Transactions Papers

Design of Efficiently Encodable Moderate-Length High-Rate Irregular LDPC Codes

Michael Yang, *Student Member, IEEE*, William E. Ryan, *Senior Member, IEEE*, and Yan Li, *Student Member, IEEE*

***Abstract*—This paper presents a new class of irregular low-density parity-check (LDPC) codes of moderate length ($10^3 \leq n \leq 10^4$) and high rate ($R \geq 3/4$). Codes in this class admit low-complexity encoding and have lower error-rate floors than other irregular LDPC code-design approaches. It is also shown that this class of LDPC codes is equivalent to a class of systematic serial turbo codes and is an extension of irregular repeat-accumulate codes. A code design algorithm based on the combination of density evolution and differential evolution optimization with a modified cost function is presented. Moderate-length, high-rate codes with no error-rate floors down to a bit-error rate of 10^{-9} are presented. Although our focus is on moderate-length, high-rate codes, the proposed coding scheme is applicable to irregular LDPC codes with other lengths and rates.**

***Index Terms*—Efficient encoding, error-rate floor, irregular repeat–accumulate codes, low-density parity-check (LDPC) codes.**

I. Introduction

THE recent literature in coding has seen an explosion of papers surrounding the design and implementation of low-density parity-check (LDPC) codes [1], [2]. As has been well reported, this class of codes is capable of operation within tenths of a decibel of the capacity limit, given sufficiently long codeword lengths, surpassing even turbo codes in many cases [3], [4]. The pioneering work of Richardson *et al.* [3], [4] presented very long rate-1/2 codes (codeword lengths $n \geq 10^6$) which are not appropriate for many applications (e.g., low-latency, bandwidth-efficient applications). Further, such codes generally require high-complexity encoders, since they lack sufficient structure to allow simple encoding (with cyclic codes representing the limit of simplicity), although Richardson *et al.* [5] have proposed a clever encoding algorithm whose complexity is approximately linear in the code length n. MacKay, on the other hand, has designed high-rate LDPC codes of moderate lengths ($10^2 \leq n \leq 10^5$) [16], but again, these codes generally require complex encoders. Important alternatives to these codes are the cyclic and quasi-cyclic LDPC codes of Kou *et al.* [15] based on finite geometries. This class of codes admits low-complexity shift-register encoders, but the decoders for these codes are generally of higher complexity as a result of their higher density parity-check matrices or, in some cases, square ($n \times n$) parity-check matrices. These codes are also regular or near regular, which limits their performance in the low signal-to-noise ratio (SNR) region.

In this paper, we focus on the design of moderate length ($10^3 < n < 10^4$), high rate ($R \geq 3/4$) irregular LDPC codes. (The only other paper of which we are aware which focuses on this regime is [10].) Our goal is the design of such codes superior in performance to alternative approaches and which allow low-complexity encoding and decoding. Our approach starts with the work of Richardson *et al.* [3], but we make some novel modifications to their design technique which constrains our codes to a specialized class of irregular LDPC codes. These modifications improve performance for the range of n and R under consideration and lead to vastly simplified encoders. We show that the complexity of these encoders is much smaller than might be achieved with a typical LDPC code, even when the technique proposed in [5] is used. We also show the connection between the proposed class of codes, serial turbo codes, and irregular repeat-accumulate (IRA) codes [11]. In fact, we call these codes extended IRA (eIRA) codes. These codes were independently studied by Narayanaswami and Narayanan [12].

The remainder of the paper is outlined as follows. Section II provides the necessary background and notation for the subsequent sections. Section III develops an important lemma which leads to the introduction of the new class of irregular LDPC codes introduced in Section IV. Section V briefly presents the (computer-based) code-design algorithm, and Section VI presents selected design results which demonstrate the validity of the preceding sections. Finally, some concluding remarks are presented in Section VII.

II. Background

Following the literature (e.g., [3]), we let n and k represent the length and dimension, respectively, of an irregular LDPC

Paper approved by M. Fossorier, the Editor for Coding and Communication Theory of the IEEE Communications Society. Manuscript received August 19, 2002; revised May 23, 2003 and October 9, 2003. This work was supported in part by the National Science Foundation under Grant CCR-9814472 and Grant CCR-9979310, in part by NASA under Grant NAG5-10643, and in part by a gift from the NSIC EHDR program. This paper was presented in part at the 2002 Allerton Conference on Communication, Control, and Computing, Monticello, IL, October 2002, and in part at the 2003 IEEE International Symposium on Information Theory, Yokohama, Japan, June 29–July 4, 2003.

The authors are with the Department of Electrical and Computer Engineering, The University of Arizona, Tucson, AZ 85721 USA (e-mail: yangsiz@ece.arizona.edu; ryan@ece.arizona.edu; yanli@ece.arizona.edu).

Digital Object Identifier 10.1109/TCOMM.2004.826367

code. (In the following, unless specified otherwise, when we say LDPC code, we shall mean the general case of an irregular LDPC code.) We also let $R = k/n$ represent its code rate and $m = n - k$ equal the number of parity bits. The generator and parity-check matrices are denoted by G and H, respectively, and encoding takes place according to the usual equation, $\mathbf{c} = \mathbf{u}G$, where $\mathbf{u}$ ($\mathbf{c}$) is the information (code) vector. Following [3], $\lambda(x) = \sum \lambda_i x^{i-1}$ represents the variable node (v-node) distribution in the code's Tanner graph, and $\rho(x) = \sum \rho_i x^{i-1}$ represents the check node (c-node) distribution. Here, λ_i is the fraction of edges connected to v-nodes of degree i and similarly for ρ_i and c-nodes. For now, we assume $i \geq 2$, but later we will allow a single v-node of degree one. It can be shown [3] that the number of v-nodes of degree i is (the integer part of)

$$N_v(i) = \frac{n\lambda_i}{i \int_0^1 \lambda(x)dx} \tag{1}$$

and the number of c-nodes of degree i is (the integer part of)

$$N_c(i) = \frac{m\rho_i}{i \int_0^1 \rho(x)dx}. \tag{2}$$

Thus, there are $N_v(i)$ columns in H of Hamming weight i and $N_c(i)$ is the number of rows in H of weight i. For v-nodes, i ranges from two to d_v, and for c-nodes i ranges from two to d_c.

With LDPC codes so parameterized, it is shown in [3] and [6] how optimum degree distributions $\lambda(x)$ and $\rho(x)$ may be obtained via algorithms centered on the evolution of the probability density functions (pdfs) of the messages passed between the two node types in the belief-propagation decoder. These distributions are optimal in the limit as $n \to \infty$ and as the number of iterations $l \to \infty$. For the practical case of finite n, the following additional design rules are proposed in [3]:

1) forbid short cycles involving only degree-two v-nodes;
2) degree-two v-nodes are made to correspond to nonsystematic bits;
3) there exist no length-four cycles in the code's graph.

III. On Degree-Two V-Nodes

We assume the (log-) sum-product decoding algorithm throughout (also called belief propagation) [1], [2]. The sum of the messages received at the qth v-node after a sufficient number of decoding iterations is known to be an approximation of the log *a posteriori* ratio (LAPR) [or log-likelihood ratio (LLR)] of the qth code bit given the received word. Degree-two v-nodes represent the weakest link in the code in the following sense. First, the bit LLRs for these v-nodes converge slower than do larger degree v-nodes. This is discussed in [6] and [7] in the context of density evolution, where it is noted that the slope of the $\mathrm{SNR_{out}}$ versus $\mathrm{SNR_{in}}$ curve for v-nodes is equal to the node degree minus one. Second, the LLRs for the degree-two v-nodes converge (on average) to a smaller magnitude than do larger degree v-nodes.

We demonstrate these characteristics in Fig. 1 for code four of *Example 2* below (the details of which are not important at this point). The figure depicts the evolution of the expected LLR magnitudes for each of the possible v-node degrees for this code at $E_b/N_0 = 3.2$ dB (corresponding to $P_b = 4 \times 10^{-5}$). As expected, the LLR magnitudes for the larger degrees converge

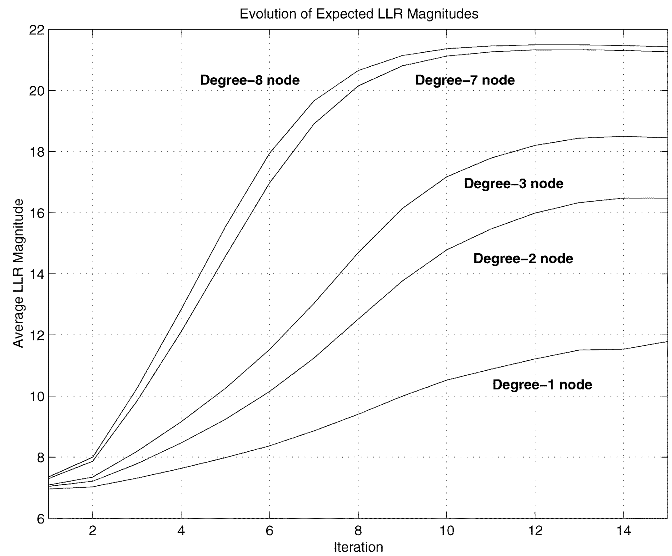


Fig. 1. Evolution of expected LLR magnitudes for code four of *Example 2*.

quickly (in about eight iterations) and to large values, whereas the degree-two node is the worst in this regard (it converges in about 14 iterations). Code four of *Example 2* also has a single degree-one v-node whose impact is negligible, as demonstrated later.

The vulnerability of the degree-two bits was made evident in the foregoing discussion. Still, as shown in [3], the presence of degree-two v-nodes is necessary to ensure optimal irregularity. Intuitively, low-degree v-nodes, which are "bad," must be present to balance out the presence of the low-degree c-nodes, which are "good." Low-degree c-nodes are advantageous because there is less opportunity for a check to fail when fewer bits are involved in the check equation.

Because degree-two v-nodes represent the weakest link in a code, we introduce in this section some results concerning the degree-two v-nodes in a code's Tanner graph (equivalently, the weight-two columns in its H matrix). Most crucial to the development is the following lemma and its implications.

Lemma: In a given Tanner graph (equivalently, H matrix), the maximum number of degree-two v-nodes possible before a cycle is created involving only these degree-two nodes is $N_{v,\max}(2) = n - k - 1 = m - 1$. Furthermore, for codes free of "degree-two cycles" and possessing this maximum, the submatrix of H composed of only its weight-two columns is simply a permutation of the following $m \times (m-1)$ parent matrix:

$$T = \begin{bmatrix} 1 & & & & & \\ 1 & 1 & & & & \\ & 1 & 1 & & & \\ & & & \cdots & & \\ & & & & 1 & 1 \\ & & & & & 1 \end{bmatrix}. \tag{3}$$

Proof: Because a Tanner graph is a bipartite graph, a length-$2s$ cycle in the graph corresponds to an $s \times s$ submatrix of H, in which each row and each column of that submatrix contains at least two ones (corresponding to the intersection of s rows and s columns in H which need not be contiguous).

The columns in H corresponding to the $N_v(2)$ degree-two v-nodes already contain two ones, so consider now the rows in H in which these ones are located, and suppose there are r such rows. It may be shown inductively that the $r \times N_v(2)$ submatrix S containing these r rows and $N_v(2)$ columns (of weight two) must contain a cycle if $r \leq N_v(2)$. (Consider, for example, such a submatrix with $N_v(2) = 2$ columns, $N_v(2) = 3$ columns, and so on. Consider also the form of T which contains no cycles and has one more row than it does columns, as it must.) Thus, in order for S to correspond to a cycle-free part of the graph, we must have $r \geq N_v(2) + 1$. But since $r \leq m$, we must have $N_v(2) \leq m - 1$. The second statement in the lemma is evident at this point. ■

We are interested in the maximum number of cycle-free weight-two columns, because the optimal v-node degree distribution (large n and large l) usually implies $N_{v,\mathrm{opt}}(2) > m-1$, as we will see in the case studies below. We know now from the lemma that degree-two cycles in the Tanner graph will be present when $N_v(2) > m - 1$. Thus, we focus on codes for which $N_v(2)$ is not at its optimum, but is instead at the smaller value prescribed by the lemma, $N_v(2) = m - 1$. Doing so will ensure that cycles among the degree-two v-nodes are avoided. Some of these points were also noted by Chiani and Ventura [10] in the design of moderate-length, high-rate irregular LDPC codes.

IV. New Class of Irregular LDPC Codes

A. Code Structure and Encoder

Motivated by the above lemma and surrounding discussion, we first consider the parity-check matrix of an $(n-1, k-1) = (n', k')$ LDPC code of the form

$$H = [H_1 \; T] \tag{4}$$

where H_1 is a sparse $(n-k) \times k$ matrix (to be randomly generated by computer) containing no weight-two columns, and T is the $(n-k) \times (n-k-1)$ matrix given in (3). Note also that $N_v(2) = n-k-1 = n'-k'-1$ so that it equals the maximum prescribed by the lemma. However, we have not yet attained the form of the H matrix which permits efficient encoding. To proceed, we require the following claim which will be verified by simulation below.

Claim: Given a randomly generated parity-check matrix for an $(n-1, k-1)$ code, appending a single weight-one column on the right to produce an (n, k) code changes the error rate only negligibly.

Remarks: The veracity of this claim should be clear, because the addition of a single weight-one column affects only one check equation. In view of Fig. 1, we can expect the error rate of this nth bit to be quite large, since it corresponds to a degree-one v-node. However, since it corresponds to a parity bit, its error rate is of no concern. As our simulation results confirm, its affect on the k information bits is negligible.

Armed with the above claim, we modify the H in (4) as

$$H = [H_1 \; H_2] \tag{5}$$

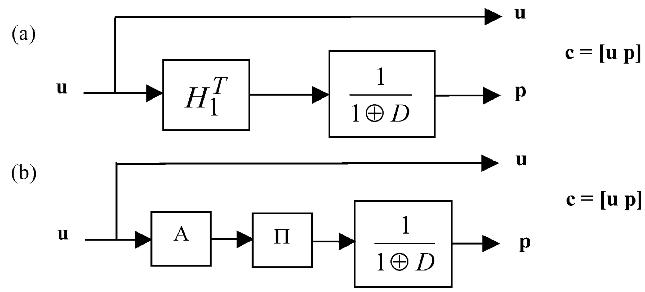


Fig. 2. Two efficient encoders for the proposed class of eIRA codes. (a) Original direct form. (b) Explicit serial turbo-code form.

where H_1 is unchanged from above, and H_2 is a full-rank $m \times m$ matrix created from T by appending to it (on the right) a weight-one column vector with a "1" at the bottom

$$H_2 = \begin{bmatrix} 1 & & & & & \\ 1 & 1 & & & & \\ & 1 & 1 & & & \\ & & & \cdots & & \\ & & & & 1 & 1 \\ & & & & & 1 \;\; 1 \end{bmatrix}. \tag{6}$$

The $(n-k) \times n$ matrix H in (5) achieves $N_v(2) = m - 1$ and corresponds to an (n, k) irregular LDPC code which allows very efficient encoding, as we now show.

Observe from the form of H, in particular, the submatrix H_2, that encoding may be performed directly from H by solving for the parity bits recursively: 1) the first check equation (first row) involves only the first parity bit; 2) the second check equation involves only the first two parity bits; 3) the third check equation involves only the next two parity bits; and so on. Thus, the form that we are forcing H to take not only avoids degree-two cycles, but it also achieves the ultimate situation in [5] in which a full $m \times m$ submatrix possesses the diagonal form which permits a recursive solution of the parity bits.

An alternative way to look at encoding for this class of codes is as follows. First, we have the following result.

Result: The systematic generator matrix G for the parity-check matrix of (5) is given by

$$G = [I \quad P] = [I \quad H_1^T H_2^{-T}].$$

Proof: It is trivial to check that $GH^T = 0$. ■

Now, it is easy to verify that

$$H_2^{-T} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ & 1 & \cdots & \cdots & 1 \\ & & 1 & \cdots & 1 \\ & & & 1 & 1 \\ & & & & 1 \end{bmatrix} \tag{7}$$

which is precisely the transformation matrix corresponding to a differential encoder whose transfer function is $1/1 \oplus D$. Thus, the encoder for this class of LDPC codes enjoys the low-complexity configuration depicted in Fig. 2(a). Observe that this encoder is indeed low complexity, for the computation of the parity bits involves multiplication of the message word $\mathbf{u}$ by the low-density matrix H_1^T followed by the differential encoding.

Observe that the encoder of Fig. 2(a) is the encoder for a type of systematic serial turbo code. The interleaver that generally accompanies a serial turbo code may be made more explicit by writing H_1^T as the product $A\Pi$, where Π is a permutation matrix and A is simply the low-density matrix $H_1^T\Pi^{-1}$. This version of the encoder is shown in Fig. 2(b). Note that while this class of LDPC codes possesses the advantage of a low-complexity encoder that a serial turbo code enjoys, it requires no interleaver and has all the advantages of an LDPC code: a low error-rate floor and a parallelizable decoder.

We remark that this class of efficiently encodable codes was independently discovered by Narayanaswami and Narayanan [12]. We mention also that these codes resemble the systematic version of the IRA codes [11], except for systematic IRA codes, the matrix H_1^T in Fig. 2(a) which has dimension $k \times (n-k)$ is replaced by a $k \times n$ low-density generator matrix $(n > k)$. For this reason, we call these codes *extended* IRA (eIRA) codes.

While our focus has been on codes for which $N_v(2) = m-1$ so that degree-two cycles are avoided, certain of the results are extendable to $N_v(2) > m-1$, as we now discuss. First, it is easy to show that the H matrix for any irregular LDPC code with $N_v(2) \geq m-1$ may be put in the form of (5), except the submatrix H_1 in this case will contain weight-two columns when $N_v(2) > m-1$. From this, it is clear that the encoding may be performed from the H matrix as described above [or via the encoder of Fig. 2(a)] when $N_v(2) \geq m-1$.

B. Encoder Efficiency

Consider first the number of binary additions required to encode one eIRA codeword, assuming encoding is performed via the H matrix. We assume the density of ones in H_1 to be δ. Then the number of binary additions required to compute the $n-k$ parity bits is approximately

$$N_1 = \delta(k+1)(n-k).$$

If we instead use the encoder of Fig. 2(a) to encode, multiplication by the matrix H_1^T results in $\delta k(n-k)$ additions, and differential encoding results in $n-k$ additions, resulting in a total of $(\delta k+1)(n-k)$, which is a small fraction larger than N_1.

For our first complexity comparison, consider encoding via a generator matrix $G = [I \quad P]$ which has been obtained from H by Gauss–Jordan elimination. In general, the $k \times (n-k)$ matrix P has a density of 0.5, and so the number of binary additions required to perform the multiplication $\mathbf{u}P$ is approximately

$$N_2 = 0.5k(n-k).$$

Thus, the proposed class of codes provides a factor of $N_2/N_1 = [0.5k/\delta(k+1)] \simeq 0.5/\delta$ reduction in the number of computations required to encode a codeword. For example, for a (fairly high) density of $\delta = 0.01$, this complexity reduction factor is 50.

As a second, more involved, comparison, we consider the encoding technique proposed in [5] assuming an *arbitrary* irregular LDPC code parity-check matrix.[1] Their technique involves the computation of the parity vector $\mathbf{p}$ in two parts (see [5, Tables I and II]). The first part requires, among other operations, multiplication by three sparse matrices whose sizes are $(n-k-g) \times k$, $g \times (n-k-g)$, and $g \times k$. (Here, g is the gap parameter defined in [5] where it was shown $g \leq O(\sqrt{n})$ with probability near one when n is large.) The second part requires, among other operations, multiplication by two sparse matrices whose sizes are $(n-k-g) \times k$ and $(n-k-g) \times g$. Thus, assuming a common density of δ for each of these matrices (we remark on this below), the number of binary additions required to perform the various sparse matrix multiplications is (after some simplification) $\delta k(n-k) + \delta(k+2g)(n-k-g)$. The first term in this expression is approximately the complexity N_1 of the proposed class of codes, and so the second term represents the *additional* number of binary additions required by the technique in [5] just for sparse matrix multiplication. Beyond these additional addition operations, the technique in [5] requires multiplication by a dense $g \times g$ matrix [complexity $O(n)$], two vector additions [each $O(n)$], and two multiplications by a triangular matrix [each $O(n)$].

In the foregoing, we assumed that the density for the full parity-check matrix H_{irreg} of an irregular LDPC code was approximately equal to that of the submatrix H_1 of the parity-check matrix H_{eIRA} for an eIRA code. We support this as follows. Note that, for high-rate codes for which H_2 is a small fraction of H_{eIRA}, the difference between the densities of H_1 and H_{eIRA} are small (e.g., within 20%). Thus, since the row and column weight distributions for H_{irreg} and H_{eIRA} are similar as seen in the examples below, we may conclude that the difference between the densities of H_{irreg} and H_1 are small.

V. Code-Design Algorithm

Based on the discussion of the previous section, our design algorithm involves (once the optimal degree distributions have been determined) appending length-m columns to the H_2 matrix in accordance with the degree distributions and some additional design rules. Note that a cycle will be created with the addition to H_2 of a column, and the addition of large-weight columns tends to create shorter cycles. Thus, in the design algorithm, we only append a column if it creates no length-four cycles. Note also that, by starting with the matrix H_2, we are abiding by the other two design criteria of [3], namely, associating the weight-two columns with the nonsystematic bits and eliminating all cycles associated with the degree-two nodes (see Section II).

The goal is to design a $(n-k) \times n$ parity-check matrix H containing E ones. The design algorithm is as follows. (A similar computer random-search algorithm for regular codes was presented in [8].)

Step 1) Use differential evolution [3], **[9]** to generate the global optimal degree distributions $\lambda(x)$ and $\rho(x)$ for the desired code rate. The cost function is modified to force $N_v(2) = n-k-1$. (A detailed differential evolution algorithm is discribed in [13, Part V], and a source program is available at [14].)

Step 2) Initialize the matrix H with H_2. Since H_2 contains $2m+1$ ones, $E-2m-1$ ones remain to be placed in H.

[1] In response to a comment of a reviewer, we acknowledge that the encoding technique in [5] is exactly that of the H-based technique described above when the code is eIRA. The encoding complexity comparison we seek here is that between an eIRA code and a typical (non-IRA) irregular LDPC code using the technique in [5].

Step 3) For both node types and for all appropriate values of d, give each degree-d node d sockets [3]. Number the sockets for the v-nodes and then for the c-nodes (there are at this point $E-2m-1$ sockets of each type to be assigned edges).

Step 4) Generate an initial permutation of $\{1, 2, \ldots\ldots, E-2m-1\}$ and use this permutation to connect the v-node sockets to the c-node sockets.

Step 5) Modify the permutation to satisfy the optimal degree distributions.

Step 6) Modify the permutation to eliminate the length-four cycles, and after each modification, go back to Step 3 to ensure the degree distributions are still satisfied. ([8] has provided detailed procedures for Step 3 and Step 4.)

Step 7) Convert the permutation into the remaining k columns of H.

A. Remarks

1) We note that the codes presented in the next section were obtained from the above algorithm without additional "manual modification" for girth control; our design prevents all length-four cycles, but ignores larger cycles. We emphasize that none of the codes presented in this paper contain length-four cycles, including the irregular LDPC codes that are not eIRA codes presented in the next section. In fact, it can be shown that for some of the high-rate codes we have designed, graphs with girths larger than six are not possible.

2) Although we have not made an effort to modify the cycle structure of our codes beyond removing the length-four cycles, we have made an effort to maximize the weight of the codewords corresponding to weight-one and weight-two encoder inputs. Considering Fig. 2(a), note that a weight-one input simply selects a row of H_1^T (a column of H_1) which then acts as input to the differential encoder. To maximize the weight of the differential encoder output corresponding to weight-one inputs, then, the ones in the columns of H_1 should be widely separated. Similarly, for weight-two encoder inputs, since such an input produces the sum of two columns of H_1 as the input to the differential encoder, the ones in all sums of pairs of columns of H_1 should be widely separated. In principle, we could continue for weight-w inputs, $w > 2$, but the algorithm becomes unwieldy at this point, and, in any case, our results below show that the error-rate floor is lower than 10^{-9} even when these larger weight inputs are ignored in the design.

VI. Code-Design Results

In this section, we present performance results for eIRA codes designed using the algorithm of the previous section. Comparisons are made to codes found in the literature. We note that computation of the bit-error rate (BER) P_b and the codeword-error rate P_{cw} involve only the k information bits.

Example 1: In this first example, we consider the design of two moderate-length rate-1/2 codes: one with (n, k) parameters (4000, 2000), and one with parameters (1000, 500). The optimal degree distributions for rate-1/2 codes with $d_v = d_c = 7$ (using the Gaussian approximation [6]) are

$$\lambda(x) = 0.30780x + 0.27287x^2 + 0.41933x^6$$
$$\rho(x) = 0.4x^5 + 0.6x^6.$$

Based on these distributions, we find for the (4000, 2000) code, the number of v-nodes of degree two is from (1)

$$N_v(2) = \frac{(4000)\left(\frac{0.30780}{2}\right)}{\frac{1}{2}0.30780 + \frac{1}{3}0.27287 + \frac{1}{7}0.41933} \simeq 2020.$$

Using (1) and (2) in similar computations, we find also $N_v(3) \simeq 1194$, $N_v(7) \simeq 786$, $N_c(6) = 876$, and $N_c(7) = 1124$. In particular, $N_v(2) > m - 1 = 1999$, so that the condition required by the lemma for cycle-free degree-two v-nodes is not satisfied. For the (1000, 500) code, we find $N_v(2) = 503 > m-1 = 499$ so that the condition is again not satisfied. However, $N_v(2)$ does not exceed the prescribed value of $m - 1$ by very much in either case, so that we would not expect there to be very many cycles among the degree-two v-nodes. Thus, we would not expect there to be much degradation due to such cycles for these particular codes.

To verify this, we first designed a (4000, 2000) eIRA code with $N_v(2) = m - 1 = 1999$ per the algorithm of the previous section. We then designed a (4000, 2000) code with $N_v(2) = 2020$ using an algorithm much like the one in the previous section, but without the $N_v(2) = m - 1$ constraint (essentially the algorithm in [3]). The two codes were simulated and had essentially identical performance curves. We repeated this for the (1000, 500) case and obtained the same result, thus confirming our expectations. The (1000, 500) curve(s) also showed close agreement with the one in [3], thus establishing the quality of our design algorithm. The BER P_b curves for the four codes are shown in Fig. 3. The two codes designed using the technique in [3] are labeled "R&U" in the figure, and two codes designed as eIRA codes per the algorithm above are labeled "eIRA." ■

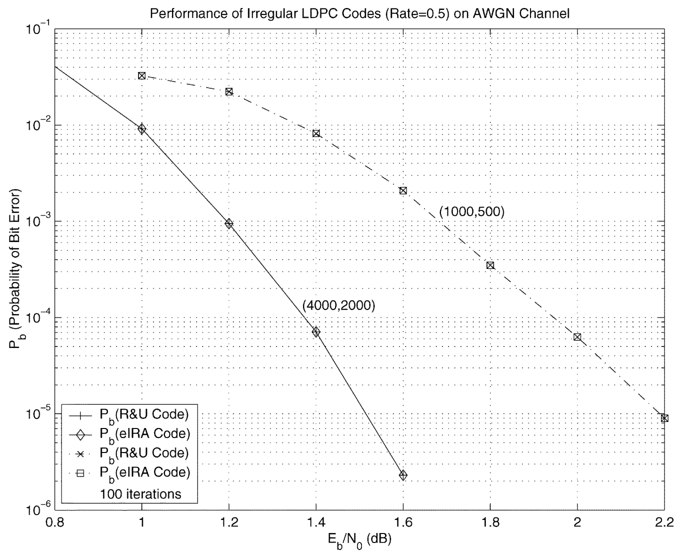


Fig. 3. Performance of $n = 4000$ and $n = 1000$ rate-1/2 irregular LDPC codes.

Example 2: In this second example, we consider the following four rate-0.82 codes.

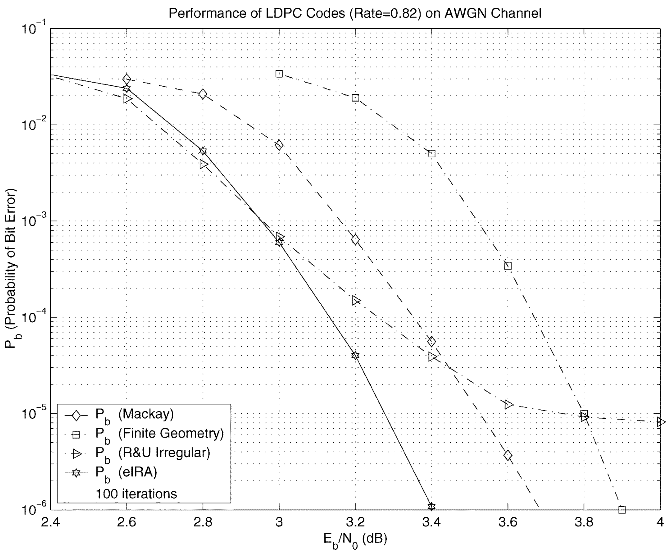


Fig. 4. Performance comparison of various $n = 4161$ rate-0.82 regular and irregular LDPC codes.

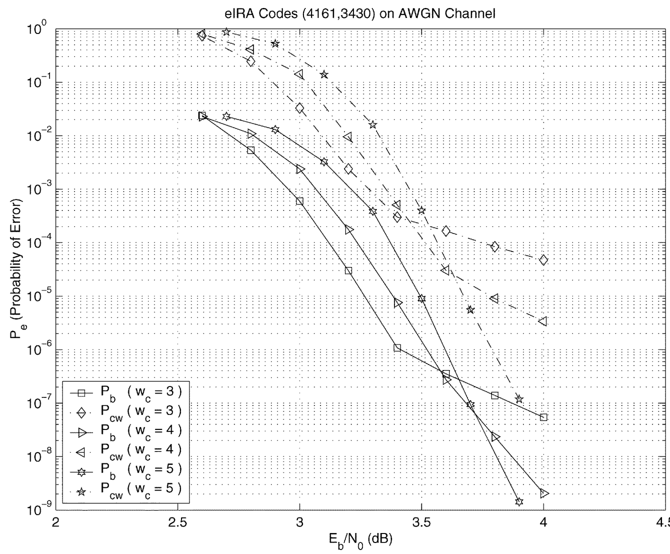


Fig. 5. Performance comparison of various $n = 4161$ rate-0.82 eIRA codes with H_1 column weights $w_c = 3$, 4, and 5.

1) A (4161, 3431) (nearly) regular LDPC code due to MacKay [16] having degree distributions

$$\lambda_{\mathrm{MacKay}}(x) = x^3$$
$$\rho_{\mathrm{MacKay}}(x) = 0.2234x^{21} + 0.7766x^{22}.$$

Note $N_v(2) = 0$.

2) A (4161, 3430) regular finite geometry-based LDPC due to Kou *et al.* [15] having degree distributions

$$\lambda_{fg}(x) = x^{64}$$
$$\rho_{fg}(x) = x^{64}.$$

Note $N_v(2) = 0$.

3) A (4161, 3430) irregular LDPC code *without* the constraint $N_v(2) = m - 1 = 730$ and with $d_v = 8$ and $d_c = 20$. The optimal degree distributions were found to be

$$\lambda_{\mathrm{wo}}(x) = 0.2343x + 0.3406x^2 + 0.2967x^6 + 0.1284x^7$$
$$\rho_{\mathrm{wo}}(x) = 0.3x^{18} + 0.7x^{19}.$$

From the v-node distribution, we compute $N_v(2) = 1636$, which is far greater than $m - 1 = 730$, and so we can expect there to be many cycles associated with the degree-two nodes, in light of the lemma.

4) A (4161, 3430) eIRA code (i.e., *with* the constraint $N_v(2) = m - 1 = 730$) and with $d_v = 8$ and $d_c = 20$. The optimal degree distributions were found to be

$$\lambda_{\mathrm{w}}(x) = 0.00007x^0 + 0.1014x + 0.5895x^2 + 0.1829x^6 + 0.1262x^7$$
$$\rho_{\mathrm{w}}(x) = 0.3037x^{18} + 0.6963x^{19}.$$

From this, we compute $N_v(1) = 1$ and $N_v(2) = 730$ so that $N_v(2)$ equals the maximum prescribed by the lemma.

Fig. 4 presents the performance of these four codes, where we first note that the unconstrained irregular code (code three) suffers from a high error-rate floor due to its large number of degree-two v-nodes and their associated cycles. The ($N_v(2)$-constrained) eIRA code (code four) has the best performance in the region simulated, since such cycles are avoided in the code design. The finite geometry code and the MacKay code possess no such cycles, but they are inferior to code four in the region simulated, since they possess far from optimal degree distributions (one is regular and the other is nearly regular). We remark that the finite geometry code likely has the lowest error-rate floor due to its large minimum distance, lower bounded as $d_{\min} \geq 66$ in [15]. Last, we point out that a code which has a value of $N_v(2)$ between those of codes three and four (i.e., $730 < N_v(2) < 1636$) will have a floor which lies between the floors of these two codes. ■

Example 3: Whereas code four has a vastly improved error-rate floor relative to code three in the previous example, it starts to hit a floor in the vicinity of $P_b = 10^{-6}$ (as will be shown shortly). This is attributable to a somewhat small $d_{\min}$. We conjecture that low-weight columns in H (specifically, the submatrix H_1) lead to a small $d_{\min}$. MacKay and Davey [17] have also made this conjecture. Further, as demonstrated in Section II, the LLR values for bits corresponding to low column weights have smaller magnitudes than those with high column weights. To study this conjecture, we designed two (4161, 3430) eIRA codes whose H_1 column weights are $w_c = 4$ and 5, and compared their performance to code four above, whose H_1 column weights are 3, 7, and 8. Degree distributions for these two additional codes are

$$\lambda_4(x) = 0.0000659x^0 + 0.0962x + 0.9037x^3$$
$$\rho_4(x) = 0.2240x^{19} + 0.7760x^{20}$$
$$\lambda_5(x) = 0.0000537x^0 + 0.0784x + 0.9215x^4$$
$$\rho_5(x) = 0.5306x^{24} + 0.4694x^{25}.$$

The error-rate curves (BER P_b and codeword-error rate P_{cw}) for these two codes are presented together with the performance of code four in Fig. 5 (where code four is labeled $w_c = 3$). We observe that the floor decreases with increasing w_c and that the code with $w_c = 5$ has no floor down to $P_b = 10^{-9}$. Further, the $w_c = 5$ code is only about 0.2 dB inferior to the $w_c = 3$ code (code four) at $P_b = 10^{-6}$. ■

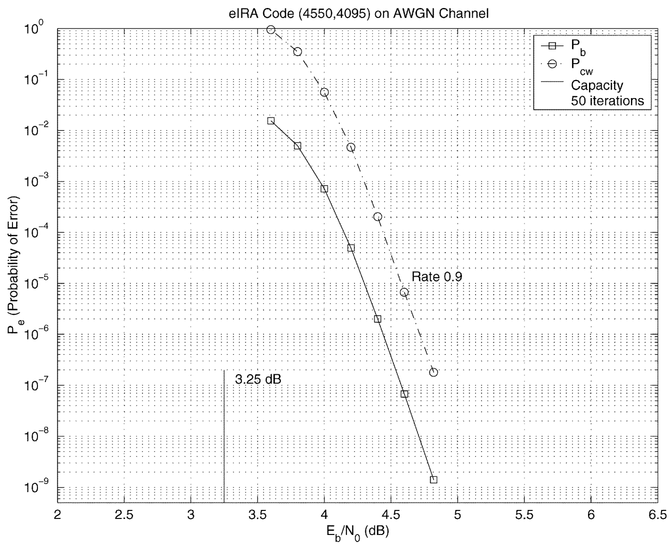


Fig. 6. Rate-0.9 eIRA code on the AWGN channel with low error-rate floor.

Example 4: As a final example which demonstrates the utility of our approach, we have designed a rate-0.9 (4550, 4095) eIRA code with $w_c = 5$. The degree distributions for this code are

$$\lambda(x) = 0.0000467x^0 + 0.0425x + 0.9574x^4$$
$$\rho(x) = 0.0000467x^{45} + 0.9999x^{46}.$$

Its performance is presented in Fig. 6, where we observe the absence of an error-rate floor down to $P_b = 10^{-9}$.

VII. Concluding Remarks

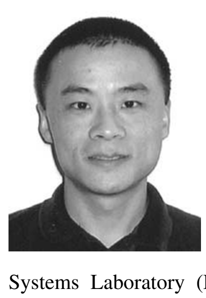
We have presented the class of extended IRA codes, making connections to irregular LDPC codes and serial turbo codes. We have presented an algorithm for the design of eIRA codes of moderate length and high rate which possess no error-rate floor down to $P_b = 10^{-9}$. To our knowledge, no other codes with such characteristics can be found in the literature. This paper represents a valuable step toward the design of codes for magnetic and optical data storage where a BER of 10^{-15} is often quoted, or optical communications where error rates below 10^{-10} are often quoted. Further research in this area includes support for the conjecture that low-weight columns in H (specifically, the submatrix H_1) lead to a small $d_{\min}$. Also, given that we have avoided only length-four cycles, the experimental results presented imply that short cycles (e.g., length six) and other graphical configurations do not have as much of an influence on the level of the floor as does $d_{\min}$, at least for high-rate codes. This was first pointed out by the work of Lin *et al.* [18]. It was shown in [19], however, that small girth values can have a tremendous effect on the performance of the code when the code rate is lower and/or the length is smaller than those considered here.

Acknowledgment

The authors would like to acknowledge J. Hou and P. Siegel of the University of California, San Diego, for help with the initial density evolution program.

References

[1] D. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, Mar. 1999.
[2] "Special issue on codes on graphs and iterative algorithms," *IEEE Trans. Inform. Theory*, vol. 47, pp. 493–849, Feb. 2001.
[3] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619–637, Feb. 2001.
[4] S. Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, pp. 58–60, Feb. 2001.
[5] T. J. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 638–656, Feb. 2001.
[6] S. Y. Chung, "On the construction of some capacity-approaching coding schemes," Ph.D. dissertation, Dept. Elect. Eng., Mass. Inst. Technol., Cambridge, MA, 2000.
[7] D. Divsalar, S. Dolinar, and F. Pollara, "Iterative turbo decoder analysis based on density evolution," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 891–907, May 2001.
[8] H. X. Song, M. Richard, and J. Cruz, "Low-density parity-check codes for magnetic recording channels," *IEEE Trans. Magn.*, vol. 36, pp. 2183–2186, Sept. 2000.
[9] K. Rice and R. Storn, "Difference evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimiz.*, vol. 11, pp. 341–359, Sept. 1997.
[10] M. Chiani and A. Ventura, "Design and performance evaluation of some high-rate irregular low-density parity-check codes," in *Proc. IEEE GlobeCom Conf.*, Nov. 2001, pp. 990–994.
[11] H. Jin, A. Khandekar, and R. McEliece, "Irregular repeat-accumulate codes," in *Proc. 2nd. Int. Symp. Turbo Codes and Related Topics*, Brest, France, Sept. 2000, pp. 1–8.
[12] R. Narayanaswami, "Coded modulation with low-density parity-check codes," M.S. thesis, Dept. Elect. Eng., Texas A&M Univ., College Station, TX, 2001.
[13] J. Hou, P. H. Siegel, and L. B. Milstein, "Performance analysis and code optimization of low-density parity-check codes on Rayleigh fading channels," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 924–934, May 2001.
[14] [Online]. Available: http://www.icsi.berkeley.edu/ storn/code.html
[15] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: A rediscovery and new results," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2711–2736, Nov. 2001.
[16] D. J. C. MacKay. Parity-check matrices. [Online]. Available: http://wol.ra.phy.cam.ac.uk/mackay
[17] D. MacKay and M. Davey, "Evaluation of Gallager codes for short block length and high-rate applications," in *Codes, Systems, and Graphical Models*. New York: Spring-Verlag, 2000, vol. 123, IMA Vols. in Math. and Applicats., pp. 113–130.
[18] H. Tang, J. Xu, S. Lin, and K. Abdel-Ghaffar, "Codes on finite geometries," *IEEE Trans. Inform. Theory*, submitted for publication.
[19] J. Campello and D. S. Modha, "Extended bit-filling and LDPC code design," in *Proc. IEEE Global Telecommunications Conf.*, Nov. 2001, pp. 985–989.



Michael Yang (S'01) received the B.S. and M.S. degrees in electrical and communication systems from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1993 and 1996, respectively. He is currently working toward the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ.

From 1996 to 1999, he was with the Shenzhen Huawei Technology, Inc., Shenzhen, China, working on SDH optical transmission systems. He is currently a Research Assistant in the Digital Communication Systems Laboratory (DCSL), University of Arizona. His current research interests include digital communications and signal processing, with emphases on error-control coding on graphs and iterative algorithms, and its application to data storage and transmission.

William E. Ryan (S'83–M'84–SM'01) received the B.S.E.E. degree from Case Western Reserve University, Cleveland, OH, in 1981, and the M.S. and Ph.D. degrees in electrical engineering from the University of Virginia, Charlottesville, in 1984 and 1988, respectively.

He was with The Analytic Sciences Corporation, Ampex Corporation, and Applied Signal Technology prior to his positions in academia. From 1993 to 1998, he was with the Electrical and Computer Engineering Department faculty at New Mexico State University, Las Cruces. Since August 1998, he has been with the Electrical and Computer Engineering Department at the University of Arizona, Tucson, where he is an Associate Professor. His research interests are in coding and signal processing for data transmission and storage.

Dr. Ryan is an Associate Editor of Modulation, Coding, and Equalization for the IEEE TRANSACTIONS ON COMMUNICATIONS.

Yan Li (S'02) received the B.S. and M.S. degrees in electrical and communication systems from Xi'an Jiaotong University, Xi'an, China, in 1996 and 1999, respectively. He is currently working toward the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Arizona, Tucson.

Previously, he was with Huawei Technology, Inc., Shanghai, China, and worked on wideband code-division multiple-access systems. From 2000 to the present, he has been a Research Assistant in the Digital Communication Systems Laboratory (DCSL), University of Arizona. His current research interests include coding, modulation, and signal processing in digital communications.