Gradient Projection Decoding of LDPC Codes

Christos Kasparis and Barry G. Evans

***Abstract*— A new practical method for decoding Low-Density Parity Check (LDPC) codes is presented. The followed approach involves reformulating the parity check equations using non-linear functions of a specific form, defined over $\Re^{\rho}$, where ρ denotes the check node degree. By constraining the inputs to these functions in the closed convex subset $[0,1]^{\rho}$ ("box" set) of $\Re^{\rho}$, and also by exploiting their form, a multimodal objective function that entails the code constraints is formulated. The gradient projection algorithm is then used for searching for a valid codeword that lies in the vicinity of the channel observation. The computational complexity of the new decoding technique is practically sub-linearly dependent on the code's length, while processing on each variable node can be performed in parallel allowing very low decoding latencies. Simulation results show that convergence is achieved within 10 iterations, although some performance degradations relative to the Belief Propagation (BP) algorithm are observed.**

***Index Terms*— LDPC, decoding, non-linear, gradient, projection.**

I. Introduction

THE rediscovery of Low Density Parity Check (LDPC) codes[1] by MacKay and Neal [2,3] (as a class of error correcting codes that achieve near Shannon-bound performances and allow manageable decoding complexities), has sparked research interest towards new powerful and efficiently encodable LDPC codes (e.g. [8]), and also towards new decoding algorithms with attractive performance-complexity features. With increasing numbers of practical communication systems employing LDPC codes, the search for efficient codes and decoding algorithms still attracts significant research interest. The current paper addresses the latter topic.

Focusing on existing types of decoding algorithms, the Belief Propagation (BP) algorithm [1,3,4] is an iterative technique for computing (approximate) marginal posterior probabilities, in which messages in the form of conditional probabilities are exchanged between variable and check nodes on the code's Tanner graph. The BP algorithm achieves best error performances amongst other practical decoding algorithms, although it also involves higher computational complexity. Reduced complexity variations of the BP [5-7] offer trade-offs between performance and computational requirements. A low complexity Bit Flipping (BF) decoding algorithm was also proposed by Gallager [1], in which the most unreliable bits, classified in terms of the number of associated parity check failures, are flipped. Various improved performance variations of the BF algorithm have been reported in [8,9]. A third decoding approach based on linear programming has been recently proposed in [10] and explored further in [11].

In this letter a new approach is followed in order to develop a decoding algorithm that is fundamentally different to any of the existing types. The central element in the proposed algorithm is the non-linear function $f(x_1, x_2) = x_1 + x_2 - 2x_1x_2$ which is equivalent to addition over GF(2), when $x_1, x_2 \epsilon \{0,1\}$. This basic function is nested: $f(..f(f(x_1,x_2),x_3)..,x_\rho)$ in order to redefine individual parity check equations over the closed convex set $[0,1]^{\rho}$ (instead of the binary finite field). This type of convex relaxation, combined with the particular form of the proposed non-linear function, allows formulation of a multimodal objective function that entails the code's constraints. Iterative algorithms, such as the Gradient Projection (GP) method [12,13], can then be used to search for a local minimum of the objective function, that lies in the vicinity of the noisy channel observation.

The number of multiplications in the new algorithm (although it is random - depending on the number of bits that have converged to 0 and 1 values) in practice is sub-linearly dependent on the code's length, whilst convergence is typically achieved within 10 iterations. Furthermore processing on individual variable nodes can be performed in parallel, allowing very small processing latencies.

Section II provides a detailed description of the proposed algorithm, comments on its computational requirements and also describes a 'denser' representation of the code, which provides improved performance and faster convergence. Section III includes simulation results that give the performance and convergence speed of the algorithm on short (96,48) and medium-length (504,252) regular systematic codes. Finally Section IV gives a summary and conclusions.

II. Gradient Projection Decoding Algorithm

A. Basics and Notations

An (n,k) binary LDPC code is a linear block code that is characterized by a parity check matrix $\mathbf{H}$ (of dimensions $(n-k) \times n$) that has a low density of 1's. Depending on whether the rows and the columns of $\mathbf{H}$ are populated by equal numbers of 1's or not, the LDPC code is categorized as regular or irregular, respectively. An LDPC code can alternatively be represented by a bipartite graph with two independent node sets: $I = \{I_1, .., I_n\}, J = \{J_1, .., J_{n-k}\}$, ('variable' and 'check' nodes respectively) that correspond to the columns and rows of $\mathbf{H}$. For convenience some I_i and J_j will hereon simply denoted as i and j, respectively. Connections between the two sets of nodes are determined by the corresponding positions of 1's on $\mathbf{H}$. The 'neighbourhood' $N_c(j)$ of check node $j \epsilon J$ is the set of variable nodes connected to j, and

Manuscript received October 31, 2006. The associate editor coordinating the review of this letter and approving it for publication was Dr. Vladimir Stankovic.

The authors are with the Centre for Communication Systems Research, University of Surrey, Guildford, GU2 7XH, Surrey, UK (e-mail: c.kasparis@surrey.ac.uk).

Digital Object Identifier 10.1109/LCOMM.2007.061780.

[1]Originally discovered by Gallager in 1962 ([1])

similarly the 'neighbourhood' $N_v(i)$ of a variable node $i\epsilon I$ is the set of check nodes connected to i. In regular LDPC codes the cardinalities (ρ_j) of the check node neighbourhoods (commonly referred to as degrees) are equal for all $j\epsilon J$, and the same is true for all variable node degrees (γ_i). On the other hand, in irregular codes the variable and check node degrees are functions of i and j, respectively.

B. Decoding Algorithm

As briefly discussed in Section I, the proposed approach involves defining and minimizing an objective function that describes directly the $m = n - k$ parity check equations that a valid codeword needs to satisfy. Starting point in the development of such objective function is the non-linear function

$$f(x_1, x_2) = x_1 + x_2 - 2x_1x_2 \tag{1}$$

which is equivalent to addition over GF(2), when $x_1, x_2\epsilon\{0, 1\}$. It is not difficult to think of alternative functions that provide this equivalency, but (1) has a number of features that make it a good choice. The most significant of these features is its shape over the relaxed ranges $x_1, x_2\epsilon[0, 1]$, which can be better appreciated from Fig. 1; assuming an AWGN channel, if a valid (2-bits long) codeword $\mathbf{c}$ had to satisfy the parity check $c_1 \oplus c_2 = 0$, and the corresponding noisy observations $\mathbf{y}= [y_1, y_2]$ were available, then a constraint gradient minimization of (1) (over $[0, 1]^2$, and by setting (y_1, y_2) as the starting solution) would converge to the valid solution closest to the channel observation, in the Euclidian distance sense. Function (1) can be generalized for more than two variables, in order to describe some j^{th} parity check, by putting it into a nested form

$$f_j(x_1, .., x_{\rho_j}) = f(..f(f(x_1, x_2), x_3).., x_{\rho_j}) \tag{2}$$

Using the fact that (2) is symmetric, in the sense that it remains unchanged by any permutation of its variables, an overall objective function is formulated as

$$F(x_1, .., x_n) = f_1(N_c(1)) + .. + f_m(N_c(m)) \tag{3}$$

A constraint local minimum on (3) that lies in the vicinity of the channel observation can be searched using the iterative GP method [12,13], which consists of the iteration:

$$\hat{\mathbf{c}}_l = P_Q[\hat{\mathbf{c}}_{l-1} - \alpha\nabla F] \tag{4}$$

where $\hat{\mathbf{c}}_l$ is the estimated codeword at iteration l, $P_Q[.]$ signifies the orthogonal projection operator on the convex set $[0, 1]^n$ and α is a positive step parameter that influences the convergence speed and the accuracy of the algorithm. Under the particular set constraints each signal component can be updated independently and in parallel [12]:

$$\hat{c}_l^i = \begin{cases} 0, & \text{if } \hat{c}_{l-1}^i - \alpha\frac{\partial F}{\partial c^i} < 0 \\ \hat{c}_{l-1}^i - \alpha\frac{\partial F}{\partial c^i}, & \text{if } 0 \le \hat{c}_{l-1}^i - \alpha\frac{\partial F}{\partial c^i} \le 1 \\ 1, & \text{if } \hat{c}_{l-1}^i - \alpha\frac{\partial F}{\partial c^i} > 1 \end{cases} \tag{5}$$

Furthermore it can easily be shown, using the symmetric property of (2), that

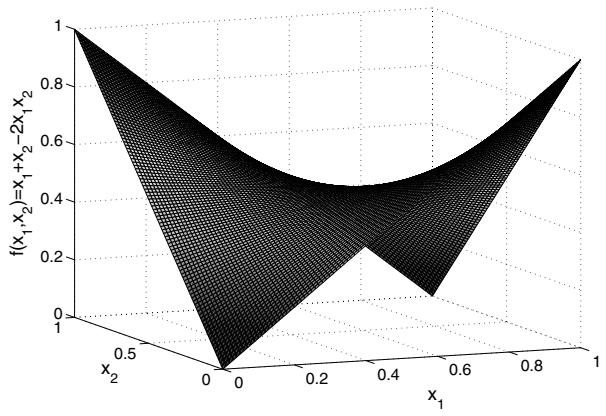

Fig. 1. Plot of the $f = x_1 + x_2 - 2x_1x_2$ function over the box set $[0, 1]^2$.

$$\frac{\partial f_j(x_1, ., x_i, ., x_{\rho_j})}{\partial x_i} = 1 - 2f_j(x_1, ., x_{i-1}, x_{i+1}, ., x_{\rho_j}) \tag{6}$$

Assuming a regular LDPC code, it can be deduced from (6) that the proposed decoding algorithm requires approximately $n(\rho-2)\gamma$ real multiplications, $4n(\rho-2)\gamma$ real additions, and n 'clipping' operations per iteration. In practice the number of computations will be random, but significantly less than the above figures; since from (1) it can be seen that no operations are needed if any of $x_1, x_2 = 0$ and only one real addition is needed if any of $x_1, x_2 = 1$.

Based on the above, the proposed decoding algorithm involves two main steps: a) Set $\hat{\mathbf{c}}_0 = P_Q[\mathbf{y}]$ and b) update individual signal estimates using (5) for a maximum number of iterations or until all parity check equations are satisfied.

C. Performance Improvements through Denser Parity Check Representation of the Code

The use of additional (redundant) higher-order parity checks, although can result in short cycles on the code's Tanner graph (and thus degrade the performance of the BP algorithm), has been shown in [10] to improve the performance of the linear programming based decoder. Simulation results, which are discussed in Section III, show that the performance and convergence speed of the proposed decoding algorithm improve just through a higher-order (denser) parity check representation of the code, i.e. not by using higher-order parity checks additionally to the first-order ones.

In order to limit the impact on the decoding complexity, only a 'second-order' representation of the code has been considered. In particular, second-order parity checks have been constructed by taking, for each i, the mod-2 summation of all possible pairs of first-order parity checks that belong in $N_v(i)$. Assuming a regular (ρ, γ) LDPC code, the second-order representation yields a $(2(\rho - 1), (\rho - 1)\gamma(\gamma - 1))$ equivalent code.

III. Simulation Results

The performance of the proposed algorithm has been evaluated on short (96,48) and medium-length (504,252) regular

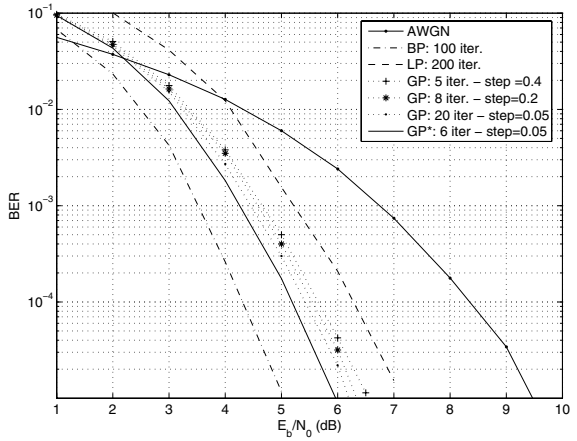

Fig. 2. Performance of proposed decoding algorithm on the (96,48) code.

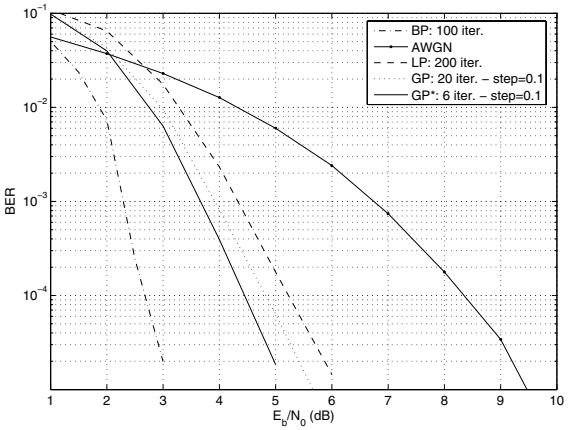

Fig. 3. Performance of proposed decoding algorithm on the (504,252) code.

systematic LDPC codes; both with $\rho = 6$ and $\gamma = 3$. The results for the (96,48) code are given in Fig. 2. The GP decoder was tested for three choices of the step parameter: $\alpha = 0.05$, 0.2 and 0.4, and also with the second-order parity check representation of the code. It is observed that larger values of the step parameter lead to faster convergence, with a small penalty on the performance (~ 0.1dB), while the denser representation of the code (denoted as GP* on the figure) gives a performance benefit of about 0.2-0.3dB at BER=10^{-5}. For comparison, the BF algorithm proposed by Liu and Pados (LP) ([9]) was also simulated. It is observed that the GP decoder gives a performance benefit of about 0.8-1dB, although it lags the performance of the BP by about the same margin at BER=10^{-5}. It is also worth noting that in the GP decoder with $\alpha = 0.2$ and $E_b/N_0 = 6dB$, on average (over the 8 iterations) 44 variables are converged to 0 and another 44 to 1. Similarly, in the denser code representation example, on average 42 variables are converged to 0 and another 42 to 1. These experimental results give a good indication that the computational requirements of the GP decoder are in practice very small.

Figure 3 gives the results for the (504,252) code. It is observed that on this code the performance gap between the BP and the GP decoder is wider: about 2.5dB at BER=10^{-5}, although the GP decoder still has a performance advantage (about 0.5dB at BER=10^{-5}) relative to the LP decoder. The second-order representation of the code yields a 0.5dB benefit (at BER=10^{-5}), and also faster convergence. In this code example the average number of variables converged to 0 and 1 in the GP decoder, is 235.15 for both signal values, and 215.9 with the denser code representation.

IV. Conclusion

A new type of decoding algorithm for LDPC codes has been proposed, which is based on the local minimization of a multi-modal non-linear objective function that entails the code constraints. A valid codeword in the vicinity of the noisy channel observation is searched by the GP method, which converges to a constraint minimum over the 'box' (convex) set. The new decoding approach has computational and implementation advantages (e.g. no noise statistics need to be estimated) relative to the BP algorithm, although the latter provides better performance. Finally it is commented that the proposed decoder, being based on different principles than with existing types of decoders, opens a new research path on the decoding of LDPC codes.

Acknowledgment

The authors would like to acknowledge the FP6 European projects MOWGLY and SatNEx2 for funding this work. The authors would also like to thank Dr. Elias Gyftodimos from the Computing Science Department, University of Aberdeen, for his valuable comments and suggestions.

References

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, pp. 21-28, Jan. 1962.

[2] D. J. C. MacKay and R. M. Neal, "Near-Shannon-limit performance of low-density parity-check codes," *Electron. Lett.*, vol. 32, pp. 1645-1646, Aug. 1996.

[3] D. J. C. MacKay, "Good error correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, pp. 399-432, Mar. 1999.

[4] T. Richardson *et al.*, "Design of capacity-approaching irregular codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 619-637, Feb. 2001.

[5] M. P. C. Fossorier *et al.*, "Reduced-complexity iterative decoding of low-density parity-check codes based on belief propagation," *IEEE Trans. Commun.*, vol. 47, pp. 673-680, May 1999.

[6] J. Chen and M. P. C. Fossorier, "Near-optimum universal belief propagation-based decoding of low-density parity-check codes," *IEEE Trans. Commun.*, vol. 50, pp. 406-414, Mar. 2002.

[7] H-Y Hu et. al., "Efficient implementation of the sum-product algorithm for decoding LDPC codes," *in Proc. IEEE GLOBCOM*, vol. 2, pp. 1036-1036E, San Antonio, 25-29 Nov. 2001.

[8] Y. Kou, *et al.*, "Low density parity check codes based on finite geometries: a rediscovery and new results," *IEEE Trans. Inf. Theory*, vol. 47, no. 7, Nov. 2001.

[9] Z. Liu and D. A. Pados, "A decoding algorithm for finite-geometry LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 3, Mar. 2005.

[10] J. Feldman *et al.*, "Using linear programming to decode binary linear codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 3, March 2005.

[11] K. Yang *et al.*, "Nonlinear programming approaches to decoding low-density parity-check codes," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, Aug. 2006.

[12] E. S. Levitin and B. T. Polyak, "Constrained minimization methods," *USSR Computational Mathematics and Math. Physics*, vol. 6, pp. 1-50, 1966.

[13] A. A. Goldstein, "Convex programming in Hilbert space," *Bulletin of the American Mathematical Society*, vol. 70, pp. 709-710, 1964.