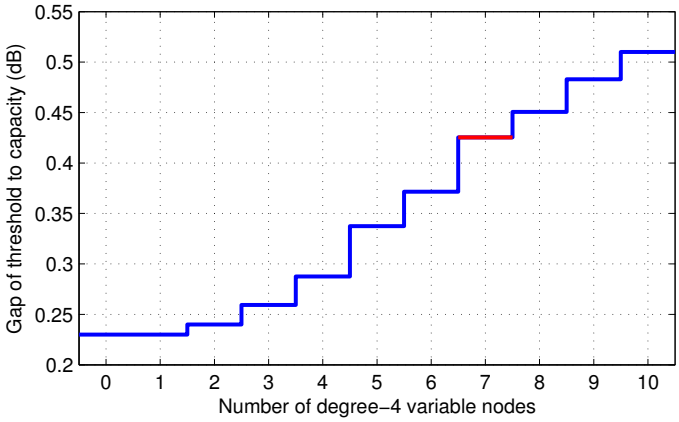

Fig. 3. Gap of threshold from binary AWGN capacity vs. number of degree-4 variable nodes for potential highest-rate-code protographs based on a (10,2,1) protograph with all variable nodes having degrees 3 or 4. Gap is shown for the rate-0.8 (11,3,1) protograph where the additional row has two parallel edges connected to the punctured node and the remaining connections are optimized to minimize the threshold under the constraint that each connection is either a single edge edge or not present. The result with seven degree-4 variable nodes corresponds to the protomatrix in (11).

The PBRL code families in [22] and [23] used CPEG alone for lifting. Combining CPEG with the ACE algorithm produces a lifting that better avoids damage to the graphical structure so that the alternating parallel edges to the punctured node could be used. This makes sense because extrinsic connections have an important influence on the error floor and the ACE algorithm [54] ensures that all small cycles have a minimum number (η) of these connections.

For this short-blocklength design, combined ACE-and-CPEG lifting allowed half the connections to the punctured node to be parallel edges, but we found that using parallel edges elsewhere in the IRC led to error floors for these short blocklengths.

We use two-step lifting as in [27] and employ CPEG and ACE algorithm jointly to lift the lowest rate code. The pre-lifting step has a lifting number of 3 to remove the parallel edges. The lifting number in the second stage is 43, giving information blocklength of $k = 1032$.

CPEG enforces a girth of 6. We used $\eta = 12$ and $d_{\text{ACE}} = 5$ for the ACE algorithm applied to the lowest rate. A pair (d_{ACE}, η) implies that every cycle consisting of up to d_{ACE} variable nodes, i.e. every cycle of length up to $2d_{\text{ACE}}$, is connected to at least η extrinsic check nodes. In other words, all the cycles with length 10 or less have at least 12 extrinsic connections for the lowest-rate (rate-1/3) code.

Fig. 4 shows that CPEG + ACE lifting for the protographs given in (11) and (12), provides larger η values than CPEG

[1] http://www.seas.ucla.edu/csl/#/publications/published-codes-and-design-tools

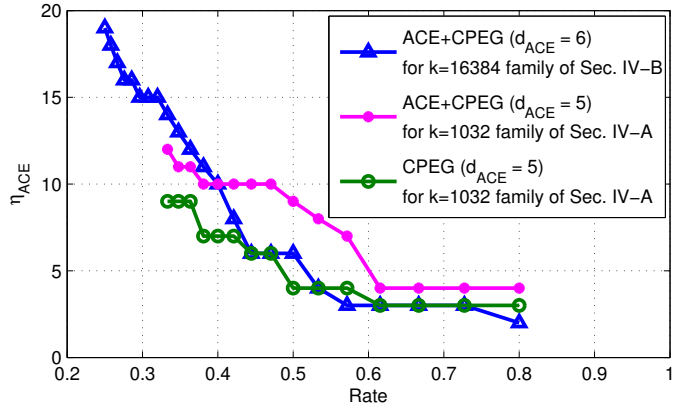

Fig. 4. η_{ACE} for the ACE+CPEG-lifted families of codes for $k = 16384$ and $k = 1032$ and CPEG-only lifted family for $k = 1032$.

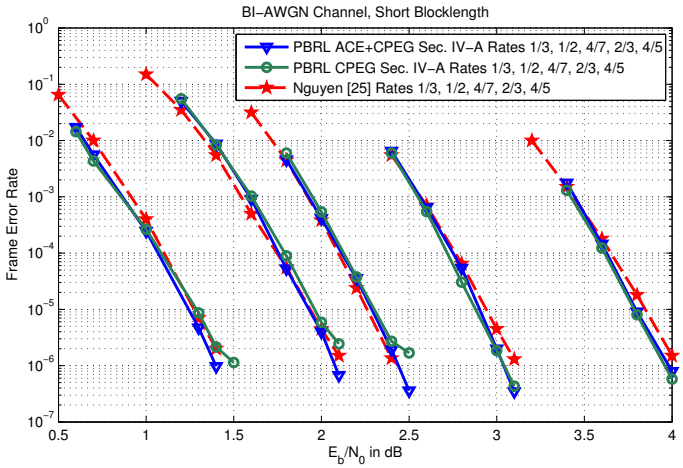

Fig. 5. Comparison of frame error rates between PBRL codes and the codes in [25] with $k = 1032$. The protographs for the PBRL code family are from (11) and (12).

alone especially at the low end of our rate range. In Fig. 5, comparing the PBRL CPEG-only curves with the ACE+CPEG curves in the lower-rate range shows how the higher η values achieved by ACE+CPEG translate to an improved error floor.

Nguyen and Nosratinia constructed a family of RC protograph codes in [25] with short blocklengths ($k = 1024$). The design process in [25] featured optimization based on EXIT analysis similar to the current paper, but did not constrain the code to have the PBRL structure. The fact that this code family has the PBRL structure even though it was not *constrained* to is a further validation of the PBRL approach.

The example in [25] uses 6 pairs of parallel edges in the IRC, all of which are connected to the punctured node in the HRC. The results agree with the observations in the current paper and also in [22] and [23] that having parallel edges connected to the punctured node improves the threshold.

The protograph family described by (11)-(12) has the same size as the family proposed in [25]. Fig. 5 compares the FER performance of the RC codes proposed in [25] and the PBRL codes described by (11)-(12). The code family described by (11)-(12) has similar performance overall to that of [25] as shown in Figs. 5 and 9. The similarity in performance is not surprising since the code family in [25] was found by a similar optimization approach optimizing over a larger family of codes that include the PBRL family as a sub-class.

However, the search in [25] required experimental testing (simulation) of numerous codes with near-optimal thresholds at each rate. In contrast, after the design of the HRC protograph, our search simply optimized the threshold for each successive rate in a greedy fashion under certain constraints on edges that were imposed to control the error floor. The surprising result is that greedy threshold optimization can produce a family that is competitive if the proper constraints on edge connections are imposed.

Now we explore the constraints we imposed on the edge connections. Recall the intuition of the IRC variable nodes providing reliability back to the original HRC. As the rate decreases we desire the check nodes of the IRC eventually to connect to all of the variable nodes in the HRC. However, density evolution threshold optimization tends to favor asymmetric degree distributions so that the rich get richer in that IRC check nodes added later tend to connect to HRC variable nodes that already connect to IRC check nodes.

Thus for short block-length codes we ensure good connectivity by requiring the IRC check nodes eventually to connect with all the HRC variable nodes as rate decreases. The specific constraints for the IRC in (11) are as follows: at rate 8/11 there can be at most two degree-3 variable nodes, at rate 8/13 there can be at most one degree-3 variable node, at rate 8/16 the minimum degree of the variable nodes is 4, at rate 8/18 there is at most one degree-4 variable node and finally at rate 8/21, the degree-4 variable nodes in HRC have at least degree five in the rate-8/21 parity-check matrix.

Fig. 2 allows comparison of the gaps from capacity for the thresholds of the protographs defined by (11)-(12) to the gaps of the $k = 1024$ protographs of [25]. The gaps are similar except at rate 8/13 where the protograph from [25] has a threshold about 0.2 dB lower than the protograph from (11)-(12) whose threshold was hampered by the constraints described above. The average gap of threshold from capacity for the protographs defined by (11)-(12) over the rate range shown in Fig. 2 is 0.3914 dB.

Our simulations use floating-point iterative decoders with a flooding schedule for message-passing unless otherwise stated. Decoding terminates early if all parity checks are satisfied before reaching the maximum number (200) of iterations.

B. A Long-Blocklength PBRL Design Example

This subsection designs a PBRL code family with blocklength $k = 16384$ and compares it to the $k = 16368$ PBRL family designed in [23] and the $k = 16384$ family designed in [24], which does not use the PBRL structure. To improve performance, this paper uses a PBRL protograph featuring an $(11, 3, 1)$ HRC protograph instead of the smaller $(8, 2, 1)$ HRC protograph used for the PBRL code family of [23]. The first variable node in the HRC is always punctured and H_{IRC} has 22 rows, each corresponding to an additional degree-one check node as described in Sec. II-A. Thus the protograph family

provides a range of rates of the form $8/(10+i)$ from $8/10$ to $8/32$. The HRC protograph for the $k = 16384$ PBRL code is

$$H_{\text{HRC}}^{(16384)} = \begin{bmatrix} 3 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 \end{bmatrix}. \quad (13)$$

Applying Algorithm 2 with constraints on the connections of the IRC check nodes to the variable nodes of the HRC produces the H_{IRC} given in (14). As in the short-blocklength example, we require every check node in the IRC to connect to the punctured variable node of the HRC. These connections alternate between a single edge and a parallel edge (alternating 1s and 2s in the first column of (14)).

Unlike [23] and in contrast to the short-blocklength design in (11)-(12), the constraints applied to Algorithm 2 for this design allow parallel edges for connections to the non-punctured nodes of the HRC. Specifically, when optimizing the IRC using Algorithm 2, each connection to a non-punctured node may have zero, one, or two edges leading to the 0, 1, and 2 values in the columns of (14). In [23] the only choices were zero and one, but the addition of ACE to the lifting process allows more parallel edges.

Nguyen et al. [24] constructed a family of RC protographs with a (19,4,2) protograph for the highest rate 15/17 and a (47,32,1) protograph for the lowest rate of 15/46. Fig. 2 shows that the protograph family defined by (13)-(14), with an (11,3,1) protograph for the highest rate of 8/10 and a (33, 25,1) protograph for the lowest rate of 8/32, gives thresholds comparable to those obtained in [24] despite the smaller protograph sizes. Fig. 2 also shows that the resulting thresholds have at least 0.1 dB of improvement at every rate compared to the thresholds of the (smaller) protographs used for the $k = 16368$ PBRL family in [23]. The gaps to capacity in Fig. 2 for the codes using (13)-(14) are all less than or equal to 0.164 dB. Furthermore, the average gap of threshold to capacity is 0.1408 dB.

We use two-step lifting as in [27] and employ CPEG and

$$H_{\text{IRC}}^{(16384)} = \begin{bmatrix} 2 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 \\ 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 \\ 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \quad (14)$$

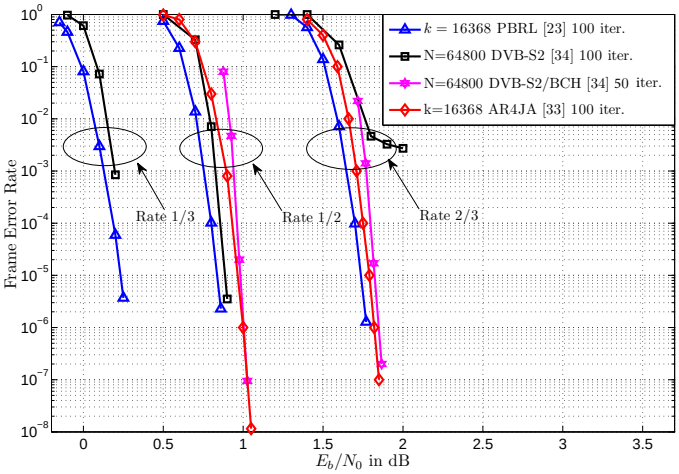

Fig. 6. This figure is from [23]. FER comparison for PBRL codes from [23], codes in the DVB-S2 standard [34], and AR4JA LDPC codes in the CCSDS standard [33].

ACE algorithm jointly to lift the lowest rate code. The pre-lifting step has a lifting number of 4 to remove the parallel edges. The lifting number in the second stage is 512, giving information blocklength of $k = 16384$. CPEG enforces a girth of 8. Fig. 4 shows the η values of the $k = 16384$ PBRL code family obtained by lifting the protographs of (13)-(14) with $d_{\text{ACE}} = 6$. Achievable η (minimum number of extrinsic edges for cycles of length 12 or smaller) increases as rate decreases.

Fig. 6 compares the FER performance of the PBRL code family in [23] at rates $1/3$, $1/2$, and $2/3$ to the DVB-S2 standard (with and without an outer BCH code) [34] and AR4JA codes from the CCSDS standard [33]. In some cases for DVB-S2 codes, only results for a maximum of 50 iterations were available. The blocklengths of the DVB-S2 codes are fixed to 64800 bits, whereas the PBRL and AR4JA codes have a fixed information length of 16368 bits and blocklengths of 32736 bits and 24552 bits for rate $1/2$ and rate $2/3$, respectively. When concatenated with the BCH code, the overall rates of DVB-S2 codes are 0.497 bits and 0.664 bits.

Fig. 6 shows that in the waterfall region, the PBRL codes from [23] outperform both the AR4JA codes and the DVB-S2 codes. Note that the PBRL codes outperform DVB-S2 even though the DVB-S2 codes have longer blocklength and benefit from concatenation with a BCH code.

Fig. 7 presents the BER and FER performance for five of the rates supported by the PBRL code family based on (13)-(14).

Fig. 8 compares the FER between our $k = 16384$ PBRL code using (13)-(14), our $k = 16368$ PBRL code from [23], and the RC protograph codes proposed in [24] at rates $1/3$ and $1/2$. The simulation results in [24] used an 8-bit quantized decoder with 200 iterations whereas our simulations used floating point decoder with 200 iterations. The simulations of the codes based on (13)-(14) in Fig. 8 used layered belief propagation, which generally performs better than flooding. Using flooding in our simulation results in less than 0.05dB of degradation at 100 iterations in our PBRL code example.

Comparing at FER around 10^{-6}, this PBRL code is 0.2 dB better than Nguyen et al.'s code at rate $1/3$ and about 0.1 dB

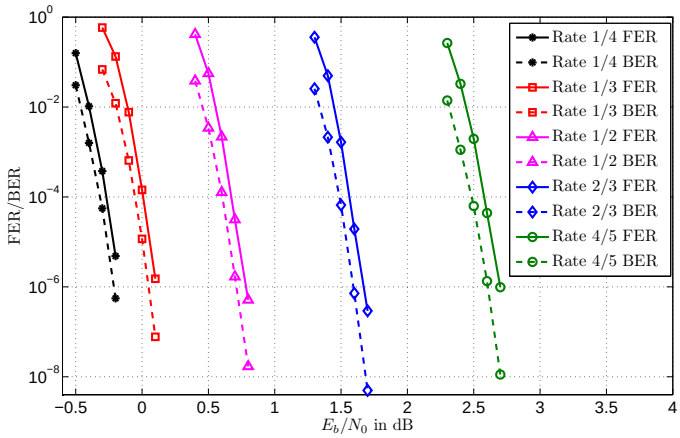

Fig. 7. Frame error rates and bit error rates for the protographs defined by (13)-(14). The lifting number is 2048, resulting in $k = 16384$.

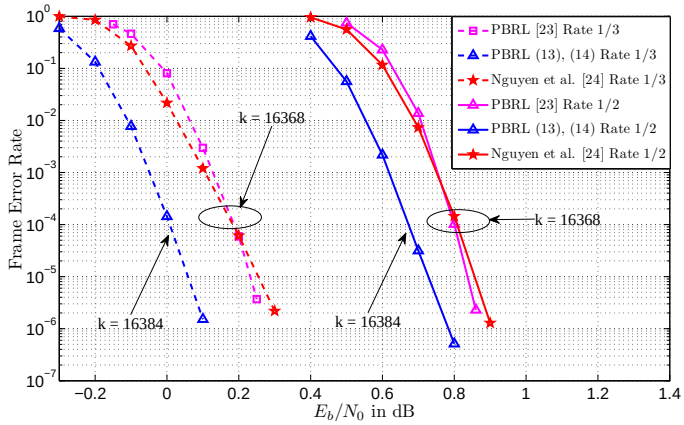

Fig. 8. Frame error rates comparison between the PBRL codes with 19 variable nodes in the protograph ($k = 16368$), the PBRL codes with 33 variable nodes in the protograph ($k = 16384$), and the RC protograph codes proposed in [24] ($k = 16368$).

better at rate-$1/2$. Note that the code family using (13)-(14) performs better even though the code family from [24] uses a larger protograph and is not constrained to the Raptor-like structure.

C. SNR Gap Analysis of PBRL Codes

Fig. 9 shows the SNR gaps in E_b/N_0 between the Shannon limit of the BI-AWGN channel and the required SNR to achieve FER of 10^{-5} in simulations of PBRL code families we have designed and the codes from [24] and [25]. Gaps from simulated PBRL codes are shown for the $k = 192$ PBRL codes from [22], $k = 1032$ PBRL codes from Section IV-A, $k = 16368$ PBRL codes from [23], and $k = 16384$ PBRL codes from Sec. IV-B. Gallager's random coding union bound and the refined normal approximation of [55], both computed for FER =10^{-5} and the specified rates and blocklengths are provided for reference.

Examining the differences between the average SNR gaps of the code families and the average SNR gaps of the computed points for the Gallager bound at the comparable blocklengths, The difference is 1.053 dB for $k = 192$ family from [22],

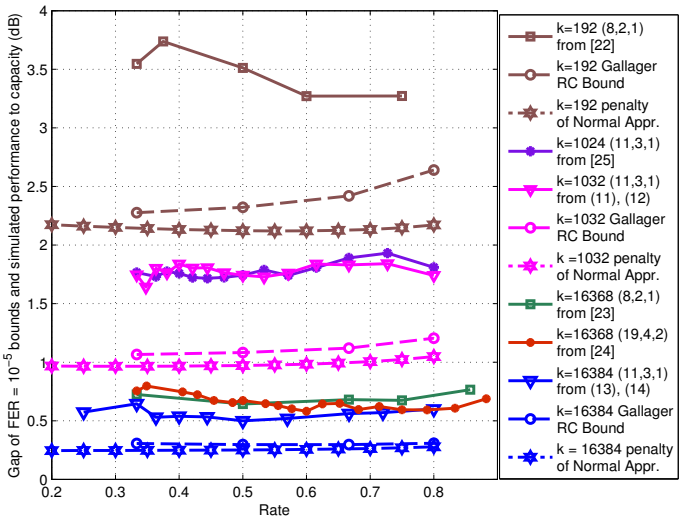

Fig. 9. SNR gaps in E_b/N_0 between the Shannon limit of the BI-AWGN channel and the required SNR to achieve FER of 10^{-5} for rates in the PBRL code families for five different PBRL code families and the codes from [24] and [25]. The code from [25] has the PBRL structure although it was not a constraint in its design. These code families correspond to protographs whose thresholds are explored in Fig. 2. Also shown are the SNRs associated with Gallager's bound [59] and associated with the refined normal approximation in [60] that correspond to an FER of 10^{-5} at the specified rates and blocklengths.

0.658 dB for the $k = 1032$ family from Sec. IV-A, 0.660 dB for the $k = 1024$ family from [25], 0.395 dB for the $k = 16386$ code from [23], 0.354 dB for the $k = 16384$ code from [24], and 0.255 dB for the $k = 16384$ code from Sec. IV-A.

The $k = 192$ example from [22] was designed without the benefit of the fast RCA algorithm and without the use of ACE in the lifting so that more improvement is likely possible at this shortest blocklength. Still, this $k = 192$ PBRL code family outperforms the RCPT codes in the 3GPP-LTE standard [56] at high SNRs and does not have error floors up to the highest SNRs studied in [22]. Other rate-compatible product codes and high-order non-binary turbo codes are designed in [57], [58]. These codes perform well in the short blocklength regime, but have much higher decoding complexity. Their blocklengths are shorter than the codes explored in this paper.

V. Concluding Remarks

This paper studies the construction and optimization of protograph-based Raptor-like (PBRL) LDPC codes. This paper designs PBRL codes by designing a highest-rate code (HRC) and sequentially adding degree-one variable nodes whose neighboring check node is connected to the variable nodes of the HRC so as to minimize the density evolution threshold. The new connections must obey constraints that control the error floor. Puncturing a single variable node in the HRC improves the threshold performance of PBRL codes.

Instead of the original density evolution, a modified reciprocal channel approximation (RCA) is used to obtain a fast and accurate approximation for the thresholds of PBRL codes to result in reasonable code-design complexity. Once the HRC is designed, the remaining code design including the

ACE+CPEG lifting takes a few hours on a typical personal computer.

Controlling the error floor is especially important for short-blocklength PBRL code families. For an example short-blocklength PBRL code family designed for $k = 1032$ we sacrificed threshold in order to improve error floor performance by increasing the number of degree-4 variable nodes in the HRC. Even in our long-blocklength ($k = 16384$) PBRL design example, a small amount of threshold performance is sacrificed in order to ensure good error floor performance by allowing only half of the connections to the punctured node to involve parallel edges.

For both long- and short-blocklength designs, we found that using CPEG and ACE jointly led to a more successful lifting of the protograph so that more threshold-reducing parallel edges could be introduced before causing a problematic error floor.

In summary, this paper provides a complete design procedure for constructing rate-compatible LDPC code families that perform uniformly close to finite blocklength performance limits for both short ($k = 1032$) and long ($k = 16384$) blocklengths.

ACKNOWLEDGMENT

The authors would like to thank Dr. Nguyen and Professor Nostratinia for sharing their numerical results of the error rate curves in [24] and [25].

REFERENCES

[1] T.-Y. Chen, A. Williamson, N. Seshadri, and R. Wesel, "Feedback communication systems with limitations on incremental redundancy," *IEEE Trans. Inf. Theory*, (submitted) 2013.

[2] J. Hagenauer, "Rate-compatible punctured convolutional codes (rcpc codes) and their applications," *IEEE Trans. Commun.*, vol. 36, no. 4, pp. 389–400, Apr. 1988.

[3] D. Rowitch and L. Milstein, "On the performance of hybrid FEC/ARQ systems using rate compatible punctured turbo (RCPT) codes," *Communications, IEEE Transactions on*, vol. 48, no. 6, pp. 948–959, 2000.

[4] R. Liu, P. Spasojevic, and E. Soijanin, "Punctured turbo code ensembles," in *Information Theory Workshop, 2003. Proceedings. 2003 IEEE*, Mar. 2003, pp. 249–252.

[5] R. G. Gallager, "Low-density parity-check codes," Ph.D. dissertation, MIT, Cambridge, MA, 1963.

[6] R. Tanner, "A recursive approach to low complexity codes," *Information Theory, IEEE Transactions on*, vol. 27, no. 5, pp. 533–547, 1981.

[7] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, Mar. 1999.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit errorcorrecting coding and decoding: Turbo codes," in *Proc. IEEE Int. Conf. Commun.*, Geneva, Switzerland, May 1993.

[9] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inform. Theory*, vol. 47, pp. 585–598, Feb. 2001.

[10] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 618–637, Feb 2001.

[11] S. Sesia, G. Caire, and G. Vivier, "Incremental redundancy hybrid ARQ schemes based on low-density parity-check codes," *IEEE Trans. Commun.*, vol. 52, no. 8, pp. 1311–1321, 2004.

[12] J. Ha, J. Kim, and S. McLaughlin, "Rate-compatible puncturing of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2824–2836, 2004.

[13] J. Ha, J. Kim, D. Klinc, and S. McLaughlin, "Rate-compatible punctured low-density parity-check codes with short block lengths," *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 728–738, 2006.

[14] M. El-Khamy, J. Hou, and N. Bhushan, "Design of rate-compatible structured LDPC codes for hybrid ARQ applications," *Selected Areas in Communications, IEEE Journal on*, vol. 27, no. 6, pp. 965–973, 2009.

[15] J. Kim, A. Ramamoorthy, and S. McLaughlin, "The design of efficiently-encodable rate-compatible LDPC codes," *IEEE Trans. Commun.*, vol. 57, no. 2, pp. 365–375, 2009.

[16] B. Vellambi and F. Fekri, "Finite-length rate-compatible LDPC codes: a novel puncturing scheme," *Communications, IEEE Transactions on*, vol. 57, no. 2, pp. 297–301, 2009.

[17] M. Yazdani and A. Banihashemi, "On construction of rate-compatible low-density parity-check codes," *Communications Letters, IEEE*, vol. 8, no. 3, pp. 159–161, 2004.

[18] S. Dolinar, "A rate-compatible family of protograph-based LDPC codes built by expurgation and lengthening," in *Proc. International Symposium on Information Theory*, 2005, pp. 1627–1631.

[19] J. Li and K. Narayanan, "Rate-compatible low density parity check codes for capacity-approaching ARQ schemes in packet data communications," in *Proc. Int. Conf. on Comm., Internet, and Info. Tech. (CIIT)*, Nov. 2002.

[20] N. Jacobsen and R. Soni, "Deign of rate-compatible irregular LDPC codes based on edge growth and parity splitting," in *Proc. IEEE Vehicular Technology Conference (VTC)*, Baltimore, MD, USA, Oct. 2007.

[21] S. Song, D. Hwang, S. Seo, and J. Ha, "Linear-time encodable rate-compatible punctured LDPC codes with low error floors," in *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, 2008, pp. 749–753.

[22] T.-Y. Chen, D. Divsalar, J. Wang, and R. D. Wesel, "Protograph-based raptor-like LDPC codes for rate-compatibility with short blocklengths," in *Proc. IEEE Global Communications Conference*, Houston, TX, Dec. 2011.

[23] T.-Y. Chen, D. Divsalar, and R. D. Wesel, "Protograph-based raptor-like LDPC codes with low thresholds," in *Proc. IEEE International Conference of Communications*, Ottawa, Canada, Jun. 2012.

[24] T. Nguyen, A. Nosratinia, and D. Divsalar, "The design of rate-compatible protograph LDPC codes," *IEEE Trans. Commun.*, vol. 60, no. 10, pp. 2841–2850, 2012.

[25] T. V. Nguyen and A. Nosratinia, "Rate-compatible short-length protograph LDPC codes," *Communications Letters, IEEE*, vol. 17, no. 5, pp. 948–951, 2013.

[26] J. Thorpe, "Low Density Parity Check (LDPC) codes constructed from protographs," *JPL IPN Progress Report*, vol. 42–154, Aug. 2003.

[27] D. Divsalar, S. Donlinar, C. R. Jones, and K. Andrews, "Capacity-approaching protograph codes," *IEEE J. Sel. Areas Commun.*, vol. 27, No. 6, pp. 876–888, Aug. 2009.

[28] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, 2006.

[29] M. Luby, "LT codes," in *Proc. 43rd Annu. IEEE Symp. Foundations of Computer Science (FOCS)*, Vancouver, BC, Canada, Nov. 2002.

[30] O. Etesami and A. Shokrollahi, "Raptor codes on binary memoryless symmetric channels," *Information Theory, IEEE Transactions on*, vol. 52, no. 5, pp. 2033–2051, 2006.

[31] W. Nitzold, M. Lentmaier, and G. Fettweis, "Spatially coupled protograph-based LDPC codes for incremental redundancy," in *Turbo Codes and Iterative Information Processing (ISTC), 2012 7th International Symposium on*, 2012, pp. 155–159.

[32] A. Jimenez Felstrom and K. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *Information Theory, IEEE Transactions on*, vol. 45, no. 6, pp. 2181–2191, 1999.

[33] "Low density parity check codes for use in near-earth and deep space. orange book." *CCSDS standard, Issue No.2*, Sep. 2007.

[34] "Digital video broadcasting; second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications," *ETSI EN 302 307 V1.2.1*, Aug. 2009.

[35] Z. Li and B. V. Kumar, "A class of good quasi-cyclic low-density parity check codes based on progressive edge growth graph," in *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Eighth Asilomar Conference on*, vol. 2. IEEE, 2004, pp. 1990–1994.

[36] J. Garcia-Frias and W. Zhong, "Approaching shannon performance by iterative decoding of linear codes with low-density generator matrix," *IEEE Commun. Lett.*, vol. 7, no. 6, pp. 266–268, June 2003.

[37] A. Eckford, J. Chu, and R. Adve, "Low-complexity cooperative coding for sensor networks using rateless and LDGM codes," in *IEEE International Conference on Communications, 2006. ICC '06.*, vol. 4, June 2006, pp. 1537–1542.

[38] K. Vakilinia, T.-Y. Chen, S. V. S. Ranganathan, A. R. Williamson, D. Divsalar, and R. D. Wesel, "Short-blocklength non-binary LDPC codes with feedback-dependent incremental transmissions," in *Proc. 2014 IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, USA, July 2014.

[39] ——, "Feedback systems using non-binary ldpc codes with a limited number of transmissions," in *Proc. 2014 IEEE Inf. Theory Workshop (ITW)*, Hobart, Tasmania, Australia, November 2014.

[40] E. Soljanin, "Punctured vs. rateless codes for hybrid ARQ," in *Proc. IEEE Inform. Theory Workshop '06*, Punta del Este, Uruguay, Mar 2006.

[41] A. Venkiah, C. Poulliat, and D. Declercq, "Analysis and design of raptor codes for joint decoding using information content evolution," in *Proc. International Symposium on Information Theory (ISIT)*, Jun. 2007, pp. 421–425.

[42] 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.

[43] S. Y. Chung, "On the construction of some capacity-approaching coding schemes," Ph.D. dissertation, MIT, Cambridge, MA, 2000.

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

[45] T. Richardson and R. Urbanke, *Modern Coding Theory*. New York, NY, USA: Cambridge University Press, Mar. 2008.

[46] R. G. Gallager, "Low density parity check codes," Ph.D. dissertation, Massachusetts Institute of Technology, 1960.

[47] C. Di, T. Richardson, and R. Urbanke, "Weight distribution of low-density parity-check codes," *Information Theory, IEEE Transactions on*, vol. 52, no. 11, pp. 4839–4855, Nov 2006.

[48] A. Orlitsky, K. Viswanathan, and J. Zhang, "Stopping set distribution of ldpc code ensembles," *Information Theory, IEEE Transactions on*, vol. 51, no. 3, pp. 929–953, March 2005.

[49] D. Burshtein and G. Miller, "Asymptotic enumeration methods for analyzing ldpc codes," *Information Theory, IEEE Transactions on*, vol. 50, no. 6, pp. 1115–1131, June 2004.

[50] ——, "Asymptotic enumeration methods for analyzing ldpc codes," *IEEE Trans. Inf. Theory*, vol. 50, no. 6, pp. 1115–1131, Jun. 2004.

[51] A. Orlitsky, K. Viswanathan, and J. Zhang, "Stopping set distribution of LDPC code ensembles," *IEEE Trans. Inf. Theory*, vol. 51, no. 3, pp. 929–953, March 2005.

[52] C.-H. Hsu and A. Anastasopoulos, "Capacity-achieving codes with bounded graphical complexity and maximum likelihood decoding," *IEEE Trans. Inf. Theory*, vol. 56, no. 3, pp. 992–1006, March 2010.

[53] D. Divsalar, C. Jones, S. Dolinar, and J. Thorpe, "Protograph based LDPC codes with minimum distance linearly growing with block size," in *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, vol. 3, Nov 2005.

[54] T. Tian, C. Jones, J. Villasenor, and R. Wesel, "Selective avoidance of cycles in irregular ldpc code construction," *IEEE Trans. Commun.*, vol. 52, no. 8, pp. 1242–1247, Aug 2004.

[55] Y. Polyanskiy, H. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307 –2359, May. 2010.

[56] 3rd Generation Partnership Project (http://www.3gpp.org), "3GPP TS 36.212 Multiplexing and channel coding," vol. Release 8, Mar. 2008.

[57] G. Liva, E. Paolini, and M. Chiani, "On optimum decoding of certain product codes," *Communications Letters, IEEE*, vol. 18, no. 6, pp. 905–908, June 2014.

[58] G. Liva, E. Paolini, B. Matuz, S. Scalise, and M. Chiani, "Short turbo codes over high order fields," *Communications, IEEE Transactions on*, vol. 61, no. 6, pp. 2201–2211, 2013.

[59] R. Gallager, "A simple derivation of the coding theorem and some applications," *Information Theory, IEEE Transactions on*, vol. 11, no. 1, pp. 3–18, Jan 1965.

[60] Y. Polyanskiy, H. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *Information Theory, IEEE Transactions on*, vol. 56, no. 5, pp. 2307–2359, May 2010.