Transactions Papers

Reduced Complexity Iterative Decoding of Low-Density Parity Check Codes Based on Belief Propagation

Marc P. C. Fossorier, *Member, IEEE,* Miodrag Mihaljević, and Hideki Imai, *Fellow, IEEE*

Abstract— **In this paper, two simplified versions of the belief propagation algorithm for fast iterative decoding of low-density parity check codes on the additive white Gaussian noise channel are proposed. Both versions are implemented with real additions only, which greatly simplifies the decoding complexity of belief propagation in which products of probabilities have to be computed. Also, these two algorithms do not require any knowledge about the channel characteristics. Both algorithms yield a good performance–complexity tradeoff and can be efficiently implemented in software as well as in hardware, with possibly quantized received values.**

Index Terms— **APP decoding, belief propagation, block codes, four-density parity check codes, iterative decoding.**

I. Introduction

LOW-DENSITY parity-check (LDPC) codes, proposed by Gallager [1], [2], appear as a class of codes which can yield very good performance on the binary symmetric channel (BSC) as well as on the additive white Gaussian noise (AWGN) channel. Recently, it was shown that the belief propagation (BP) algorithm [3] provides a powerful tool for iterative decoding of LDPC codes, by noting that the original Gallager's iterative probabilistic decoding of LDPC codes is a particular BP-based decoding approach [5]–[8]. As in [4], this probabilistic decoding algorithm is based on evaluating the likelihood ratios associated with each information bit from information provided by disjoint parity check equations. Iterative decoding techniques in general have received significant attention recently and various results have been reported (see [5]–[18], for example).

The aim of this paper is to develop an iterative decoding algorithm for LDPC codes as an approximation of the standard BP decoding procedure, so that the modified algorithm performs close to the standard BP but with a significant reduction of complexity. Accordingly, a goal of this paper is to obtain a good performance–complexity tradeoff. First, a simplified version of the BP algorithm is considered. This modified algorithm, which corresponds to the approach taken in [12]–[14], is then further simplified so that it performs real value additions only. This second simplification, which is equivalent to the approximation presented in [19] for majority logic decoding based on the APP algorithm of [4], simply consists of expressing all the steps of the algorithm with respect to logarithms of probabilities rather than probabilities. The same standard approximation was used to derive the Max-Log-MAP algorithm from the MAP algorithm [10], [11]. This simple algorithm allows the processing of low-complexity iterative decoding of LDPC codes, but at the expense of about 1-dB degradation in error performance with respect to the BP algorithm at the bit error rate (BER) 10^{-5}. A second simplified algorithm is then considered. For this algorithm, the standard approximation is directly applied to the BP algorithm. Although not as straightforward as in the previous case, due to the fact that for the BP algorithm the probability values considered at iteration-i do not necessarily correspond to the hard decisions made at iteration-$(i-1)$, the application of the standard approximation achieves a better tradeoff between error performance and decoding complexity, with real value additions only and a performance degradation of few tenths of a decibel at the BER 10^{-5} for the LDPC codes simulated.

The paper is organized as follows. The characteristics of LDPC codes and their decoding based on BP are briefly reviewed in Section II. Then the two reduced-complexity BP-based decoding algorithms are described in Section III. Finally, these algorithms are compared with BP in Section IV and concluding remarks are given in Section V.

Paper approved by J. Huber, the Editor for Coding and Coded Modulation of the IEEE Communications Society. Manuscript received May 25, 1998; revised September 9, 1998 and November 23, 1998. This work was supported by the National Science Foundation under Grant NCR-94-15374 and Grant CCR-97-32959 and by the Japan Society for Promotion of Science (JSPS) under Contract JSPS-RFTF-96P00604.

M. P. C. Fossorier is with the Department of Electrical Engineering, University of Hawaii, Honolulu, HI 96822 USA (e-mail: marc@aravis.eng.hawaii.edu).

M. Mihaljević is with the Mathematical Institute, Academy of Science and Arts, Belgrade, Yugoslavia.

H. Imai is with the Institute of Industrial Science, University of Tokyo, Tokyo 106, Japan.

Publisher Item Identifier S 0090-6778(99)03899-4.

0090–6778/99$10.00 © 1999 IEEE

II. Background

A. Low-Density Parity Check Codes

LDPC codes are specified by a parity-check matrix containing mostly zeros and only a small number of ones. A binary (N, J, K) LDPC code has block length N and a parity-check matrix with exactly J ones in each column and K ones in each row, assuming $J \geq 3$ and $K > J$. In the following, we refer to the elements of an LDPC codeword $\mathbf{x} = [x_n]$ as bits, and the rows of the parity-check matrix $\mathbf{H} = [H_{mn}]$, as checks. Accordingly, in a binary LDPC code, every code bit is checked by precisely J parity checks, and every parity check involves precisely K code bits. The typical minimum distance of these codes increases linearly with N for a fixed rate and fixed J. For $J > 3$ and a sufficiently low rate, a simple decoding procedure exists such that the error rate decreases at least exponentially with a root of the block length, assuming a BSC [1], [2].

B. Standard Belief Propagation-Based Decoding

This section summarizes, according to [6] and [7], the iterative decoding of LDPC codes based on the BP algorithm. The decoding problem consists of finding the most likely vector $\mathbf{x}$ (represented as a column matrix) such that $\mathbf{H}\mathbf{x}[\bmod 2] = \mathbf{0}$. The likelihood of $\mathbf{x}$ is given by $\prod_n f_n^x$, with $f_n^x = P(x_n = x)$, so that $f_n^1 = 1 - f_n^0$. We denote the set of bits n that participate in check m by $N(m) = \{n\colon H_{mn} = 1\}$. Similarly, we define the set of checks in which bit n participates as $M(n) = \{m\colon H_{mn} = 1\}$. We denote a set $N(m)$ with bit n excluded by $N(m)\backslash n$, and a set $M(n)$ with parity check m excluded by $M(n)\backslash m$. The cardinalities of the sets $N(m)$ and $M(n)$ are denoted by $|N(m)|$ and $|M(n)|$, respectively.

The iterative decoding algorithm has two alternating parts, in which certain quantities q_{mn} and r_{mn}, associated with each nonzero element in the matrix $\mathbf{H}$, are iteratively updated. The quantity q_{mn}^x is meant to be the probability that bit n of $\mathbf{x}$ is x, given the information obtained via checks other than check m. The quantity r_{mn}^x is meant to be the probability of check m is satisfied if bit n of $\mathbf{x}$ is considered fixed at x and the other bits have a separable distribution given by the probabilities $\{q_{mn'}\colon n' \in N(m)\backslash n\}$. The algorithm would produce the exact posterior probabilities of all the bits if the bipartite graph[1] defined by the matrix $\mathbf{H}$ contained no cycles [3].

The standard iterative decoding algorithm based on the BP approach consists of the following main steps.

- *Initialization*: The variables q_{mn}^0 and q_{mn}^1 are initialized to the values f_n^0 and f_n^1, respectively.
- *Iterative Processing*
  1) *Step 1*: Define $\delta q_{mn} = q_{mn}^0 - q_{mn}^1$ and for each m, n, and for $x = 0, 1$, compute
  $$\delta r_{mn} = \prod_{n' \in N(m)\backslash n} \delta q_{mn'} \tag{1}$$
  $$r_{mn}^x = (1/2)(1 + (-1)^x \delta r_{mn}). \tag{2}$$
  2) *Step 2*: For each n and m, and for $x = 0, 1$, update
  $$q_{mn}^x = \alpha_{mn} f_n^x \prod_{m' \in M(n)\backslash m} r_{m'n}^x \tag{3}$$
  where $\alpha_{m,n}$ is chosen such that $q_{mn}^0 + q_{mn}^1 = 1$.
  For each n and $x = 0, 1$, update the "pseudoposterior probabilities" q_n^0 and q_n^1 given by
  $$q_n^x = \alpha_n f_n^x \prod_{m \in M(n)} r_{mn}^x \tag{4}$$
  where α_n is chosen such that $q_n^0 + q_n^1 = 1$.
  3) *Step 3*:
     a) create $\hat{\mathbf{x}} = [\hat{x}_n]$ such that $\hat{x}_n = 1$ if $q_n^1 > 0.5$, and $\hat{x}_n = 0$ if $q_n^1 \leq 0.5$,
     b) do the following:
        —If $\mathbf{H}\hat{\mathbf{x}} = \mathbf{0}$ then the decoding algorithm halts, and $\hat{\mathbf{x}}$ is considered as a valid decoding result.
        —Otherwise, the algorithm repeats from Step 1.
        —A failure is declared if some maximum number of iteration stages (e.g., 100) occurs without a valid decoding.

III. Two Reduced-Complexity Decoding Algorithms

In the following, we assume that the binary (N, J, K) LDPC code C considered is used for error control over the AWGN channel, with BPSK signaling. Let $s(C)$ represent the image of C under the usual componentwise mapping from {0, 1} to $\{\pm 1\}$. If $\mathbf{x} = [x_n]$ is a codeword in C and $s(\mathbf{x}) = \mathbf{s} = [s_n]$ is the corresponding transmitted sequence, then the received sequence is $\mathbf{s} + \mathbf{w} = \mathbf{y} = [y_n]$, with $y_n = s_n + w_n$, where for $1 \leq n \leq N$, w_n's are statistically independent Gaussian random variables with zero mean and variance $N_0/2$.

A. APP-Based Decoding Algorithm

Define q_n as the *a priori* probability that bit n is in error. Then the probability r_{mn} that for check sum $m \in M(n)$, the sum of all bits $n' \in N(m)\backslash n$ mismatches the transmitted bit n is given by [4, p. 53]

$$r_{mn} = (1/2)\left(1 - \prod_{n' \in N(m)\backslash n} (1 - 2q_{n'})\right). \tag{5}$$

In other words, r_{mn} represents the probability of having an odd number of errors in the hard decisions of the bits of $N(m)$. For $m \in M(n)$, define σ_m as the result of check sum-m evaluated from the hard decisions corresponding to q_n, and $\overline{\sigma}_m$ as its modulo-2 complement. Note that σ_m is computed based on the whole symbol set $N(m)$. Furthermore, define $\tilde{q}_n$ as the *a posteriori* probability that bit n is in error based on the results of the check sums intersecting in position-n. It follows that

$$\frac{1 - \tilde{q}_n}{\tilde{q}_n} = \left(\frac{1 - q_n}{q_n}\right) \prod_{m \in M(n)} \left(\frac{1 - r_{mn}}{r_{mn}}\right)^{\overline{\sigma}_m} \left(\frac{r_{mn}}{1 - r_{mn}}\right)^{\sigma_m} \tag{6}$$

[1] A bipartite graph G is defined as a graph whose vertices can be partitioned into two subsets V_1 and V_2 such that every edge of G joins V_1 to V_2.

or equivalently

$$\tilde{q}_n = \left(1 + \left(\frac{1-q_n}{q_n}\right) \prod_{m\in M(n)} \left(\frac{1-r_{mn}}{r_{mn}}\right)^{\overline{\sigma}_m} \times \left(\frac{r_{mn}}{1-r_{mn}}\right)^{\sigma_m}\right)^{-1}. \tag{7}$$

As proposed in [12]–[14], all check sums can be re-evaluated based on the hard decisions corresponding to the values $\tilde{q}_n$, which are used as new *a priori* probabilities q_n. Consequently, we obtain an iterative decoding algorithm, which as mentioned in [7], can be viewed as a simplified version of BP since instead of (3), only (4) needs to be updated.

For the AWGN channel model considered, the probability q_n can be expressed as

$$q_n = \frac{\mathrm{e}^{-|L_n|}}{1+\mathrm{e}^{-|L_n|}} \tag{8}$$

where $L_n = 4y_n/N_0$ represents the log-likelihood ratio associated with the hard decision value based on y_n. By approximating $\prod_{n'\in N(m)}(1-2q_{n'}) \approx 1 - 2\max_{n'\in N(m)}\{q_{n'}\}$, (5) becomes

$$r_{mn} \approx \frac{\mathrm{e}^{-4|y_{mn}|_{\min}/N_0}}{1+\mathrm{e}^{-4|y_{mn}|_{\min}/N_0}} \tag{9}$$

where $|y_{mn}|_{\min} = \min_{n'\in N(m)\backslash n}\{|y_{n'}|\}$. The effects of the relative errors introduced by approximation (9) on the overall error performance become less and less significant as the SNR (i.e., $1/N_0$) increases. By substituting (9) into (6), we obtain

$$\ln\left(\frac{1-\tilde{q}_n}{\tilde{q}_n}\right) = \frac{4}{N_0}\left(|y_n| + \sum_{m\in M(n)} (\overline{\sigma}_m - \sigma_m)|y_{mn}|_{\min}\right). \tag{10}$$

After normalizing (10) by the factor $N_0/4$, we derive the following algorithm for computing $z_n = (N_0/4)\ \ln((1-\tilde{q}_n)/\tilde{q}_n)$:

- *Initialization*: The hard decisions $\hat{x}_n$ are initialized to the hard decisions of the received symbols y_n, and denote the *a priori* log-likelihood ratio of error by $|r_n| = |y_n|$.
- *Iterative Processing*
  1) *Step 1*: For each n and each $m \in M(n)$, evaluate

  $$\sigma_m = \sum_{n'\in N(m)} \hat{x}_{n'}\ [\mathrm{mod}\, 2] \tag{11}$$

  and identify

  $$|y_{mn}|_{\min} = \min_{n'\in N(m)\backslash n}\{|y_{n'}|\}. \tag{12}$$

  2) *Step 2*: For each n, compute

  $$z_n = |r_n| + \sum_{m\in M(n)} (\overline{\sigma}_m - \sigma_m)|y_{mn}|_{\min}. \tag{13}$$

  3) *Step 3*:
     a) If $z_n < 0$, then $\hat{x}_n = \hat{x}_n \oplus 1$.
     b) Set $|y_n| = |z_n|$ and repeat Step 1 after the same termination procedure as in Section II-B.

Since this algorithm does not depend on N_0 and therefore does not require any *a priori* information about the AWGN channel, it is referred to as the *uniformly most powerful (UMP) APP-based iterative decoding algorithm*. This algorithm can be viewed as an iterative implementation of the algorithm presented in [19], which uses the same standard approximation. It also represents a simplified version of the iterative decoding algorithm presented in [14] for the AWGN channel.

B. BP-Based Decoding Algorithm

In this section, we investigate how to simplify the BP algorithm based on the same approximation as in Section III-A. To this end, for $x = 0, 1$ we first rewrite r_{mn}^x based on (1) and (2) as

$$r_{mn}^x = (1/2)\left(1 + (-1)^x \prod_{n'\in N(m)\backslash n} (1 - 2q_{mn'}^1)\right) \tag{14}$$

which is of the same form as (5). Consequently, by considering (3) and (14) in conjunction with (9), the same simplification as in Section III-A seems possible. However, particular care has to be brought to the following problem. In Section III-A, all check sums described by (11) are evaluated at Step 1 of each iteration stage based on the same hard decision values $\hat{\mathbf{x}}$. On the other hand, after the first iteration stage, for a given n, the values q_{mn}^x used by the BP algorithm to compute (1) may define hard decision values $\hat{x}_{mn}$ different from the value $\hat{x}_n$ corresponding to q_n^x for some $m \in M(n)$. In other words, it is possible to obtain $q_{mn}^x \leq 0.5$ and $q_n^x > 0.5$, or inversely, at the same iteration stage. As a result, the same check sum m may take different values depending on whether for $n \in N(m)$, the hard decision values $\hat{x}_{mn}$ corresponding to the values q_{mn}^x or the hard decision values $\hat{x}_n$ corresponding to the values q_n^x are considered. At Step 3 of the UMP APP-based decoding algorithm, the value $\hat{x}_n$ at the ith stage of the iterative process (referred to as iteration-i in the sequel of the paper) is updated based on the value $\hat{x}_n$ obtained at iteration-$(i-1)$. Consequently, a different updating rule has to be chosen in simplifying the BP decoding algorithm as for each n, the values $\hat{x}_{mn}$, $m \in M(n)$, and $\hat{x}_n$ obtained at iteration-$(i-1)$ are not necessarily equal in general. Since iteration-1 is the only iteration stage for which all check sums are evaluated based on the same hard decisions, the hard decision values $\hat{x}_n$ and $\hat{x}_{mn}$ have to be updated at iteration-i based on the initial hard decisions $\hat{x}_n = \hat{x}_{mn}$ corresponding to $q_{mn}^x = f_n^x$ for all $m \in M(n)$. Hence, iteration-1 becomes the reference for all subsequent iteration stages. Based on this important remark, we derive the following algorithm:

- *Initialization*: For all n and each $m \in M(n)$, the hard decisions $\hat{x}_{mn}$ and $\hat{x}_n$ are initialized to the hard decisions of the received symbols y_n and recorded by $\hat{x}_n^r = \hat{x}_n$; also $|r_n| = |y_n|$ and for each $m \in M(n)$, $|y_{mn}| = |y_n|$.

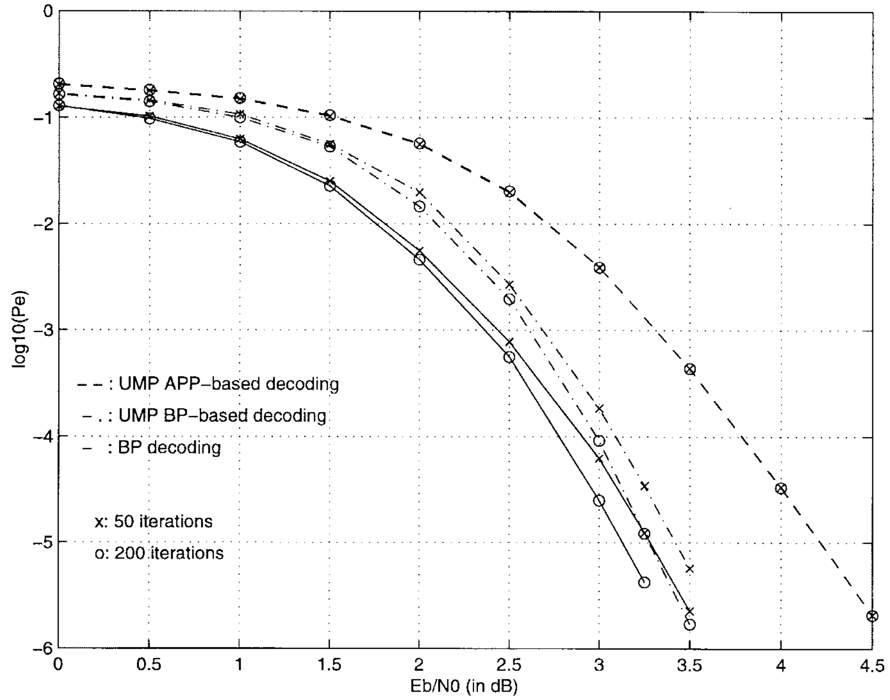

Fig. 1. Error performance for iterative decoding of the (504, 252) LDPC codewith BP, UMP BP-based, and UMP APP-based decoding algorithms, and at most 50 and 200 iterations.

- *Iterative Processing*

1) *Step 1*: For each n and each $m \in M(n)$, evaluate the check sums

$$\sigma_{mn} = \hat{x}_n^r \oplus \left(\sum_{n' \in N(m)\backslash n} \hat{x}_{mn'} \, [\text{mod } 2] \right) \tag{15}$$

$\overline{\sigma}_{mn} = \sigma_{mn} \oplus 1$, and identify

$$|y_{mn}|_{\min} = \min_{n' \in N(m)\backslash n} \{|y_{mn'}|\}. \tag{16}$$

2) *Step 2*: For each n and each $m \in M(n)$, compute

$$z_{mn} = |r_n| + \sum_{m' \in M(n)\backslash m} (\overline{\sigma}_{m'n} - \sigma_{m'n})|y_{m'n}|_{\min}. \tag{17}$$

For each n, compute

$$z_n = |r_n| + \sum_{m \in M(n)} (\overline{\sigma}_{mn} - \sigma_{mn})|y_{mn}|_{\min}. \tag{18}$$

3) *Step 3*:

a) Create $\hat{\mathbf{x}} = [\hat{x}_n]$ such that $\hat{x}_n = \hat{x}_n^r$ if $z_n > 0$, and $\hat{x}_n = \hat{x}_n^r \oplus 1$ if $z_n < 0$.

b) For each $m \in M(n)$, create $\hat{\mathbf{x}}_m = [\hat{x}_{mn}]$ such that $\hat{x}_{mn} = \hat{x}_n^r$ if $z_{mn} > 0$, and $\hat{x}_{mn} = \hat{x}_n^r \oplus 1$ if $z_{mn} < 0$.

c) For each $m \in M(n)$, set $|y_{mn}| = |z_{mn}|$ and repeat Step 1 after the same termination procedure as in Section II-B.

Note that in (15), the initial decision $\hat{x}_n^r$ is always used to evaluate σ_{mn}. Although surprising, this is easily justified by the fact that once all $\hat{x}_{mn'}$ with $n' \in N(m)\backslash n$, evaluated at the previous iteration stage are correct, $\sigma_{mn} = 1$ if $\hat{x}_n^r$ is initially in error. In that case, (17) and (18) can be rewritten as

$$z_{mn} = |r_n| - \sum_{m' \in M(n)\backslash m} |y_{m'n}|_{\min} \tag{19}$$

$$z_n = |r_n| - \sum_{m \in M(n)} |y_{mn}|_{\min} \tag{20}$$

respectively. It follows that the hard decisions $\hat{x}_{mn}$ and $\hat{x}_n$ evaluated at Step 3 from (19) and (20) are correct, unless the reliability associated with the initial decision about bit n is still larger than the sum of the reliabilities associated with each check sum intersecting on bit n.

As in Section III-A, since this algorithm does not depend on N_0, it is referred to as the *UMP BP-based iterative decoding algorithm*. The UMP property is advantageous since in [20], it is shown that a poor estimate of the noise characteristics may result in some performance degradation for the BP algorithm.

IV. Comparisons with the BP Algorithm

In this section, we compare the two algorithms presented in Section III with the BP algorithm with respect to the tradeoff between error performance and decoding complexity.

A. Error Performance

In the following, simulation results for the (504, 252) and (1008, 504) LDPC codes taken from [7] are presented. Figs. 1 and 2 depict the bit error performance for iterative decoding

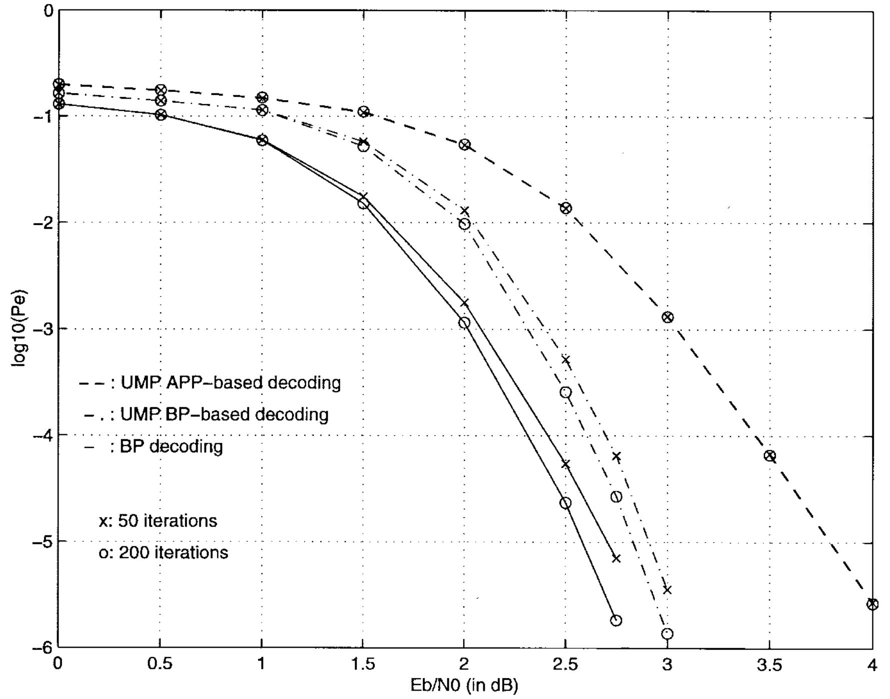

Fig. 2. Error performance for iterative decoding of the (1008, 504) LDPC codewith BP, UMP BP-based, and UMP APP-based decoding algorithms, and at most 50 and 200 iterations.

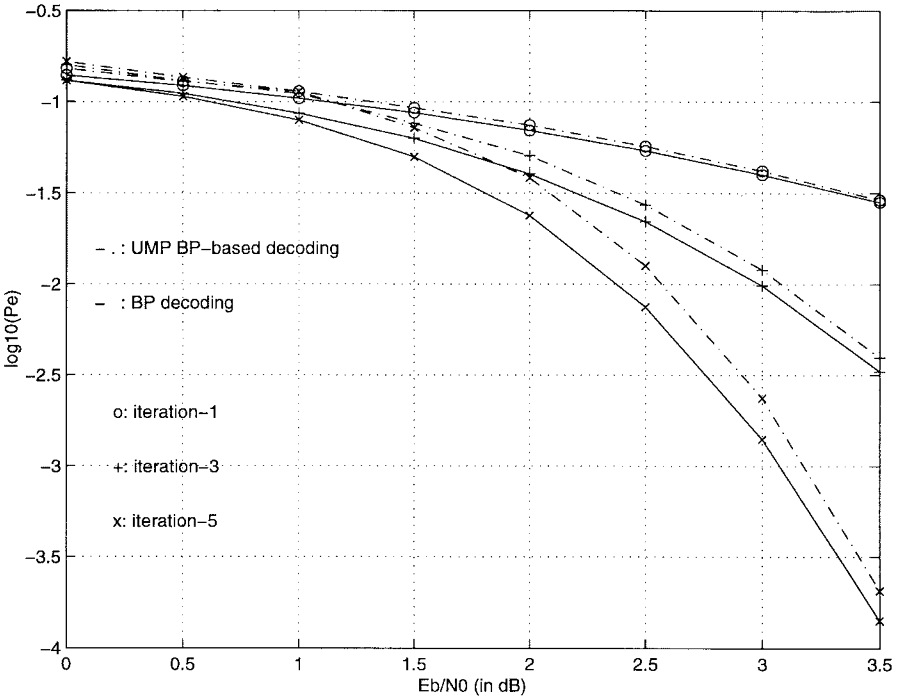

Fig. 3. Error performance after the first, third, and fifth iterations for iterative decoding of the (1008, 504) LDPC code with BP and UMP BP-based decoding algorithms.

of the (504, 252) and (1008, 504) LDPC codes, respectively, with the BP, UMP BP-based, and UMP APP-based decoding algorithms, and at most 50 and 200 iterations. The results are obtained by Monte Carlo simulations, with at least 1000 bit errors for each recorded point. For both codes, we observe that at the BER 10^{-5}, the UMP APP-based algorithm performs at least 1 dB worse than the BP algorithm. On the other hand, for both 50 and 200 iterations, only about 0.15 and 0.25 dB separate the error performance curves of the BP and UMP BP-based algorithms for the (504, 252) and (1008, 504) LPDC codes, respectively, at the BER 10^{-5}. Finally, we observe that while for both the BP and UMP BP-based algorithms, a nonnegligible error performance improvement is achieved at medium to high SNR values by increasing the maximum number of iterations from 50 to 200, approximatively no improvement is made for the UMP APP-based algorithm.

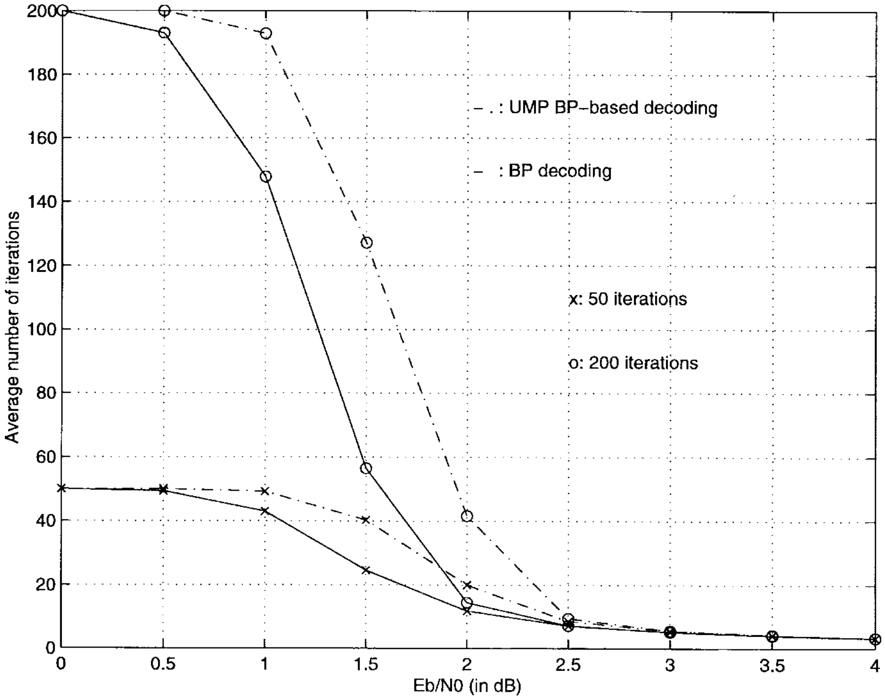

Fig. 4. Average number of iterations for iterative decoding of the (1008, 504) LDPC code with BP and UMP BP-based decoding algorithms, and at most 50 and 200 iterations.

Fig. 3 depicts the error performance after the first, third, and fifth iterations for iterative decoding of the (1008, 504) LDPC code with BP and UMP BP-based decoding algorithms. A similar figure was obtained for the (504, 252) LDPC code. Based on this figure, we observe that at very low SNR values, the error performance of the BP algorithm improves as the number of iterations increases, while it becomes worse for the UMP BP-based algorithm. Due to these different behaviors, the gap in error performance increases with the number of iterations considered in this figure at SNR values corresponding to BER values used in practice. Finally, for the same number of iterations, both curves seem to converge at high SNR values. Based on these observations, a slower convergence (or equivalently a higher number of iterations) is expected for the UMP BP-based algorithm with respect to the BP algorithm. On the other hand, the decoding complexity associated with each iteration of the UMP BP-based algorithm is significantly smaller than that of the BP algorithm, as shown next.

TABLE I
DECODING COMPLEXITY FOR ONE ITERATION OF THE BP DECODING ALGORITHM AND A RATE—1/2 $(N, J, 2J)$ LDPC CODE

Operation	Number of Computations
δq_{mn}	NJ additions
δr_{mn} (forward and backward recursions [7])	$3N(J-1)$ multiplications
r_{mn}^x	NJ additions
$f_n^x \prod_{m'} r_{m'n}^x$ (forward and backward recursions [7])	$2N(3J-4)$ multiplications
α_{mn}	NJ additions and NJ divisions
q_{mn}^x	$2NJ$ multiplications
α_n	N additions and N divisions
q_n^x	$2N$ multiplications

TABLE II
DECODING COMPLEXITY FOR ONE ITERATION OF THE UMP APP-BASED DECODING ALGORITHM AND A RATE—1/2 $(N, J, 2J)$ LDPC CODE

Operation	Number of Computations
$\min_{n'}\{\lvert y_{mn'}\rvert\}$ [21]	$N/2\,(2J + \lceil \log_2 2J \rceil - 2)$ additions
z_n	NJ additions

TABLE III
DECODING COMPLEXITY FOR ONE ITERATION OF THE UMP BP-BASED DECODING ALGORITHM AND A RATE—1/2 $(N, J, 2J)$ LDPC CODE

Operation	Number of Computations
$\min_{n'}\{\lvert y_{mn'}\rvert\}$ [21]	$N/2\,(2J + \lceil \log_2 2J \rceil - 2)$ additions
z_{mn} and z_n (forward and backward recursions)	$3N(J-1)$ additions

B. Decoding Complexity

The decoding complexities associated with the BP, the UMP APP-based, and the UMP BP-based decoding algorithms are summarized in Tables I–III, respectively. In these tables, all operations associated with modulo-2 arithmetic have been neglected as conventionally done. The decoding complexity associated with BP is evaluated based on the clever forward and backward recursions proposed in [7] to compute both δr_{mn} and q_{mn}^x for $x = 0, 1$. Based on Table I, for a rate-1/2 $(N, J, 2J)$ LDPC code, the total complexity associated with one iteration of BP consists of $11NJ - 9N$ *real multiplications,* $N(J+1)$ *real divisions, and* $N(3J+1)$ *real additions*. This is larger than the $6NJ$ complexity proposed in [7], which can be achieved by computing only either q_{mn}^0 (instead of both q_{mn}^0 and q_{mn}^1), or equivalently, the probabilities of each bit being in error (as described in Section III-A), so that no normalization by α_{mn} is needed. However, it is not clear at that time whether this modified algorithm will perform as well as the algorithm considered in Section II-B and [7], especially when the probabilities are limited to prevent overflow problems.

The minimum values $\min_{n' \in N(m)\backslash n}\{\lvert y_{mn}\rvert\}$ associated with each check sum-m of both the UMP APP-based and the UMP BP-based decoding algorithms can be determined by identifying the two minimum values corresponding to this check sum. This method is more efficient than ordering the entire set of reliability values for small values of K and can be achieved in a straightforward manner with at most $4J-3$ comparisons, or with at most $2J+\lceil\log_2 2J\rceil-2$ comparisons with the help of a binary tree, as described in [21]. Note finally that if quantized values are considered, then the cost for ordering becomes negligible [22]. Then the decoding complexity associated with the UMP APP-based directly follows from the description of the algorithm given in Section III-A. Consequently, this algorithm requires at most $2NJ+N/2\lceil\log_2 2J\rceil-N$ *real additions,* as shown in Table II. For the UMP BP-based decoding algorithm, the values z_{mn} and z_n are evaluated simultaneously based on the forward and backward recursions of [7]. Consequently, as summarized in Table III, the UMP BP-based decoding algorithm is performed with at most $4N(J-1)+N/2$ $\lceil\log_2 2J\rceil$ *real additions.* Based on these tables, it follows that both algorithms described in Section III require real additions only and, therefore, achieve significant computations savings with respect to the BP algorithm. Also, the UMP BP-based decoding algorithm requires only about twice as many real additions as the UMP APP-based algorithm as well as about twice the decoding delay of the UMP APP-based algorithm due to the forward and backward recursions associated with each check sum.

Fig. 4 depicts the average number of iterations for iterative decoding of the (1008, 504) LDPC code with BP and UMP BP-based decoding algorithms, and at most 50 and 200 iterations. Again, a similar figure was obtained for the (504, 252) LDPC code. Based on this figure, we conclude that the UMP BP-based algorithm requires significantly more iterations (of smaller decoding cost) than the BP algorithm, especially at medium SNR values. This conclusion confirms the observations made in Section IV-A.

V. Conclusion

In this paper, two simple iterative algorithms for decoding LDPC codes have been proposed. Both algorithms require real additions only, and therefore achieve a good tradeoff between error performance and decoding complexity as well as fit hardware implementation with quantized received values. In particular, for the LDPC codes considered, the UMP BP-based decoding algorithm performs within a few tenths of a decibel of the BP algorithm at the BER 10^{-5}. Based on these results, we conclude that the UMP BP-based decoding algorithm provides an attractive solution to implement iterative decoding of LDPC codes.

The UMP BP-based decoding algorithm has been derived from the BP algorithm by considering only the dominant contribution when evaluating the reliability associated with each check sum. Therefore, the performance of this algorithm can be further enhanced by adding correction values, as described in [23]–[25] for MAP-based decoding algorithms. Furthermore, since for an (N, J, K) LDPC code, each check sum consists of K bits, with K small, this approach provides an alternative way to implement the BP algorithm as at most $K-1$ correcting values have to be added. This method becomes attractive for hardware implementation of the BP algorithm since the corrective terms can be stored in a ROM [23]–[25].

Acknowledgment

The authors wish to thank D. J. C. MacKay for providing the generator matrices of the LDPC codes used in this paper by allowing access to his website, as well as the reviewers for improving the presentation of the paper.

References

[1] R. G. Gallager, “Low-density parity-check codes,” *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–28, Jan. 1968.

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

[3] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann, 1988.

[4] J. L. Massey, *Threshold Decoding*. Cambridge, MA: M.I.T. Press, 1963.

[5] F. R. Kschischang and B. J. Frey, “Iterative decoding of compound codes by probability propagation in graphical models,” *IEEE J. Select. Areas Commun.*, vol. 16, pp. 219–230, Feb. 1998.

[6] D. J. C. MacKay and R. M. Neal, “Near shannon limit performance of low density parity check codes,” *Electron. Lett.*, vol. 33, pp. 457–458, Mar. 1997.

[7] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, Mar. 1999.

[8] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, “Turbo decoding as an instance of Pearl’s ‘Belief Propagation’ algorithm,” *IEEE J. Select. Areas Commun.*, vol. 16, pp. 140–152, Feb. 1998.

[9] G. Battail, “A conceptual framework for understanding turbo codes,” *IEEE J. Select. Areas Commun.*, vol. 16, pp. 245–254, Feb. 1998.

[10] J. Hagenauer, E. Offer, and L. Papke, “Iterative decoding of block and convolutional codes,” *IEEE Trans. Inform. Theory*, vol. 42, pp. 429–445, Mar. 1997.

[11] P. Jung, “Novel low complexity decoder for turbo-codes,” *Electron. Lett.*, vol. 31, pp. 86–87, Jan. 1995.

[12] R. Lucas, M. Bossert, and M. Breitbach, “On iterative soft-decision decoding of linear binary block codes and product codes,” *IEEE J. Select. Areas Commun.*, vol. 16, pp. 276–298, Feb. 1998.

[13] W. Meier and O. Staffelbach, “Fast correlation attacks on certain stream ciphers,” *J. Cryptol.*, vol. 1, pp. 159–176, 1989.

[14] M. Mihaljević, “An iterative probabilistic decoding algorithm for binary linear block codes beyond the half minimum distance,” *Lecture Notes in Computer Science*, Applied Algebra, Algorithms and Error Correcting Codes—AAECC 12, vol. 1255, pp. 237–249, 1997.

[15] H. Nickl, J. Hagenauer, and F. Burkert, “Approaching Shannon’s capacity limit by 0.27 dB using simple Hamming codes,” *IEEE Commun. Lett.*, vol. 1, pp. 130–132, Sept. 1997.

[16] B. Radosavljević, E. Arikan, and B. Hajek, “Sequential decoding of low-density parity-check codes by adaptive reordering of parity checks,” *IEEE Trans. Inform. Theory*, vol. 38, pp. 1833–1839, Nov. 1992.

[17] M. Sipser and D. A. Spielman, “Expander codes,” *IEEE Trans. Inform. Theory*, vol. 42, pp. 1710–1722, Nov. 1996.

[18] R. M. Tanner, “A recursive approach to low complexity codes,” *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 533–547, Sept. 1981.

[19] V. D. Kolesnik, “Probability decoding of majority codes,” *Prob. Peredachi Inform.*, vol. 7, pp. 3–12, July 1971.

[20] D. J. C. MacKay and C. P. Hesketh, “Performance of low density parity check codes as a function of actual and assumed noise levels,” *IEEE Trans. Commun.*, to be published.

[21] J. Snyders, “Reduced lists of error patterns for maximum likelihood soft decoding,” *IEEE Trans. Inform. Theory*, vol. IT-37, pp. 1194–1200, July 1991.

[22] B. G. Dorsch, “A decoding algorithm for binary block codes and J-ary output channels,” *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 391–394, May 1974.

[23] S. S. Pietrobon, “Implementation and performance of a serial MAP decoder for use in an iterative turbo decoder,” in *Proc. IEEE Int. Symp. Information Theory*, Whistler, B.C., Canada, Sept. 1995, p. 471.

[24] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal decoding algorithms in the log domain," in *Proc. ICC,* Seattle, USA, June 1995, pp. 1009–1013.

[25] A. J. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 260–264, Feb. 1998.

Marc P. C. Fossorier (S'90–M'95) 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 at Manoa in 1991 and 1994, all in electrical engineering.

He is currently with the Department of Electrical Engineering of the University of Hawaii at Manoa, where he has been an Assistant Professor since January 1996, and worked as a Postdoctoral Fellow in 1995. His research interests include decoding techniques for linear codes, communication algorithms, ISI channels, and statistics. He coauthored (with S. Lin, T. Kasami, and T. Fujiwara) *Trellises and Trellis-Based Decoding Algorithms* (Kluwer, 1998).

Dr. Fossorier is a member of the IEEE Information Theory, Communications, and Magnetics Societies. Since 1996, he has served as Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS, and is currently the Treasurer of the IEEE Information Theory Society.

Miodrag Mihaljević received the B.E. degree in electrical engineering from University of Belgrade, Yugoslavia, in 1979 and the M.S. and Ph.D. degrees in 1981 and 1990, respectively.

He is currently a Research Associate Professor and a Project Leader at the Mathematical Institute, Academy of Science and Arts, Belgrade. His research area is information processing including cryptology, computer security, and coding theory.

Hideki Imai (M'74–SM'88–F'92) was born in Shimane, Japan, on May 31, 1943. He received the B.E., M.E., and Ph.D. degrees in electrical engineering from the University of Tokyo, Japan, in 1966, 1968, and 1971, respectively.

From 1971 to 1992 he was on the faculty of Yokohama National University. In 1992 he joined the faculty of the University of Tokyo, where he is currently a Full Professor in the Institute of Industrial Science. His current research interests include information theory, coding theory, cryptography, spread spectrum systems, and their applications.

Dr. Imai received Excellent Book Awards from IEICE in 1976 and 1991. He also received the Best Paper Award (Yonezawa Memorial Award) from IEICE in 1992, the Distinguished Services Award from the Association for Telecommunication Promotion in 1994, the Telecom System Technology Prize from the Telecommunication Advancement Foundation and Achievement Award from IEICE in 1995. In 1998 he was awarded the Golden Jubilee Paper Award by the IEEE Information Theory Society. He was elected an IEEE Fellow for his contributions to the theory of coded modulation and two-dimensional codes in 1992. He chaired several committees of scientific societies such as the IEICE Professional Group on Information Theory. He served as the Editor of several scientific journals of IEICE, IEEE, etc. He chaired many international conferences such as the 1993 IEEE International Theory Workshop and the 1994 International Symposium on Information Theory and Its Applications (ISITA'94). He has been on the board of IEICE, the IEEE Information Theory Society, Japan Society of Security Management (JSSM), and the Society of Information Theory and Its Applications (SITA). At present, he serves as President of the IEICE Engineering Sciences Society.