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

Iterative Decoding of Multi-step Majority Logic Decodable Codes

Marc Fossorier, Ravi Palanki and Jonathan Yedidia

TR2003-107 August 2003

Abstract

The performance of iterative decoding algorithms for multi-step majority logic decodable (MSMLD) codes of intermediate length is investigated. 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. 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).

Proceedings of the 2003 International Symposium on Turbo Codes and Related Codes

Publication History:–

1. First printing, TR-2003-107, August 2003

Iterative Decoding of Multi-Step Majority Logic Decodable Codes

Marc Fossorier
Dept. of Electrical Engineering, University of Hawaii, Honolulu HI96822, USA
Ravi Palanki
Dept. of Electrical Engineering, California Institute of Technology, Pasadena CA 91125, USA
Jonathan Yedidia
MERL, 201 Broadway Av., 8th Floor, Cambridge, MA 02139, USA
E-mail:marc@spectra.eng.hawaii.edu, ravi@magica.systems.caltech.edu, yedidia@merl.com

Abstract: *The performance of iterative decoding algorithms for multi-step majority logic decodable (MSMLD) codes of intermediate length is investigated. 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. 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).*

Keywords: iterative decoding, majority logic decoding, bit flipping decoding, Reed Muller codes, Euclidean geometry codes.

1. Introduction

Recently, iterative decoding algorithms for low density parity check (LDPC) codes have received a great deal of attention. In [1], [2], a (J, L) LDPC code is defined as an (N, K, d) linear block code whose M x N parity check matrix H has J ones per column and L ones per row, where J and L are relatively small numbers. For large N, it is quite easy to avoid the occurrence of two check sums intersecting on more than one position when constructing H. In that case, the check sums are called orthogonal and the Tanner graph representation [3] of H has girth at least six. This is often considered to be an important feature for good performance of iterative decoding [4], [5].

In [6]–[9], it was shown that iterative decoding of one-step majority logic decodable codes also performed very well; indeed often better than for ordinary LDPC codes of similar blocklength and rate for lengths up to a few thousand bits. Despite the fact that the parity check matrix of these codes has a higher density of ones than that of the original LDPC codes, the geometric structure guarantees a girth of six. Perhaps even more importantly, the matrix H used for decoding is highly redundant, i.e. $M > N - K$, and this feature seems to significantly help iterative decoding algorithms.

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 [10]–[14]. However, whereas on the AWGN channel the performance of iterative decoding does not approach that of maximum likelihood decoding (MLD), we find that on the BSC, fast and low complexity bit flipping (BF) algorithms can achieve near MLD performance.

The paper is organized as follows. After a brief review of MSMLD codes in Section 2, an improved version of the Gallager's bit flipping algorithm B is presented and analyzed in Section 3. Different decoding approaches exploiting the structure of MSMLD codes are proposed in Section 4 and simulation results are reported in Section 5. Possible extensions to iterative decoding of these codes for the AWGN channel are discussed in Section 6 and concluding remarks are finally given in Section 7.

2. A Brief Review of Multi-Step Majority Logic Decodable Codes

The most famous MSMLD codes are the Reed-Muller (RM) codes introduced in [15]. Motivated by the efficient multi-step majority logic decoding algorithm proposed in [16], several other classes of MSMLD codes were developed in the 1960's and 70's. Many of these are based on constructions derived from finite geometries [17]–[20]. Unfortunately, the minimum distance d of these codes does not compare favorably to that of their counterpart BCH codes. Consequently, when decoded using a t-bounded distance decoding (t-BDD) algorithm (i.e., when decoded up to the guaranteed error correcting capability t of the code), they are outperformed by BCH codes also decoded by a t-BDD algorithm.

One-step majority logic decodable codes can also be viewed as a special class of LDPC codes with orthogonal check sums. For example, a one step majority logic decodable Euclidean geometry (EG) code of length $N = 2^{ms} - 1$ over the finite field $GF(2^s)$ is also an LDPC code with

$$\begin{aligned} J &= \frac{2^{ms} - 1}{2^s - 1} - 1, \\ L &= 2^s. \end{aligned}$$

Iterative decoding of these codes has been shown to perform very well and most importantly for the BSC, is able to correctly decode many error patterns with considerably more than t errors.

The main feature in the construction of one-step majority logic decodable codes is the same as that of LDPC codes, that is the fact that each bit can be estimated by J check sums orthogonal on it. In constructing a μ-step majority logic decodable code, this principle is generalized into μ steps as follows: at step i, $1 \leq i \leq \mu$, the modulo-2 sum of K_i bits is estimated by J_i check sums orthogonal on these K_i positions, with $K_\mu = 1$, $K_i < K_{i-1}$, and $J_i \geq d-1$. While μ-step majority logic decoding directly follows this construction method, its extension to an iterative decoding method is not straightforward for $\mu \geq 2$ because any graphical representation of H necessarily contains many four-cycles corresponding to check sums intersecting on K_1 positions.

In the following, we consider the family of μ-step majority logic decodable EG codes since the same developments apply to other families of majority logic decodable codes.

3. Three-state Decoding Algorithm

In [1], [2], Gallager proposed two different BF algorithms. These algorithms are designed for LDPC codes with few check sums of low weight orthogonal on each bit and therefore, careful attention must be paid to the introduction of correlations in the iterative process. In particular, in Gallager's bit-flipping algorithms, he takes care that the "message" from a bit to its neighboring check should not directly depend on the message sent by that check back to the bit and vice versa. In our case, because of the very large number of check sums intersecting on each bit, we can neglect that refinement with negligible performance degradation, and obtain the following algorithm, which simplifies Gallager's algorithm-B:

- For each check sum m and for each bit n in check sum 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 check sums 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. While for the codes considered in [1], [2], N_u can take only a few different values, this is no longer 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 propose to modify the algorithm described above into the following "three-state" algorithm, which also allows bits to be erased and check sums to be de-activated.

- For each check sum m and for each bit n in check sum 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 check sum is de-activated.
- For each bit n, determine the number N_{ua} of unsatisfied activated check sums 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, there are many ways to do this, and we only could roughly optimize to find the best schedules, but fortunately the performance seems to be a rather insensitive function of the choice made. For our schedules, 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 [21, 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 [21] can be used to analyze the three-state algorithm.

4. Decoding Approaches

4.1. Fixed Cost Approaches

4.1.1. Direct Approach

A μ-step majority logic decodable EG code can be represented by its M x N incidence matrix H in which rows represent μ-flats and columns points, with $h_{ij} = 1$ if the j-th point belongs to the i-th μ-flat.

A straightforward approach is to run the BF algorithm based on H. This matrix will be plagued by many four-cycles, but fortunately it can also be made very redundant with $M >> N$, and the weight of each row of the parity check matrix need not be too high. Furthermore, by exploiting the cyclic structure of the code, a very balanced graph is obtained so that the same speed of convergence can be expected in all parts of the graph.

4.1.2. Multi-Step Approach

In [14], a general method was presented for modifying the parity check matrix of a code to make it more suitable for iterative message-passing algorithms. Using this method on a two-step majority logic decodable EG code, one obtains a new parity check matrix whose graphical representation contains no four-cycles. It is a $(M_1 + M_2)$ x $(N_1 + N_2)$ matrix

$$H = \begin{bmatrix} A & B \\ D & C \end{bmatrix}, \quad (1)$$

in which the M_1 and M_2 rows represent the plane constraints and line constraints, respectively, and the N_1 and N_2 columns represent the points and lines, respectively. As a result, $M_2 = N_2$ and C represents the identity matrix, A is the all-0 matrix while the remaining matrices B and D are free of four-cycles (and so is H). Generalization of (1) to μ-step majority logic decodable EG codes is straightforward.

Decoding based on (1) can be realized in at least two ways. First the BF algorithm can be run on H with the N_2 nodes corresponding to the lines initialized without a-priori knowledge. The drawback of this approach is that nodes with no a priori information from the channel directly exchange highly unreliable information with each other.

To overcome this problem, H can be modified so that each row of B has weight one. If a plane is composed of l lines, this corresponds to duplicating each plane l times and viewing it as the union of one line and of the points composing the remaining $l-1$ lines. As a result, nodes without a-priori information no longer directly exchange information, but the graph representation of the resulting matrix A now contains many four-cycles. The BF algorithm can then be decomposed in two steps based on the following scheduling: in step-1, only the top part $[AB]$ of H is used to estimate the N_2 lines, while in step-2, the bottom part $[CD]$ is used to estimate the N_1 points. We notice that this scheduling "mimics" two-step majority logic decoding and can be easily generalized to μ steps for μ-step majority logic decodable codes.

4.1.3. Decomposable Approach

By their construction, several μ-step majority logic decodable codes have a decomposable structure. For example, Reed-Muller (RM) codes can be constructed by the $|u|u \oplus v|$ construction or the iterative squaring construction [22]. For simplicity, we consider the $|u|u \oplus v|$ construction in the following. If C_1 and C_2 are two codes with parity check matrices H_1 and H_2, respectively, then $C = |C_1|C_1 \oplus C_2|$ has parity check matrix

$$H = \begin{bmatrix} H_2 & H_2 \\ H_1 & 0 \end{bmatrix}. \quad (2)$$

Following the approach described in [23], [24], two stage decoding based on (2) is performed as follows. Assuming the received sequence corresponding to the codeword $|u_1|u_1 \oplus u_2|$ is $y = |y_1|y_2|$, first $y_1 \oplus y_2$ is decoded by a BF algorithm based on H_2 to estimate $\hat{u}_2$. Then $|y_1|y_2 \oplus \hat{u}_2|$ is decoded by the three-state BF algorithm of Section 2 based on H_1 to estimate $\hat{u}_1$. At the initialization of this second decoding stage, the values which coincide in y_1 and $y_2 \oplus \hat{u}_2$ are conserved, while the other values are erased.

4.2. Variable Cost Approach

The matrix H used for decoding is generally highly redundant, so that $M >> N$. If a sufficient number of check sums is used, then the BF algorithm converges rapidly to its final solution while if not enough check sums 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 check sums 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 check sums during N_b iterations. This process is repeated until either a codeword is found, or all M check sums have been used without success, in which case the decoding fails.

5. 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.

5.1. (255,127,21) EG Code

In Figure 1, the simulated error performance of three-state BF decoding of the (255,127,21) EG code

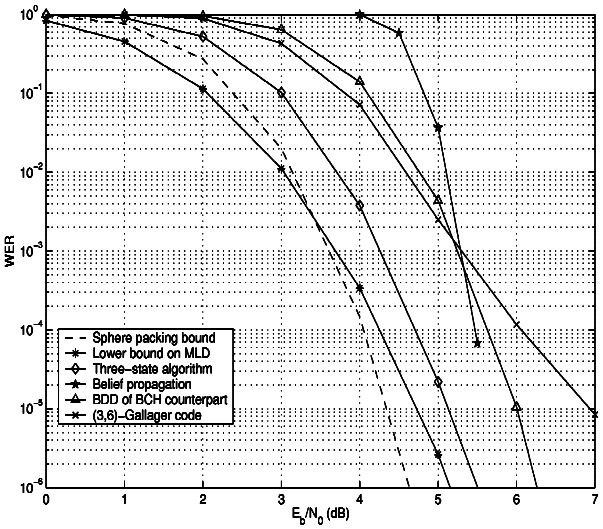

Figure 1: BF decoding of the (255,127,21) EG code; (a) low SNR regime.

with the direct approach of Section 4.1.1. is compared to t-BDD of its (255,123,39) BCH code counterpart as well as its $(3, 6)$ Gallager LDPC code counterpart. This EG code corresponds to a $\mu = 2$ Euclidean geometry with 255 points and 5355 planes, so we can construct a parity check matrix H with 5355 rows and 255 columns. We observe that three-state 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 much worse performance of BP at low SNR's are elaborated in Section 6.

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.

Since this code is two-step majority logic decodable, the two-step approach of Section 4.1.2. was also implemented. The decomposition of [14] gives an 32,130 x 5610 matrix. Each row corresponding to one of the of 26,775 plane constraints in this matrix has to be duplicated four times to have weight one in the B-part of (1). The final matrix H given by (1) becomes an 112,455 x 5610 matrix. Unfortunately, despite the large increase in complexity, only a tiny improvement was obtained by this approach. One explanation is that the multi-step approach can be viewed as a particular scheduling of the direct approach in which hidden nodes are introduced as intermediary states. As a result, the information initially available is used in successive steps rather than at once as in the direct approach. In the case of a binary erasure channel (BEC), the increased number of constraints and erasures associated with the multi-step approach helps in improving the decoding as information can only be improved [14]. However, for the BSC (or other channels introducing errors), erroneous decisions can propagate through the hidden nodes so that using all available information at once in a suboptimum way becomes as good as using it partially in a more optimum (but still suboptimum after iteration-1) way. The only advantage of the multi-step approach is its guarantee to perform no worse than t-BDD since its first iteration can be made equivalent to multi-step majority logic decoding with $b_1 = b_2 = \lceil J/2 \rceil$.

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. These plots actually show the performance of the two-step algorithm described above, but as mentioned already, the difference in performance between the direct 3-state algorithm and the more complex two-step algorithm is tiny. At all word error rates (WERs) down to 10^{-20}, this difference is less than 0.1 dB.

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

$$P_s \quad = \quad \sum_{w=t+1}^{N} P_s(w) \binom{N}{w} p_0^w (1-p_0)^{N-w}. \quad (3)$$

The results are reported in Figure 3. Since for WERs larger than 10^{-6}, no reliable evaluation of $P_s(w)$ is possible, we computed: (a) an upper bound on (3) by assuming the same $P_s(w_{min})$ as the smallest simulated for weights $w^{'}$, $t < w^{'} < w_{min}$; (b) a lower bound on (3) 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 from the lower bound on P_s. 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^{-13} for the two-step ap-

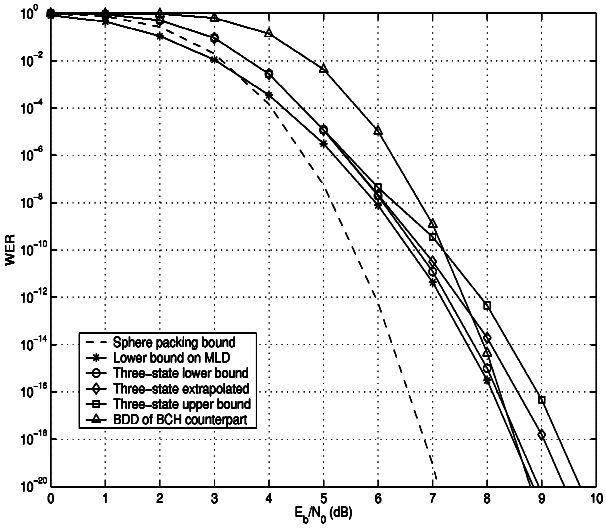

Figure 2: BF decoding of the (255,127,21) EG code; (b) high SNR regime.

proach (and 10^{-12} for the direct approach).

5.2. (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 direct approach of Section 4.1.1. and the decomposable approach of Section 4.1.3. based on the $|u|u \oplus v|$ construction. For comparison, the SPB and t-BDD of the counterpart (511,250,63) BCH code have also been represented.

For the direct approach, $M = 76,650$ and $M = 511,000$ have been considered (corresponding to 150 and 1000 different cyclic shifts of weight-32 codewords of the dual code, respectively). Also the progressive method was used to speed up each decoding. In both case, we chose five different sizes of the set of check sums 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 check sums 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 progressive 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 check sums 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

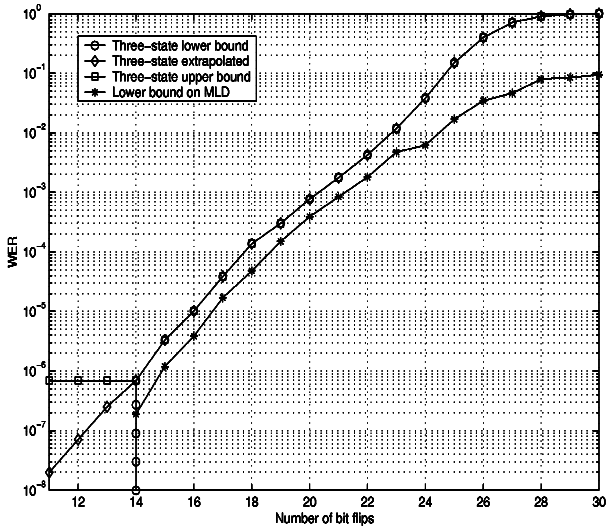

Figure 3: BF decoding of the (255,127,21) EG code-for fixed number of errors.

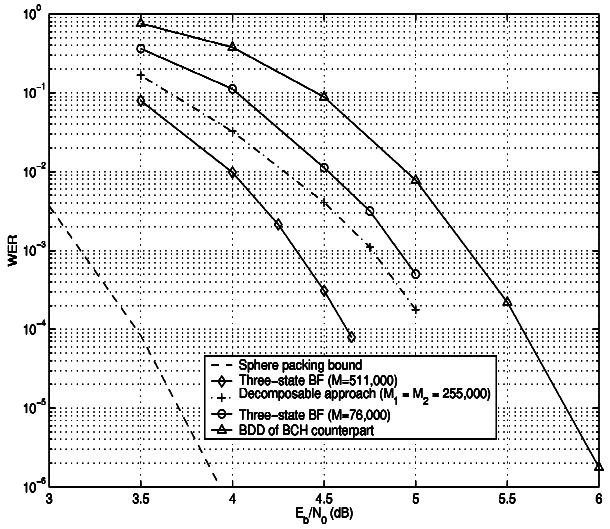

Figure 4: BF decoding of the (511,256,31) EG (or RM) code.

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 represented in Figure 4.

The decomposable approach of Section 4.3 was also tried with C_1 and C_2 being the (255,163,15) and (255,93,31) RM codes, respectively (resulting in a (510,256,30) code). At each stage, at most $M_1 = M_2 = 255,000$ check sums were considered. Again the progressive approach was used with all previous sizes of check sum sets divided by two. When decoded separately with $M = 255,000$, about 95% of the errors are undetectable errors, and about 40% of the errors are MLD errors for the (255,163,15) RM code. For the (255,93,31) RM code, about 80-90% of the errors are undetectable errors while about 10% are MLD errors. However, despite these near MLD individual performances, the resulting two-stage decoding is not as good as expected. This is mostly due to the dominance of undetected errors at stage-1 in conjunction with the suboptimality of this approach (a slight improvement can be obtained by choosing $M_2 > M_1$ since the performance of stage-1 dominates the overall error performance). Hence, the applications of the techniques developed in [25]–[28] to iterative decoding should provide interesting error performance improvements.

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.

6. 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 [29], 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 ($f \circ g = \alpha f$, $0 < \alpha \leq 1$), offsetting ($f \circ g = \text{sgn}(f)\max\{|f| - \beta, 0\}$), damping ($f \circ g = \alpha f_i + (1-\alpha) f_{i-1}$, $0 < \alpha \leq 1$), or clipping ($f \circ g = \text{sgn}(f)\min\{|f|, C\}$). 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 felt short of MLD by about 0.8 dB.

7. 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 check sums. 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 [30], [31].

REFERENCES

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

[2] R.G. Gallager, "Low-Density Parity-Check Codes," *IRE Trans. Inform. Theory*, vol. 8, pp.21-28, Jan. 1968.

[3] R.M. Tanner, "A Recursive Approach to Low Complexity Codes," *IEEE Trans. Inform. Theory*, vol. 27, pp. 533-547, Sept. 1981.

[4] D.J.C. MacKay and R.M. Neal, "Near Shannon Limit Performance of Low Density Parity Check Codes," *Electron. Let.*, vol. 32, pp. 1645-1646, Aug. 1996.

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

[6] K. Yamaguchi, H. Iizuka, E. Nomura and H. Imai, "Variable Threshold Soft Decision Decoding", *IEICE Trans. Elect. and Comm.*, vol. 72, pp. 65-74, Sept. 1989.

[7] K. Karplus and H. Krit, "A Semi-Systolic Decoder for the PDSC-73 Error-Correcting Code," *Discrete Applied Mathematics 33*, North-Holland, pp. 109-128, 1991.

[8] 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.

[9] Y. Kou, S. Lin and M. Fossorier, "Low Density Parity Check Codes Based on Finite Geometries: A Rediscovery and New Results," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2711-2736, Nov. 2001.

[10] 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.

[11] S. Lin, H. Tang and Y. Kou, "On a Class of Finite Geometry Low Density Parity Check Codes," *The Proc. IEEE Intern. Sympos. Inform. Theory*, Washington, USA, June 2001.

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

[13] 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.

[14] 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.

[15] 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.

[16] I.S. Reed, "A Class of Multiple-Error-Correcting Codes and the Decoding Scheme," *IRE Trans. Inform. Theory*, vol. 4, pp. 38-49, Sept. 1954.

[17] W.W. Peterson and E.J Weldon Jr., *Error-Correcting Codes (Second Edition)*, Cambridge: M.I.T. Press, 1972.

[18] I.F. Blake and R.C. Mullin, *The Mathematical theory of Coding*, Academic Press, New York, 1975.

[19] F.J. Mac Williams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam: North-Holland Mathematical Library, 1977.

[20] S. Lin and D.J. Costello, Jr., *Error Control Coding, Fundamentals and Applications*, Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1983.

[21] 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.

[22] G.D. Forney Jr., "Coset Codes II: Binary Lattices and Related Codes," *IEEE Trans. Inform. Theory*, vol. 34, pp. 1152-1187, Sept. 1988.

[23] F. Hemmati, "Closest Coset Decoding of $|u|u+v|$ Codes," *IEEE Jour. Select. Areas Commun.*, vol. 7, pp. 982-988, Aug. 1989.

[24] M. Fossorier and S. Lin, "Generalized Coset Decoding," *IEEE Trans. Commun.*, vol. 45, pp. 393-395, Apr. 1997.

[25] D. Stojanovic, M. Fossorier and S. Lin, "Iterative Multi-stage Maximum Likelihood Decoding of Multi-level Concatenated Codes," *The Proc. Workshop on Coding and Cryptography*, Paris, France, Jan. 1999.

[26] I. Dumer and K. Shabunov, "Recursive Decoding of Reed-Muller Codes," *The Proc. IEEE Intern. Sympos. Inform. Theory*, Sorrento, Italy, June 2000.

[27] I. Dumer and K. Shabunov, "Near-Optimum Decoding for Subcodes of Reed-Muller Codes," *The Proc. IEEE Intern. Sympos. Inform. Theory*, Washington, USA, June 2001.

[28] I. Dumer and K. Shabunov, "Recursive and Permutation Decoding for Reed-Muller Codes," *The Proc. IEEE Intern. Sympos. Inform. Theory*, Lausanne, Switzerland, June 2002.

[29] 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.

[30] J. Stern, "A Method for Finding Codewords of Small Weight", *Lecture Notes in Computer Science*, vol. 388, pp 106-113, Springer Verlag, 1989.

[31] A. Canteaut and F. Chabaud, 'A New Algorithm for Finding Minimum-Weight Words in a Linear Code: Application to McEliece's Cryptosystem and to Narrow-Sense BCH Codes of Length 511", *IEEE Trans. Inform. Theory*, vol. 44, pp. 367-378, Jan. 1998.