MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Iterative Decoding of Multiple-Step Majority Logic Decodable Codes

Ravi palanki, Marc Fossorier, Jonathan Yedidia

TR2007-038 June 2007

Abstract

We investigate the performance of iterative decoding algorithms for multistep majority logic decodable (MSMLD) codes of intermediate length. We introduce a new bit-flipping algorithm that is able to decode these codes nearly as well as a maximum-likelihood decoder on the binary-symmetric channel. We show that MSMLD codes decoded using bit-flipping algorithms can outperform comparable Bose-Chaudhuri-Hocquenghem (BCH) codes decoded using standard algebraic decoding algorithms, at least for high bit-flip rats (or low and moderate signal-to-noise ratios).

IEEE Transactions on Communications

This is a revised version of MERL TR2003-107. Published in IEEE Transactions on Communications, vol. 55, number 6, June 2007, pp. 1099-1102.

Iterative Decoding of Multi-Step Majority Logic Decodable Codes

Ravi Palanki

Qualcomm Inc.

San Diego, CA 92121, USA

Email: `ravi@systems.caltech.edu`

Marc Fossorier

Dept. of Electrical Engineering,

University of Hawaii,

Honolulu HI96822, USA

Email: `marc@spectra.eng.hawaii.edu`

Jonathan S. Yedidia

Mitsubishi Electric Research Laboratories

Cambridge, MA 02139, USA

Email: `yedidia@merl.com`

Abstract

We investigate the performance of iterative decoding algorithms for multi-step majority logic decodable (MSMLD) codes of intermediate length. We introduce a new bit-flipping algorithm that is able to decode these codes nearly as well as a maximum likelihood decoder on the binary symmetric channel. We show that MSMLD codes decoded using bit-flipping algorithms can out-perform comparable BCH codes

decoded using standard algebraic decoding algorithms, at least for high bit flip rates (or low and moderate signal to noise ratios).

1 Introduction

In [1], it was shown that iterative decoding of one-step majority logic decodable (OSMLD) codes performed very well; the performance was often better than that of ordinary low density parity check (LDPC) codes [2, 3] of similar blocklength N and rate R for values of N up to a few thousand bits, despite the fact that the parity check matrix of OSMLD codes has a higher density than that of ordinary LDPC codes. The reason for the improved performance was mainly because the $M \times N$ matrix H used for decoding was highly redundant, i.e., $M \gg N(1-R)$.

In this paper, we investigate iterative decoding of multi-step majority logic decodable (MSMLD) codes for transmission over a binary symmetric channel (BSC). With the use of redundant H matrices, these codes have already been shown to perform relatively well on the additive white Gaussian noise (AWGN) channel [4, 5]. However, unlike on the AWGN channel where the performance of iterative decoding does not approach that of maximum likelihood decoding (MLD), we find that fast and low complexity bit flipping (BF) algorithms can achieve near MLD performance on the BSC.

2 Three-state Decoding Algorithm

Two different BF algorithms designed for LDPC codes with a few low-weight checksums per bit were proposed by Gallager in [2]. In these algorithms, the "message" from a bit to its neighboring check does not directly depend on the message sent by that check back to the bit and vice versa. This is done in order to prevent the introduction of correlations in the iterative process. In our case, because of the very large number of checksums corresponding to each bit, we can neglect that refinement with negligible performance degradation and we can obtain the following algorithm, which simplifies Gallager's algorithm-B:

- For each checksum m and for each bit n in checksum m, compute the modulo-2 sum σ_{mn} of the initial value of bit n and of the other bit values computed at iteration-$(i-1)$.

- For each bit n, determine the number N_u of unsatisfied checksums σ_{mn} intersecting on it. If N_u is larger than some predetermined threshold b_1, invert the original received bit n, otherwise keep this value.

The use of a single threshold b_1 implies that bits with very different values N_u are viewed with the same reliability at the next iteration. For the codes considered in [2], N_u can take only a few different values. This is not the case for the codes considered in this paper. It seems reasonable to try to reflect the differing reliablities of the bits in our algorithm. Consequently, we modify the algorithm described above into the following "three-state" algorithm, which also allows bits to be erased and checksums to be de-activated.

- For each checksum m and for each bit n in checksum m, compute the modulo-2 sum σ_{mn} of the initial value of bit n and of the other bit values computed at iteration-$(i-1)$. If any of these bits is erased, the checksum is de-activated.

- For each bit n, determine the number N_{ua} of unsatisfied activated checksums σ_{mn} intersecting on it.
  If $N_{ua} \geq b_1$, invert the original received bit n.
  If $b_1 > N_{ua} \geq b_2$, erase bit n.
  Otherwise keep the original received bit n.

Empirically, we find that the three-state algorithm performs best when the thresholds b_1 and b_2 are functions of the iteration number. Unfortunately, we could only do a rough optimization; however, this appears to be sufficient since the performance is a rather insensitive function of the thresholds. We typically chose to begin at the first iteration with b_1 equal to the maximum possible number of unsatisfied checks J, and with $b_2 \approx b_1 - J/15$, and then to decrease b_1 and b_2 by the same small fixed integer (say one to five) at each iteration, continuing to decrease their values until they reach zero.

The proposed three-state approach can also be applied in a straightforward way to Gallager's original algorithm-B. In fact, for a theoretical analysis, only this version is meaningful since the simplified algorithm introduces correlation and it is not known how to handle correlated values in the analysis of an iterative decoding algorithm in general. In that case, the three-state algorithm becomes a generalized version of the algorithm described in [7, Example 5], where $b_2 = b_1 - 1$. Consequently, if we assume the graph representation of the code is a tree, the same approach as in [7] can be used to analyze the three-state algorithm.

3 Decoding Approaches

A $(\mu+1)$-step majority logic decodable Euclidean geometry (EG) code over EG$(m, 2^s)$ can be represented by an M x N incidence matrix H [8, p.309-319]. The matrix H is also a parity-check matrix of the EG code. Its M rows represent the $(\mu+1)$-flats of the Euclidean geometry EG$(m, 2^s)$ not going through the origin and its $2^{ms} - 1$ columns represent the points other than the origin, with $h_{ij} = 1$ if the j-th point belongs to the i-th $(\mu+1)$-flat. Note that for $s = 1$, we obtain the subclass of Reed-Muller (RM) codes.

A straightforward approach is to run the BF algorithm based on H. This matrix contains many four-cycles, but it is redundant with $M >> N$. Futhermore, it is possible to use an $M_a \times N$ sub-matrix H_a of H for decoding. The M_a rows of H_a are chosen in a manner that exploits the cyclic nature of the code. That is, if H_a contains a row X, it also contains all cyclic shifts of

X. No noticeable difference in performance has been observed for different choices of these M_a rows.

If a sufficient number of checksums M_a is used, then the BF algorithm converges rapidly to its final solution while if not enough checksums are used, the BF algorithm generally never converges to a codeword. In this latter case, a decoding failure is detected. This observation suggests a "call by the need" algorithm in which, for $M_a < M_b < \cdots < M$, M_a checksums are initially used for N_a iterations. If the algorithm converges to a codeword, correct decoding is assumed; otherwise, the algorithm is reinitialized (not continued) and performed based on M_b checksums during N_b iterations. This process is repeated until either a codeword is found, or all M checksums have been used without success, in which case the decoding fails.

4 Simulation Results

We assume a BSC obtained from BPSK signaling, so that for a code of rate R, we have $p_0 = \mathrm{Q}\left(\sqrt{RE_b/N_0}\right)$, where E_b/N_0 is the signal to noise ratio (SNR) per information bit.

4.1 (255,127,21) EG Code

In Figure 1, the simulated error performance of three-state BF decoding of the (255,127,21) EG code with the direct approach of Section 3 is compared to t-bounded distance decoding (BDD) with the Berlekamp-Massey algorithm of its (255,123,39) BCH code counterpart as well as bit flipping with Gallager algorithm-B of its $(3,6)$ Gallager LDPC code counterpart. This EG code corresponds to $\mu + 1 = 2$ and the Euclidean geometry $EG(4,4)$ with 255 points other than the origin and 5355 planes not going through the origin. Hence we can construct a parity check matrix H with 5355 rows and 255 columns. A maximum of 200 iterations was selected, while on average much less are needed, especially at high SNR values. We observe that three-state

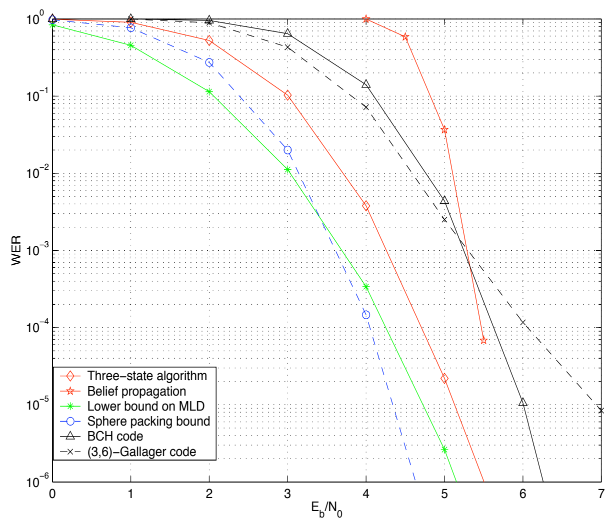

Figure 1: BF decoding of the (255,127,21) EG code; low SNR regime.

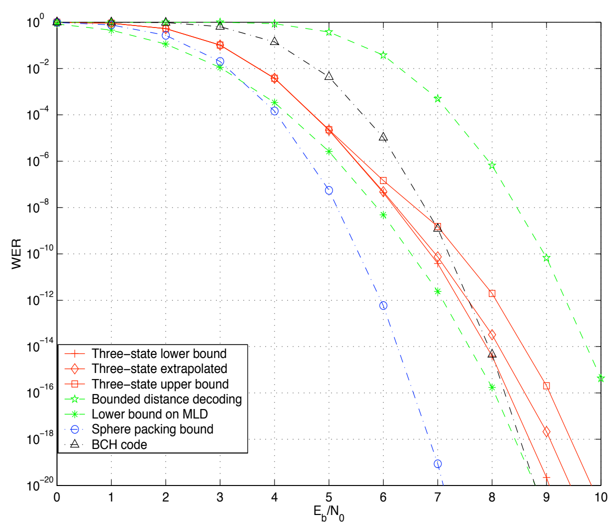

Figure 2: BF decoding of the (255,127,21) EG code; high SNR regime.

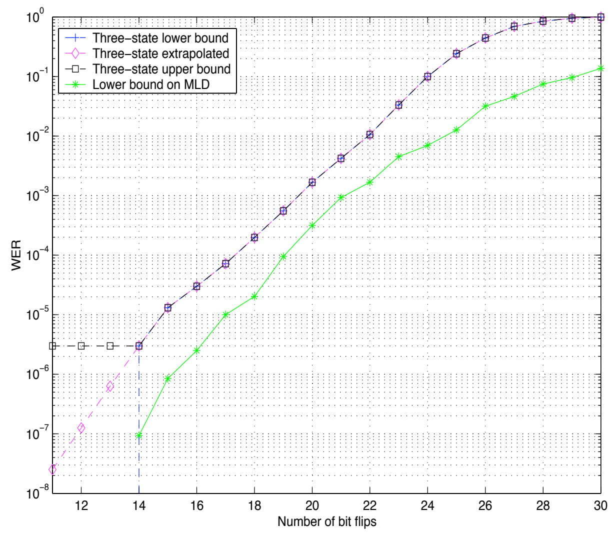

Figure 3: BF decoding of the (255,127,21) EG code for fixed number of errors.

BF decoding of the EG code not only outperforms its counterparts at the SNR values represented, but also remains quite close to the sphere packing bound (SPB), also represented in Figure 1. In fact, a lower bound on the MLD failure rate for this code was computed by checking whether the decoding errors were also MLD errors (with unbiased recording of the ties). This bound is represented in Figure 1. One can see that the performance of the three-state BF algorithm must be very close (within a few tenths of a dB) of MLD performance. The error performance of the standard sum-product or "belief propagation" (BP) algorithm, initialized with the crossover probability p_0 of the BSC is also shown in Figure 1. The reasons for the degraded performance of BP at low SNR's are elaborated in Section 5.

We also mention that the advantage of the three-state BF algorithm over Gallager's algorithm B is a reduction factor that ranges between two and five in the number of errors. This gain is small, but remains non-negligible in approaching MLD performance, especially since the three-state algorithm is not much harder to implement than Gallager's algorithm B.

In Figure 2, we plot the performance of the three-state BF decoding algorithm for the (255,127,21) EG code into the very high SNR, or low decoding failure, regime. To obtain these performance curves, we randomly generated random errors of fixed weight w, $w > t$ and for each weight w, evaluated the corresponding error performance $P_s(w)$. The overall error performance P_s was then obtained by the average

$$P_s = \sum_{w=t+1}^{N} P_s(w) \binom{N}{w} p_0^w (1-p_0)^{N-w}. \qquad (1)$$

The results are reported in Figure 3. Since for WERs smaller than 10^{-6}, no reliable evaluation of $P_s(w)$ is possible, we simulated weights $w \geq w_{min}$, where w_{min} is the smallest weight for which $P_s(w) \geq 10^{-6}$. Based on these results, we computed: (a) an upper bound on (1) by assuming the same $P_s(w) = P_s(w_{min})$ for weights w, $t < w < w_{min}$; (b) a lower bound on (1) by assuming $P_s(w) = 0$ for weights w, $t < w < w_{min}$; and (c) an approximation by extrapolating $P_s(w)$ for weights w, $t < w < w_{min}$. A pessimistic lower bound on MLD was also obtained by recording only the MLD errors for weight $w \geq w_{min}$ and assuming $P_s(w) = 0$ for weights w, $t < w < w_{min}$. From Figure 2, we conclude that the three-state BF for the (255,127,21) EG code outperforms t-BDD of its BCH counterpart down to a WER of about 10^{-12}. We must also mention that this performance is nearly the same as that of a more complicated approach based on generalized parity check (GPC) matrices [9]. At all word error rates down to 10^{-20}, the difference between the straightforward method and the GPC matrix based method is less than 0.1 dB.

4.2 (511,256,31) EG (RM) Code

Figure 4 depicts the error performance of three-state BF decoding of the (511,256,31) EG (or RM) code with the variable cost approach of Section 3. For comparison, the SPB and t-BDD with the Berlekamp-Massey algorithm

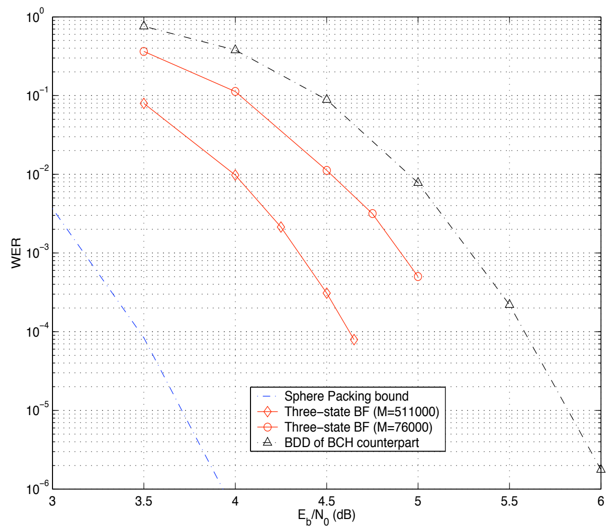

Figure 4: BF decoding of the (511,256,31) EG (or RM) code.

of the counterpart (511,250,63) BCH code have also been presented.

Two different numbers of total parity checks, $M = 76,650$ and $M = 511,000$ have been considered (corresponding to 150 and 1000 different cyclic shifts of weight-32 codewords of the dual code, respectively). In both cases, we chose five different sizes of the set of checksums used, namely, $M_a = 5110$; $M_b = 12,775$; $M_c = 22,550$; and $M_d = 51,000$. For each size, at most 10 iterations were performed. The value b_1 was set to the maximum number of unsatisfied checksums at each initial iteration and decreased by one (or a small number) at each subsequent iteration while we chose $b_2 = b_1 - 20$. Again these values were not thoroughly optimized so that additional secondary gains should be achievable.

The application of the variable cost method is validated by the fact that for $M = 76,650$, no undetected error was recorded at all simulated SNR values. For $M = 511,000$, at the SNR value of 4.5 dB, about 10% of the errors were undetected (all of them occurring when all checksums were considered) and at this SNR value, one out of the 100 errors recorded was recognized as an MLD error. At lower SNR values, no undetected errors and no MLD errors were recorded. While a reasonably good error performance is achieved, we are clearly not able to obtain a tight bound on MLD performance. Because the three-state BF algorithm has a very low word error rate even for error patterns with a number of bit flips far beyond the guaranteed error-correcting capability t of the code, we are also not able to meaningfully repeat the analysis of the very high SNR regime. We also observe that despite the fact that the minimum distance of this code is about half of that of its BCH counterpart, iterative BF decoding of this EG code can easily outperform t-BDD of its BCH counterpart and approaches relatively closely the SPB at the WERs presented in Figure 4. We must also mention that a more complicated approach based on the decomposable structure of RM codes yielded no improvement [9].

At a given code rate, as N increases, the weight of the rows of the parity check matrix H also increases for the class of MSMLD codes. This causes the number of redundant rows in H to grow to a very large number if near MLD peformance is required, as is already apparent for the results we present for the (511,256,31) code. Consequently, this approach does not seem to scale up very well with N despite the fact that iterative decoding is used. This is not totally surprising, as in general, the decoding complexity of MLD increases exponentially with N. On the other hand, near MLD of EG codes of length $N \leq 511$ based on their parity check matrix H given in Section 3 is possible with this approach and was verified by simulation for many shorter codes.

5 Extension to Iterative Decoding for the AWGN Channel

A very natural extension of these results is to replace the BSC by an AWGN channel. Although as already stated in the introduction, relatively good results for iterative decoding of MSMLD codes have been previously reported for the AWGN channel, all these results fall short of near MLD. The main reason we believe is the large dynamical range taken by the a-posteriori values evaluated after few iterations due to the large correlation propagated by feedback (note that in the BF algorithms, the values at the bit nodes are always the same at the beginning of each iteration). As a result, there is no longer much difference between soft information and hard information with erasure. Indeed, the same conclusions also hold for BP decoding over the BSC, although in that case, no significant degradation can be expected at high enough SNR, as observed in Figure 1.

Using a heuristic extension of the decomposition proposed in [10], the a-posteriori information L_{i+1} evaluated at iteration-$(i+1)$ can be represented as the sum of the a-priori information L_0 and a function of approximated extrinsic information values $\tilde{L}_i^e$ derived (and observable) at iteration-i. In graphs with cycles, $\tilde{L}_i^e$ can be viewed as the sum of the true extrinsic information L_i^e and additional correlated values L_i^c, so that

$$\begin{aligned} L_{i+1} &= L_0 + f(\tilde{L}^e) \\ \text{with} \quad \tilde{L}_i^e &= L_i^e + L_i^c, \end{aligned}$$

Consequently, the influence of correlation can be reduced by modifying the function $f()$ in several ways $g()$ such as scaling ($g \circ f = \alpha f$, $0 < \alpha \leq 1$), off-setting ($g \circ f = \text{sgn}(f)\max\{|f| - \beta, 0\}$), damping ($g \circ f = \alpha f_i + (1 - \alpha) f_{i-1}$, $0 < \alpha \leq 1$), or clipping ($g \circ f = \text{sgn}(f)\min\{|f|, C\}$). However, these modifications affect both L_i^e and L_i^c while hypothetically, it would be desirable to reduce L_i^c only. This is indeed a much difficult task as we have direct access to $\tilde{L}_i^e$ only. For example, all best approaches used to iteratively decode the (255,127,21) EG code over the AWGN channel fell short of MLD by about 0.8 dB [11].

6 Conclusion

In this paper, we have shown that iterative BF algorithms can achieve near MLD of intermediate length MSMLD codes despite the presence of four-cycles in their graph representation. This drawback is overcome by the very large number of redundant low weight checksums. The most straightforward parity check matrix representation of these codes in conjunction with a "call by the need" decoding seems to provide the best compromise between error performance and decoding complexity.

In principle, the three-state BF decoding approach could be applied to any other intermediate length linear code. One "merely" needs to find a sufficient number of redundant low weight codewords in the dual code to construct a useful parity check matrix H. Unfortunately, this does not appear to be an easy task for codes that are not as nicely structured as the families of codes considered in this paper.

References

[1] R. Lucas, M. Fossorier, Y. Kou and S. Lin, "Iterative Decoding of One-Step Majority Logic Decodable Codes Based on Belief Propagation," *IEEE Trans. Commun.*, vol. 48, pp. 931-937, June 2000.

[2] R.G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: M.I.T. Press, 1963.

[3] D.J.C. MacKay, "Good Error-Correcting Codes Based on Very Sparse Matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399-431, Mar. 1999.

[4] R. Lucas, M. Bossert, and M. Breitbach, "On Iterative soft-decision decoding of linear binary block codes and product codes," *IEEE Jour. Select. Areas Commun.*, vol. 16, pp. 276-298, Feb. 1998.

[5] H. Tang, J. Xu, Y. Kou, S. Lin, and K. Abdel-Ghaffar, "On Algebraic Construction of Gallager Low Density Parity Check Codes," *The Proc. IEEE Intern. Sympos. Inform. Theory*, Lausanne, Switzerland, June 2002.

[6] D.E. Muller, "Application of Boolean Algebra to Switching Circuit Design and to Error Detection, IRE Trans. on Electronic Computation," *IRE Trans. Electron. Comput.*, vol. 3, pp.6-12, Jan. 1954.

[7] 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, Feb. 2001.

[8] S. Lin and D.J. Costello, Jr., *Error Control Coding, Second Edition*, Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 2004.

[9] M. Fossorier, R. Palanki, and J. S. Yedidia, "Iterative Decoding of Multi-Step Majority Logic Decodable Codes," *Proc. 3rd Intl. Symp. on Turbo Codes and Related Topics*, Brest, France, Sep. 2003.

[10] J. Hagenauer, E. Offer, and L. Papke, "Iterative Decoding of Block and Convolutional Codes," *IEEE Trans. Inform. Theory*, vol. 42, March 1997, pp. 429-445.

[11] J. Yedidia, J. Chen and M. Fossorier, "Generating Code Representations Suitable for Belief Propagation Decoding," *The Proc. 40-th Annual Allerton Conf.*, Monticello, USA, Oct. 2002.