Binary Message Passing Decoding of Product Codes Based on Generalized Minimum Distance Decoding: (Invited Paper)

Binary Message Passing Decoding of Product Codes Based on Generalized Minimum Distance Decoding

Alireza Sheikh[§], Alexandre Graell i Amat[§], and Gianluigi Liva[†]
[§] Department of Electrical Engineering, Chalmers University of Technology, Sweden
[†]Institute of Communications and Navigation of the German Aerospace Center (DLR), Germany

(Invited Paper)

***Abstract*—We propose a binary message passing decoding algorithm for product codes based on generalized minimum distance decoding (GMDD) of the component codes, where the last stage of the GMDD makes a decision based on the Hamming distance metric. The proposed algorithm closes half of the gap between conventional iterative bounded distance decoding (iBDD) and turbo product decoding based on the Chase–Pyndiah algorithm, at the expense of some increase in complexity. Furthermore, the proposed algorithm entails only a limited increase in data flow compared to iBDD.**

I. Introduction

Applications requiring very high throughputs, such as fiber-optic communications and high-speed wireless communications, have recently triggered a significant amount of research on low-complexity decoders. While codes-on-graphs such as low-density parity-check (LDPC) codes and turbo codes have been shown to provide close-to-capacity performance under belief propagation (BP) decoding, scaling their BP decoders to yield throughtputs of the order of several Gbps or even 1 Tbps, as required for example for the the future optical metro-networks, is a very challenging task. One of the main bottlenecks is the data flow required by the exchange of soft information in the iterative BP decoding. This has spurred a great deal of research in novel low-complexity decoding algorithms.

Several works have attempted to reduce the decoding complexity of BP decoding of LDPC codes, see, e.g., [1]–[4]. For high-throughput applications, an alternative to LDPC codes with (BP) soft decision decoding (SDD) is to consider hard decision decoding (HDD). Product codes (PCs) [5], half-product codes [6], staircase codes [7], braided codes [8], and other product-like code structures [9] with HDD based on bounded distance decoding (BDD) of the component codes (which we refer here to as iterative BDD (iBDD)) yield excellent performance with a significantly reduced data flow, hence achieving very high throughputs. However, this comes at the expense of a performance loss (typically larger than 1 dB) compared to SDD.

To close the performance gap between iBDD of product-like codes and SDD of LDPC codes or product-like codes, yet with throughputs and energy consumption close to that of iBDD, another line of research recently explored is to improve the performance of the conventional iBDD. In [10], an algorithm that exploits conflicts between component codes in order to assess their reliabilities even when no channel reliability information is available, was proposed. The algorithm, dubbed anchor decoding (AD), improves the performance of iBDD at the expense of some increase in decoding complexity. In [11], a decoding algorithm based on marking the least reliable bits was proposed for staircase codes. In [12], we proposed a decoding algorithm based on BDD of the component codes, named iBDD with scaled reliability (iBDD-SR). The algorithm in [12] improves the performance of iBDD by exploiting channel reliabilities as proposed in [13] for LDPC codes, while still only exchanging binary (i.e., hard-decision) messages between component decoders, similar to iBDD. iBDD-SR improves upon iBDD and AD, and achieves the same throughput of iBDD with a slight increase in energy consumption [14]. In [15], we proposed an algorithm based on generalized minimum distance decoding (GMDD) of the component codes. The proposed algorithm closes over 50% of the performance gap between iBDD and turbo product decoding (TPD) based on the Chase–Pyndiah algorithm [16], with lower complexity than TPD. However, the algorithm, which we referred to as iterative GMDD with scaled reliability (iGMDD-SR), requires the exchange of soft information between the component decoders and hence entails a decoder data flow equivalent to that of TPD and significantly higher than that of iBDD.

In this paper, we propose a novel binary message passing (BMP) decoding algorithm for product codes based on GMDD of the component codes, which we refer to as BMP-GMDD. The proposed algorithm follows the same principle as the iGMDD-SR algorithm proposed in [15], but a crucial difference is that the Hamming distance metric is used at the final stage of the GMDD of the component codes. In contrast to iGMDD-SR, the resulting algorithm does not require the exchange of soft information, but the exchange of the hard decisions on the code bits (as conventional iBDD) and an ordered list of the $d_{\mathsf{min}} - 1$ least reliable code bits for each component code, where d_{min} is the minimum Hamming distance of the component code. This list can be represented by a small number of bits. The proposed algorithm yields performance very close to that of iGMDD-SR, closing 50% of

This work was financially supported by the Knut and Alice Wallenberg Foundation, the Swedish Research Council under grant 2016-04253, and the Ericsson Research Foundation.

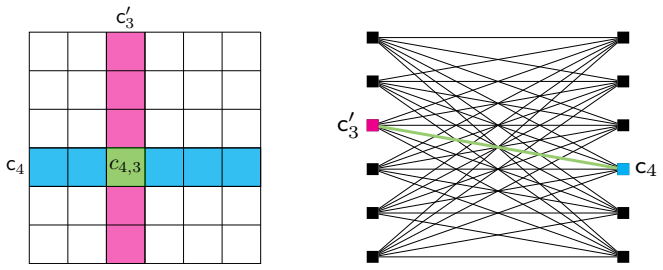

Fig. 1. Code array (left) and simplified Tanner graph (right) of a PC with identical component code of length $n = 6$ for row and column codes. In the simplified Tanner graph, the CNs are represented by squares (the CNs on the left represent the column codes and the CNs on the right represent the row codes) and degree-2 VNs are represented as simple edges. The third column code and the fourth row code are highlighted.

the performance gap between iBDD and TPD, while entailing only a small increase in data flow compared to iBDD (between 8.5% and 34.3%, depending on the code parameters).

Notation: We use boldface letters to denote vectors and matrices, e.g., $\boldsymbol{x}$ and $\boldsymbol{X} = [x_{i,j}]$. The i-th row and j-th column of matrix $\boldsymbol{X}$ is denoted by $\boldsymbol{X}_{i,:}$ and $\boldsymbol{X}_{:,j}$, respectively. $|a|$ denotes the absolute value of a, $\lfloor a \rfloor$ the largest integer smaller than or equal to a, and $\lceil a \rceil$ the smallest integer larger than or equal to a. A Gaussian distribution with mean μ and variance σ^2 is denoted by $\mathcal{N}(\mu, \sigma^2)$.

II. Preliminaries

Let $\mathcal{C}$ be an (n, k, d_{min}) binary linear code, where n, k, and d_{min} are the code length, dimension, and minimum distance, respectively. We consider two-dimensional PCs with identical binary Bose–Chaudhuri–Hocquenghem (BCH) component code $\mathcal{C}$ for the row and column codes. Such a PC, of parameters $(n^2, k^2, d_{\mathsf{min}}^2)$ and rate $R = k^2/n^2$, is defined as the set of all $n \times n$ arrays such that each row and each column of the array is a codeword of $\mathcal{C}$. Thus, a codeword of the PC can be represented by an $n \times n$ binary matrix $\boldsymbol{C} = [c_{i,j}]$. Alternatively, a PC can be defined via a Tanner graph with $2n$ constraint nodes (CNs), where n CNs correspond to the row codes and n CNs correspond to the column codes. The graph has n^2 variable nodes (VNs) corresponding to the n^2 code bits. The code array and (simplified) Tanner graph of a two-dimensional PC with $n = 6$ is shown in Fig. 1.

We assume transmission over the binary-input additive white Gaussian noise (AWGN) channel. The channel observation corresponding to code bit $c_{i,j}$ is given by

$$y_{i,j} = x_{i,j} + z_{i,j},$$

where $x_{i,j} = (-1)^{c_{i,j}}$, $z_{i,j} \sim \mathcal{N}(0, (2R\mathsf{E_b}/\mathsf{N_0})^{-1})$, with $\mathsf{E_b}/\mathsf{N_0}$ being the signal to noise ratio. We denote by $\boldsymbol{L} = [L_{i,j}]$ the matrix of channel log-likelihood ratios (LLRs) and by $\boldsymbol{R} = [r_{i,j}]$ the matrix of hard decisions at the channel output, where $r_{i,j}$ is obtained by mapping the sign of $L_{i,j}$ according to $1 \mapsto 0$ and $-1 \mapsto 1$. We denote this mapping by $\mathsf{B}(\cdot)$, i.e., $r_{i,j} = \mathsf{B}(L_{i,j})$. With some abuse of notation, we also write $\boldsymbol{R} = \mathsf{B}(\boldsymbol{L})$.

A. Generalized Minimum Distance Decoding

Consider the decoding of a BCH component code of length n and the vector of channel LLRs $\boldsymbol{l} = (L_1, \ldots, L_n)$ corresponding to the received vector $\boldsymbol{r} = (r_1, \ldots, r_n)$. GMDD is based on multiple algebraic error-erasure decoding attempts [17]. In particular, the decoder ranks the coded bits in terms of their reliabilities $|L_1|, \ldots, |L_n|$. Then, the m least reliable bits in $\boldsymbol{r}$ are erased, where $m \in \mathcal{M}_{\mathsf{odd}} \triangleq \{d_{\mathsf{min}} - 1, d_{\mathsf{min}} - 3, ..., 2\}$ if d_{min} is odd and $m \in \mathcal{M}_{\mathsf{even}} \triangleq \in \{d_{\mathsf{min}} - 1, d_{\mathsf{min}} - 3, ..., 3\}$ if d_{min} is even. For later use, we denote by $\mathcal{L}$ the ordered list of $d_{\mathsf{min}} - 1$ least reliable code bits. It can be readily checked that $|\mathcal{M}_{\mathsf{odd}}| = |\mathcal{M}_{\mathsf{even}}| = t$, where $t = \left\lfloor \frac{d_{\mathsf{min}}-1}{2} \right\rfloor$ is the error correcting capability of the code. Together with the received vector $\boldsymbol{r}$, this gives a list of $t + 1$ trial vectors $\tilde{\boldsymbol{r}}_i$, $i = 1, \ldots, t + 1$, out of which t vectors contain both erasures and (possibly) errors. Finally, algebraic error-erasure decoding [18, Sec. 6.6] is applied to each trial vector $\tilde{\boldsymbol{r}}_i$, resulting in a set of candidate codewords, of size at most $t + 1$, denoted by $\mathcal{S}$. If decoding fails for all $t + 1$ vectors in the list, a failure is declared. Otherwise, the decoder picks among all candidate codewords in $\mathcal{S}$ the one that minimizes the generalized distance $\mathsf{d_{GD}}(\boldsymbol{r}, \boldsymbol{c})$, [17]

$$\begin{aligned} \hat{\boldsymbol{c}} &= \arg\min_{\boldsymbol{c} \in \mathcal{S}} \mathsf{d_{GD}}(\boldsymbol{r}, \boldsymbol{c}) \\ &= \arg\min_{\boldsymbol{c} \in \mathcal{S}} \sum_{i: r_i = c_i} (1 - \alpha_i) + \sum_{i: r_i \neq c_i} (1 + \alpha_i), \end{aligned} \tag{1}$$

where $\alpha_i \triangleq |L_i| / \max\limits_{1 \leq j \leq n} |L_j|$. Note that if all input LLRs $L_1, \ldots, L_n$ have the same magnitude, we have $\alpha_i = 1$ for all $i = 1, \ldots, n$ and (1) reverts to $2\mathsf{d_H}(\boldsymbol{r}, \hat{\boldsymbol{c}})$, where $\mathsf{d_H}(\boldsymbol{r}, \hat{\boldsymbol{c}})$ is the Hamming distance between $\boldsymbol{r}$ and $\hat{\boldsymbol{c}}$.

By introducing erasures and performing multiple error-erasure component decoding attempts, GMDD can decode beyond half the minimum distance of the code.

III. Binary Message Passing Decoding based on Generalized Minimum Distance Decoding

In this section, we propose a BMP decoding algorithm for PCs based on GMDD of the component codes. We refer to it as BMP-GMDD. The algorithm follows the same principle as the iGMDD-SR algorithm that we proposed in [15]. However, compared to iGMDD-SR, the proposed BMP-GMDD does not require the exchange of the reliabilities on the code bits between the row and column decoders. To achieve that, rather than considering the generalized distance in (1) to perform the decision at the last stage of GMDD of the row and column decoders as in [15], we perform the decision based on the Hamming distance, i.e., among all candidate codewords in $\mathcal{S}$ (see Section II-A), the decoder selects the one that minimizes $\mathsf{d_H}(\boldsymbol{r}, \boldsymbol{c})$, i.e., the decision in (1) is substituted by

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c} \in \mathcal{S}} \mathsf{d_H}(\boldsymbol{r}, \boldsymbol{c}). \tag{2}$$

Making the decision based on the Hamming distance instead of the generalized distance entails a small performance loss, as the decision does not take into consideration the normalized

reliabilities α_i. However, this allows to significantly reduce the decoder data flow, as explained later.

The proposed BMP-GMDD algorithm works as follows. Without loss of generality, assume that the decoding starts with the row codes and let us consider the decoding of the i-th row code at iteration ℓ. Let $\boldsymbol{\Psi}^{\mathsf{c},(\ell-1)} = [\psi_{i,j}^{\mathsf{c},(\ell-1)}]$ be the matrix of hard decisions on code bits $c_{i,j}$ after the decoding of the n column codes at iteration $\ell-1$. Also, let $\mathcal{L}_i^{\mathsf{r},(\ell-1)}$ be the ordered list of $d_{\mathsf{min}}-1$ least reliable bits of codeword $\boldsymbol{C}_{i,:}$ from the decoding of the column codes at iteration $\ell-1$. Note that in the first iteration the list $\mathcal{L}_i^{\mathsf{r},(\ell-1)}$ is built according to the ordering of the channel reliabilities $\boldsymbol{L}_{i,:} = (L_{i,1}, \ldots, L_{i,n})$. Row decoding of the i-th row code is then performed based on $\boldsymbol{\Psi}_{i,:}^{\mathsf{c},(\ell-1)}$ and $\mathcal{L}_i^{\mathsf{r},(\ell-1)}$. First, GMDD of the i-th row code based on the Hamming distance is performed based on $\boldsymbol{\Psi}_{i,:}^{\mathsf{c},(\ell-1)}$ and $\mathcal{L}_i^{\mathsf{r}}$, as explained in Section II-A (see (2)). Note that GMDD does not provide reliability information about the decoded bits, i.e., it is a *soft-input hard-output* decoding algorithm. In order to provide the column decoders with the list of m least reliable bits for each codeword $\boldsymbol{C}_{:,j}$ after the decoding of the row codes at iteration ℓ, we do the following. The output bits of GMDD are mapped according to $0 \mapsto +1$ and $1 \mapsto -1$ if GMDD is successful and mapped to 0 if GMDD fails. Let $\bar{\mu}_{i,j}^{\mathsf{r},(\ell)} \in \{\pm 1, 0\}$ be the result of this mapping for the decoded bit corresponding to code bit $c_{i,j}$. The reliability information is then formed according to

$$\mu_{i,j}^{\mathsf{r},(\ell)} = w_i^{\mathsf{r},(\ell)} \cdot \bar{\mu}_{i,j}^{\mathsf{r},(\ell)} + L_{i,j}, \tag{3}$$

where $w_i^{\mathsf{r},(\ell)} > 0$ is a scaling factor than needs to be optimized. Then, the hard decision on $c_{i,j}$ made by the i-th row decoder is formed as

$$\psi_{i,j}^{\mathsf{r},(\ell)} = \mathsf{B}(\mu_{i,j}^{\mathsf{r},(\ell)}). \tag{4}$$

The hard decision $\psi_{i,j}^{\mathsf{r},(\ell)}$ is the binary message on code bit $c_{i,j}$ passed from the i-th row code to the j-th column code, i.e., from the i-th row CN to the j-th column CN (see Fig. 1). In particular, after applying this procedure to all row codes, the matrix $\boldsymbol{\Psi}^{\mathsf{r},(\ell)} = [\psi_{i,j}^{\mathsf{r},(\ell)}]$ is formed and used as the input for the n column decoders. Furthermore, after decoding of all row codes, for each column codeword $\boldsymbol{C}_{:,j}$, the corresponding code bits are ranked according to the reliabilities $(\mu_{1,j}^{\mathsf{r},(\ell)}, \ldots, \mu_{n,j}^{\mathsf{r},(\ell)})$. Then the m least reliable bits are stored in the list $\mathcal{L}_j^{\mathsf{c},(\ell)}$, which is passed to the j-th column decoder.

The decoding of the n column codes at iteration ℓ is then performed based on the hard decisions $\boldsymbol{\Psi}^{\mathsf{r},(\ell)}$ and the lists of least reliable bits $\mathcal{L}_1^{\mathsf{c},(\ell)}, \ldots, \mathcal{L}_n^{\mathsf{c},(\ell)}$ as explained for the i-th row decoder above. After decoding of the n column codes at decoding iteration ℓ, the matrix $\boldsymbol{\Psi}^{\mathsf{c},(\ell)} = [\psi_{i,j}^{\mathsf{c},(\ell)}]$ of hard decision bits and the lists $\mathcal{L}_1^{\mathsf{r},(\ell)}, \ldots, \mathcal{L}_n^{\mathsf{r},(\ell)}$ are passed to the n row decoders for the next decoding iteration. The iterative process continues until a maximum number of iterations is reached. The BMP-GMDD of PCs is schematized in Fig. 2.

Remark: With reference to Fig. 2, the iGMDD-SR algorithm proposed in [15] passes the soft information $\mu_{i,j}^{\mathsf{r},(\ell)}$ to the j-th column decoder, which entails a significantly higher decoder data flow compared to BMP-GMDD.

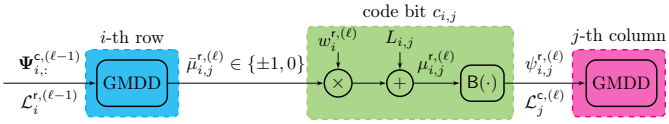

Fig. 2. Block diagram showing the information flow from the i-th row decoder to the j-th column decoder in BMP-GMDD. The message at the input of the i-th row decoder is the vector of hard decisions on the code bits $\boldsymbol{\Psi}_{i,:}^{\mathsf{c},(\ell-1)}$ and the ordered list of the $d_{\mathsf{min}}-1$ least reliable bits $\mathcal{L}_j^{\mathsf{r},(\ell-1)}$ from the decoding of the column codes at the previous iteration.

IV. Decoding Complexity Discussion

A thorough complexity analysis of BMP-GMDD should include, besides pure algorithmic aspects, implementation implications in terms of memory requirements, wiring, and transistor switching activity [14], and is beyond the scope of this paper. We however provide a high-level discussion of the complexity and data flow of BMP-GMDD compared to that of conventional iBDD, AD [10], iBDD-SR [12], and iGMDD-SR [15].

Conventional iBDD, iBDD-SR, and AD are based on BDD of the component codes and are characterized by a similar complexity and data flow. In particular, it was shown in [14] that for the same data throughput (up to 1 Tbps), iBDD-SR provides 0.2–0.25 dB gain with respect to iBDD with only slightly higher energy consumption.

Both GMDD-SR and the proposed BMP-GMDD are based on GMDD of the component codes. In this case, t error-erasure decoding attempts and one BDD attempt are required. Each error-erasure decoding attempt has a cost close to a run of BDD. Each decoding attempt may result in a candidate codeword that is used to form a list of size up to $t+1$, as explained in Section II-A. The minimization of the generalized distance in (1) for GMDD-SR and the Hamming distance in (2) for BMP-GMDD has a negligible cost with respect to the $t+1$ decoding attempts. On the other hand, both BMP-GMDD and iGMDD-SR require finding the $d_{\mathsf{min}}-1$ least reliable bits and sorting them according to their reliabilities, which adds some further complexity.

Note that GMDD-SR requires the component decoders to be provided with soft information by the previous decoding iteration. Therefore, its data flow is significantly higher than that of iBDD, iBDD-SR, and AD, and is the same of soft decision TPD. For an a-bit representation of the soft information, the data flow is roughly a times that of BDD, iBDD-SR, and AD. In contrast, BMP-GMDD requires only the exchange of the hard decisions and the ordered list of $d_{\mathsf{min}}-1$ least reliable bits for each row and column codeword. For a component code of length n, the index of each code bit can be represented with $\lceil \log_2(n) \rceil$ bits. Furthermore, for each of the $d_{\mathsf{min}}-1$ least reliable bits we need to provide their ordering in terms

Table I
COMPARISON OF DIFFERENT PRODUCT DECODING ALGORITHMS. CODING GAINS AND CAPACITY GAPS ARE MEASURED AT A BER OF 10^{-6}

acronym	decoding algorithm	channel reliabilities	exchanged messages	gain over iBDD (dB)	gap from capacity (dB)
iBDD	iterative bounded distance decoding	no	hard	-	1.03 (HD)
iBDD (ideal)	iterative bounded distance decoding without miscorrections	no	hard	0.28	0.75 (HD)
iBDD-SR	iterative bounded distance decoding with scaled reliability [12]	yes	hard	0.27	2.3 (SD)
AD	anchor decoding [10]	no	hard	0.18	0.85 (HD)
BMP-GMDD	binary message passing decoding based on GMD decoding	yes	hard	0.51	1.79 (SD)
iGMDD-SR	iterative generalized minimum distance decoding with scaled reliability [15]	yes	soft	0.58	1.72 (SD)
TPD	turbo product decoding (Chase–Pyndiah) [16]	yes	soft	1.08	1.22 (SD)

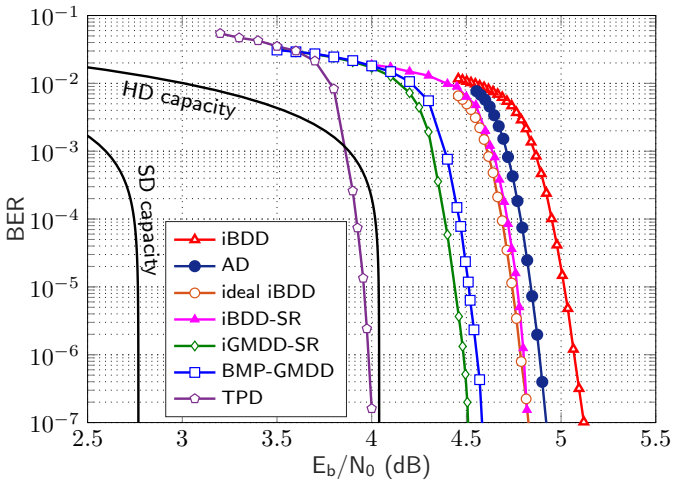

Fig. 3. BER performance of different decoding algorithms for a PC with $(256, 239, 6)$ eBCH component codes and transmission over the AWGN channel. The PC rate is 0.8716 and the maximum number of decoding iterations is 10.

of reliabilities. Thus, each ordered lists $\mathcal{L}_i^{\mathsf{r},(\ell)}$, $i = 1, \ldots, n$, and $\mathcal{L}_j^{\mathsf{c},(\ell)}$, $= j, \ldots, n$, can be represented with

$$\left(\lceil \log_2(n) \rceil + \lceil \log_2(d_{\mathsf{min}} - 1) \rceil\right)(d_{\mathsf{min}} - 1)$$

bits each. This is the additional data flow (per row and column code decoding) compared to conventional iBDD. For instance, for a component code of code length $n = 256$ bits, the data flow of BMP-GDD is 15.625% and 34.375% higher than that of iBDD for $d_{\mathsf{min}} = 5$ $(t = 2)$ and $d_{\mathsf{min}} = 9$ $(t = 4)$, respectively. For a component code of length $n = 512$ bits, the increase in data flow is reduced to 8.593% and 18.75%, respectively. Thus, the increase in data flow of BMP-GMDD compared to iBDD is very limited and is much lower than the data flow of iGMDD-SR and conventional TPD.

V. NUMERICAL RESULTS

In Fig. 3, we give the bit error rate (BER) performance of BMP-GMDD for a PC with double-error-correcting extended BCH (eBCH) codes with parameters $(256, 239, 6)$ as component codes and transmission over the AWGN channel. The resulting PC has rate $R = 239^2/256^2 \approx 0.8716$. For comparison purposes, we we also plot the performance of conventional iBDD, AD [10], iBDD-SR [12], iGMDD-SR [15], and TPD based on the Chase-Pyndiah decoding [16]. For all algorithms, a maximum of $\ell_{\max} = 10$ decoding iterations is performed. As a reference, the Shannon limit for SDD and HDD is also plotted in the figure.

Both BMP-GMDD and iGMDD-SR require a proper choice of the scaling factors $w_i^{(\ell)}$. For simplicity, we consider the same scaling factor for all row and column codes, i.e., $w_i^{\mathsf{r},(\ell)} = w_j^{\mathsf{c},(\ell)} = w^{(\ell)}$ for all $i, j = 1, \ldots, n$, and jointly optimize the vector $\boldsymbol{w} = (w^{(1)}, \ldots, w^{(\ell_{\max})})$ by using Monte–Carlo estimates of the BER for a fixed $\mathsf{E_b/N_0}$ as the optimization criterion. Intuitively, one would expect that the decisions of the component decoders become more reliable with increasing number of iterations, whereas the channel observations become less informative. Therefore, in order to reduce the optimization search space, we only consider vectors $\boldsymbol{w}$ with monotonically increasing entries. iBDD-SR also requires scaling factors (see [12], [19]). In this case, the scaling factors can be derived using density evolution [13], [19].

The two reference curves are conventional iBDD (red curve with empty triangle markers) and TPD (purple curve with pentagon markers), with the latter performing 1.1 dB better at a BER of 10^{-7}. AD (dark blue curve with filled circle markers) and iBDD-SR (pink curve with filled triangle markers) outperform conventional iBDD by 0.18 dB and 0.27 dB, respectively, at the same BER. As a reference, we also show the performance of *ideal* iBDD (brown curve with empty circle markers), where a genie prevents miscorrections. Interestingly, at a BER of 10^{-7} iBDD-SR yields the same performance as ideal iBDD.[1] iGMDD-SR (green curve with diamond markers) outperforms iBDD, iBDD-SR, and AD and closes $\approx 54\%$ of the gap between iBDD and TPD, at the expense of an increased complexity and data flow.

The performance of the proposed BMP-GMDD is given by the blue curve with square markers. The proposed decoding algorithm yields performance very close to that of iGMDD-SR (a performance degradation compared to iGMDD-SR of only 0.074 dB is observed at a BER of 10^{-7}), while achieving a significantly lower data flow. BMP-GMDD closes around

[1]We remark that the performance of iBDD-SR in Fig. 3 is improved compared to [15], since in this paper we use the optimized scaling factors based on the density evolution derived in [19], rather than based on Monte-Carlo simulations as in [15].

50% of the performance gap between iBDD and TPD, while requiring only a 21.48% higher data throughput than iBDD.

The coding gain improvements of all considered decoding algorithms over iBDD are summarized in Table I (fifth column). In the table we also indicate whether the algorithms exploit the channel reliabilities or not, the nature of the messages exchanged in the iterative decoding (hard or soft), as well as the gap to capacity for all schemes (sixth column). Note that the performance of iBDD and AD should be compared to the hard decision (HD) capacity, while the performance of iBDD-SR, iGMDD-SR, BMP-GMDD, and TPD should be compared to the soft decision (SD) capacity since the channel LLRs are exploited in the decoding. Overall, one can see a clear trade-off between BER performance and decoding complexity for the different algorithms.

We remark that if the channel LLRs are highly reliable but with wrong sign, one can expect that the decoding rule in (3) will be unable to recover from these errors. In this situation, although $\bar{\mu}_{i,j}^{\mathsf{r}}$ may correspond to a correct decision, it is overridden by the channel channel, i.e., the hard decision on code bit $c_{i,j}$ made by the i-th row decoder, $\psi_{i,j}^{\mathsf{r},(\ell)}$, becomes $\psi_{i,j}^{\mathsf{r},(\ell)} = \mathsf{B}(w_i^{\mathsf{r},(\ell)} \cdot \bar{\mu}_{i,j}^{\mathsf{r},(\ell)} + L_{i,j}) = \mathsf{B}(L_{i,j})$ (cf. (3) and (4)), leading to an erroneously decoded bit. Therefore, one needs to be careful when applying BMP-GMDD to avoid the appearance of an error floor. In particular, to avoid such errors and avoid a high error floor, we run BMP-GMDD for some iterations and then we append a few conventional iBDD iterations, where the channel reliabilities are disregarded when making the decision on a given code bit. The appended iBDD iterations increase the chance to correct transmission errors with high channel reliability. By doing so, an error floor is avoided. The same discussion applies to iBDD-SR [12], [19] and iGMDD-SR [15]. For the simulation of BMP-GMDD, iBDD-SR, and iGMDD-SR in Fig. 3 we considered 8 decoding iterations of the algorithms appended with 2 iBDD iterations.

VI. Conclusion

We proposed a new message passing decoding algorithm for product codes based on generalized minimum distance decoding, i.e., error and erasure decoding, of the component codes, where the last stage of GMDD is based on the Hamming distance metric. The proposed algorithm, dubbed BMP-GMDD, exploits soft information but requires to exchange only hard decisions and a short ordered list of the least reliable bits between component decoders, hence introducing a limited increase in data flow compared to conventional iterative bounded distance decoding. For the considered scenario based on $(256, 239, 6)$ double-error-correcting eBCH component codes, the proposed algorithm closes about 50% of the performance gap between iBDD and turbo product decoding and yields performance very close to that of the algorithm iGMDD-SR introduced in [15], with a much lower data flow, only 21.48% higher than that of iBDD. The increase in data flow is even lower for longer component codes. While in this paper we considered PCs, the proposed algorithm can be extended to other classes of product-like codes such as staircase codes. Overall, the proposed BMP-GMDD algorithm provides a very good performance-complexity tradeoff and is appealing for very high-throughput applications such as fiber-optic communications.

Acknowledgment

The authors would like to thank Dr. Christian Häger for providing the simulation results of anchor decoding in Fig. 3.

References

[1] A. Darabiha, A. C. Carusone, and F. R. Kschischang, "Power reduction techniques for LDPC decoders," *IEEE J. Solid-State Circ.*, vol. 43, no. 8, pp. 1835–1845, Aug. 2008.

[2] T. Mohsenin, D. N. Truong, and B. M. Baas, "A low-complexity message-passing algorithm for reduced routing congestion in LDPC decoders," *IEEE Trans. Circ. and Sys. I: Regular Papers*, vol. 57, no. 5, pp. 1048–1061, May 2010.

[3] F. Angarita, J. Valls, V. Almenar, and V. Torres, "Reduced-complexity Min-Sum algorithm for decoding LDPC codes with low Error-Floor," *IEEE Trans. Circ. and Sys. I: Regular Papers*, vol. 61, no. 7, pp. 2150–2158, Jul. 2014.

[4] K. Cushon, P. Larsson-Edefors, and P. Andrekson, "Low-power 400-Gbps soft-decision LDPC FEC for optical transport networks," *IEEE/OSA J. Lightw. Technol.*, vol. 34, no. 18, pp. 4304–4311, Sep. 2016.

[5] P. Elias, "Error-free coding," *Trans. IRE Professional Group on Inf. Theory*, vol. 4, no. 4, pp. 29–37, Sep. 1954.

[6] J. Justesen, "Performance of Product Codes and Related Structures with Iterated Decoding," *IEEE Trans. Commun.*, vol. 59, no. 2, pp. 407–415, Feb. 2011.

[7] B. P. Smith, A. Farhood, A. Hunt, F. R. Kschischang, and J. Lodge, "Staircase codes: FEC for 100 Gb/s OTN," *IEEE/OSA J. Lightw. Technol.*, vol. 30, no. 1, pp. 110–117, Jan. 2012.

[8] Y. Jian, H. D. Pfister, and K. R. Narayanan, "Approaching capacity at high rates with iterative hard-decision decoding," *IEEE Trans. Inf. Theory*, vol. 63, no. 9, pp. 5752–5773, Sep. 2017.

[9] C. Häger, H. D. Pfister, A. Graell i Amat, and F. Brännström, "Density Evolution for Deterministic Generalized Product Codes on the Binary Erasure Channel at High Rates," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4357–4378, Jul. 2017.

[10] C. Häger and H. D. Pfister, "Approaching Miscorrection-free Performance of Product Codes with Anchor Decoding," *IEEE Trans. Commun.*, vol. 66, no. 7, pp. 2797–2808, Jul. 2018.

[11] Y. Lei, A. Alvarado, B. Chen, X. Deng, Z. Cao, J. Li, and K. Xu, "Decoding staircase codes with marked bits," in *Proc. IEEE Int. Symp. Turbo Codes & Iterative Information Processing (ISTC)*, Hong Kong, Dec. 2018.

[12] A. Sheikh, A. Graell i Amat, and G. Liva, "Iterative bounded distance decoding of product codes with scaled reliability," in *Proc. Eur. Conf. Opt. Commun. (ECOC)*, Rome, Italy, Sep. 2018.

[13] G. Lechner, T. Pedersen, and G. Kramer, "Analysis and design of binary message passing decoders," *IEEE Trans. Commun.*, vol. 60, no. 3, pp. 601–607, Mar. 2012.

[14] C. Fougstedt, A. Sheikh, A. Graell i Amat, G. Liva, and P. Larsson-Edefors, "Energy-efficient soft-assisted product decoders," in *Proc. OSA Optical Fiber Commun. Conf. (OFC)*, San Diego, CA, Mar. 2019.

[15] A. Sheikh, A. Graell i Amat, G. Liva, C. Häger, and H. D. Pfister, "On low-complexity decoding of product codes for high-throughput fiber-optic systems," in *Proc. IEEE Int. Symp. Turbo Codes & Iterative Information Processing (ISTC)*, Hong Kong, Dec. 2018.

[16] R. M. Pyndiah, "Near-optimum decoding of product codes: block turbo codes," *IEEE Trans. Commun.*, vol. 46, no. 8, pp. 1003–1010, Aug. 1998.

[17] G. Forney, "Generalized minimum distance decoding," *IEEE Trans. Inf. Theory*, vol. 12, no. 2, pp. 125–131, Apr. 1966.

[18] S. Lin and D. J. Costello Jr., *Error Control Coding, Second Edition*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2004.

[19] A. Sheikh, A. Graell i Amat, and G. Liva, "Binary message passing decoding of product-like codes," *IEEE Trans. Commun. (submitted)*, 2019.