Combinatorial Constructions of LDPC Codes

P.S. Guinand and J. Lodge
Communications Research Centre, 3701 Carling Avenue, Ottawa Canada K2H 8S2
(tel) 613-998-2646, (fax) 613-990-6339, e-mail: paul.guinand@crc.ca.

Abstract — In this paper we study low density parity check (LDPC) codes where the structure of the equations arises from high girth graphs. We also discuss two modifications, recoding and optimization of scale factors, to the standard decoding methods for LDPC codes.

1 Introduction

Although codes consisting of systems of relatively short single parity check equations has been around for some time [1] there has been a recent resurgence of interest in these codes [2, 3]. As with Turbo codes, for high code rates these codes tend to suffer from an"error floor/flare" caused by low weight error events. It is possible [4, 5, 6] to design codes of this type which have greater minimum distance than randomly constructed codes of this type. Previously [7, 8] we have used a class of high girth bipartite graphs constructed by Lazebnik et. al. [9] to construct iteratively decodable codes whose constituent codes were Hamming codes. In this paper we use the same class of graphs to construct LDPC codes. Here LDPC is being used somewhat generically to mean codes which are defined by systems of single parity check equations which are relatively short compared to the total size of the code. We also describe some modified decoding strategies that are applicable to LDPC codes in general. The first of these involves decoding using one of the standard iterative decoding algorithms and if the decoding has failed to converge after a fixed number of iterations recoding the codeword based upon a set of log likelihood ratios (LLR) that indicate high reliability. The second modification involves the use of a scaling of the extrinsic which is iteration dependent in a Max Log APP decoder.

2 Description of the Codes and Graphs

The graphical representation of LDPC codes as bipartite graphs was introduced by Tanner [10]. One set of vertices in the graph corresponds to the bits in the code and the other set of vertices correspond to the parity checks. The edges of the graph indicate which bits are involved in which parity checks. Error patterns for codes specified in this manner correspond to subgraphs where on the variable side all edges coming out of a vertex involved in the error pattern are in the subgraph and on the check node side, the subgraph's nodes are all of even degree. Thus any error pattern will involve cycles in the graph. The notion being pursued in this paper is that by driving up the minimum length of a cycle in the graph (the *girth*) one will tend to improve the minimum distance properties of the code. Increasing the girth may also improve the convergence properties of the decoding as independence will be maintained for more iterations. Relatively dense graphs of high girth have been constructed by Ustimenko, Lazebnik and others [11, 9, 12, 13]. Their major result was the following: For each odd $m \geq 1$ and any prime power q a bipartite, q-regular, edge transitive graph $CD(m,q)$ of order at most $2q^{m-\lfloor \frac{m+2}{4} \rfloor+1}$ can be constructed whose girth is at least $m+5$. The graphs with even m also tend to have good girth properties and are used for some of the codes in this paper. These graphs arise as a connected component of a graph $D(m,q)$. The $D(m,q)$ graphs come from point-line incidence structures which are in some sense analogues of generalized m-gons. The incidence structure that the $D(m,q)$ come from have a structure which makes it quite simple to remove lines and points to reduce the degrees of the vertices on the two sides as desired. This process is described in [11]. One can also directly remove edges and vertices from the graph to modify the degrees. These operations cannot decrease the girth and in fact may well increase it.

3 Example Code

An example of these codes is a (2401,1372) (rate $=\sim .58$) code with a minimum distance of 24. By way of comparison a 4-D product code of similar size, (2401,1296) (rate $=\sim .53$) has a minimum distance of 16. It should also be noted that the decoding complexity of the presented code is also lower as each bit is involved in only three equations. Simulations indicate that these codes do indeed have good error floor performance. This code is obtained by taking a $D(4,7)$ graph, which is connected, and reducing the degrees of the variable nodes to 3.

4 Decoding Approaches

The baseline decoding strategy we employ to decode these codes is enhanced Max-Log APP[14]. This is Max-Log APP processing with a scale-factor applied to the extrinsic.

Because the usual soft output decoding strategies for LDPC codes produce reliability estimates for all bits they lend themselves to a modified decoding strategy that may improve packet error rate performance and has some benefits for evaluating the weight spectra of the codes via

simulation. In the normal off line setup of the encoding for an (n, k) LDPC code a set of $n-k$ linearly independent parity check equations are determined and then reduction is done so that $n-k$ bits (the "parity" bits) are defined in terms of the remaining k bits ("the information bits"). In this modification a similar procedure to this encoding process is carried out in the decoder with respect to a set of 'reliable' bits. The first step is the normal iterative decoding. If the code has not converged to a legitimate codeword (i.e. some of the parity equations are not satisfied) the reliability estimates on the bits produced by the decoding are sorted and, starting at the most reliable bit, a set of k bits sufficient to span the code space is determined. Note that these may not be the k most reliable bits but the bits that will have to be omitted because of dependencies will tend to be amongst the less reliable bits. Hard decisions are made on these reliable bits and a recoding is performed using these bits. In effect the block is coded as if the most reliable bits were the information bits. The positions corresponding to the original information bits are then read off. This procedure may degrade the bit error rate performance but it always has a beneficial effect on the packet error rate performance. Because the result of this process is always a codeword this process is of some value in evaluating the weight spectra of a code. For various of the LDPC constructions designed to produce high minimum distance codes, including the codes described here and some skewcodes[5, 6], it is exceedingly rare for the standard decoding to converge to an erroneous codeword. Thus it is difficult to find low weight codewords by observing error patterns coming out of the decoding. Since this recoding procedure leads to legitimate codewords, error patterns can be determined by comparing the transmitted bits and the recoded bits The resulting error patterns tend to be of low weight because of the initial decoding and hence they tend to be indicative of the low weight spectra of the code. Obviously this procedure is computationally intensive as it involves a significant matrix inversion and multiplication. The matrix inversion being equivalent to that normally required off line for the preparation of LDPC encoders and the multiplication being the encoding process. In a practical situation one might well be better off to apply the computational effort at hand to doing more iterations of the initial decoding. However, there do appear to be packets for which the iterative decoding process will not converge even after an extremely large number of iterations but will be corrected by recoding. This procedure is probably most suitable for structured as opposed to random LDPC codes because structured LDPC codes tend to have substructures whose reliabilities after decoding are significantly larger than the average for bits in the code. This is a characteristic that can be exploited [4] by stopping processing of certain equations.

The other modification in the decoding strategy that is applicable to the codes in this paper and more generally to LDPC codes is the use of a scale factor which is iteration dependent. Frequently the best tradeoff between decoding complexity and performance for LDPC codes is obtained by using a Max-Log APP decoder where the extrinsic has been scaled [14]. This is sometimes referred to as enhanced Max-Log APP. Asymptotically (high SNR) one knows that a scale factor of 1 is appropriate. But at SNRs of interest experience has shown that a smaller scale factor works better. For example *ad hoc* testing showed that for many hypercodes and product codes a scale factor of .625 worked well. Given that the iterative decoding process can be regarded as improving the SNR each iteration a variable scale factor may give an improvement in performance. Two approaches to determining appropriate variable scale factors have been used. The first approach is to simply go the route of running simulations to evaluate performance. To restrict the class of scale factor vectors (one entry per iteration) attention was restricted to scale factor ramps specifying an initial scale factor and an associated per iteration increment. Once the scale factor reaches 1 it is saturated at 1. The other approach is to use a form of density evolution [3]. To do the density evolution a Gaussian assumption [15] on the message distributions was used. Because the enhanced Max-Log APP algorithms do not preserve consistency (true APP does) it is necessary to consider the evolution of both the mean and the variance of the message distribution. Doing this for a specific distribution of vertex degrees in the graph associated to the code and a specific set of scale factors allows the determination of threshold on the noise variance below which arbitrarily long codes with this vertex distribution will correctly decode under this algorithm. Standard optimization procedures were then applied to maximize this threshold with respect to the scale factors used. Note that the code is not being optimized here. Rather, that the decoding algorithm is being optimized within a class of algorithms for a specific ensemble of codes. These optimizations are typically only applied to the first ten iterations so as to keep the dimensionality of the space being optimized over tractable. The normal procedure is to then saturate the scale factor at 1. The typical observed result of the optimization is close to the type of scale factor ramp described above.

The variable scale factor has been used on a variety of LDPC codes and can for some codes substantially close the gap between single scale factor decoding and true APP decoding. The benefit seems to be most pronounced for irregular LDPC codes.

5 Simulation Results

All points on the curves represent at least 100 packet errors except for the lowest points on the curves which may represent as few as 20 packet errors. For all simulations the maximum number of iterations was set to 92. Figure 1 shows the performance of a number of these codes under true APP decoding. The (16807,9750) code is showing distinct flaring but still a relatively steep BER curve.

There may be some trace of flaring in the (2401,1372) code; flaring usually manifests itself earlier in the PER curve than in the BER curve. Figure 2 shows the performance of the various decoding algorithms applied to the (2401,1372) code described above. The loss in performance of Max-Log APP relative to true APP is approximately .4 dB which is in keeping with previously observed results. Scale factor Max-Log APP reduces this to approximately .075 dB and scale vector Max-Log APP reduces it to approximately .02 dB.

6 Conclusions

A class of LDPC codes based on high girth bipartite graphs has been presented. By varying the parameters of the graphs to increase their girth one can improve the distance properties of the code without increasing the number of parity checks in which each bit is involved. However the size of the graph, and hence size of the code, required grows rapidly. Some modifications to standard decoding algorithms for LDPC codes have also been presented. One of which achieves near true APP performance with approximately Max-Log APP complexity.

References

[1] Robert G. Gallagher. *Low-density parity-check codes.* M.I.T. Press, 1963. Available from `http://justice.mit.edu/people/gallager.html/`.

[2] D. J. C. MacKay and R. M. Neal. Near Shannon limit performance of low density parity check codes. *Electronics Letters*, 32(18):1645–1646, August 1996. Reprinted *Electronics Letters*, vol 33, no 6, 13th March 1997, p.457–458.

[3] T. Richardson, Amin Shokrollhi, and Rudiger Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Trans. on Info. Thy.*, 47(2):619–637, 2000.

[4] P-P. Sauve, A. Hunt, S. Crozier, and P. Guinand. Hyper-codes: High-performance, low-complexity codes. In *Intnl. Symp on Turbo Codes*, pages 121–124, Brest, 2000.

[5] J. Lodge, A. Hunt, and P. Guinand. High code rate iteratively decodable FEC codes with low complexity and high minimum distance. In *Proc. Of Queen's University 20th Biennial Symp. On Communications*, pages 8–12, Kingston, 2000.

[6] J. Lodge, A. Hunt, and P. Guinand. High code rate iteratively decodable FEC codes for applications requiring low packet error rates. In *Intnl. Symp on Turbo Codes*, pages 117–120, Brest, 2000.

[7] P. Guinand and J. Lodge. Design of generalized product codes suitable for iterative decoding. In *Proc. of the 1996 Symp. on Info. Thy. and its Applications, Victoria BC*, 1996.

[8] P. Guinand and J. Lodge. Graph theoretic construction of generalized product codes. In *Proc. of the Intnl. Symp. on Info. Thy.*, Ulm, Germany, 1997.

[9] F. Lazebnik, V.A. Ustimenko, and A.J. Woldar. A new series of dense graphs of high girth. *Bull. Amer. Math. Soc. (N.S.)*, 32(1):73–79, 1995.

[10] R. Michael Tanner. A recursive approach to low complexity codes. *IEEE Trans on Info. Thy.*, IT-27(5):533–547, Sept. 1981.

[11] Z. Furedi, F. Lazebnik, A. Seress, V.A. Ustimenko, and A.J. Woldar. Graphs of prescribed girth and bi-degree. *J. of Combinatorial Theory (B)*, 64:228–239, 1995.

[12] F. Lazebnik and V.A. Lazebnik. New examples of graphs without small cycles and of large size. *Europ. J. Combinatorics*, 14:445–460, 1993.

[13] F. Lazebnik and V.A. Ustimenko. Explicit construction of graphs with an arbitrary girth and of large size. *Discrete Appl. Math.*, 60:275–284, 1995.

[14] K. Gracie, S. Crozier, and A. Hunt. Performance of a low-complexity turbo decoder with a simple early stopping criterion implemented on a sharc processor. In *Sixth International Mobile Satellite Conference (IMSC '99)*, pages 281–286, Ottawa, Canada, June 1999. Available at www.crc.fec in .pdf format.

[15] S.-Y. Chung, T.J. Richardson, and R.L. Urbanke. Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation. *IEEE Trans. on Info. Thy.*, 47:657–671, Feb. 2001.

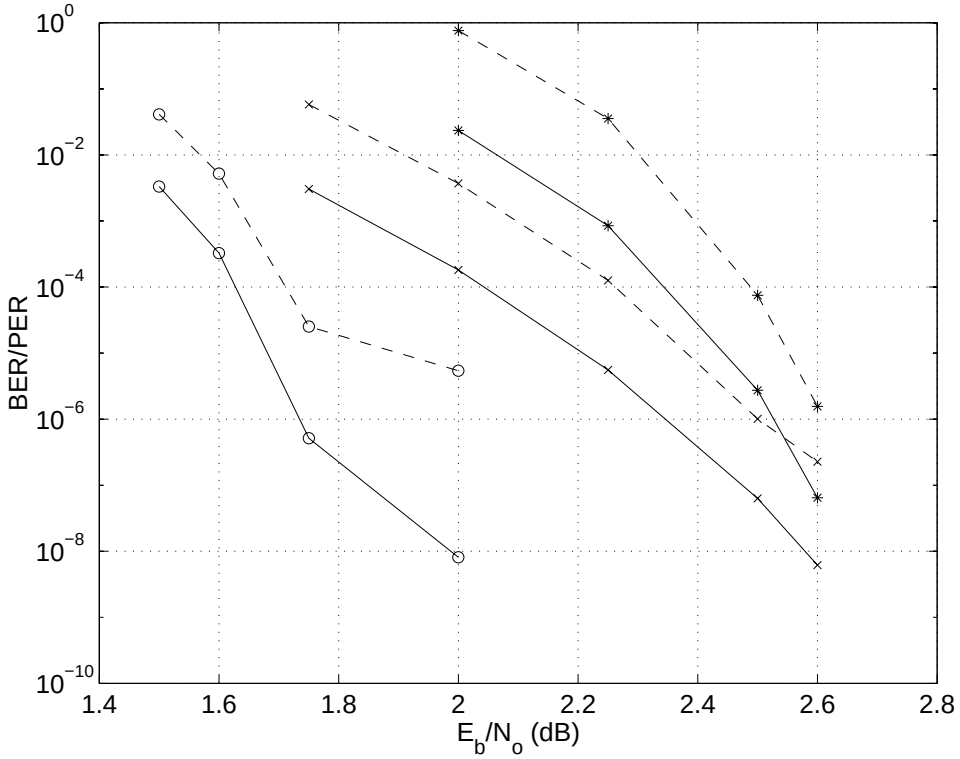

Figure 1: Performance of a various codes under True APP: (2401,1372) code (x) rate $=\sim .5801$, (14641,10680) code (*) rate$=\sim .7295$, (16807,9750) code (o) rate$=\sim .5801$. Solid line indicates BER, dashed line indicates PER.

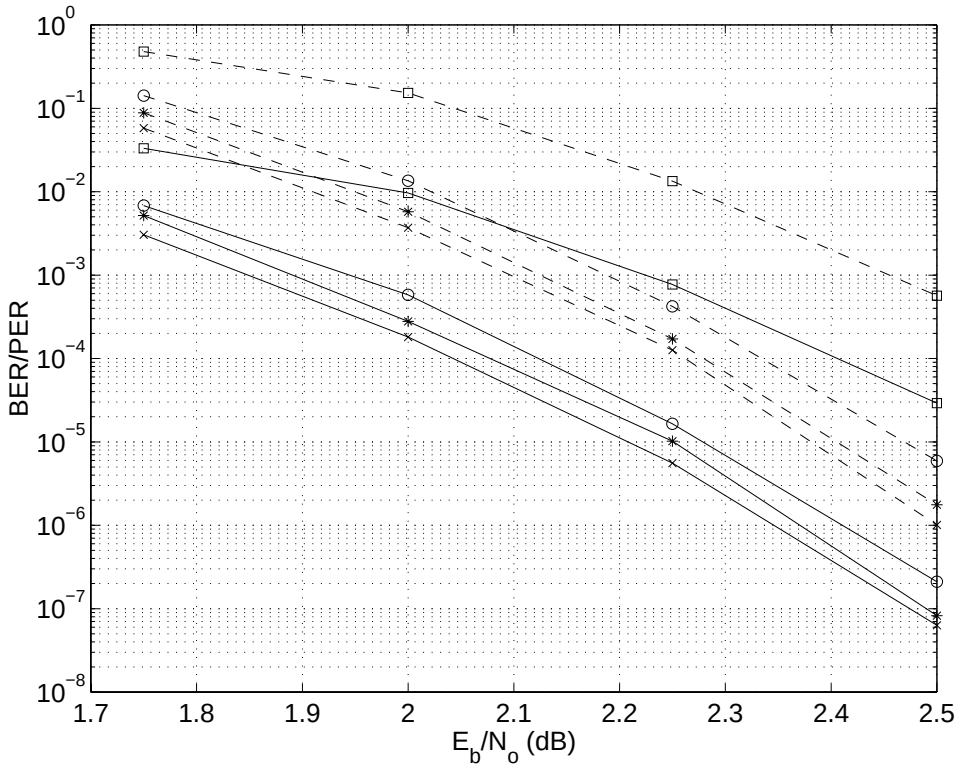

Figure 2: Performance of a (2401,1372) code under True APP (x), optimized scale factor vector(*), optimized fixed scale factor (o) and standard Max-Log APP (□). Solid line indicates BER, dashed line indicates PER.