Multiple Rate Low-Density Parity-Check Codes with Constant Blocklength

Andres I. Vila Casado, Wen-Yen Weng and Richard D. Wesel
Department of Electrical Engineering, University of California, Los Angeles, CA 90095-1594
Email: avila@ee.ucla.edu, wenyen@ee.ucla.edu, wesel@ee.ucla.edu

Abstract— **This paper presents a new method to design low-density parity-check codes for a variety of different rates that all share the same fundamental decoder architecture. Combining rows of the parity-check matrix for the lowest rate code produces the parity-check matrices for higher rates. An important advantage of this approach is that all effective code rates have the same blocklength. These LDPC codes also share the same variable degree distribution. The proposed design method maintains good graphical properties and hence low error floors for all rates. Furthermore, an imposed matrix structure facilitates a low complexity encoding and decoding of the codes.**

I. Introduction

PRACTICAL communication systems often need to operate at several different rates. To keep the implementation as simple as possible, the same basic hardware architecture should be able to decode the encoded data at all possible rates. One way to achieve this with low-density parity-check (LDPC) codes is to generate higher-rate codes by puncturing lower-rate codes as proposed in [1] and [2]. However, puncturing reduces the code blocklength, which degrades performance. For the highest rate codes where the puncturing is most severe, the performance degradation is significant when compared to an LDPC code with the original blocklength.

Another way to achieve this is to generate lower-rate codes by shortening higher-rate codes, as described in [2]. As with puncturing, shortening reduces the code blocklength, which degrades performance. For the lowest-rate codes where the shortening is most severe, the performance degradation is significant when compared to an LDPC code with the original blocklength.

This paper introduces the concept of Constant Blocklength Multiple Rate (CBMR) codes which are LDPC codes that share the same fundamental structure while having an identical code blocklength and different code rates. The basic idea is to generate higher rate codes (called "effective" codes in this paper) from a low-rate code (called the "mother" code in this paper) by reducing the number of rows in its parity check matrix. From an implementation point of view, rows in the parity check matrix correspond to check nodes. Reducing the number of rows by linearly combining rows is equivalent to replacing a group of check nodes with a single check node that sums all the edges coming into each of the original check nodes as seen in Fig. 1.

This work was supported by the state of California and three corporations; ST Microelectronics, Conexant, and Texas Instruments, through UC discovery grant COMO3-10142, UC MICRO grant 03-92, and UC MICRO grant 03-93 respectively

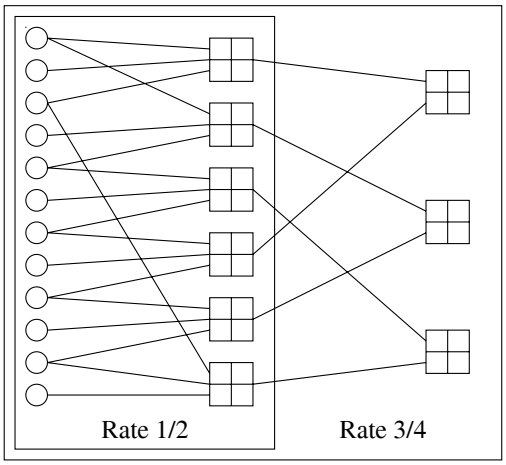

Fig. 1. Graph of a rate-3/4 LDPC code obtained from a rate 1/2 LDPC code via row-combining

Section II describes in detail the row combining approach used to generate CBMR codes. Section III explains how to apply graph conditioning techniques in the design of CBMR codes. Section IV describes how to lower the complexity of the encoder and decoder of CBMR codes. Section V explains how information nulling combined with row-combining can help to generate different blocklength codes and how this can be applied to the problem of bit padding. Section VI compares the performance of CBMR codes with the performance of single rate codes. Section VII delivers the conclusions.

II. Constant Blocklength Multiple Rate (CBMR) Codes

Consider the example "mother" LDPC matrix shown in (1),

$$H_{\frac{1}{2}} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \quad (1)$$

This is a rate-1/2 "mother" LDPC matrix with blocklength 12 whose graph representation can be seen in Fig.

1. This is by no means a good LDPC code but the reader should see it as an example to explain row-combining. Fig. 1 also shows that replacing each pair of nodes with a new single node transforms this rate-1/2 code into a rate-3/4 code. This is equivalent to summing the rows of the "mother" LDPC matrix that correspond to the check nodes that were combined if the check nodes do not have any common neighbors. This means that the "mother" matrix has to be designed so that the rows that will be combined don't have ones in the same column.

The following is the "effective" rate-3/4 LDPC matrix that resulted from the row-combining described in Fig. 1, where the resulting row 1 comes from combining rows 1 and 3 of the "mother" matrix, row 2 comes from combining rows 2 and 4 and row three results from the combination of rows 3 and 6,

$$H_{\frac{3}{4}} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \quad (2)$$

It is easy to see that many different rates can be obtained from the same "mother" code by changing the way rows are combined. For example using the "mother" code described in (1), three rows at a time can be combined to generate the rate-5/6 LDPC matrix shown in (3). In this rate-5/6 matrix the first row results from combining the odd rows of the "mother" matrix and the second row results from combining the even rows of the "mother" matrix,

$$H_{\frac{5}{6}} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}. \quad (3)$$

In the previous examples all the rows of the "mother" LDPC code have been combined in order to generate the new codes. However more rates can be achieved by combining just a fraction of the rows. For example to produce an "effective" code of rate 2/3 from the "mother" code described in (1), connect two thirds of the check nodes two at a time and leave the remaining check nodes alone. A possible rate-2/3 LDPC matrix that can result from the previously described row-combining is shown in (4),

$$H_{\frac{2}{3}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \quad (4)$$

This method changes the rate without changing the blocklength or the basic architecture of the decoder. Furthermore, the variable degree distribution remains the same as the rate changes and although in principle different rates may require different variable-node degree distributions for theoretical optimality, as stated in [3], a single variable node degree distribution can be found in practice that works well for the "mother" code and all the "effective" codes.

A concentrated degree distribution is a degree distribution in which every node has the same degree. In principle, concentrated check node degree distributions are desirable for theoretical optimality [3]. If the check node degree distribution of the "mother" code is concentrated, then the check node degree distribution for the higher-rate "effective" code will also be concentrated if all the rows in the "effective" LDPC matrix result from combining the same number of rows of the "mother" LDPC matrix. In the previous examples, the "effective" codes of rate 3/4 and rate 5/6 have a concentrated degree distribution. However, for many rates it may not be possible to maintain a concentrated check-node degree distribution. For example, the rate-2/3 code described previously does not maintain a concentrated check node degree distribution.

III. Graph Conditioning

The performance of the LDPC codes is limited by the fact that their graphs contain cycles which compromise the optimality of the belief propagation decoding. These cycles generate error floors in the performance of LDPC codes in the high SNR regions. However, the negative effect of the cycles can be reduced using graph conditioning techniques as those described in [4], [5] and [6]. This section will explain a couple of graph conditioning techniques and show how they can be used in CBMR codes.

As explained in [4] not all cycles degrade the performance of the code in the same way. Among the most dangerous structures that can be found in LDPC bi-partite graphs are stopping sets. A stopping set is a variable node set where all its neighbors are connected to the set at least twice. This implies that if all the variable nodes that belong to a stopping set are unreliable, the decoding will fail.

Small stopping sets must be avoided. This is a very complex problem to attack directly. To indirectly increase the size of stoping sets, [4] proposes the ACE algorithm which is based on maximizing the ACE metric of small cycles. The ACE metric of a cycle is the sum of the number of neighbors of each of the variable nodes in the cycle minus two times the number of variable nodes in the cycle.

According to this algorithm, the LDPC matrix is constructed by generating a single column randomly until one is found where all the cycles of length less than or equal to a previously specified threshold (denoted as $2d_{ACE}$) that contain its corresponding variable node have an ACE metric higher than or equal to another previously specified threshold (denoted as η_{ACE}). This process is done with all the columns starting from the one with the lowest degree.

The constraints specified in [5] also help to avoid small stopping sets in the graph, particularly if applied to the high-degree columns. In [5] two new metrics are defined called β_c and β_p and they are the number singly-connected check nodes to a cycle or a path respectively. In the same manner as the ACE algorithm, random columns are generated and they must satisfy some constraints based on previously specified thresholds which are denoted here as d_{SS}, γ_c and γ_p. Specifically, a randomly generated column is valid if all the cycles of length less than or equal to $2d_{SS}$ that contain its corresponding variable node have a β_c metric of value higher than or equal to γ_c and if all

the paths of length less than or equal to d_{SS} that contain its corresponding variable node have a β_p metric of value higher than or equal to γ_p.

CBMR codes can be generated so that all the effective codes have good graph properties by naturally extending the previously described graph conditioning algorithms. Graph conditioning must be applied simultaneously to all the codes. This implies that once the blocklength, rates and degree distribution have been chosen, the "mother" matrix rows to be combined in order to generate the "effective" matrices must be selected. Section IV will explain some important criteria to be taken into account when selecting these rows in order to maintain the structure of the code.

The generation of the "mother" and "effective" matrices is done using simultaneous graph conditioning. This means that the previously described column by column generation is still used and every column generated must satisfy the graph constraints specified for the "mother" code and all the "effective" codes. Different graph constraints can be used for different matrices which is necessary because the achievable values of such constraints change with the rate as shown in [7].

IV. Low Complexity Encoding and Decoding

A. Block Structure for Parallel Decoding

In order to have parallel processing to decrease decoder complexity, LDPC codes can be designed to have an inherent structure based on the ideas presented by Mansour and Shanbag in [8]. In [8] the LDPC matrices have a block structure so that it consists of a plurality of square sub-matrices of size p. Each said square sub-matrix is either a zero sub-matrix or a structured sub-matrix. This overall block structure facilitates parallel processing.

An example that illustrates the structured sub-matrices proposed in [8] is shown in (5). The sub-matrix labelled as S_2, is a cyclic shift of the columns of the identity matrix. Each sub-matrix S_i is produced by cyclically shifting the columns of an identity matrix to the right i places,

$$S_2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (5)$$

This structured LDPC allows the decoder to use at least p processors in parallel. It also diminishes the amount of memory needed to store the code since the only information necessary will be the size of the sub-matrices (p), the position of the S_i sub-matrices and the value of their cyclic shift (i).

B. Linear Encoding Using Back Substitution

The parity check matrix should also allow a simple encoder implementation. In [9] and [10] a linear encoder for LDPC codes is found if its matrix is composed by two matrices $H_0 = [H_1|H_2]$ where H_2 has a particular structure. In [9] H_2 is a double diagonal matrix so the encoding process is simply done by multiplying the input vector by H_1^T and then processed by an accumulator.

Unfortunately it is difficult or impossible to maintain a bi-diagonal structure for the H_2 portion of the "mother" LDPC matrix and all of the "effective" LDPC matrices in the context of row combining. The solution found to this problem was to use a model presented in [10] where H_2 is a lower triangular matrix. This will allow a linear encoder based on back-substitution as explained in [10]. Back substitution consists on solving a sequence of linear equations in order to obtain the parity bits. This structure can be maintained for all the rates, as will be shown in the following subsection.

C. Structured CBMR Codes

The challenge presented when trying to design a structured CBMR code is that the "mother" code and all the "effective" codes must have both the sub-matrix and the lower triangular structures. There is an easy way to maintain the sub-matrix structure for all the "effective" codes. Instead of combining individual rows, rows of sub-matrices will be combined so that if sub-matrix A and sub-matrix B are combined it implies that row i of A and row i of B will be combined for $i = \{1...p\}$. The resulting sub-matrix will be equivalent to the superposition of sub-matrix A and sub-matrix B.

If the exact sub-matrix structure of [8] is to be maintained for all the "effective" codes, the "mother" matrix has to be designed so that among the combined sub-matrices at most only one of them has the S_i structure and the rest of them are zero sub-matrices. However, the sub-matrix that results from the superposition of two or more S_i sub-matrices also has good parallel properties so this design criteria can be relaxed to include such superpositions.

Now with a careful selection of the rows to be combined a block lower triangular structure can be maintained on all the "effective" codes. The following is a possible way to select the rows to combine in order to preserve the block lower triangular structure of H_2 through all the rates, although the dimensions are changing with each rate. In this example the "mother" matrix is a rate-1/2 LDPC matrix and the "effective" LDPC codes will have rates 2/3, 3/4 and 5/6.

Row i of the parity check matrix of the rate 3/4 code will be generated by summing row i of the parity check matrix of the rate 1/2 code plus the row with index $i+M/2$ where M is the total number of check equations in the rate 1/2 code. The rate 2/3 matrix is generated by doing the previously described sum only for the rows where $i < M/3$ and leaving the rest of the rows intact. Likewise row i of the parity check matrix of the rate 5/6 code will be generated by summing row i of the parity check matrix of the rate

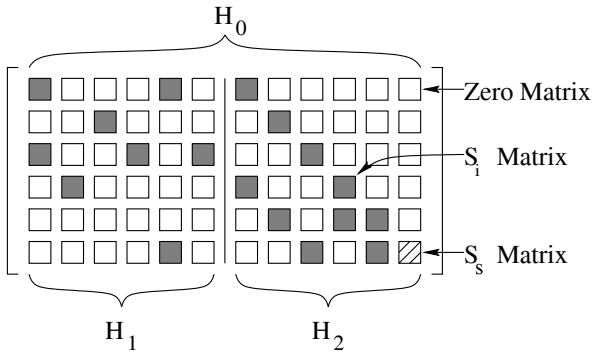

Fig. 2. Structured LDPC matrix

$1/2$ code plus the row with index $i+M/3$ plus the row with index $i+2*M/3$.

The overall sub-matrix structure with a block lower triangular H_2 is shown in Fig. 2 where the sub-matrices along the diagonal of H_2 will have the S_i structure except for the bottom right matrix which will have a staircase structure described in (6). The staircase structure is necessary in order to maintain the back substitution encoding without having p degree one nodes,

$$S_s = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \quad (6)$$

Fig. 3 shows the BER and FER of a structured CBMR code generated using the simultaneous graph conditioning algorithm described in section III. The "mother" code has rate $1/2$ while the three "effective" codes have rates $2/3$, $3/4$ and $5/6$. The blocklength of the codes is 1944 bits, the size of the sub-matrices is 27 and a maximum of 200 iterations were used in the simulation. These are hardware simulation results obtained using the JPL Universal Decoder for Sparse Codes. The codes show a good performance specially in the error floor region which remains consistently low in all the rates. Table I shows the gap from capacity of the previously described CBMR codes.

V. Information Nulling

Deleting one or more columns in a LDPC matrix generates a lower rate code with less blocklength. This approach is known as information nulling or shortening and is well documented in [2]. The code generated will satisfy the graph conditioning constraints of the "mother" code and its girth will be higher than or equal to the girth of the "mother" code. The decoding hardware can remain the same by assuming that a zero was received with infinite reliability in the bits that were shortened.

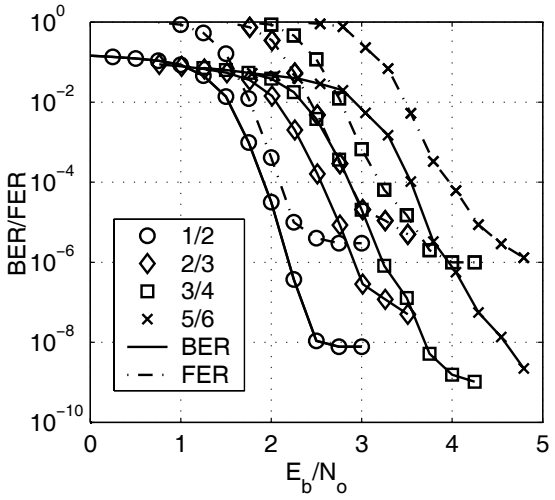

Fig. 3. Performance of a blocklength 1944 CBMR code on an AWGN channel

TABLE I
Gaps in SNR from Shannon capacity and excess mutual information of a blocklength 1944 CBMR code

Rate	FER=10^{-4}		BER=10^{-6}	
	SNR[dB]	EMI[bits]	SNR[dB]	EMI[bits]
1/2	2.1	.39	2.2	.41
2/3	1	.22	1.1	.23
3/4	.58	.13	.61	.13
5/6	.59	.14	.59	.14

This implies that wide variety of codes with different blocklengths and rates can be generated from the same "mother" matrix by using the row-combining approach and the information nulling technique. A careful selection of the variable nodes to cancel will generate codes with good graph properties and good degree distributions. It is important to notice that all the codes generated using row-combining and information nulling share the same decoding hardware.

Shortened codes can maintain the good structural properties of the "mother" code. In order to maintain the sub-matrix structure columns should be deleted in groups of p so that whole sub-matrices are deleted. Also H_2 must remain block lower triangular so none of its columns can be deleted

An important application of information nulling is bit padding. In communication systems, sometimes there are fewer bits to send than the number of information bits of the code, for example in the last frame of a data packet. A solution to this problem for LDPC codes is to shorten the remaining information bits. During the decoding this last frame the code has a lower information rate, which implies that fewer iterations can be used and still obtain the same performance at the same SNR. Fig. 4 shows the performance on a AWGN channel of the rate-5/6 CBMR code

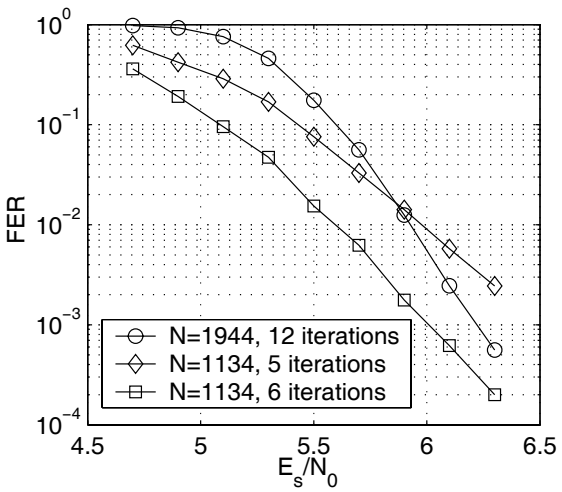

Fig. 4. Performance comparison of a CBMR rate-5/6 code and the rate-5/7 code generated by shortening half the information bits

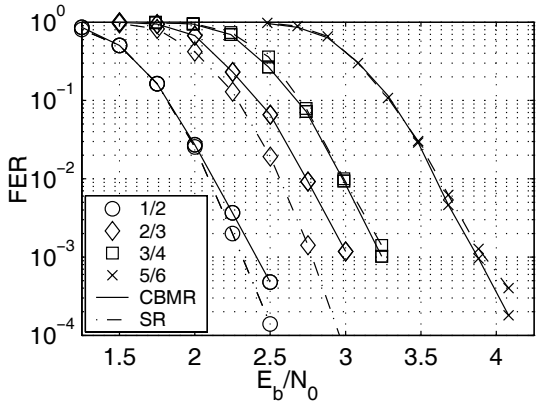

Fig. 5. Performance of CBMR and SR codes with blocklength 1944 on an AWGN channel

previously generated. It can also be seen that by shortening half the information bits the resulting code with rate 5/7 and blocklength 1134 can obtain the same performance using half the iterations.

VI. Performance Comparison

This section presents a comparison in the performance on an AWGN channel of structured CBMR codes with respect to the performance of unstructured single rate (SR) LDPC codes. Fig. 5 shows the performance of the previously designed structured CBMR code and the performance of four unstructured single rate codes with rates corresponding to those of the "mother" and all the "effective" codes of the CBMR code. These single rate codes have the same blocklength of the CBMR code and were generated using graph conditioning. A maximum of 15 iterations was used in the simulation of the codes.

As observed in Fig. 5 the CBMR codes perform very well at this blocklength. It is also noticeable that the CBMR code at rate-2/3 does not perform as well as the CBMR codes for the other rates. This is probably due to the fact that in order to generate the "effective" rate-2/3 LDPC matrix, 2/3 of the check nodes are combined and 1/3 of the nodes are left intact thus half the rate-2/3 check nodes will have a degree twice as big as the other half. This goes against density evolution results which state that the check node degree distribution should be as concentrated as possible.

VII. Conclusions

Multiple-rate LDPC codes can be generated using a row-combining approach. The advantage of the approach is that the codes have the same blocklength thus maintain a good performance at all the rates. Furthermore, structures that reduce complexity of both encoder and decoder can be applied and maintained through all the rates. Finally, graph conditioning algorithms can be applied to the design of the codes to produce good error floor performance at all the rates.

VIII. Acknowledgments

The authors would like to thank Dr. Christopher R. Jones of JPL for providing the hardware simulation of the CBMR codes, and thank Massimiliano Siti, Stefano Valle and Nicola Moschini of ST Microelectronics for all their help and encouragement.

References

[1] J.Ha and S.W. McLaughlin. Rate-Compatible Puncturing of Low-Density Parity-Check Codes. In *IEEE Trans. on Info. Th.*, volume 50, pages 2824–2836, November 2004.

[2] T. Tian, C. Jones, and J. Villasenor. Rate-Compatible Low-Density Parity-Check Codes. In *ISIT 2004*, Chicago, July 2004.

[3] T. Richardson, A. Shokrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. In *IEEE Trans. Inform. Theory*, volume 47, pages 619–637, February 2001.

[4] T. Tian, C. Jones, J. Villasenor, and R. Wesel. Avoidance of Cycles in Irregular LDPCC Construction. In *IEEE Transactions on Communications*, August 2004.

[5] A. Ramamoorthy and R. D. Wesel. Construction of Short Block Length Irregular LDPCCs. In *Proc. IEEE ICC 2004*, Paris, France, June 2004.

[6] Xiao Yu Hu, Evangelos Eleftherioua, and Dieter Michael Arnold. Progressive edge-growth tanner graphs. In *GLOBECOM, The Evolving Global Communications Network*, pages 995–1001, San Antonio, Texas, November 2001.

[7] W. Weng, A. Ramamoorthy, and R. Wesel. Lowering the error floors of irregular high-rate ldpc codes by graph conditioning. In *VTC*, Los Angeles, California, September 2004.

[8] M. M. Mansour and N. R. Shanbhag. Low Power VLSI Decoder Architectures for LDPCCs. In *2002 International Low Power Electronics and Design*, pages . 284–289, June 2002.

[9] M. Yang, W. E. Ryan, and Y. Li. Design of Efficiently Encodable Moderate-LengthHigh-Rate Irregular LDPC Codes. In *IEEE Trans. on Comm.*, volume 52, pages 564–571, April 2004.

[10] T. Richardson and R. Urbanke. Efficient encoding of low-density parity-check codes. In *IEEE Trans. Inform. Theory*, volume 47, pages 638–656, February 2001.