Low-rate LDPC codes with simple protograph structure

Dariush Divsalar
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109-8099, USA
Email: Dariush.Divsalar@jpl.nasa.gov

Sam Dolinar
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109-8099, USA
Email: sam@shannon.jpl.nasa.gov

Christopher Jones
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109-8099, USA
Email: christop@jpl.nasa.gov

***Abstract*— This paper provides a construction method for low-rate low density parity check codes. Inspired by recently proposed Accumulate-Repeat-Accumulate (ARA) codes, and hybrid concatenated codes, in this paper we extend the construction to low rates. Such codes can be viewed as hybrid concatenations of simple modules such as accumulators, repetition codes, differentiators, and punctured single parity check codes. These codes constitute a subclass of LDPC codes with very fast encoder structure. They also have a projected graph or protograph representation that allows for high-speed decoder implementation. Based on density evolution, we show through some examples that low iterative decoding thresholds close to the channel capacity limits can be achieved, as the block size goes to infinity. Iterative decoding simulation results for short blocks are provided for a few examples that show near-capacity performance and very low error floor.**

I. Introduction

Low-density parity-check (LDPC) codes were proposed by Gallager [1] in 1962. After introduction of turbo codes by Berrou et al [2] in 1993, researchers revisited LDPC codes, and extended the work of Gallager using the code graphs introduced by Tanner [3] in 1981. After 1993 there have been many contributions to the analysis of LDPC codes; see for example [9], [11], [12], and references there.

Repeat-Accumulate (RA) [5], Irregular Repeat-Accumulate (IRA) [6] and recently Accumulate-Repeat-Accumulate (ARA) [14] codes have been proposed as simple subclasses of LDPC codes with fast encoder structures. For high-speed decoding, it is advantageous for an LDPC code to be constructed from a projected graph [8], or protograph [7]. Multi-edge-type LDPC codes were introduced in [15], [10] and should be considered as superclass for unstractured irregular LDPC and Prograph based LDPC codes. A protograph is a Tanner graph with a relatively small number of nodes.

As a simple example, we consider the protograph shown in Fig. 1. This graph consists of 3 variable nodes and 2 check nodes, connected by 5 edges. We can obtain a larger graph by a "copy-and-permute" operation as shown in Fig. 1. This operation consists of first making T copies of the protograph, and then permuting the endpoints of each edge among the T variable and T check nodes connected to the set of T edges copied from the same edge in the protograph. The derived graph is the graph of a code T times as large as the code corresponding to the protograph, with the same rate and the same distribution of variable and check node degrees. Thresholds for infinite block size are computed using density evolution on the protograph.

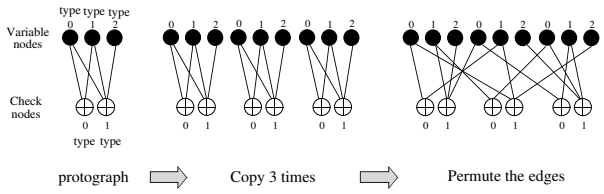


Fig. 1. Copy and Permute operation to generate larger graphs

RA, IRA, and ARA codes, all have simple protograph representations. In this paper we define further extensions to low-rate LDPC codes. These extensions provide greater flexibility to construct codes with lower decoding thresholds and lower error floors.

A. Richardson-Urbanke construction of low-rate LDPC codes

Richardson and Urbanke [15] proposed to construct low-rate LDPC codes by serial concatenation of a high-rate regular LDPC code as an outer code with an LT code [16] as an inner code; this creates a Raptor-type code [17]. All nodes of the outer LDPC code are also transmitted through the channel. They provided one example of a rate-1/10 multi-edge-type LDPC code with threshold -1.09 dB on the AWGN channel. We constructed several similar protograph codes with degree distributions matching the multi-edge-type degree distributions listed in their table in [15]. For example, Figure 2 gives a simple rate-1/10 protograph matching their degree distributions and achieving a threshold of -0.976 dB. (In this graph, double and triple parallel edges are denoted by single lines with adjacent multiplicity labels.) This particular protograph has 40 variable nodes and 35 check nodes and a total of 155 edges. Other protograph realizations may come closer to Richardson and Urbanke's threshold of -1.09 dB, but we show the one in Fig. 2 because its semi-regular structure makes it relatively easy to draw. Still, this is a fairly large and complex protograph. In this paper we propose alternative simpler low-rate protographs that achieve near-capacity thresholds.

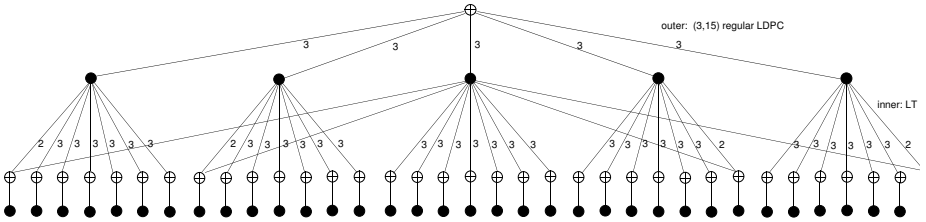


Fig. 2. protograph of rate-1/10 version of Richardson-Urbanke LDPC code

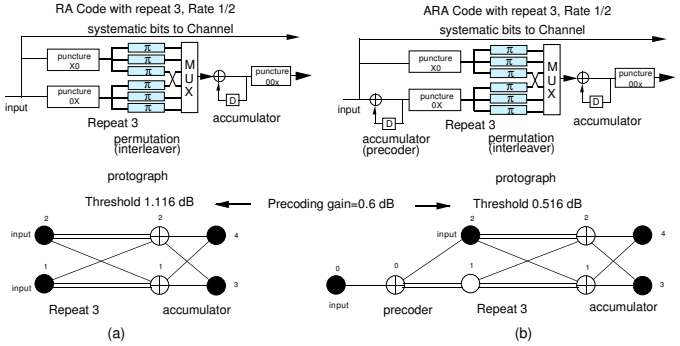
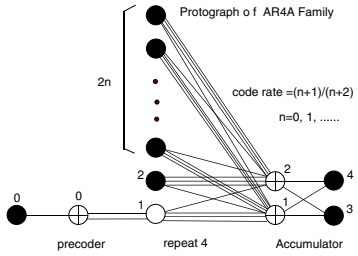


Fig. 3. (a) A rate-1/2 systematic punctured RA code,(b) A rate-1/2 AR3A code and their protographs.



Code Rate	Protograph Threshold	Capacity	Difference
1/2	0.560	0.187	0.373
2/3	1.414	1.059	0.355
3/4	1.980	1.626	0.354
4/5	2.396	2.040	0.356
5/6	2.717	2.362	0.355
6/7	2.980	2.625	0.355
7/8	3.197	2.845	0.352
8/9	3.385	3.033	0.352

Fig. 4. Protograph and thresholds for AR4A Family.

B. Background on ARA codes

Classical RA codes, in addition to simplicity, have reasonably good performance, with iterative decoding threshold within 1 dB from capacity for rates less than or equal to 1/3. RA codes use fixed repetition for the input bits.

Jin et al [6] generalized the notion of RA codes by allowing irregular repetition of the input bits. An Irregular RA (IRA) code can be viewed as a serial concatenation [13] of a simple low density generator matrix (LDGM) code with different-degree variable nodes (irregular repetition) as an outer code, and an accumulator as an inner code.

For IRA codes the node degree distribution can be optimized to achieve low thresholds. However, to achieve a very low threshold, the maximum repetition for some portion of the input bits can be very high, Similar requirements on the maximum variable node degree have been noted for a general irregular LDPC code [4] to achieve a very low threshold. ARA codes, on the other hand, can achieve a very low threshold (e.g., within 0.08 dB from the capacity limit for rate-1/2 codes) with variable and check nodes of low maximum degree.

Consider the rate-1/2 systematic punctured RA code with repetition 3, and puncturing period 3, shown in Fig. 3(a). In [14] it was shown that the threshold can be further improved by precoding the repetition code by an accumulator. An RA code with an accumulator precoder is called an Accumulate-Repeat-Accumulate (ARA) code [14].

An example of a simple rate-1/2 ARA code, its protograph, and the corresponding threshold are shown in Fig. 3(b). Here the protograph in Fig. 3(b) is called AR3A, because it uses repetition-3. It uses a punctured accumulator as the precoder.

An ARA code with repetition-4 should achieve higher minimum distance and better error floor performance. We call these codes AR4A. Figure 4 shows protographs for a family of AR4A codes of rates 1/2 and higher, and their corresponding thresholds. Figure 5 shows FPGA simulated performance for AR4A codes expanded to size $k = 4096$.

II. Low-rate ARA-type LDPC codes

For a given number of nodes and checks in a protograph one can search over all possible connections between variable and check nodes to obtain a protograph with the lowest threshold. For a rate-1/3 LDPC with 4 variable nodes and 3 checks where one variable node is punctured, we found a protograph with threshold of $E_b/N_0 = -0.326$ dB. The same protograph can be encoded in two ways as shown in Figure 6. The first encoder is similar to an ARA encoder except for an additional single-parity-check code. The second encoder is a simple serial concatenation [13] of a rate-1/2 two-state convolutional code as an outer code and a punctured accumulator as inner code, where the parity output of the outer code is also connected through a permutation to a single-parity-check code.

Rather than searching we propose the following constructions extending the ARA families to low rates. In the AR3A or AR4A protographs shown in Figs. 3(b) and 4, respectively, we keep only the punctured variable node in the middle column of variable nodes, and delete the transmitted variable(s) in this column along with their associated edges. This produces

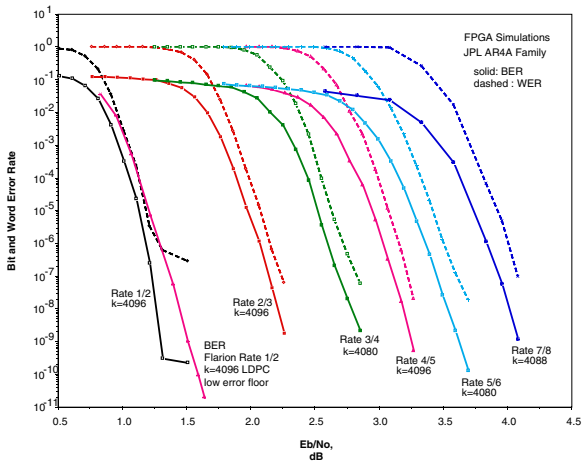


Fig. 5. Bit and word error rates for AR4A family with input block size $k = 4096$ bits, and rates 1/2 to 7/8.

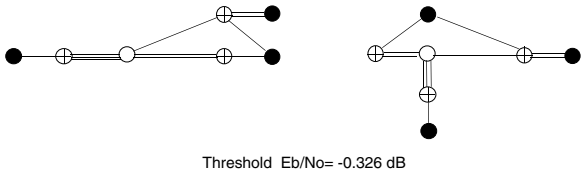


Fig. 6. A low-threshold rate-1/3 protograph.

rate-1/3 ARA protographs having 4 variables and 3 checks with one variable punctured. The two checks on the right are still connected to two variables forming an accumulator. The single check and single degree-1 variable on the left can then be replaced by a constellation of such check-variable pairs to achieve lower rates. Figures 7 through 12 show our constructed protographs in the AR3A and AR4A families, and their corresponding thresholds for rates 1/3 through 1/10. For the low-rate AR4A family, we have also replaced the single accumulator in Fig. 4 with multiple parallel accumulators.

For rates 1/6 through 1/10, it becomes profitable to connect the degree-4 punctured variable node in the AR4A protograph

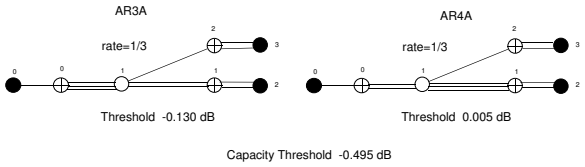


Fig. 7. Rate-1/3 AR3A and AR4A protographs.

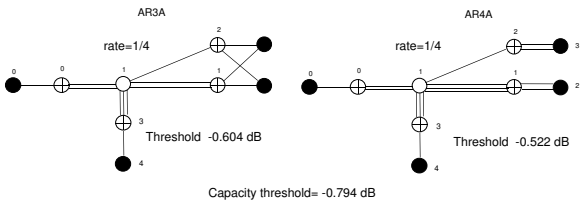


Fig. 8. Rate-1/4 AR3A and AR4A protographs.

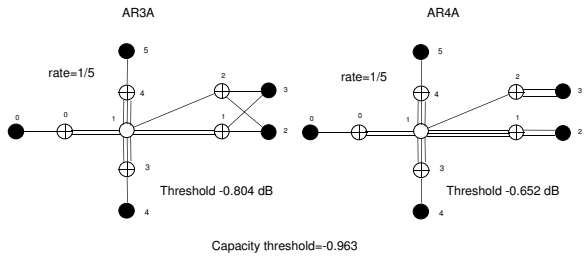


Fig. 9. Rate-1/5 AR3A and AR4A protographs.

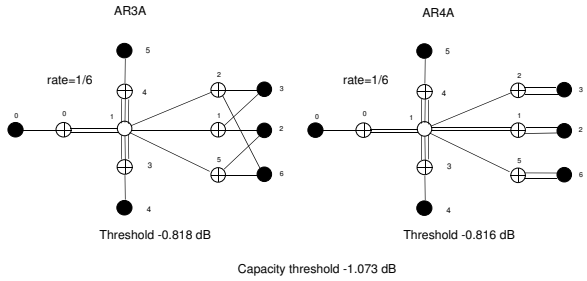


Fig. 10. Rate-1/6 AR3A and AR4A protographs.

to more than two check nodes and parallel accumulators. A three-accumulator configuration achieves the best threshold for rate 1/6, and a four-accumulator configuration is best for rates 1/8 and 1/10.

The constructions in Figures 7 through 12 can be regarded as hybrid concatenated codes [18] where the outer code is a repetition code, the inner code is an accumulator with possible puncturing, and the parallel code is a low-density generator matrix (LDGM) code. The simplest version of LDGM code is implemented by a differentiator and single-parity-check codes with 3 inputs and one parity bit. In our construction we used repetition-3 (AR3A family) for lowest threshold and repetition-4 (AR4A family) for lower error floor performance.

To lower the threshold further, we should use more complex LDGM code. In this case it is better to start with a rate 1/2 ARA code. Here we provide an example for rate 1/3 as shown in Figures 13, using more complex LDGM codes. Codes with lower than 1/3 rates can be constructed in a similar way.

III. Low-rate ARJA-type LDPC codes

In an ARA code protograph the number of degree 2 variable nodes is equal to the number of inner checks (checks that are

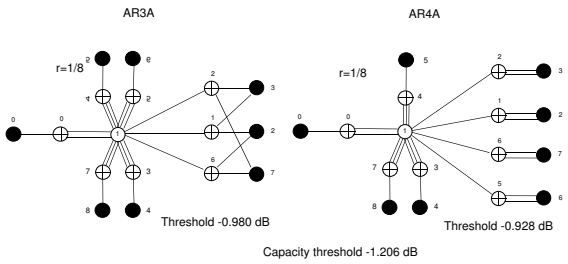


Fig. 11. Rate-1/8 AR3A and AR4A protographs.

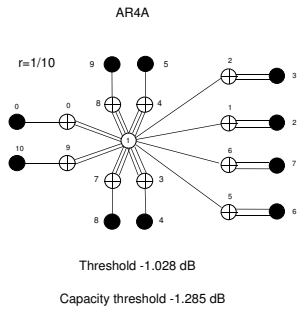


Fig. 12. Rate-1/10 AR4A protograph.

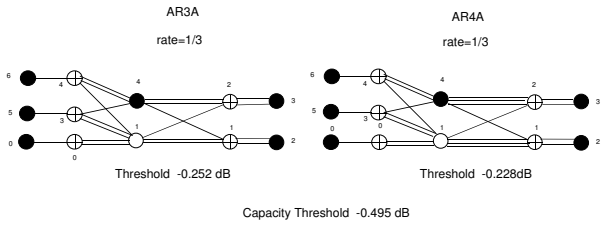


Fig. 13. Rate-1/3 more complex AR3A and AR4A protographs.

connected to these degree 2 variable nodes). If we decrease the number of degree 2 variable nodes with respect to inner checks, then the ensemble asymptotic minimum distance of code may grow with n. For example if we replace 50% of degree 2 variable nodes with degree 3 variable nodes, then the minimum distance grows with n. We call such constructed codes Accumulate Repeat Jagged Accumulate (ARJA) codes.

An example of a simple rate-1/2 ARJA code, its protograph, and the corresponding threshold are shown in Fig. 14. we have found that asymptotically $d_{min} = 0.015 \times$ block size for this protograph code. Higher code rates are constructed similar to the protograph in Fig. 4, by replacing rate-1/2 AR4A base protograph with rate-1/2 ARJA base protograph in Fig. 14.

We use the ARJA protograph shown in Fig. 14 to construct lower rate codes, analogous to the construction method described for rate 1/3 codes in Figures 13. This construction produces a rate-1/3 ARJA protograph having 7 variables and 5 checks with one variable punctured. The two checks on the right are still connected to two variables forming a jagged accumulator. The single check and single degree-1 variable on the left representing the precoder is also untouched. Thus the rate 1/2 ARJA base protograph is unchanged to preserve the code family structure. Figures 15 shows our constructed protographs in the ARJA family, and the corresponding thresholds and Shannon capacities for rates 1/3 and 1/4.

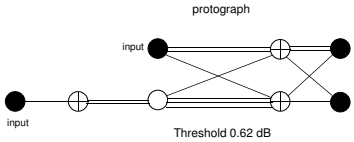


Fig. 14. A rate-1/2 ARJA code and its protograph. The asymptotic minimum distance over block size of this code is 0.015

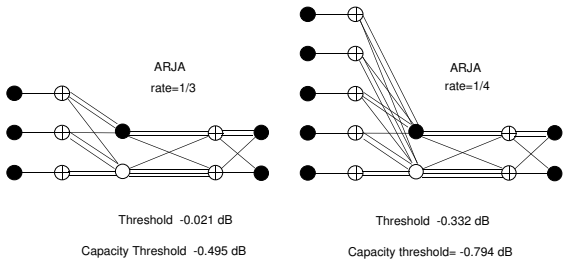


Fig. 15. Rate-1/3 and Rate-1/4 ARJA protographs.

IV. Performance Results for short blocks

The AR4A code family for input lengths $k = 1024$ and $k = 4096$ were "lifted" from protographs using a hybrid of the Progressive Edge Growth (PEG) algorithm [21] and the ACE algorithm [19]. Specifically PEG is used to expand the initial protograph by a small factor, typically 4 or 8. This initial expansion is done randomly with the only constraints being that connectivity is dictated by the protograph structure and that graph girth should be maximized greedily per the PEG algorithm. The second expansion employs shifted identity matrices, or circulants, where a given circulant phase is accepted or rejected based on a criterion known as Approximate Cycle EMD. For instance, the rate-1/6 $k = 1024$ code is lifted by a factor of 8 from a protograph that has $k = 1$ to form a small code with $k = 8$. This small code is then lifted using circulants of size 128 with ACE criteria $D_{ACE} = 26$ and $\eta_{ACE} = 24$. Because the reader may be less familiar with ACE than with PEG, we provide a brief description of the ACE algorithm.

The Approximate Cycle Extrinsic message degree (ACE) technique was developed in [19]. The extrinsic message degree (EMD) of a variable node set equals the number of constraint nodes that are singly connected to the set. The ACE algorithm increases the expected EMD of a variable node sets of a given cardinality [20] by examining graph cycles during bipartite graph construction and rejecting cycles that do not incur a threshold level of "extrinsic" connections. A key approximation in the algorithm is that all external connections to a variable node involved in the cycle are deemed extrinsic. Therefore, the ACE of a cycle can be found by simply summing the degree of each variable node in the cycle minus two. This ignores the fact that connections from a variable node that are immediately external to the cycle may in fact reconnect with the cycle via other graph loops. Constraints that connect back to the current cycle were clearly not singly connected to the cycle to begin with. It is in this way that ACE is only an approximation to the EMD of a given cycle structure. However, as shown in [19] the complexity of an ACE construction is linear in the block length of the code and its application has been shown to reduce error floors by 2 to 3 orders of magnitude as compared to randomly constructed graphs. In certain portions of the construction, the selected ACE criteria may not actually be feasible. In these cases, the ACE construction "backs off" by lowering D_{ACE} until the criteria can be satisfied. Table I summarized the random

TABLE I

rate	PEG	ACE	k	D_{ACE}	η_{ACE}
1/6	8	128	1024	26	24
1/4	8	128	1024	22	16
1/3	8	128	1024	18	14
1/2	4	128	1024	12	7
2/3	4	64	1024	9	5
4/5	4	32	1024	8	5
5/6	4	26	1040	8	4
7/8	4	19	1064	9	5

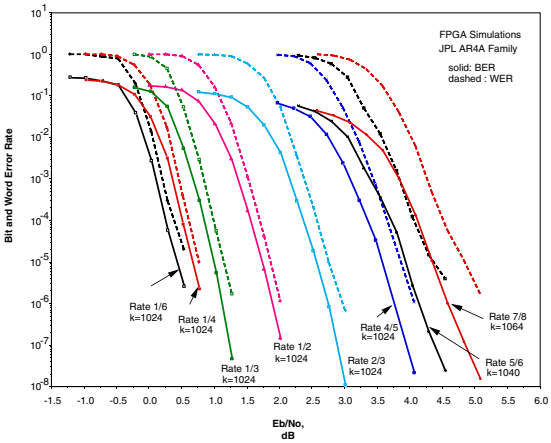


Fig. 16. Bit and word error rates for AR4A family for input block size $k = 1024$ bits, rates 1/6 to 7/8.

(PEG) and circulant (ACE) lifting steps used to construct the $k = 1024$ codes in this paper. The $k = 4096$ codes were lifted using a similar strategy. Note that the codes which result from this hybrid lifting are "structured" codes where the structure is inherent (from the point of view of a hardware decoder implementation) in the graph representation that results at the end of the first stage of lifting rather than in the initial protograph. FPGA simulation results for input block sizes of $k = 1024$ for codes in the AR4A family of rates 1/6 to 7/8 are shown in Figure 16.

FPGA simulation results for input block sizes of $k = 1024$ for codes in the ARJA family of rates 1/4 to 4/5 are shown in Figure 17. In the figure the performance of rate 1/2 ARJA with input block size of $k = 4096$ bits is also shown.

V. Conclusion

We have designed a set of very simple small ARA-type and ARA-type protographs for low-rate codes of rates 1/3 to 1/10 that achieve asymptotic low decoding thresholds.

Acknowledgment

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

References

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

[2] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. Commun.*, Vol. 44, pp. 1261–1271, October 1996.

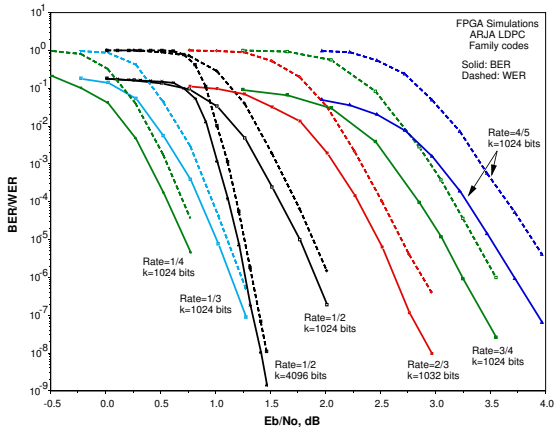


Fig. 17. Bit and word error rates for ARJA family for input block size $k = 1024$ bits, rates 1/4 to 4/5, and rate 1/2, $k = 4096$ bits for comparison

[3] M. R. Tanner, "A recursive approach to low complexity codes," IEEE Trans. Inform. Theory, vol. 27, pp. 533-547, 1981.

[4] 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, 2001.

[5] D. Divsalar, H. Jin, and R. McEliece, " Coding theorems for Turbo-like codes," in Proceedings of the 1998 Allerton Conference, pp. 201-210, 1998.

[6] H. Jin, A. Khandekar, and R. McEliece, " Irregular repeat-accumulate codes," in Proc. 2nd International Symposium on Turbo Codes, pp. 1-8, 2000.

[7] Jeremy Thorpe, " Low Density Parity Check (LDPC) Codes Constructed from Protographs," JPL INP Progress Report 42-154, August 15, 2003.

[8] Richardson , et al. , " Methods and apparatus for decoding LDPC codes," United States Patent 6,633,856, October 14, 2003.

[9] D.J.C. MacKay, R.M. Neal, "Near Shannon limit performance of low density parity check codes," Electronics Letters ,Vol. 32, Issue 18, 29 Aug. 1996, Page(s) 1645.

[10] T. Richardson and R. Urbanke,"The Renaissance of Gallager's Low-Density Parity-Check Codes," IEEE Communications Magazine, pages 126-131, August 2003

[11] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message- passing decoding," IEEE Trans. Inform. Theory, vol. 47, pp. 599-618, 2001.

[12] F.R. Kschischang, "Codes defined on graphs," IEEE Communications Magazine, Vol. 41, Issue 8 , Aug. 2003, Pages 118 -125.

[13] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, " Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," IEEE Trans. Info. Theory, vol. 44, pp. 909-926, May 1998.

[14] A. Abbasfar, D. Divsalar, and K. Yao,"Accumulate Repeat Accumulate Codes," ISIT 2004, and IEEE Globecom 2004.

[15] T. J. Richardson and R. Urbanke, "Multi-edge type LDPC codes," preprint. [Online]. Available: http://lthcwww.ep.ch/papers/multiedge.ps.

[16] M. Luby,"LT-codes," in Proceedings of the ACM Symposium on Foundations of Computer Science (FOCS), 2002.

[17] A. Shokrollahi, "Raptor codes," Proceedings 2004 IEEE International Symposium on Information Theory (ISIT2004), p. 36, Chicago, IL, USA, June 2004.

[18] D. Divsalar,and F. Pollara, "Hybrid concatenated codes and iterative decoding," Proceedings of 1997 IEEE International Symposium on Information Theory, page 10, June 29–July 4, 1997.

[19] Tian T., Jones C., Villasenor J., Wesel R. D.,"Characterization and selective avoidance of cycles in irregular LDPC code construction," IEEE Trans. on Comm., vol. 52, pages 1242–1247, Aug. 2004.

[20] Ramamoorthy A., Wesel R. D.,"Analysis of an Algorithm for Irregular LDPC Code Construction ," In Proceedings ISIT 2004, page 69, Chicago, Illinois.

[21] Arnold D. M., Eleftheriou E., Hu. X. Y., "Progressive edge-growth Tanner graphs,"In Proceedings IEEE Global Telecommunications Conf. 2001, Pages:995–1001 vol.2, San Antonio, TX, 25-29 Nov. 2001.