CHALMERS
UNIVERSITY OF TECHNOLOGY

On Low-Complexity Decoding of Product Codes for High-Throughput Fiber-Optic Systems

On Low-Complexity Decoding of Product Codes for High-Throughput Fiber-Optic Systems

Alireza Sheikh[§], Alexandre Graell i Amat[§], Gianluigi Liva[†], Christian Häger[§,‡], and Henry D. Pfister[‡]
[§] Department of Electrical Engineering, Chalmers University of Technology, Sweden
[†]Institute of Communications and Navigation of the German Aerospace Center (DLR), Germany
[‡] Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA

(Invited Paper)

***Abstract*—We study low-complexity iterative decoding algorithms for product codes. We revisit two algorithms recently proposed by the authors based on bounded distance decoding (BDD) of the component codes that improve the performance of conventional iterative BDD (iBDD). We then propose a novel decoding algorithm that is based on generalized minimum distance decoding 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 at a bit error rate of 10^{-5}. Moreover, the algorithm only leads to a limited increase in complexity with respect to iBDD and has significantly lower complexity than TPD. The studied algorithms are particularly interesting for high-throughput fiber-optic communications.**

I. Introduction

The advent of codes-on-graphs and advances in digital electronics have spurred a great deal of research on soft-decision forward error correction (SD-FEC) for fiber-optic communications in the last decade, see, e.g., [1]–[5] and references therein. Contrary to other applications such as wireless communications—where SD-FEC is the de-facto standard—research on SD-FEC in fiber-optic communications has been paralleled by a revival of research on hard-decision FEC (HD-FEC). The reason is that SD-FEC entails a significantly higher decoding complexity and data flow compared to HD-FEC. Thus, for applications where very high throughputs and low power consumption are required, HD-FEC is still an appealing alternative. HD-FEC can also be combined with multilevel modulation formats to achieve high spectral efficiency [6]–[8].

Recent research on HD-FEC for fiber-optic communications has been largely fuelled by the proposal of several new classes of product-like codes, such as staircase codes [9] and braided codes [10], [11], which we refer to as generalized product codes (GPCs). Similar to the original product codes (PCs) [12], GPCs are built from smaller component codes, typically Reed–Solomon or Bose–Chaudhuri–Hocquenghem (BCH) codes, which can be efficiently decoded via algebraic bounded distance decoding (BDD). The overall GPC can then be decoded by iteratively applying BDD to the component codes. This algorithm is referred here to as iterative BDD (iBDD). Code designs that employ iBDD can achieve an excellent performance-complexity trade-off.

GPCs can also be decoded iteratively by employing soft-input soft-output (SISO) component decoding. This is referred to as turbo product decoding (TPD) and typically implemented in practice via the Chase–Pyndiah algorithm [13].[1] TPD yields larger net coding gains than iBDD but it has a significantly higher decoding complexity. In order to (roughly) quantify the complexity increase, one may rely, for example, on decoder data flow considerations [9] or compare existing implementations (e.g., [14], [15]) in terms of gate counts. Both approaches reveal that the complexity and potential power consumption increases by around 1–2 orders of magnitude when switching from iBDD to TPD. Given that commercially available implementations of TPD already consume around 8 W to achieve a throughput of 100 Gb/s [15], it remains a significant challenge to scale such high net coding gain implementations to even higher throughputs. On the other hand, staircase decoders based on iBDD remain feasible for throughputs as large as 1 Tb/s [16].

The recent years have seen an increasing interest in the research community in closing the gap between the performance of HD-FEC and SD-FEC, while keeping the decoding complexity low. An interesting line of research is to concatenate an inner SD-FEC code, e.g., a low-density parity-check (LDPC) code decoded via belief propagation, with an outer staircase code [4], [5]. Another alternative, investigated by the authors in [17] and [18], is to improve the performance of iBDD. In [17], a new anchor decoding (AD) algorithm that exploits conflicts between component codes in order to assess their reliabilities, even when no channel reliability information is available, is proposed. The algorithm in [18] improves performance by exploiting channel reliabilities, while still

This work was financially supported by the Knut and Alice Wallenberg and the Ericsson Research Foundations. This work is also part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant No. 749798. The work was also supported by the National Science Foundation (NSF) under grant No. 1609327. Any opinions, findings, recommendations, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of these sponsors.

[1]TPD can also be based on other component decoders, e.g., the forward-backward algorithm applied to the component code trellis. In this paper, we use the term TPD to refer to the iterative SISO decoding based on the Chase–Pyndiah algorithm.

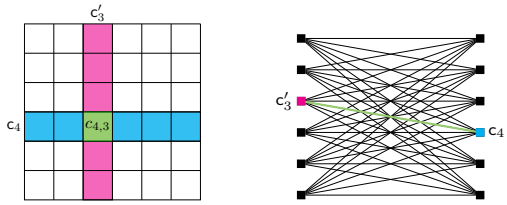

Fig. 1. Code array (left) and simplified Tanner graph (right) for a PC assuming a component code of length $n = 6$. In the simplified Tanner graph, degree-2 VNs are omitted and instead represented as simple edges.

only exchanging binary (hard-decision) messages between component codes, similar to iBDD.[2]

In this paper, we study and compare several decoding algorithms for PCs based on algebraic decoding of the component codes. While our focus is on PCs, the considered algorithms can also be applied to GPCs such as staircase and braided codes. We start by reviewing the two decoding algorithms that were proposed in [17] and [18], respectively. It is shown that for the considered scenario, both algorithms offer sizable net coding gain improvements of 0.18 dB and 0.25 dB, respectively, at a bit error rate (BER) of 10^{-5}, with only a small complexity increase compared to iBDD. We then propose a novel iterative decoding algorithm based on generalized minimum distance (GMD) decoding of the component codes, which we refer to as iterative GMD decoding with scaled reliability (iGMDD-SR). Using iGMDD-SR, a significant coding gain improvement of around 0.60 dB can be achieved compared to iBDD. This closes over 50% of the performance gap to TPD, while maintaining significantly lower complexity.

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 $\boldsymbol{X}$ are denoted by $\boldsymbol{X}_{i,:}$ and $\boldsymbol{X}_{:,j}$, respectively. $|a|$ denotes the absolute value of a, and $\lfloor a \rfloor$ the maximum integer value less than or equal to a. A Gaussian distribution with mean μ and variance σ^2 is denoted by $\mathcal{N}(\mu, \sigma^2)$. The Hamming distance between vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is denoted by $\mathsf{d_H}(\boldsymbol{a}, \boldsymbol{b})$.

II. Preliminaries

Let $\mathcal{C}$ be a binary linear $(n, k, d_{\min})$ code, where n, k, and $d_{\min}$ are the code length, dimension, and minimum distance, respectively. A (two-dimensional) PC with parameters $(n^2, k^2, d_{\min}^2)$ and rate $R = k^2/n^2$ is defined as the set of all $n \times n$ arrays such that each row and column of the array is a codeword of $\mathcal{C}$. A codeword of the product code can be represented as a binary matrix $\boldsymbol{C} = [c_{i,j}]$ of size $n^2 \times n^2$. 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 PC with $n = 6$ is shown in Fig. 1.

[2]A similar approach was analyzed in the context of low-complexity decoding algorithms for LDPC codes in [19].

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

$$y_{i,j} = x_{i,j} + z_{i,j}, \tag{1}$$

where $x_{i,j} = (-1)^{c_{i,j}}$, $z_{i,j} \sim \mathcal{N}(0, (2RE_\mathrm{b}/N_0)^{-1})$, and E_b/N_0 is the signal to noise ratio. Let $\boldsymbol{L} = [L_{i,j}]$ be the matrix of channel log-likelihood ratios (LLRs) and $\boldsymbol{R} = [r_{i,j}]$ 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 $+1 \mapsto 1$. This mapping is denoted 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. Bounded Distance Decoding

Consider now the decoding of an arbitrary row or column component code, assuming that the codeword $\boldsymbol{c} = (c_1, \ldots, c_n)$ is transmitted and only hard-detected channel observations $\boldsymbol{r} = (r_1, \ldots, r_n)$ are available. BDD corrects all error patterns with Hamming weight up to the error-correcting capability of the code, $t = \left\lfloor \frac{d_{\min}-1}{2} \right\rfloor$. If the weight of the error pattern is larger than t and there exists another codeword $\tilde{\boldsymbol{c}} \in \mathcal{C}$ with $\mathsf{d_H}(\tilde{\boldsymbol{c}}, \boldsymbol{r}) \leq t$, then BDD maps $\boldsymbol{r}$ to $\tilde{\boldsymbol{c}}$ and thus introduces a miscorrection. Otherwise, if no such codeword exists, BDD fails and we use the convention that the decoder outputs $\boldsymbol{r}$. Thus, the decoded vector $\hat{\boldsymbol{r}}$ for BDD can be written as

$$\hat{\boldsymbol{r}} = \begin{cases} \boldsymbol{c} & \text{if } \mathsf{d_H}(\boldsymbol{c}, \boldsymbol{r}) \leq t \\ \tilde{\boldsymbol{c}} \in \mathcal{C} & \text{if } \mathsf{d_H}(\boldsymbol{c}, \boldsymbol{r}) > t \text{ and } \exists \tilde{\boldsymbol{c}} \text{ such that } \mathsf{d_H}(\tilde{\boldsymbol{c}}, \boldsymbol{r}) \leq t \\ \boldsymbol{r} & \text{otherwise} \end{cases}. \tag{2}$$

B. Generalized Minimum Distance Decoding

Consider again the component decoding but now assume that the vector of channel LLRs $\boldsymbol{l} = (L_1, \ldots, L_n)$ is available. In that case, GMD decoding, which is based on multiple algebraic error-erasure decoding attempts [20], can be employed. 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 $m \in \{d_{\min} - 1, d_{\min} - 3, ..., 2\}$ if $d_{\min}$ is odd and $m \in \{d_{\min} - 1, d_{\min} - 3, ..., 3\}$ if $d_{\min}$ is even. Together with $\boldsymbol{r}$, 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 [21, Sec. 6.6] is applied to each $\tilde{\boldsymbol{r}}_i$. If error-erasure decoding fails for all $t + 1$ vectors in the list, an overall failure is declared for the GMD decoding. On the other hand, if some of the error-erasure decoding attempts did not fail, the decoder picks among all decoded candidate codewords $\hat{\boldsymbol{r}}$ the one that minimizes the generalized distance [20]

$$\mathsf{d_{GD}}(\boldsymbol{r}, \hat{\boldsymbol{r}}) = \sum_{i: r_i = \hat{r}_i} (1 - \alpha_i) + \sum_{i: r_i \neq \hat{r}_i} (1 + \alpha_i), \tag{3}$$

where $\alpha_i \triangleq |L_i| / \max\limits_{1 \leq j \leq n} |L_j|$. Note that if all input LLRs L_j have the same magnitude, we have $\alpha_i = 1$ for all $i = 1, \ldots, n$ and (3) reverts to $2\mathsf{d_H}(\boldsymbol{r}, \hat{\boldsymbol{r}})$.

III. Iterative Decoding of Product Codes

A. Anchor Decoding

When a miscorrection occurs during iBDD, it is possible that two component codes disagree on the value of a particular bit, leading to a conflict. Conflicts are conventionally ignored in the sense that row and column codes are decoded sequentially and previous decoding decisions are simply overridden. The main idea in AD is to introduce status information for each component code and designate certain "reliable" component codes as anchors. Then, no further additional corrections from other component codes are allowed if this would lead to a conflict and overturn the decision of an anchor. Since some anchors may actually be miscorrected, AD also allows for the backtracking of the decoding decisions of anchors. This happens whenever too many other component codes are in conflict with a particular anchor. Pseudocode for AD can be found in [17, Alg. 2].

B. Iterative Bounded Distance Decoding With Scaled Reliability

Both iBDD and AD do not take potentially available channel reliability information into account. In [18], we proposed a modification of iBDD where channel reliabilities are exploited, while only binary messages between component decoders are exchanged. This algorithm is referred to as iBDD with scaled reliability (iBDD-SR). In particular, assume that the i-th row code has been decoded via BDD. In order to combine the BDD output with the channel LLRs, the decoded bits are mapped according to $0 \to +1$ and $1 \to -1$ if BDD is successful and mapped to 0 if a decoding failure occurs. 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 $c_{i,j}$ in iteration ℓ. Then, we compute

$$\psi_{i,j}^{\mathsf{r},(\ell)} = \mathsf{B}(w_\ell \cdot \bar{\mu}_{i,j}^{\mathsf{r},(\ell)} + L_{i,j}), \tag{4}$$

where $w_\ell > 0$ is a scaling parameter. $\psi_{i,j}^{\mathsf{r},(\ell)}$ can be interpreted as the message passed from the i-th row code to the j-th column code. In particular, after applying this procedure to all row codes, the matrix $\boldsymbol{\Psi}^{\mathsf{r},(\ell)} = [\psi_{i,j}^{\mathsf{r},(\ell)}]$ is used as the input for the column codes, where BDD based on $\boldsymbol{\Psi}^{\mathsf{r},(\ell)}$ is performed. The binary output messages for the column codes are then formed in a similar fashion as for the row codes. Intuitively, the mapping (4) helps to alleviate the effect of miscorrections by allowing the outcome of BDD at certain bit positions to be overturned if the corresponding channel observation is very reliable (i.e., $|L_{i,j}|$ is large).

IV. Iterative Generalized Minimum Distance Decoding with Scaled Reliability

In this section, we propose a novel iterative decoding algorithm for PCs based on GMD decoding of the component codes and the exchange of soft information between component codes, which we refer to as iGMDD-SR. The 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 $\tilde{\boldsymbol{\mu}}_i^{\mathsf{r},(\ell)}$ be the vector of soft information corresponding to $\boldsymbol{C}_{i,:}$ at the input of the i-th row decoder at iteration ℓ, resulting from the decoding of the n column codes at decoding iteration $\ell - 1$, where, initially, $\tilde{\boldsymbol{\mu}}_i^{\mathsf{r},(1)} = \boldsymbol{L}_{i,:}$. Also, let $\boldsymbol{R}_{i,:}^{(\ell)} = \mathsf{B}(\tilde{\boldsymbol{\mu}}_i^{\mathsf{r},(\ell)})$ be the corresponding hard-decoded vector. Then, GMD decoding of the i-th row code is performed as explained in Section II-B based on $\boldsymbol{R}_{i,:}^{(\ell)}$ and the reliabilities $|\tilde{\boldsymbol{\mu}}_i^{\mathsf{r},(\ell)}|$. Note that GMD decoding does not provide reliability information about the decoded bits, i.e., it is "soft-input, hard-output". In order to provide reliability information to the column decoders, we resort to a heuristic approach that is similar to the approach used for iBDD-SR. In particular, the output bits of GMD decoding are mapped according to $0 \to +1$ and $1 \to -1$ if GMD decoding is successful and mapped to 0 if GMD decoding 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

$$\mu_{i,j}^{\mathsf{r},(\ell)} = w_\ell \cdot \bar{\mu}_{i,j}^{\mathsf{r},(\ell)} + L_{i,j}, \tag{5}$$

where $w_\ell > 0$ is a scaling parameter. We let $\boldsymbol{\mu}_i^{\mathsf{r},(\ell)} = (\mu_{i,1}^{\mathsf{r},(\ell)}, \ldots, \mu_{i,n}^{\mathsf{r},(\ell)})$ denote the entire soft-output vector of the i-th row decoder.

Visualizing the decoding over the Tanner graph of the code, $\mu_{i,j}^{\mathsf{r},(\ell)}$ corresponds to the message from row CN i to column CN j. Now assume that all row codes have been decoded. The vector of soft information corresponding to $\boldsymbol{C}_{:,j}$ at the input of the j-th column decoder at decoding iteration ℓ is then defined as $\tilde{\boldsymbol{\mu}}_j^{\mathsf{c},(\ell)} = (\tilde{\mu}_{j,1}^{\mathsf{c},(\ell)}, \ldots, \tilde{\mu}_{j,n}^{\mathsf{c},(\ell)}) = (\mu_{1,j}^{\mathsf{r},(\ell)}, \ldots, \mu_{n,j}^{\mathsf{r},(\ell)})$. GMD decoding is then performed using $\boldsymbol{R}_{:,j}^{(\ell)} = \mathsf{B}(\tilde{\boldsymbol{\mu}}_j^{\mathsf{c},(\ell)})$ and $|\tilde{\boldsymbol{\mu}}_j^{\mathsf{c},(\ell)}|$. The soft output of the j-th column decoder is formed similar to (5) and the resulting soft output vector is denoted by $\boldsymbol{\mu}_j^{\mathsf{c},(\ell)} = (\mu_{j,1}^{\mathsf{c},(\ell)}, \ldots, \mu_{j,n}^{\mathsf{c},(\ell)})$, where $\mu_{j,i}^{\mathsf{c},\ell}$ corresponds to the message from column CN j to row CN i. After decoding all column component codes, we set $\tilde{\boldsymbol{\mu}}_i^{\mathsf{r},(\ell+1)} = (\tilde{\mu}_{i,1}^{\mathsf{r},(\ell+1)}, \ldots, \tilde{\mu}_{i,n}^{\mathsf{r},(\ell+1)}) = (\mu_{1,i}^{\mathsf{c},(\ell)}, \ldots, \mu_{n,i}^{\mathsf{c},(\ell)})$ and the iterative process continues until a maximum number of iterations is reached. The information flow from the row to column codes in iGMDD-SR is schematically illustrated in Fig. 2.

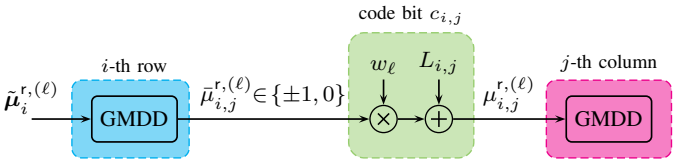

Fig. 2. Block diagram showing the information flow from the i-th row to the j-th column code in iGMDD-SR.

V. Decoding Complexity Discussion

A thorough complexity analysis of the different decoding algorithms presented in Sections II-A, III-B, and IV is a formidable task that should include, besides the pure algorithmic aspects, the implications in terms of internal data

flow. A complete analysis is therefore out of the scope of this paper. We however provide a high-level discussion of complexity aspects associated with the different choices of the component code decoders, focusing on the complexity per decoding iteration. Additional remarks on the decoding complexity of AD and iBDD-SR can be found in the respective papers [17] and [18].

The use of BDD to decode the component codes represents the simplest approach among those considered. From this viewpoint, iBDD, AD, and iBDD-SR are characterized by a similar complexity. For iBDD-SR, the combination of the BDD output and the channel LLRs in (4) yields a small complexity increase with respect to iBDD and AD. With respect to BDD of the component codes, GMD decoding entails more substantial changes in the decoder. In this case, the component decoder has to be provided with soft decisions by the previous decoding step. Finally, 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 $t+1$. The minimization of the distance in (3) has a negligible cost with respect to the $t+1$ decoding attempts.

The decoding complexity of TPD exceeds the complexity of AD, iBDD-SR, and iGMDD-SR. In particular, the Chase–Pyndiah algorithm requires the construction of a list of binary test sequences for each component decoding, where the list size depends on a design parameter. Typically, the list size is set to at least 16 [13], and BDD is applied to each test sequence. Thus, for component codes where t is small (e.g., 2, 3, or 4), as the ones considered in this paper and of practical use for high-throughput fiber-optic communications, the list size and corresponding cost per component decoding in GMD decoding is only a fraction of that of TPD. Moreover, iGMDD-SR also relaxes the computational requirements when computing the extrinsic soft-output information for each code bit compared to TPD by using the heuristic update equation in (5).

VI. Simulation Results

In this section, we compare the product decoding algorithms in terms of their performance. For the simulations, we consider double-error-correcting extended BCH (eBCH) codes with parameters $(256, 239, 6)$ as component codes. The resulting PC has rate $R = 239^2/256^2 \approx 0.8716$, corresponding to an FEC overhead of $1/R - 1 \approx 15\%$. For all algorithms, a maximum of $\ell_{\max} = 10$ decoding iterations is performed.

Both iBDD-SR and iGMDD-SR require a proper choice for the scaling factors w_ℓ in each iteration. We jointly optimize all scaling factors $\boldsymbol{w} = (w_1, \ldots, w_{\ell_{\max}})$ by using Monte–Carlo estimates of the bit error rate (BER) for a fixed E_b/N_0 as the optimization criterion. Intuitively, one would expect that the decisions of the component decoders become more reliable with 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. For iBDD-SR, we found that the optimized vector $\boldsymbol{w}$ is relatively insensitive to the targeted E_b/N_0. On the other hand, iGMDD-SR is more sensitive to a mismatch between the optimized and actual E_b/N_0 and the optimization is thus performed for each value of E_b/N_0 separately.

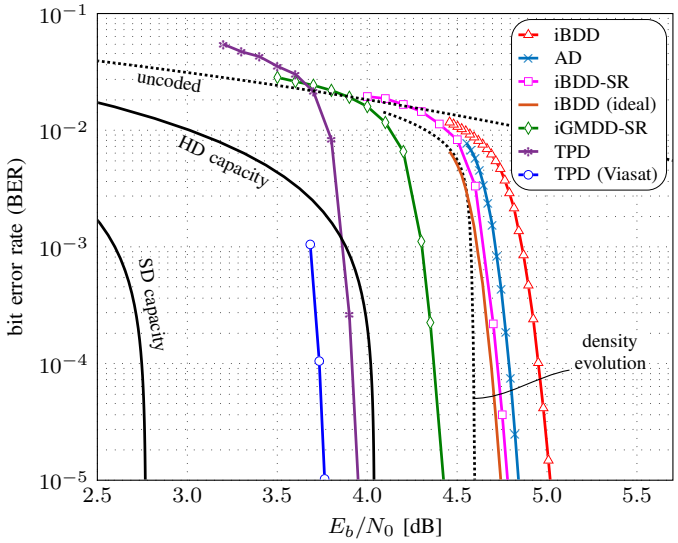

Fig. 3. Performance of different product decoding algorithms for $(256,239,6)$ eBCH component codes and 10 iterations. The PC rate is 0.8716 corresponding to 15% FEC overhead. For TPD (Viasat), results are taken from [15] for the same overhead, but different component codes may be employed.

In Fig. 3, the performance of iBDD, AD, iGMDD-SR, and iBDD-SR is shown. We also plot the performance of TPD via off-the-shelf Matlab toolbox functions and compare to a commercially available 100 Gb/s SD-FEC solution implementing TPD for the same overhead [15]. The data points are directly extracted from [15], but we remark that the component code details are not disclosed in [15]. Thus, a different PC may be used. Moreover, pre- and post-processing steps are employed, which may also explain the performance difference.

AD and iBDD-SR outperform the conventional iBDD by 0.18 dB and 0.25 dB, respectively, at a BER of 10^{-5}. As a reference, we show the performance of idealized iBDD, where a genie prevents all miscorrections. The asymptotic performance of idealized iBDD can be analytically predicted by using density evolution [22], [23], which is shown by the dotted line. It can be seen that both AD and iBDD-SR are effective algorithms to combat miscorrections. The performance degradation of iBDD-SR compared to ideal iBDD is very small (< 0.01 dB), implying that by properly tuning the scaling parameters $\boldsymbol{w}$, iBDD-SR can alleviate the effect of miscorrections to a large extent.

One can also see that iGMDD-SR outperforms iBDD, AD, and iBDD-SR. In particular, the performance gain of iGMDD-SR over iBDD is 0.60 dB at a BER of 10^{-5}. The observed additional coding gain is expected, since GMD decoding can decode beyond half the minimum distance by introducing erasures and performing multiple error-erasure component decoding attempts. Furthermore, iGMDD-SR performs 0.52

Table I
COMPARISON OF DIFFERENT PRODUCT DECODING ALGORITHMS. CODING GAINS AND CAPACITY GAPS ARE MEASURED AT BER $= 10^{-5}$.

acronym	decoding algorithm	channel reliabilities	exchanged messages	gain over iBDD [dB]	gap from capacity [dB]
iBDD	iterative bounded distance decoding	no	hard	-	0.98 (HD)
AD	anchor decoding [17]	no	hard	0.18	0.80 (HD)
iBDD-SR	iterative bounded distance decoding with scaled reliability [18]	yes	hard	0.25	2.00 (SD)
iBDD (ideal)	iterative bounded distance decoding without miscorrections	no	hard	0.28	0.70 (HD)
iGMDD-SR	iterative generalized minimum distance decoding with scaled reliability	yes	soft	0.60	1.66 (SD)
TPD	turbo product decoding (Chase–Pyndiah) [13]	yes	soft	1.08	1.18 (SD)
TPD (Viasat)	commercially available decoder (undisclosed component code details) [15]	yes	soft	1.26	1.00 (SD)

dB away from TPD, i.e., it closes over 50% of the performance gap between iBDD and TPD.

The net coding gain improvements of all considered decoding algorithms over iBDD are summarized in Table I. We also indicate the gap to capacity for all schemes. Note that the performance of iBDD and AD should be compared to the HD capacity, whereas the performance of iBDD-SR, iGMDD-SR, and TPD should be compared to the SD capacity since channel LLRs are exploited during decoding. Overall, one can see a clear trade-off between performance and complexity for the different algorithms, e.g., using iGMDD-SR, with higher complexity than iBDD, yet less complexity than TPD, the gap between iBDD and TPD is approximately halved.

VII. CONCLUSION

We studied several low-complexity iterative decoding algorithms for PCs that outperform the conventional iBDD. In particular, we reviewed two previously proposed algorithms, AD and iBDD-SR, and we proposed a novel algorithm called iterative GMD decoding with scaled reliability (iGMDD-SR). For the considered scenario based on double-error-correcting eBCH component codes, AD and iBDD-SR outperform iBDD by 0.18 dB and 0.25 dB, respectively, with only a small increase in complexity. The complexity increase for iGMDD-SR is larger, but the algorithm achieves a more significant performance gain of 0.60 dB over iBDD. This closes over 50% of the performance gap to TPD, at a significantly lower complexity.

REFERENCES

[1] A. Leven and L. Schmalen, "Status and recent advances on forward error correction technologies for lightwave systems," *IEEE/OSA J. Lightw. Technol.*, vol. 32, no. 16, pp. 2735–2750, Aug. 2014.

[2] L. Schmalen, V. Aref, J. Cho, D. Suikat, D. Rösener, and A. Leven, "Spatially coupled soft-decision error correction for future lightwave systems," *IEEE/OSA J. Lightw. Technol.*, vol. 33, no. 5, pp. 1109–1116, Mar. 2015.

[3] C. Häger, A. Graell i Amat, F. Brännström, A. Alvarado, and E. Agrell, "Terminated and tailbiting spatially coupled codes with optimized bit mappings for spectrally efficient fiber-optical systems," *IEEE/OSA J. Lightw. Technol.*, vol. 33, no. 7, pp. 1275–1285, Apr. 2015.

[4] L. M. Zhang and F. R. Kschischang, "Low-complexity soft-decision concatenated LDGM-staircase FEC for high-bit-rate fiber-optic communication," *IEEE/OSA J. Lightw. Technol.*, vol. 35, no. 18, pp. 3991–3999, Sep. 2017.

[5] M. Barakatain and F. R. Kschischang, "Low-complexity concatenated LDPC-staircase codes," *IEEE/OSA J. Lightw. Technol.*, vol. 36, no. 12, pp. 2443–2449, Jun. 2018.

[6] B. Smith and F. R. Kschischang, "A Pragmatic Coded Modulation Scheme for High-Spectral-Efficiency Fiber-Optic Communications," *J. Lightw. Technol.*, vol. 30, no. 13, pp. 2047–2053, Jul. 2012.

[7] C. Häger, A. Graell i Amat, H. D. Pfister, and F. Brännström, "Density Evolution for Deterministic Generalized Product Codes with Higher-Order Modulation," in *Proc. Int. Symp. Turbo Codes and Iterative Information Processing (ISTC)*, Brest, France, Sep. 2016, pp. 236–240.

[8] A. Sheikh, A. Graell i Amat, G. Liva, and F. Steiner, "Probabilistic Amplitude Shaping with Hard Decision Decoding and Staircase Codes," *IEEE/OSA J. Lightw. Technol.*, vol. 36, no. 9, pp. 1689–1697, May 2018.

[9] 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.

[10] A. J. Feltström, D. Truhachev, M. Lentmaier, and K. S. Zigangirov, "Braided Block Codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 6, pp. 2640–2658, Jul. 2009.

[11] Y. Y. Jian, H. D. Pfister, K. R. Narayanan, R. Rao, and R. Mazahreh, "Iterative hard-decision decoding of braided BCH codes for high-speed optical communication," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2013, pp. 2376–2381.

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

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

[14] C. Condo, P. Giard, F. Leduc-Primeau, G. Sarkis, and W. J. Gross, "A 9.52 dB NCG FEC scheme and 162 b/Cycle low-complexity product decoder architecture," *IEEE Trans. Circuits and Systems I*, vol. 65, no. 4, pp. 1420–1431, Apr. 2018.

[15] Viasat Inc., "ECC66100 series SD-FEC encoder/decoder cores," 2017. [Online]. Available: https://www.viasat.com/sites/default/files/media /documents/ecc_66100_datasheet_2_pgr_007_web_0.pdf

[16] C. Fougstedt and P. Larsson-edefors, "Energy-Efficient High-Throughput Staircase Decoders," in *Proc. Optical Fiber Commun. Conf. (OFC)*, San Diego, CA, Mar. 2018.

[17] 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.

[18] 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.

[19] 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.

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

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

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

[23] 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.