Iterative Reliability-Based Decoding of Low-Density Parity Check Codes

Marc P. C. Fossorier, Senior Member, IEEE

Abstract—In this paper, reliability based decoding is combined with belief propagation (BP) decoding for low-density parity check (LDPC) codes. At each iteration, the soft output values delivered by the BP algorithm are used as reliability values to perform reduced complexity soft decision decoding of the code considered. This approach allows to bridge the error performance gap between belief propagation decoding which remains suboptimum, and maximum likelihood decoding which is too complex to be implemented for the codes considered. Trade-offs between decoding complexity and error performance are also investigated. In particular, a stopping criterion which reduces the average number of iterations at the expense of very little performance degradation is proposed for this combined decoding approach. Simulation results for several Gallager LDPC codes and different set cyclic codes of hundreds of information bits are given and elaborated.

Index Terms—Block code, four density parity check code, iterative decoding, reliability decoding, soft decision decoding.

I. INTRODUCTION

R

Ecently, iterative decoding of low-density parity check (LDPC) codes, first introduced in [1], [2], has received a great deal of interest. The main motivations of these works are the excellent performances achieved by such codes for long enough lengths, and their relatively simple iterative decoding methods based on belief propagation (BP) [3]–[6]. Although belief propagation iterative decoding of such codes does not achieve maximum likelihood decoding (MLD), it has been shown both theoretically and experimentally that its application to improved constructions derived from Gallager original codes [7]–[10] can closely approach channel capacity for the additive white Gaussian noise (AWGN) channel. Interestingly, other families of codes such as quasicyclic based codes [11] and one-step majority logic decodable codes [12], [13] can also be decoded iteratively with BP1. On the other hand, the derivation of these theoretical limits assumes that the influences of loops in the Tanner graph representation of the codes [14], [15] are irrelevant, which may need very long code lengths to be satisfied with a sufficiently high probability [7]. Consequently, BP decoding of codes of medium lengths (say between hundreds and few thousand bits) may fall short of MLD, not only due to the suboptimum nature of iterative decoding but also due to the suboptimum graph structure of the code because of the relevant presence of short loops.

On the other hand, MLD of such codes is generally too complex to be implemented. Reliability based near optimum probabilistic decoding is another possible approach to tradeoff error performance, and decoding complexity. In [16], a reprocessing strategy in successive phases has been proposed. This decoding method is referred to as ordered statistic decoding (OSD). However, for each reprocessing phase \( i \), \( i \geq 0 \), \( \binom{K}{i} \) codewords need to be processed, where \( K \) is the dimension of the code. As a result, for LDPC codes of dimensions larger than hundreds of bits, only few reprocessing phases of OSD can be implemented and the resulting error performance remains far from that of MLD.

Phase-0 of the OSD algorithm can be viewed as performing information set decoding in the most reliable information set obtained from the noisy received sequence (i.e., the \( K \) most reliable independent positions (MRIPs) in the received sequence). Consequently, no information about the code considered is used to construct this information set. On the other hand, the algorithm of [17] can be viewed as the optimum reliability based information set decoding since the entire code is taken into account to estimate the \( K \) information bits. In [18], a generalized strategy is proposed in which a supercode of the code considered is used to determine the most reliable information set. As a result, phase-0 of [16] and the decoding method of [17] can be viewed as the extreme cases in which the supercode is the universal code and the code considered itself, respectively. Unfortunately, for a given code rate, near-optimum information set decoding with the algorithm of [18] rapidly becomes too complex due to the exponential increase of the computational complexity with respect to \( K \). In addition, the reprocessing approach of [16] in successive phases does not perform well when combined with the algorithm of [18]. This can be easily explained by the fact that if the most reliable information set determined from a supercode of the code considered is erroneous, then the corresponding number of bit errors in the MRIPs is likely to be larger than that of the most reliable information set determined from the universal code due to the correlations between some information bits. Another generalization of the algorithm of [16] based on several reliable information sets has been presented in [19] but this generalization would achieve near MLD only for relatively low code rates. Consequently, all information set probabilistic decoding methods previously discussed remain too complex to achieve near-optimum decoding of LDPC codes, despite their important computational complexity reduction with respect to MLD of these codes. Note that the main drawback of these MRP-based decoding methods

1In this paper, we restrict our attention to codes constructed without an interleaver, although the proposed approach can be readily extended to “turbo-like” codes.
is the fact that reliabilities from the positions outside the MRIPs are either ignored [16] or only partially considered [18], [19] when constructing the initial state of the reduced search to determine the list of candidate codewords. As a result, these algorithms are in general efficient for codes with relatively small codimension only. It follows that reliability based algorithms which include reliability informations from all positions in a balanced way when determining the initial state of the reduced search are more promising for codes of medium to long lengths. The algorithm considered in the following implicitly achieves this goal in successive steps.

In this paper, we propose to combine BP decoding with reliability based decoding. At each iteration, the soft-output values delivered by the BP algorithm are viewed as reliability values by the soft decision decoding algorithm. As a result, at each iteration, BP decoding is completed by a low complexity reliability based soft decision decoding algorithm. In order to keep a manageable decoding complexity, we choose phase-0 and phase-1 reprocessing of the algorithm of [16], so that \( O(K \log K) \) computations are required to determine the most reliable information set based on the soft-output values delivered by BP, and \( O(K) \) candidate codewords are processed at each iteration. This approach allows to bridge the gap between the error performance achieved by the BP algorithm and that of MLD, which is much too complex to be implemented for the codes considered. Simulation results for LDPC codes and different set cyclic (DSC) codes of hundreds of information bits are reported. In all cases, nonnegligible improvements over BP decoding are obtained with the proposed method for the LDPC codes considered. As expected, for a fixed code rate, the relative gains achieved by the proposed method over BP decoding increase as the code length decreases. Finally, a new stopping criterion which takes into account the joint decoding method is proposed to reduce the average number of iterations, and consequently, the average number of OSDs. Other tradeoffs between decoding complexity and error performance are also investigated.

The paper is organized as follows. The proposed combined algorithm and a simple stopping criterion for this algorithm are introduced in Section II. Simulation results of the proposed combined scheme are presented and discussed in Section III, while tradeoffs between error performance and decoding cost are investigated in Section IV. Finally, concluding remarks are given in Section V.

II. COMBINED BELIEF PROPAGATION AND ORDERED STATISTIC DECODINGS

In the following, binary error control coding for BPSK signaling over the AWGN channel is considered.

A. Decoding Algorithm

BP decoding iteratively updates the \( \text{a posteriori} \) probabilities of each bit of the code sequence. For each bit, its associated \( \text{a posteriori} \) probability is evaluated based on a predetermined set of check sums assumed to be orthogonal\(^2\) on the position of that bit, and the \( \text{a priori} \) probabilities of the bits contributing to this set of check sums. The new information provided to the original \( \text{a priori} \) probability of the bit considered by the corresponding computed \( \text{a posteriori} \) probability is then used as additional \( \text{a priori} \) information at the next iteration. A detailed description of the algorithm can be found for example in [6]. It follows that BP decoding is suitable to decode codes for which a set of check sums orthogonal on each coded bit position can be constructed. Such is the case for LDPC codes [1], [2] and one-step majority logic decodable codes [20, Ch. 7 and 8]. Furthermore, the influence of correlations between values introduced by the iterative decoding process has to be minimized in the code construction. This can be achieved by maximizing the girth of the graph representation of the code [7], [21], although the density of short loops in this graph also influences the error performance. Note that all codes of practical interest have loops in their graph representation [22].

The OSD algorithm consists of two main parts [16]: 1) construction of the most reliable information set [also referred to as the most reliable basis (MRB)] and 2) systematic reprocessing of candidate codewords expressed in the MRB. For an \((N,K)\) binary linear block code, the \( K \) MRIP’s define the MRB, which is constructed as follows.

a) Order the reliabilities associated with the hard decisions of the received values in decreasing order, which defines a first permutation \( \lambda_1 \).

b) Apply \( \lambda_2 \) to the received sequence and to the columns of the generator matrix associated with the code considered.

Let \( y_1 \) and \( G_1 \) denote the resulting received sequence and generator matrix, respectively.

c) Perform a Gaussian elimination on \( G_1 \) from left to right. The dependent columns found during the Gaussian elimination are then permuted after the last independent column found to obtain a matrix \( G_2 \) in systematic form, which defines a second permutation \( \lambda_2 \).

d) Apply \( \lambda_2 \) to the permuted received sequence \( y_1 \), which defines a second permuted received sequence \( y_2 \).

The MRB consists of the \( K \) positions corresponding to the leftmost set of independent columns found in step (c). Note that if \( c_2 \) represents a codeword in the code defined by \( G_2 \), then \( \lambda_2^{-1}(\lambda_2^{-1}(c_2)) \) represents the corresponding codeword in the original code, where \( \lambda_i^{-1} \) represents the inverse permutation to \( \lambda_i \). Once the MRB has been identified, the order-\( i \) OSD algorithm is conducted as follows.

a) For \( 0 \leq i \leq 3 \), make all possible changes of \( i \) of the \( K \) most reliable bits in the hard decision decoding of the permuted noisy received sequence \( y_2 \).

b) For each change, reencode these \( K \) bits based on \( G_2 \) (in systematic form), and compute the decoding metric associated with each constructed codeword.

c) Select the most likely codeword among the \( L(i) = \sum_{i=0}^{K} \binom{K}{i} \) constructed candidate codewords.

Each value of \( i \) determines the reprocessing phase-\( i \) of order-\( i \) reprocessing. Note finally that an equivalent dual implementa-

\(^2\)Note that if quantized received values are considered, the MRB is determined in a straightforward way based on the numbers of values quantized to each level within a block.
tion of OSD based on the processing of the parity check matrix of the code considered rather than its generator matrix is also possible [23].

In this paper, we propose to process the OSD at the end of each iteration of the BP decoding. The \textit{a posteriori} probabilities delivered by the BP decoding at each iteration are considered as reliability values by the OSD and are used to construct the corresponding MRB. Let $B_K$ represent the set of the $K$ locations (indices) associated with the MRIP’s determined from the \textit{a posteriori} probabilities delivered by BP decoding. Furthermore, let $I(B_K) = \{\hat{x}_j : j \in B_K\}$ define the set of binary values corresponding to the largest \textit{a posteriori} probabilities computed by the BP decoding for each position in $B_K$ at the iteration considered. Then, OSD is realized based on the set $I(B_K)$ as original information set and the original received sequence $y$. More specifically, phase-0 reprocessing corresponds to the codeword associated with $I(B_K)$, while phase-1 reprocessing considers the $K$ codewords obtained by inverting a single position in $I(B_K)$. Since phase-1 of order-$i$ decoding requires the reprocessing of $\binom{K}{i}$ candidate codewords and codes of hundreds of information bits are considered, we limit the OSD to order-1 reprocessing to keep the overall decoding complexity manageable. For each constructed codeword, its associated decoding metric is computed based on the original noisy received sequence $y$. After a predetermined maximum number of iterations, the most likely codeword recorded by the combined decoding algorithm is delivered as the final solution. Note that this algorithm always delivers a codeword as its final estimate. Consequently, as opposed to BP decoding, no decoding error can be detected based on the decoded sequence. The flow-chart of this combined algorithm is given in Section II-C.

Unfortunately, an error performance analysis of this combined algorithm seems very difficult as for the medium length codes considered, no theoretical framework to derive the decoding error probabilities is available. Furthermore, these codes are too large and generally, not structured enough to compute their weight distribution, so that a tight approximation of their MLD error performance is also not feasible.

### B. Stopping Criterion

In [6], it is proposed to check whether the hard decision decoding of the \textit{a posteriori} probabilities delivered at each iteration defines a codeword by checking whether the corresponding syndrome is zero. This stopping criterion can be justified by the fact that for the word error rates (WERs) considered, an undetectable error is much less likely to occur than a decoding error. Indeed, this stopping criterion can still be used in our combined decoding method.

In [6], [24], and [25], sufficient conditions on the optimality of a candidate codeword expressed in the MRB of the code have been derived. These conditions improve upon the universal conditions derived in [26]–[28] by exploiting the OSD reprocessing strategy in the MRB. Unfortunately, the conditions of [16], [24], [25] can not be used in a straightforward way in our approach since after the first iteration, the hard decisions of the MRIPs based on the \textit{a posteriori} probabilities delivered at each iteration do not necessarily equal the hard decisions of the corresponding noisy received values. On the other hand, the general sufficient conditions [26]–[28] remain valid but are unlikely to offer significant computation savings for this decoding approach.

In order to further reduce the average number of iterations and consequently, the average number of OSDs, we propose the following test, denoted $T(\alpha)$, $\alpha \geq 2$. “If the most likely codeword found by the combined algorithm at a given iteration is also delivered by OSD at $\alpha - 1$ subsequent iterations, then this codeword is chosen as the final estimate.” This simple test relies on the fact that if at a given iteration, near-convergence is not achieved by the iterative decoding, the reliability values at subsequent iterations are likely to be modified sufficiently to define different MRBs. The test $T(\alpha)$ is easily implemented with modulo-2 arithmetic operators. For all codes simulated in this paper, the test $T(2)$ provides a good tradeoff as significant computation savings are achieved with little error performance degradation.

### C. Algorithm Flow-Chart

The detailed realization of the combined algorithm proposed in Section II-A in conjunction with the stopping criteria discussed in Section II-B are given in the following. To clearly distinguish how the reliability values computed at each iteration by BP and the initial received values are used by the OSD algorithm, the BP decoding has been reviewed in details following [6]. Note also that the approach presented in [23] perfectly fits this combined algorithm as OSD is expressed in the dual code defined by $H$, the parity check matrix of the code considered.

Define $\Omega_n$ as the set of check sums containing bit $n$, and $\omega(m)$ as the set of bits forming check sum $m$. The iterative BP decoding algorithm has two alternating parts, in which certain quantities $q_{mn}$ and $r_{mn}$ are iteratively updated. The quantity $q_{mn}$ is meant to be the probability that bit $n$ is taking the value $x$, given the information obtained via checks other than check $m$ (i.e., $\Omega_n \setminus \{m\}$). The quantity $r_{mn}$ is meant to be the probability that check sum $m$ is satisfied if bit $n$ is considered fixed at $x$, and the other bits have a separable distribution given by the probabilities $\{q_{m'n} : n' \in \omega(m) \setminus \{n\}\}$.

Initially, for $n = 1, \ldots, N$, the bit decisions $\hat{x}_n$ and $\tilde{x}_n$ are initialized to the symbols $z_n$, corresponding to the hard decision decoding of the noisy received values $y_n$, and the variables $q_{0mn}$ and $r_{0mn}$ are initialized to the \textit{a priori} probabilities $\rho_n$ and $j_n = 1 - \rho_n$ associated with $z_n$ and $z_n \oplus 1$. The value $D_{\text{init}}$ is initialized to a large number, the value $\alpha$ is initialized to a predetermined value to be used in the test $T(\alpha)$, and the counter $N(\alpha)$ is set to zero. Then each iteration is processed as follows:

- **BP Decoding:**
  
  \textbf{Step 1:} Define $\delta_{0mn} = q_{0mn} - q_{0mn}$ and for each $m, n$, and for $x = 0, 1$, compute:
  
  \begin{align*}
  a) & \quad \delta r_{mn} = \prod_{n' \in \omega(m)} \delta q_{m'n'} \\
  b) & \quad r_{mn}^x = (1/2)(1 + (-1)^x \delta r_{mn}).
  \end{align*}


Step 2:
a) For each $n$ and each $m$, and for $x = 0, 1$, update:

$$q_{nm}^{x} = \alpha_{nm} f_{n}^{x} \prod_{m' \in \Omega_{n} \setminus m} r_{m' n}^{x}$$

where $\alpha_{nm}$ is chosen such that $q_{nm}^{0} + q_{nm}^{1} = 1$.
b) For each $n$ and $x = 0, 1$, update the “pseudo-posterior probabilities” $q_{n}^{x}$ and $q_{n}^{1-x}$ given by:

$$q_{n}^{x} = \alpha_{n} f_{n}^{x} \prod_{m \in \Omega_{n}} r_{m n}^{x}$$

where $\alpha_{n}$ is chosen such that $q_{n}^{0} + q_{n}^{1} = 1$.

Step 3:
a) Create $\hat{x} = [\hat{x}_{n}]$ such that $\hat{x}_{n} = 1$ if $q_{n}^{1} > 0.5$, and $\hat{x}_{n} = 0$ if $q_{n}^{0} \leq 0.5$.
b) do the following:
- If $\Pi \hat{x} = 0$ then the decoding algorithm halts and $\hat{x}$ is considered as a valid decoding result.
- Else, the algorithm starts the “Order-i Reprocessing.”

• **Order-i Reprocessing:**
  
  Step 1: Based on the reliability values $\Pi \max \{q_{n}^{0}, q_{n}^{1}\}$, determine the $K$ corresponding MRIPs. Define $B_{K}$ as the set of locations of these MRIPs.
  
  Step 2: Take $I(B_{K}) = \{\hat{x}_{j} : j \in B_{K}\}$ created at Step 3-(a) of BP decoding as initial information sequence.
  
  b) For $0 \leq l \leq i$, make all possible $\binom{i}{l}$ changes of $l$ values in $I(B_{K})$ and for each change, determine the corresponding codeword $c$.
  
  Step 3: Let $\hat{c}$ denote the codeword with minimum discrepancy $D_{\min}$ constructed so far by the iterative combined decoding algorithm.
  
  a) For each codeword $c$ constructed at Step 2:
  - If $c = \hat{c}$, check whether test $T(\alpha)$ is satisfied based on the counter $N(\alpha)$ (automatically satisfied for $\alpha = 2$):
    * If $N(\alpha) = \alpha - 1$, then $T(\alpha)$ is satisfied; the decoding algorithm halts and $\hat{c}$ is considered as a valid decoding result.
    * Else, increment $N(\alpha) = N(\alpha) + 1$.
  - Else, evaluate the discrepancy:
    $$D(y, c) = \sum_{i \in c_{i} \neq \hat{c}_{i}} |y_{i}|$$
  
  between $c$ and the original noisy received sequence $y$.
  - If $D(y, c) < D_{\min}$, set $D_{\min} = D(y, c)$. $\hat{c} = c$ and $N(\alpha) = 1$.
  c) Start the next iteration with BP decoding.

The algorithm stops if some maximum number of iterations is reached. The codeword $\hat{c}$ is delivered as the final solution in that case. The OSD can also be processed based on the initial values $f_{n}^{0}$ and $f_{n}^{1}$, in which case iteration-1 becomes equivalent to the conventional order-i reprocessing. Finally, note that if the syndrome test at Step 3-(b) of the BP decoding introduces a nonnegligible number of undetectable errors, then $\hat{x}$ can be considered as a valid result only if it satisfies $D(y, \hat{x}) < D_{\min}$.

It is important to notice that at each iteration, both the MRB defined by $B_{K}$ and the corresponding information sequence $I(B_{K})$ are determined from the updated values $\Pi \max \{q_{n}^{0}, q_{n}^{1}\}$. $n \in B_{K}$, while the discrepancy values are computed-based on the original noisy received sequence $y$. As a result, for $n \in B_{K}$, it is possible to have $\hat{x}_{n} \neq \hat{z}_{n}$.

### TABLE I

**Decoding Complexity for Order-i Reprocessing**

<table>
<thead>
<tr>
<th>Operations</th>
<th>Real additions</th>
<th>Binary additions</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\xi_{\min}$</td>
<td>$NJ$ additions</td>
<td>$Nk$ additions</td>
</tr>
<tr>
<td>$\delta_{\min}$</td>
<td>$3NJ - 1$ multiplications</td>
<td>$Nk$ additions</td>
</tr>
<tr>
<td>$\alpha_{\min}$</td>
<td>$Nk$ additions</td>
<td>$NJ$ additions</td>
</tr>
<tr>
<td>$\beta_{\min}$</td>
<td>$2NJ$ multiplications</td>
<td>$Nk$ additions</td>
</tr>
<tr>
<td>$\gamma_{\min}$</td>
<td>$2Nk$ multiplications</td>
<td>$Nk$ additions</td>
</tr>
</tbody>
</table>

### TABLE II

**Decoding Complexity for One Iteration of BP**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Number of computations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\eta_{\min}$</td>
<td>$NJ$ additions</td>
</tr>
<tr>
<td>$\delta_{\min}$</td>
<td>$3NJ - 1$ multiplications</td>
</tr>
<tr>
<td>$\alpha_{\min}$</td>
<td>$Nk$ additions</td>
</tr>
<tr>
<td>$\beta_{\min}$</td>
<td>$2NJ$ multiplications</td>
</tr>
<tr>
<td>$\gamma_{\min}$</td>
<td>$2Nk$ multiplications</td>
</tr>
</tbody>
</table>

### D. Computational Analysis

Consider an $(N, K)$ linear LDPC block code with $J$ check sums of weight $L$ orthogonal on each bit position. For such a code, Tables I and II summarize the operations required for order-i reprocessing and for one iteration of BP decoding, respectively, based on the analyzes of [16] and [6], [29]. Consequently, the computational cost of OSD is dominated by about $NJk + (N - K)^{2}$ binary additions and $Kk(N - K)$ real additions, while one iteration of BP decoding requires $11NJ - 9N$ real multiplications, $N(J + 1)$ real divisions and $N(3J + 1)$ real additions.

Due to the different natures of the operations involved in each decoding algorithm, any computational comparison becomes arguable. Furthermore, several steps of the OSD can be processed in parallel and it has even been traditionally assumed in the literature that binary additions can be discarded from a computational cost analysis as long as they do not highly dominate the total number of operations. If these arguments seem mostly applicable to hardware implementations, it is worthy mentioning that many binary additions can also be performed in one real addition in a software implementation. Similarly, the values reported in Table II assume forward/backward recursions to compute the values $\delta_{\min}$ and $\gamma_{\min}$ [6]. In hardware implementation, a parallel realization of these computations may become more attractive, in which case decoding speed is traded for a larger computational cost than that of Table II.

---

4 In Table I, a comparison is considered as a real addition since it can be obtained from the sign of a real subtraction.
Fig. 1. WER comparison between BP decoding and BP decoding in conjunction with order-1 reprocessing for a (3,6) (504 252) LDPC code.

Fig. 2. WER comparison between BP decoding and BP decoding in conjunction with order-1 reprocessing for a (3,6) (1 008 504) LDPC code.

III. SIMULATION RESULTS

In this section, simulation results for different types of LDPC codes are given. In all cases, the maximum numbers of iterations have been chosen as the minimum numbers of iterations for which near-convergence is achieved. Also, if not stated otherwise, the test \( T(\alpha) \) was omitted in order to reach the best achievable error performance.

A. LDPC Gallager Codes

Figs. 1 and 2 depict the simulation results for (3,6) (i.e., each bit is associated with three orthogonal check sums of Hamming weight six) Gallager codes with \((N,K)\) parameters (504 252) and (1 008 504), respectively. In both figures, BP decoding is compared with the combined scheme presented in Section II. Based on Fig. 1, we notice that the combined decoding scheme achieves near convergence for a smaller maximum number of iterations than BP decoding, which is a desirable feature. We also observe that for the WER \( 10^{-3.5} \), order-1 reprocessing of at most 50 BP decoding iterations achieves 0.55-dB coding gain over BP decoding of at most 200 iterations. An additional 0.05-dB coding gain is achieved at this WER by increasing the maximum number of order-1 reprocessings from 50 to 200. Furthermore, both order-0 reprocessing with at most 50 BP decoding iterations and order-1 reprocessing with at most ten-BP decoding iterations achieve a good tradeoff between error performance and decoding complexity. For the (1 008 504) LDPC depicted in Fig. 2, at the WER \( 10^{-3.5} \), order-1 reprocessing of at most 50 BP decoding iterations achieves 0.25-dB coding gain over BP decoding of at most 200 iterations. As expected, as the code length increases, smaller coding gains are achieved by the combined scheme since the BP decoding becomes closer and closer to the ideal behavior discussed in [7]. On the other hand, these gains remain important relatively since LDPC codes should approach channel capacity as their code length increases [6], [7], [9].

Fig. 3 compares the effectiveness of the proposed stopping criterion \( T(2) \) used in conjunction with syndrome checking with that of syndrome checking alone for the (504 252) and (1 008 504) LDPC codes. In both cases, we observe a very small performance degradation, as shown in Figs. 1 and 2. On the other hand, about 25% to 30% iterations are saved by introducing the test \( T(2) \) in both cases. Note that for the (1 008 504) LDPC code, the proposed test seems highly unreliable at the SNR of 0.00 dB but in that case, all words are likely to be in error anyway. Finally, it is worth mentioning that the test of [26] used in conjunction with syndrome checking provides negligible savings upon syndrome checking alone.

B. DSC Codes and Corresponding LDPC Codes

Fig. 4 depicts simulation results for the (73,45) DSC code. The first term of the union bound on the WER for MLD has also been represented in this figure for comparison purposes. For this code, order-3 reprocessing practically achieves MLD [16] (i.e., the error performance curve of order-3 reprocessing
is on top of that of MLD at all WERs considered in practical applications). We observe that order-1 reprocessing of at most two BP decoding iterations achieves near-MLD while order-1 reprocessing of at most three BP decoding iterations practically achieves MLD. While order-3 reprocessing considers at most 15,226 candidate codewords, only 92 and 138 candidate codewords are considered by order-1 reprocessing of at most two BP decoding iterations and three BP decoding iterations, respectively. For this code, $J = L = 9$, so that based on Section II-D, one iteration of BP requires 810 multiplications, 90 divisions, and 252 additions. Assuming an integer addition can include up to 128 binary additions in software, the Gaussian elimination step of OSD is realized in $28^2 \cdot 784$ additions. The sorting of the reliability values requires about 320 additions and order-1 reprocessing is achieved with less than $(1 + 45) \cdot 28 = 1,288$ additions. Note finally that although the parameters of this code are relatively small compared to that of the other codes considered in this paper, MLD of this code remains a complex task. For example, the minimum trellis representation of this code has at least $2^{23}$ states [30].

In [12], it was shown that BP decoding of the (273, 191) and (1,057, 813) DSC codes significantly outperforms BP decoding of their (3,L) and (4,L) LDPC counterparts. The same codes are considered in Figs. 5 and 6. Surprisingly enough, we observe that in both cases, a much larger coding gain is achieved for decoding of LDPC codes than for decoding DSC codes with the proposed combined scheme. At the WER $10^{-3.2}$, coding gains of 0.9 dB and 0.4 dB are achieved by the proposed scheme to decode the (4,L) LDPC codes with parameters (273, 191) and (1,057, 813), respectively. Good gains are also achieved for the (3,L) LDPC codes, but these codes are subject to an error floor at the WERs represented. As a result, the (4,L) LDPC codes become as efficient as their DSC counterparts.

This behavior can be explained by the fact that despite the large number of short loops in the Tanner graph of the DSC codes, the balanced structure of this graph and the relatively large number of check sums considered are quite favorable for BP decoding of these codes. On the other hand, due to the relatively high rate and short length of the randomly generated LDPC codes considered, their Tanner graph is likely to contain short loops in an unbalanced way. As a result, residual errors are left by BP decoding which is unlikely to return a codeword. On the other hand, the density of short loops in the graph representation of LDPC codes is much smaller than that of DSC codes. Consequently, the correlation values introduced by the iterative decoding have less influence for LDPC codes and the iterative decoding may diverge slower from its ideal trajectory (obtained when assuming no loops in the graph representation of the code, or equivalently, by ideally canceling the influences of correlations in the iterative decoding). As a result, a reduced search around the $N$-tuple delivered by BP decoding at each iteration is more likely to succeed for LDPC codes.
C. \textit{(155,64,20) LDPC Tanner Code}

In [31], a regular \textit{(155,64)} LDPC code with $J = 3$ and $L = 5$ is constructed in a very special way. This code has both very good minimum Hamming distance $d_H = 20$ and very good girth $g = 8$ (i.e., no loops of length four or six are present in its graph representation). Fig. 7 compares the simulation results for BP decoding of this code with at most 500 iterations, and combined BP decoding with order-1 reprocessing with at most 50 iterations. Order-4 reprocessing, which is the largest reprocessing order for which OSD remains feasible for this code, BP decoding with at most 100 iterations and combined BP decoding with order-1 reprocessing with at most 10 iterations are also depicted in Fig. 7. We observe that at the WER, the combined proposed scheme outperforms BP decoding by about 1.2 dB. Furthermore, the combined proposed scheme, which considers at most 65 candidate codewords at each iteration rapidly outperforms order-4 reprocessing which considers at most 679 121 candidate codewords. In fact, the combined scheme achieves about the same error performance as order-4 reprocessing after only ten iterations. These results further emphasize that the smaller the code length $N$, the larger the potential gain to be achieved by the combined decoding approach over standard BP decoding.

IV. \textsc{Tradeoffs Between Performance and Complexity}

A. \textit{Simulation Results}

Based on Section II-D, we readily conclude that order-1 reprocessing requires in general more computations than one iteration of BP. This suggests to investigate a hybrid scheme in which order-1 reprocessing is performed periodically after $I$ iterations of BP decoding, so that $I = 1$ corresponds to the combined scheme presented in Section II-A and $I$ set to the maximum number of iterations considered corresponds to a serial implementation of BP decoding followed by order-1 reprocessing. Fig. 8 depicts the simulation results of iterative BP decoding with a maximum of 200 iterations, and combined iterative BP with periodic order-1 decoding every $I$ iterations with a maximum of 50 iterations and values $I = 1, 5, 10$ and 50, for a \textit{(3,6) (504,252)} LDPC code. We observe that the value $I = 10$ provides a reasonable trade-off between error performance and computation complexity at medium to high SNR values. On the other hand, little improvement is achieved by reducing $I$ from 10 to 5 and most importantly, a serial decoding by BP followed by order-1 reprocessing after 50 iterations provides negligible improvement over the standard BP decoding. Consequently, postreliability based decoding is unlikely to provide significant improvement whenever the standard BP decoding does not deliver the MLD codeword.

The error performance of serial decoding with BP followed by order-1 reprocessing depicted in Fig. 8 may suggest that little improvement is achieved during the last iterations of the proposed combined scheme. Consequently, another tradeoff between error performance and decoding complexity can be achieved by processing the combined scheme composed of BP decoding and order-1 reprocessing during the first $I$ iterations only, and then terminating the decoding by the standard BP decoding. Fig. 9 depicts the simulation results of iterative BP decoding with a maximum of 200 iterations, and combined iterative BP with order-1 decoding during the first $I$ iterations only with a maximum of 50 iterations and values $F = 5, 10, 15$ and 50, for a \textit{(3,6) (504,252)} LDPC code. Again, a good trade-off between error performance and computation complexity is achieved at all SNR values but the error performance of the combined scheme described in Section II cannot be approached closely with a small value of $F$.

B. \textit{Discussion}

The simulation results of Sections III and Section IV-A suggest that the convergence of the standard BP decoding can be decomposed into two phases. During the first phase, the BP iterative decoding is expected to approach the MLD codeword at some iteration while during the second phase, the BP iterative decoding is unlikely to approach the MLD codeword. Consequently, the combined decoding scheme is efficient only during the first phase. The turning point between these two phases cor-
responds to starting having non negligible influences introduced in the iterative process by correlation values due to the loops in the graph representation of the code considered. For BP decoding of codes of medium lengths, the existence of the second phase is further justified by the negligible improvement achieved by a serial decoding as depicted in Fig. 8. On the other hand, the simulation results of Fig. 8 also suggest that during the first phase, the MLD codeword may be approached during few iterations only, so that a periodic reprocessing of the a posteriori probabilities delivered by BP decoding results in a non negligible performance degradation with respect to the combined scheme of Section II. Furthermore, the results of Fig. 9 suggest that the turning point is more likely to occur during the first iterations of BP decoding, as expected from the small girth of the graphs associated with the codes considered. Nevertheless, neglecting the turning point occurrences after few iterations only results in a nonnegligible performance degradation, as shown in Fig. 9.

Due to the large density of loop of length 6 in the graph representation of DSC codes, the turning point associated with BP decoding of these codes is expected to occur quite fast typically. As a result, the combined approach is likely to provide little improvement over BP decoding for these codes, as observed in Section III-B. In many cases, the parity check matrix of a cyclic code can be obtained by cyclically shifting \(N\) times a minimum weight codeword of the dual code. However, the graph representation associated with this construction is rich in length-4 loops. As a result, the proposed combined decoding method is expected to perform quite poorly for such codes. We verified this fact by simulating BP decoding combined with order-1 reprocessing for the (127,64) BCH code. Fig. 10 compares the results obtained after 50 iterations with order-\(i\) reprocessing for \(i = 1, 2,\) and 4. While order-4 reprocessing practically achieves MLD at all SNR values represented in this figure [16], the combined iterative decoding method is outperformed by order-2 reprocessing (which can be viewed as 32 order-1 reprocessings of the same MRB) and falls far from MLD, as expected. On the other hand, if the density of length-4 loops in the graph representation of the code is small, the proposed combined decoding method can still work well. This is verified in Fig. 11, in which a (3,6) (504 252) LDPC code with girth \(g = 6\) is compared with a (3,6) (504 252) LDPC code whose graph representation contains 25 loops of length 4. Although this latter code is subject to a higher error floor than the former code, the combined approach achieves about the same coding gains for both codes with respect to BP decoding. A (4,8) (504 253) LDPC code with girth \(g = 6\) has also been represented in Fig. 11. As expected, this code should have a better asymptotic error performance than many (3,6) LDPC codes due to a larger expected minimum Hamming distance. However, we also observe that the improvement achieved by the combined algorithm over standard BP decoding is not as important as for the (3,6) LDPC codes. This can be justified by the larger density of short loops in the graph representation of (4,8) LDPC codes.
V. CONCLUSION

In this paper, we have presented a near-MLD decoding method which combines BP decoding with the OSD algorithm. This approach allows to bridge the error performance gap between BP decoding and MLD. In general, for a fixed code rate and a fixed code structure, the performance improvements increase as the code length decreases, as expected from the analytical results of [11, 21, 26, 27, 28]. In general, the proposed combined scheme converges faster than BP decoding to a better limit. Indeed, the computational cost of one iteration of the combined algorithm is higher than that of BP decoding. This approach also motivates the search for low complexity probabilistic soft decision decoding algorithms of relatively long block codes.

It can be anticipated that for some fixed code parameters, a smaller error performance gap between BP decoding and MLD is achieved if particular attention is paid to the code construction and its Tanner graph representation. To this end, the proposed approach provides a way to measure the effectiveness of the construction, assuming it achieves near-MLD. Furthermore, although for the combined proposed approach, we limited the reprocessing order to one and the maximum number of iterations to 50 for complexity issues, further secondary gains could be achieved with higher reprocessing phases and a larger maximum number of iterations if a particular target code is being investigated.

This combined approach can also be extended in several ways: 1) the BP decoding can be replaced by suboptimum iterative decoding methods as those discussed in [14]–[15], [29], and [32]–[33]; and 2) the OSD algorithm can be replaced by other reliability based decoding methods such as Chase or GMD decodings [34]–[36], especially for DSC codes for which efficient algebraic decoding methods have been devised [20]. Further refinements in the tradeoffs between error performance and decoding complexity are possible with such methods.

Finally, it is worth mentioning that the OSD algorithm can be combined in a similar way as that described in this paper to further improve the error performance of iterative decoding of “turbo-like” codes [37]–[39].

ACKNOWLEDGMENT

The author wishes to thank M. Tanner for numerous valuable comments, the preprints of [11], [21], [31], suggesting the use of the combined algorithm for his (155,64,20) LDPC code [31], and anticipating the large gap in error performance compared with BP decoding, as well as S. Lin for many inspiring discussions and D. MacKay for providing the parity check matrices of most LDPC codes simulated in this paper. He is also very grateful to the three reviewers and the editor, E. Eleftheriou, whose constructive comments have greatly improved this paper.

REFERENCES


Marc P. C. Fossorier (S’90–M’95–SM’00) was born in Annemasse, France, on March 8, 1964. He received the B.E. degree from the National Institute of Applied Sciences (I.N.S.A.), Lyon, France, in 1987, and the M.S. and Ph.D. degrees from the University of Hawaii, Manoa, Honolulu, in 1991 and 1994, all in electrical engineering.

In 1996, he joined the Faculty of the University of Hawaii, Honolulu, as an Assistant Professor of Electrical Engineering. He was promoted to Associate Professor in 1999. His research interests include decoding techniques for linear codes, communication algorithms, combining coding and equalization for ISI channels, magnetic recording and statistics. He coauthored (with S. Lin, T. Kasami, and T. Fujiwara) the book, *Trellises and Trellis-Based Decoding Algorithms*, (Kluwer, 1998).

Dr. Fossorier is a recipient of a 1998 NSF Career Development award. He has served as Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS since 1996, as Editor for the IEEE COMMUNICATIONS LETTERS since 1999, and he is currently the Treasurer of the IEEE Information Theory Society.