

Binary Message Passing Decoding of Product-like Codes

Alireza Sheikh *Student Member, IEEE*, Alexandre Graell i Amat, *Senior Member, IEEE*,
and Gianluigi Liva, *Senior Member, IEEE*

Abstract—We propose a novel binary message passing decoding algorithm for product-like codes based on bounded distance decoding (BDD) of the component codes. The algorithm, dubbed iterative BDD with scaled reliability (iBDD-SR), exploits the channel reliabilities and is therefore soft in nature. However, the messages exchanged by the component decoders are binary (hard) messages, which significantly reduces the decoder data flow. The exchanged binary messages are obtained by combining the channel reliability with the BDD decoder output reliabilities, properly conveyed by a scaling factor applied to the BDD decisions. We perform a density evolution analysis for generalized low-density parity-check (GLDPC) code ensembles and spatially coupled GLDPC code ensembles, from which the scaling factors of the iBDD-SR for product and staircase codes, respectively, can be obtained. For the white additive Gaussian noise channel, we show performance gains up to 0.29 dB and 0.31 dB for product and staircase codes compared to conventional iterative BDD (iBDD) with the same decoder data flow. Furthermore, we show that iBDD-SR approaches the performance of ideal iBDD that prevents miscorrections.

Index Terms—Binary message passing, bounded distance decoding, complexity, hard decision decoding, product codes, staircase codes.

I. INTRODUCTION

A RENEWED interest in the design of iterative coding schemes has been recently triggered by the need of efficient error correction mechanisms for very high throughput applications. Turbo and low-density parity-check (LDPC) codes have been shown to be capable of approaching the channel capacity under belief propagation (BP) decoding [1], [2]. However, applications requiring transmission at data rates of several hundreds of Gbps (such as optical transport systems [3]) pose a challenge to the implementation of fast BP decoders. The difficulty is especially due to the handling of the internal decoder data flow when soft messages are exchanged within the iterative decoder. In this context, attempts to reduce to complexity of soft-input soft-output (SISO) LDPC decoders have been undertaken, e.g., in [4]–[7]. An alternative line of research is to resort to hard decision decoding (HDD) for such applications. HDD reduces the decoder data flow [8] at

the expense of some performance loss compared to (BP) soft decision decoding (SDD) [9].

Among the coding schemes that are particularly suited for high throughput HDD, product codes (PCs) [10] gained a large attention. The iterative decoding of PCs dates back to 1968 [11], and it regained attention after employing SDD based on the Chase-Pyndiah decoding algorithm, which improved the coding gain extensively [12]. Several works tackled the complexity reduction of the Chase-Pyndiah decoder [13]–[17]. However, these approaches still require the exchange of soft messages between component decoders, which prevents from achieving very high throughputs. Recently, HDD of product-like codes with bounded distance decoding (BDD) of the component codes, which we refer here to iterative BDD (iBDD), has been considered for high-throughput optical communications due to their excellent performance-complexity tradeoff, see, e.g., [18]–[20]. For instance, PCs have been adopted by the optical submarine standard [21].

To keep up with the increasing demand of coding gains, performance improvements of the coding scheme coupled with a reasonable decoding complexity are necessary. In [22], an algorithm that exploits conflicts between component decoders in order to assess their reliabilities even when no channel reliability information is available, was proposed. The algorithm, named anchor decoding (AD), improves the performance of iBDD at the expense of some increase in decoding complexity. However, the performance is still limited by the binary (i.e., hard) nature of the decoder input. The use of HDD is typically motivated by the need of operating with low-complexity analog to digital converters (ADCs) and the above mentioned limitations in the internal decoder data flow. Whereas the latter motivation is hard to circumvent, the former motivation is often irrelevant for several practical applications. In this case, the use of soft information at the decoder input may be considered if the complexity of the decoding algorithms is kept close to that of HDD. Following this paradigm, one proposal is to concatenate an inner code for which SDD can be performed with limited complexity, with an outer HDD code [23], [24].

In this paper, we propose a low-complexity binary message passing decoding algorithm for product-like codes that relies on BDD of the component codes. The proposed algorithm exploits the channel reliabilities and thus is *soft* in nature. However, the component decoders exchange only hard decisions (i.e., binary messages), which yields a decoder data flow equal to that of conventional iBDD. The binary messages are obtained by combining the BDD outputs with the channel

This paper was presented in part at the European Conference on Optical Communications (ECOC), Rome, Italy, 2018, and the International Symposium on Turbo codes & Iterative Information Processing, Hong Kong, 2018.

This work was funded by the Knut and Alice Wallenberg Foundation.

A. Sheikh and A. Graell i Amat are with the Department of Electrical Engineering, Chalmers University of Technology, SE-41296 Gothenburg, Sweden (email: asheikh, alexandre.graell@chalmers.se).

G. Liva is with the Institute of Communications and Navigation of the German Aerospace Center (DLR), Münchner Strasse 20, 82234 Weßling, Germany (email: gianluigi.liva@dlr.de).

soft information similar to the approach proposed in [25] for the decoding of LDPC codes. The algorithm, which we dub iterative bounded distance decoding with scaled reliability (iBDD-SR), relies on the weighted sum of the BDD output with the channel log-likelihood ratio (LLR), where the BDD decoder output reliability is conveyed by a scaling factor applied to the BDD outbound messages.

This work is the extension of the conference paper [26], where we originally proposed iBDD-SR for PCs, and where the scaling factors were derived based on simulation results. This prevented the extension of iBDD-SR to other product-like codes, as the search of the scaling factors based on simulations for product-like codes such as staircase codes proved infeasible. Here, we derive a density evolution under iBDD-SR for generalized LDPC (GLDPC) code ensembles and spatially coupled GLDPC (SC-GLDPC) code ensembles, based on extrinsic BDD of the component codes, that allows us to derive the scaling factors. The derived density evolution analysis rigorously takes care of miscorrections, and follows the same principles as the density evolution proposed in [27] for the iBDD case. In particular, the density evolution for GLDPC and SC-GLDPC ensembles allows to obtain the optimal (asymptotically in the large blocklength limit) scaling factors for iBDD-SR of PCs and *spatially coupled* product-like codes such as staircase codes, respectively. Our simulation results show that iBDD-SR remarkably outperforms iBDD for both PCs and staircase codes, and performs very close to the (genie-aided) miscorrection-free iBDD.

The proposed algorithm is a promising solution for very high-throughput applications such as fiber-optic communications. For instance, the 400G ZR standard for transmission at 400Gbps over data center interconnect links up to 100 km, has agreed on an FEC scheme consisting of the concatenation of an inner Hamming code decoded soft and an outer staircase code with hard decision decoding.

Related work includes [28], [29], and [30]. All these algorithms require the knowledge of the least reliable bits in the decoding process, and are significantly more complex than the proposed algorithm.

Notation: We use boldface letters to denote vectors and matrices, e.g., \mathbf{x} and \mathbf{X} , with $x_{i,j}$ representing the element corresponding to the i -th row and j -th column of \mathbf{X} . $|a|$ 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 $\mathcal{N}(\mu, \sigma^2)$. The Hamming distance between vectors \mathbf{a} and \mathbf{b} is denoted by $d_H(\mathbf{a}, \mathbf{b})$.

II. PRELIMINARIES

Our focus is on two main classes of product-like codes, namely two-dimensional PCs and staircase codes.

A. Product Codes and Staircase Codes

We consider two-dimensional PCs with the same component code for the row and column codes. However, we remark that the proposed decoding algorithm extends in a straightforward

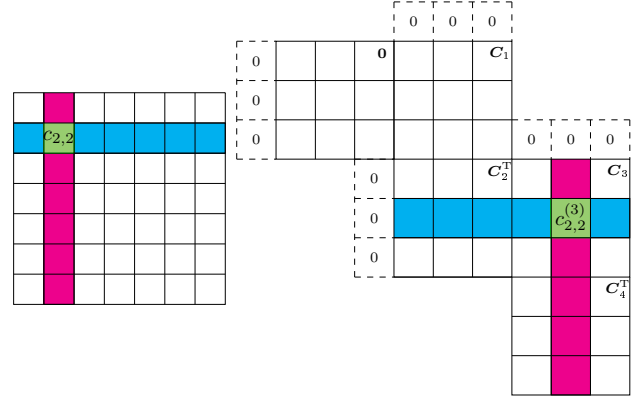


Fig. 1: PC and staircase codes with $n = 7$ and $n = 6$ and the corresponding code bits $c_{2,2}$ and $c_{2,2}^{(3)}$. The shortened bits are dashed.

manner to more general PCs, where the row and column codes are not necessarily the same.

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 Hamming distance, respectively. A (two-dimensional) PC with parameters (n^2, k^2, d_{\min}^2) and code rate $R = k^2/n^2$ based on component code \mathcal{C} 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 PC can thus be represented as a binary matrix $\mathbf{C} = [c_{i,j}]$ of size $n \times n$. Fig. 1 shows the code array of a PC with component codes of length $n = 7$, where the code bit $c_{2,2}$ and the row and column constraints in which participates are highlighted.

PCs can be represented by a Tanner graph, where variable nodes (VNs) represent code bits and constraint nodes (CNs) represent row and column codes, respectively. For a PC with component code length n , the corresponding Tanner graph has n^2 degree-2 VNs (each of the n^2 code bits participates in 2 code constraints, a row constraint and a column constraint) and $2n$ degree- n CNs (n bits participate in each code constraint). PCs are contained in the ensemble of GLDPC codes described by the Tanner graph in Fig. 2, where each CN corresponds to a block code of length n (the component code of the PC). Note that the Tanner graph of a PC is a particular instance of the Tanner graph in Fig. 2, where the connectivity between VNs and CNs is deterministic.

A binary staircase code is defined by a two-dimensional code array that has the form of a staircase. Formally, given a component code of length n , a staircase code is defined as the set of all matrices \mathbf{B}_i of size $\frac{n}{2} \times \frac{n}{2}$, $i = 1, 2, \dots$, such that each row of the matrix $[\mathbf{B}_{i-1}^T, \mathbf{B}_i]$ is a codeword of \mathcal{C} . \mathbf{B}_0 is initialized to all zeros and the code rate is $R = 1 - \frac{2(n-k)}{n}$ [8]. We will refer sometimes to the matrices \mathbf{B}_i as blocks. In Fig. 1, we plot the code array corresponding to the first four spatial positions of a staircase code with component codes of length $n = 6$. The code bit $c_{2,2}^{(3)}$ and the row and column constraints in which participates are highlighted.

A staircase code can be seen as a class of *spatially coupled codes* and as such it can be interpreted as spanning over a number of spatial positions, where the code bits in position i correspond to the code bits $[\mathbf{B}_{i-1}^T, \mathbf{B}_i]$, i.e., each spatial position contains two blocks of code bits. We will denote the code

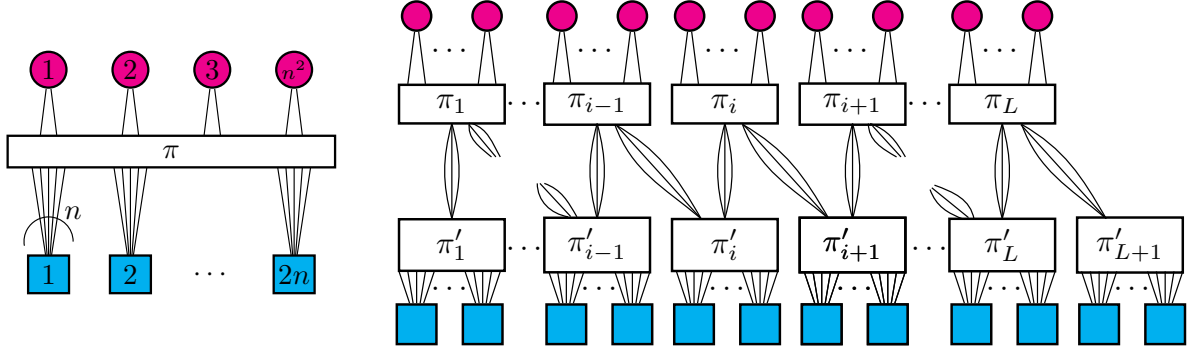


Fig. 2: (Left) Schematic of the GLDPC code ensemble containing PCs with component codes of length n as a special case. Circles and squares correspond VNs and CNs, respectively, and π denotes the interleaver. (Right) The schematic of the SC-GLDPC code ensemble with L spatial positions and $u = 2$ containing staircase codes as a special case. π_i and π'_i represent the random permutations for VNs and CNs, respectively, at spatial position i .

bits in position i by the matrix $\mathbf{C}^{(i)} \triangleq [\mathbf{B}_{i-1}^T, \mathbf{B}_i] = [c_{p,j}^{(i)}]$, of dimensions $\frac{n}{2} \times n$.

As PCs, staircase codes can also be represented by a Tanner graph and as such can be seen as an instance of a SC-GLDPC code ensemble. The Tanner graph of a SC-GLDPC code ensemble of coupling memory u is constructed by placing L copies of a regular GLDPC code of VN degree d_v and CN degree d_c in L spatial positions in the set $\mathcal{L} = \{1, \dots, L\}$. Each spatial position consists of M CNs and N VNs. The L copies are then coupled as follows: each VN at position $i \in \mathcal{L}$ is connected to d_v CNs in positions in the range $[i, \dots, i + u - 1]$, chosen uniformly at random. Likewise, each CN at position $i \in \mathcal{L}$ is connected to d_c randomly chosen VNs at positions in the range $[i - u + 1, \dots, i]$. This chain of coupled codes may be terminated by appending $u - 1$ spatial positions with CNs only at the end of the chain. The Tanner graph of a staircase code is contained in the Tanner graph of a SC-GLDPC code ensemble with VN degree 2, coupling memory $u = 2$, and $N = \frac{n^2}{2}$ degree-2 VNs and $M = n$ degree- n CNs per spatial position. The Tanner graph of such a SC-GLDPC code ensemble, originally considered in [27], is depicted in Fig. 2.

The connection between PCs and GLDPC codes and between staircase codes and SC-GLDPC codes, allows the use of tools for the analysis of codes-on-graphs, such as density evolution, to analyze PCs.

B. Channel Model

We assume transmission over the binary-input additive white Gaussian noise (bi-AWGN) channel. For simplicity, for the definitions below we assume transmission using a PC. The channel output corresponding to code bit $c_{i,j}$ is given by

$$y_{i,j} = x_{i,j} + z_{i,j}, \quad (1)$$

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

For the case of staircase codes, the matrices of LLRs and hard decisions must be defined per each spatial position, i.e., we define $\mathbf{L}^{(i)}$ and $\mathbf{R}^{(i)}$, of dimensions $\frac{n}{2} \times n$, in accordance to $\mathbf{C}^{(i)}$.

Remark: The target application for the proposed decoding algorithm is very high throughput applications. A salient application is next generation fiber-optic links, which will support throughputs up to 1 Tb/s. The optical channel in practical scenarios such as wavelength division multiplexing can be approximated with the well-established Gaussian noise (GN) model [31]. The gist of the GN model is that the interplay between Kerr nonlinearity and chromatic dispersion in the optical channel can be accurately modeled as a Gaussian noise under some mild conditions. Thus, the bi-AWGN channel considered in this paper is also relevant for this particularly interesting application.

C. Bounded Distance Decoding

Consider now the decoding of an arbitrary row or column component code, assuming that the codeword $\mathbf{c} = (c_1, \dots, c_n)$ is transmitted and decoding is based on the hard-detected bits at the channel output, $\mathbf{r} = (r_1, \dots, r_n)$. BDD corrects all error patterns with Hamming weight up to the error-correcting capability of the code $t = \lfloor \frac{d_{\min} - 1}{2} \rfloor$. If the weight of the error pattern is larger than t and there exists another codeword $\tilde{\mathbf{c}} \in \mathcal{C}$ with $d_H(\tilde{\mathbf{c}}, \mathbf{r}) \leq t$, then BDD erroneously decodes \mathbf{r} onto $\tilde{\mathbf{c}}$ and a so-called *mis-correction* occurs. Otherwise, if such codeword does not exist, BDD fails and we use the convention that the decoder outputs \mathbf{r} . Thus, the decoded vector $\hat{\mathbf{r}}$ for BDD can be written as

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

The decoding of product-like codes can then be accomplished in an iterative fashion based on BDD of the component codes and iterating between the row and column decoders. Here, we refer to iterative decoding of product-like codes based on BDD of the component codes as iBDD.

III. ITERATIVE BOUNDED DISTANCE DECODING WITH SCALED RELIABILITY

In this section, we propose a modification of the conventional iBDD of product-like codes. The main idea is to exploit the channel reliabilities in the decoding of the component codes, while only binary messages are exchanged between the component decoders. We refer to this algorithm as iBDD with scaled reliability (iBDD-SR). The proposed algorithm applies to product-like codes in general, including, e.g., PCs, half PCs [32], staircase codes, and braided codes. For illustration purposes, we focus on PCs and staircase codes.

A. Iterative Bounded Distance Decoding with Scaled Reliability for Product Codes

Without loss of generality, assume that decoding starts with the decoding of the row codes and, in particular, consider the decoding of the i th row code at iteration ℓ .

Let $\Psi^{c,(\ell-1)} = [\psi_{i,j}^{c,(\ell-1)}]$ be the matrix of hard decisions on code bits $c_{i,j}$ after the decoding of the n column codes at iteration $\ell - 1$. Row decoding is then performed based on $\Psi^{c,(\ell-1)}$. First, BDD is performed. The output of the BDD stage of the i th row component code corresponding to code bit $c_{i,j}$, denoted by $\bar{\mu}_{i,j}^{r,(\ell)} \in \{\pm 1, 0\}$, takes values on a ternary alphabet, $\bar{\mu}_{i,j}^{r,(\ell)} \in \{\pm 1, 0\}$, where the decoded bits are mapped according to $0 \mapsto +1$ and $1 \mapsto -1$ if BDD is successful, and the output is 0 if a decoding fails.

The reliability information on code bit $c_{i,j}$ is then formed according to

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

where $w_i^{r,(\ell)}$ is a scaling factor corresponding to the reliability of the decision on $c_{i,j}$ at the output of the BDD (it may be different for each iteration) that should be properly optimized. In [26] we optimized the scaling factors for PCs based on simulations. Unfortunately a simulation-based approach becomes infeasible for spatially-coupled product-like codes such as staircase codes. In Section IV-A, we show that the scaling factors can be found via density evolution, which renders the search (and hence the decoder design) feasible.

Finally, the hard decision on code bit $c_{i,j}$ made by the i th row decoder is formed as

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

where ties can be broken with any policy.

The hard decision $\psi_{i,j}^{r,(\ell)}$ is 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 $\Psi^{r,(\ell)} = [\psi_{i,j}^{r,(\ell)}]$ is formed and used as the input for the n column decoders, and column decoding based on $\Psi^{r,(\ell)}$ is performed. As before, we assume that the output of the BDD stage of the j th column component code corresponding to code bit $c_{i,j}$, denoted by $\bar{\mu}_{i,j}^c$, takes values on $\{\pm 1, 0\}$. Then, the hard decision on $c_{i,j}$ made by the j th column decoder at iteration ℓ is formed as

$$\psi_{i,j}^{c,(\ell)} = B(\mu_{i,j}^{c,(\ell)}),$$

where $\mu_{i,j}^{c,(\ell)} = w_j^{c,(\ell)} \cdot \bar{\mu}_{i,j}^c + L_{i,j}$ is the reliability of code bit $c_{i,j}$. After decoding of the n column codes at decoding

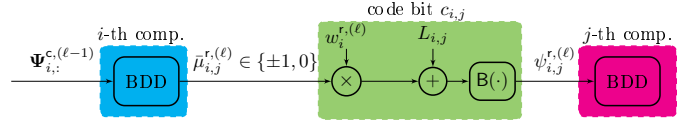


Fig. 3: Block diagram of iBDD-SR for PCs.

iteration ℓ , the matrix $\Psi^{c,(\ell)} = [\psi_{i,j}^{c,(\ell)}]$ is passed to the n row decoders for the next decoding iteration. The iterative process continues until a maximum number of iterations is reached. The iBDD-SR of PCs is schematized in Fig. 3.

The crucial modification in iBDD-SR with respect to conventional iBDD is that the hard decisions passed between component decoders are not simply the result of the BDD of the component codes, but are made on the sum of a scaled version of the output of the BDD decoder and the channel LLR. Therefore, the channel reliabilities are exploited to make the final hard decisions at each row and column decoding stage. Intuitively, since the channel reliability is exploited in the hard decisions at each row and column decoding, in the case that the reliability of the channel is high and conventional iBDD introduces miscorrections, the modified algorithm may combat the possible miscorrections.

B. Iterative Bounded Distance Decoding with Scaled Reliability for Staircase Codes

Similar to PCs, staircase codes can be decoded iteratively. Decoding of staircase codes is typically performed in a sliding-window fashion, iterating between the row and column decoders within a window containing a number of staircase blocks and after a given number of iterations shifting the window by one staircase block [8]. Thus, the iBDD-SR of PCs described in the previous subsection readily extends to staircase codes. However, in this case, the scaling factors may be different for each spatial position, and this needs to be considered when optimizing them.

IV. DENSITY EVOLUTION OF iBDD-SR

In this section, we provide the density evolution analysis for GLDPC codes and SC GLDPC codes, which contain PCs and staircase codes as particular cases for a certain choice of parameters. The analysis provides, as a byproduct, the optimal scaling factors of iBDD-SR (optimal in an asymptotic sense, for infinitely long block length [25]). Note that for finite block length, the scaling factors derived from the density evolution analysis are not necessarily the ones that minimize the error probability. However, selecting the scaling factors based on density evolution avoids resorting to an optimization based on Monte-Carlo simulations, which may be very time-consuming, and is unfeasible for staircase codes. Furthermore, interestingly, comparing the performance of iBDD-SR with scaling factors derived from the density evolution to that with scaling factors found by simulations¹, we have observed that the finite length performance of the former is always slightly better.

¹Note that due to the limitation of resources, a full exhaustive search is infeasible, i.e., only a grid search with a certain accuracy (step size) is possible.

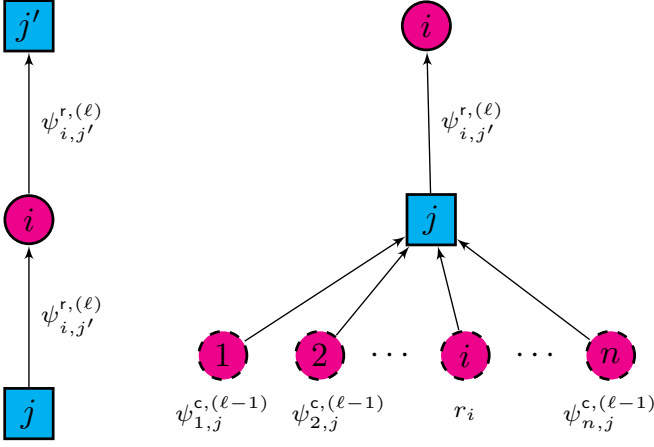


Fig. 4: Block diagram of the message passing algorithm for GLDPC codes. Circles and squares correspond to VNs and CNs, respectively.

As discussed in Section II-A, PCs and staircase codes are contained in a particular GLDPC and SC-GLDPC code ensemble, respectively, whose asymptotic behavior can be rigorously analyzed via density evolution. In [27], the density evolution of iBDD for transmission over the binary symmetric channel (BSC) was derived. In this section, we derive the density evolution of iBDD-SR for GLDPC code ensembles and SC-GLDPC code ensembles by extending the density evolution of conventional iBDD in [27] to the iBDD-SR case. It is worth mentioning that product-like codes are *structured* codes and therefore their decoding threshold is not necessarily fully characterized by the threshold of the corresponding GLDPC code ensemble. In [33], the rigorous density evolution analysis that characterizes the asymptotic decoding performance over the binary erasure channel (BEC) of a deterministic construction of product-like codes that encompasses several classes of product-like codes such as irregular PCs, block-wise braided codes, and staircase codes, was derived. Its extension to the BSC, however, is very cumbersome. In practice, for the BEC the thresholds predicted by the rigorous density evolution and those predicted by the density evolution of the corresponding GLDPC ensemble are virtually identical [33], [34]. Thus, in this paper we opt to derive the density evolution of iBDD-SR for the GLDPC and SC-GLDPC ensembles encompassing PCs and staircase codes.

The GLDPC codes that we consider here can be represented graphically by a Tanner graph where each VN is connected to two CNs, corresponding to one row and one column code (see Section II-A and Fig. 2). We consider the iterative hard decision decoding algorithm based on extrinsic BDD of the component codes proposed in [27]. In particular, at the i -th VN, the input message from the j -th CN is forwarded to the j' -th CN. At the j -th CN, the input corresponding to the i -th VN is replaced by the hard-detected bit r_i at the channel output. This message passing algorithm is illustrated in Fig. 4. Note that substituting the i -th VN message with r_i makes the decoding rule extrinsic, i.e., the decision on the i -th VN does not depend on previous decisions inside the decoder. The extrinsic decoding rule is essential for the density evolution analysis.

A. Density Evolution Analysis of iBDD-SR for GLDPC Code Ensembles

We consider transmission over the bi-AWGN channel and analyze the decoder behavior by assuming the transmission of the all-zero codeword. At the CNs, we assume a length- n binary component code with error-correcting capability t .

We denote the channel output error probability, i.e., the error probability that would be attained by applying a hard detection to the bi-AWGN channel output, as p_{ch} . For an arbitrary row/column BDD stage, we denote by $P^e(i)$ the probability that a randomly selected bit in the component code's codeword is decoded incorrectly when it was initially in error and there are i errors in the other $n-1$ positions, and by $P^c(i)$ the probability that a randomly selected bit in the component code's codeword is decoded correctly when it was initially in error and there are i errors in the remaining $n-1$ positions. The probability that a randomly selected bit in the component code's codeword is erased when it was initially in error and there are i errors in the remaining $n-1$ positions is denoted by $P^e(i)$. We have obviously that $P^e(i) = 1 - P^e(i) - P^c(i)$. Similarly, we denote by $Q^e(i)$, $Q^c(i)$, and $Q^e(i)$ the probability that a randomly selected bit in the component code's codeword is decoded incorrectly, correctly, and erased, respectively, when the bit was initially correct and there are i errors in the remaining $n-1$ positions. Note that $Q^e(i) = 1 - Q^e(i) - Q^c(i)$.

We have that

$$P^e(i) = \begin{cases} 0 & \text{if } 0 \leq i \leq t-1 \\ \sum_{\delta=1}^t \sum_{j=0}^{\delta} \frac{h+1}{n} A_{h+1} F_h^{(1)} & \text{if } t \leq i \leq n-t-2 \\ 1 & \text{otherwise} \end{cases} \quad (5)$$

and

$$Q^c(i) = \begin{cases} 1 & \text{if } 0 \leq i \leq t \\ \sum_{\delta=1}^t \sum_{j=0}^{\delta} \frac{n-h}{n} A_h F_h^{(1)} & \text{if } t+1 \leq i \leq n-t-1 \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

where $h = i - \delta + 2j$ and $F_h^{(1)}$ is defined as

$$F_h^{(1)} \triangleq \frac{\binom{h}{n-j} \binom{n-h-1}{\delta-j}}{\binom{n-1}{i}}. \quad (7)$$

In (5), A_h is the weight enumerator of the component code, which can be tightly approximated as

$$A_h \approx \begin{cases} 2^{-\nu t} \binom{n}{h} (1 + o(1)) & \text{if } 2t+1 \leq h \leq n-2t-1 \\ 1 & \text{if } h=0, h=n \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

Similarly, one can compute $P^c(i)$ and $Q^e(i)$ as

$$P^c(i) = \begin{cases} 1 & \text{if } 0 \leq i \leq t-1 \\ \sum_{\delta=1}^t \sum_{j=0}^{\delta-1} \frac{n-h}{n} A_h F_h^{(2)} & \text{if } t \leq i \leq n-t-2 \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

and

$$Q^e(i) = \begin{cases} 0 & \text{if } 0 \leq i \leq t \\ \sum_{\delta=1}^t \sum_{j=0}^{\delta-1} \frac{h+1}{n} A_{h+1} F_h^{(2)} & \text{if } t+1 \leq i \leq n-t-1 \\ 1 & \text{otherwise} \end{cases} \quad (10)$$

where $h = i - \delta + 2j + 1$ and $F_h^{(2)}$ is defined as

$$F_h^{(2)} \triangleq \frac{\binom{h}{h-j} \binom{n-h-1}{\delta-j-1}}{\binom{n-1}{i}}. \quad (11)$$

For ease of understanding, we briefly explain the derivation of $P^e(i)$ in (5) in the appendix. The derivation of $Q^c(i)$, $P^c(i)$, and $Q^e(i)$ follows a similar reasoning.

For the considered GLDPC code ensemble, CNs are divided into two sets of equal size to capture the serial row/column decoding schedule of PCs. Each set defines a CN type. We refer to the two CN types as row and column CN types. Each VN is connected to one row-type CN and to one column-type CN. Each decoding iteration consists of a row CN elaboration, followed by a column CN elaboration. In the following, we denote by \mathbf{x} the error probability associated to the messages exchanged by the component decoders. In particular, we denote by $\mathbf{x}^{r,(\ell)}$ and $\mathbf{x}^{c,(\ell)}$ the message error probability at the output of the row component decoder (row-type CN) and column component decoder (column-type CN), respectively, at the ℓ th iteration. The message error probability at the input of a row-type CN at the ℓ th iteration is given by $\mathbf{x}^{c,(\ell-1)}$, whereas the message error probability at the input of a column-type CN during the ℓ th iteration is $\mathbf{x}^{r,(\ell)}$. At the first iteration, we have $\mathbf{x}^{c,(0)} = p_{\text{ch}}$, i.e., the input of the row-type CNs is initialized with the channel observations.

In iBDD-SR, the output of the BDD decoder is from a ternary alphabet, i.e., $\bar{\mu}_{i,j}^{r,(\ell)} \in \{\pm 1, 0\}$ (see Section (III-A)), where (under the all-zero codeword assumption) $\bar{\mu}_{i,j}^{r,(\ell)} = 0$, $\bar{\mu}_{i,j}^{r,(\ell)} = -1$, and $\bar{\mu}_{i,j}^{r,(\ell)} = +1$ correspond to erasure (failure), erroneous decoding, and correct decoding, respectively. In the following, the probabilities of $\bar{\mu}_{i,j}^{r,(\ell)} = 0$, $\bar{\mu}_{i,j}^{r,(\ell)} = -1$, and $\bar{\mu}_{i,j}^{r,(\ell)} = +1$ conditioned on $x_{i,j} = +1$ are denoted as $f^e(\mathbf{x}; p_{\text{ch}})$, $f^e(\mathbf{x}; p_{\text{ch}})$, and $f^c(\mathbf{x}; p_{\text{ch}})$, respectively, for a given input message error probability \mathbf{x} and a channel error probability p_{ch} . One can check that

$$f^e(\mathbf{x}; p_{\text{ch}}) = \sum_{i=0}^{n-1} \binom{n-1}{i} \mathbf{x}^i (1-\mathbf{x})^{n-i-1} \cdot (p_{\text{ch}} P^e(i) + \bar{p}_{\text{ch}} Q^e(i)), \quad (12)$$

$$f^c(\mathbf{x}; p_{\text{ch}}) = \sum_{i=0}^{n-1} \binom{n-1}{i} \mathbf{x}^i (1-\mathbf{x})^{n-i-1} \cdot (p_{\text{ch}} P^c(i) + \bar{p}_{\text{ch}} Q^c(i)), \quad (13)$$

$$f^e(\mathbf{x}; p_{\text{ch}}) = \sum_{i=0}^{n-1} \binom{n-1}{i} \mathbf{x}^i (1-\mathbf{x})^{n-i-1} \cdot (p_{\text{ch}} P^e(i) + \bar{p}_{\text{ch}} Q^e(i)), \quad (14)$$

where $\bar{p}_{\text{ch}} \triangleq 1 - p_{\text{ch}}$.

Under the all-zero codeword assumption, one can readily find that $L_{i,j} \sim \mathcal{N}(2/\sigma^2, 4/\sigma^2)$. Note that for the density evolution, as all row CNs and all column CNs are of the same type (i.e., they behave identically), we can assume that $w_i^{r,(\ell)} = w^{r,(\ell)}$ and $w_i^{c,(\ell)} = w^{c,(\ell)}$ for all i . We are now interested in finding $\mathbf{x}^{r,(\ell)} \triangleq p(w^{r,(\ell)} \cdot \bar{\mu}_{i,j}^{r,(\ell)} + L_{i,j} < 0)$ (alternatively $\mathbf{x}^{c,(\ell)} \triangleq p(w^{c,(\ell)} \cdot \bar{\mu}_{i,j}^{c,(\ell)} + L_{i,j} < 0)$), as the iBDD-SR output error probability for the row (column) decoding. To proceed further, one needs to find the distribution of $w^{r,(\ell)} \cdot \bar{\mu}_{i,j}^{r,(\ell)} + L_{i,j}$. The task of finding this distribution is rendered complicated by observing that the messages $\bar{\mu}_{i,j}^{r,(\ell)}$ and $\bar{\mu}_{i,j}^{c,(\ell)}$ are statistically dependent on $L_{i,j}$. In fact, the evolution of the error probability at the output of the CNs discussed so far is based on the approach proposed in [27], where the CN operation is modified (with respect to the classical behavior) to account for the channel observation $r_{i,j}$ when computing the extrinsic terms $\bar{\mu}_{i,j}^{r,(\ell)}$ and $\bar{\mu}_{i,j}^{c,(\ell)}$. In our model, the BDD output message is modified according to the sum $w^{r,(\ell)} \cdot \bar{\mu}_{i,j}^{r,(\ell)} + L_{i,j}$, hence the channel observation for code bit $c_{i,j}$ is used at both the BDD input and output. Here instead of finding directly the distribution of $w^{r,(\ell)} \cdot \bar{\mu}_{i,j}^{r,(\ell)} + L_{i,j}$, we first expand $\mathbf{x}^{r,(\ell)}$ using the auxiliary RV $\hat{L}_{i,j}$ giving the sign of $L_{i,j}$, i.e., $\hat{L}_{i,j} = \text{sign}(L_{i,j})$. Employing Bayes' rule, $\mathbf{x}^{r,(\ell)}$ can be written as

$$\begin{aligned} \mathbf{x}^{r,(\ell)} &= p(\mu_{i,j}^{r,(\ell)} < 0) \\ &= \sum_{\substack{\bar{\mu}_{i,j}^{r,(\ell)} \in \{0, \pm 1\} \\ \hat{L}_{i,j} \in \{\pm 1\}}} p(\mu_{i,j}^{r,(\ell)} < 0 | \bar{\mu}_{i,j}^{r,(\ell)}, \hat{L}_{i,j}) p(\bar{\mu}_{i,j}^{r,(\ell)} | \hat{L}_{i,j}) p(\hat{L}_{i,j}). \end{aligned} \quad (15)$$

It is easy to see that $L_{i,j} \rightarrow \hat{L}_{i,j} \rightarrow \bar{\mu}_{i,j}^{r,(\ell)}$ form a Markov chain, hence, conditioned on $\hat{L}_{i,j}$, $\bar{\mu}_{i,j}^{r,(\ell)}$ is independent of $L_{i,j}$. Using this,

$$\begin{aligned} p(\mu_{i,j}^{r,(\ell)} < 0 | \bar{\mu}_{i,j}^{r,(\ell)}, \hat{L}_{i,j}) &= p(w_i^{r,(\ell)} \cdot \bar{\mu}_{i,j}^{r,(\ell)} + L_{i,j} < 0 | \bar{\mu}_{i,j}^{r,(\ell)}, \hat{L}_{i,j}) \\ &= p(L_{i,j} < -w_i^{r,(\ell)} \cdot \bar{\mu}_{i,j}^{r,(\ell)} | \hat{L}_{i,j}) \end{aligned}$$

One can easily check that $p(L_{i,j} < -w^{r,(\ell)} | \hat{L}_{i,j} = 1) = p(L_{i,j} < 0 | \hat{L}_{i,j} = 1) = 0$ and $p(L_{i,j} < -w^{r,(\ell)} | \hat{L}_{i,j} = -1) =$

$p(L_{i,j} < 0 | \hat{L}_{i,j} = -1) = 1$. Using this in (15) yields

$$\begin{aligned} \mathbf{x}^{r,(\ell)} = & p(0 < L_{i,j} < w^{r,(\ell)})p(\bar{\mu}_{i,j}^{r,(\ell)} = -1 | \hat{L}_{i,j} = 1) + \\ & p(L_{i,j} < -w^{r,(\ell)})p(\bar{\mu}_{i,j}^{r,(\ell)} = 1 | \hat{L}_{i,j} = -1) + \\ & (1 - p(\bar{\mu}_{i,j}^{r,(\ell)} = 1 | \hat{L}_{i,j} = -1))p(L_{i,j} = -1). \end{aligned} \quad (16)$$

Further, $p(\bar{\mu}_{i,j}^{r,(\ell)} = -1 | \hat{L}_{i,j} = 1)$ and $p(\bar{\mu}_{i,j}^{r,(\ell)} = -1 | \hat{L}_{i,j} = -1)$ can be obtained based on $Q^e(i)$ and $P^c(i)$ as

$$\begin{aligned} f^{Q^e}(\mathbf{x}) & \triangleq p(\bar{\mu}_{i,j}^{r,(\ell)} = -1 | \hat{L}_{i,j} = 1) \\ & = \sum_{i=0}^{n-1} \binom{n-1}{i} \mathbf{x}^i (1-\mathbf{x})^{n-i-1} \cdot Q^e(i), \end{aligned} \quad (17)$$

$$\begin{aligned} f^{P^c}(\mathbf{x}) & \triangleq p(\bar{\mu}_{i,j}^{r,(\ell)} = 1 | \hat{L}_{i,j} = -1) \\ & = \sum_{i=0}^{n-1} \binom{n-1}{i} \mathbf{x}^i (1-\mathbf{x})^{n-i-1} \cdot P^c(i). \end{aligned} \quad (18)$$

Finally, by substituting (17) and (18) in (16) and using the Gaussian distribution of $L_{i,j}$ to compute $p(0 < L_{i,j} < w^{r,(\ell)})$ and $p(L_{i,j} < -w^{r,(\ell)})$, we can track the evolution of the message error probabilities as

$$\begin{aligned} \mathbf{x}^{r,(\ell)} = & f^{Q^e}(\mathbf{x}^{c,(\ell-1)}) \cdot \left(Q\left(\frac{1}{\sigma} - \frac{\sigma w^{r,(\ell)}}{2}\right) - p_{\text{ch}} \right) \\ & + f^{P^c}(\mathbf{x}^{c,(\ell-1)}) \cdot Q\left(\frac{1}{\sigma} + \frac{\sigma w^{r,(\ell)}}{2}\right) + (1 - f^{P^c}(\mathbf{x}^{c,(\ell-1)}))p_{\text{ch}} \end{aligned} \quad (19)$$

and

$$\begin{aligned} \mathbf{x}^{c,(\ell)} = & f^{Q^e}(\mathbf{x}^{r,(\ell)}) \cdot \left(Q\left(\frac{1}{\sigma} - \frac{\sigma w^{c,(\ell)}}{2}\right) - p_{\text{ch}} \right) \\ & + f^{P^c}(\mathbf{x}^{r,(\ell)}) \cdot Q\left(\frac{1}{\sigma} + \frac{\sigma w^{c,(\ell)}}{2}\right) + (1 - f^{P^c}(\mathbf{x}^{r,(\ell)}))p_{\text{ch}}. \end{aligned} \quad (20)$$

where $Q(\cdot) \triangleq \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{\xi^2}{2}} d\xi$ is the familiar tail distribution function of the standard Gaussian distribution and $p_{\text{ch}} = Q(\frac{1}{\sigma})$.

One can numerically search for the optimal scaling factors $w^{c,(\ell)}$ and $w^{r,(\ell)}$ in the sense of minimizing $\mathbf{x}^{r,(\ell)}$ and $\mathbf{x}^{c,(\ell)}$, respectively. Alternatively, by neglecting the statistical dependence between $\bar{\mu}_{i,j}^{r,(\ell)}$ and $L_{i,j}$ one can approximate $w^{r,(\ell)}$ and $w^{c,(\ell)}$ as the LLR of the output of a binary error and erasure channel with error probability f^e and erasure probability f^c , given as

$$w^{r,(\ell)} = \log\left(\frac{f^c(\mathbf{x}^{c,(\ell-1)}; p_{\text{ch}})}{f^e(\mathbf{x}^{c,(\ell-1)}; p_{\text{ch}})}\right) \quad (21)$$

and

$$w^{c,(\ell)} = \log\left(\frac{f^c(\mathbf{x}^{r,(\ell)}; p_{\text{ch}})}{f^e(\mathbf{x}^{r,(\ell)}; p_{\text{ch}})}\right). \quad (22)$$

Employing (21) and (22) yields very similar scaling factors as the ones obtained performing a numerical search. Furthermore, the code threshold given by the scaling factors in (21) and (22) is roughly the same as the one computed based on the numerically optimized scaling factors. Therefore, (21) and

(22) provide a good approximation for $w^{r,(\ell)}$ and $w^{c,(\ell)}$, respectively. The obtained scaling factors can then be used in (3) to implement iBDD-SR for a particular finite-length PC.

B. Density Evolution Analysis of iBDD-SR for SC-GLDPC Code Ensembles

The density evolution for GLDPC codes derived in the previous subsection can be readily extended to SC-GLDPC codes with smoothing parameter u . For SC-GLDPC codes, we need to track the probabilities of the messages exchanged in the iterative decoding for each spatial position. Let $\mathbf{x}_a^{(\ell)}$ be the average bit error probability from VNs at spatial position a to the connected CNs at spatial positions $[a, a+u-1]$. Also, let $\mathbf{x}_a^{c,(\ell)}$ be the average bit error probability from CNs at spatial position a to connected VNs at positions $[a-u+1, a]$. $\mathbf{x}_a^{(\ell)}$ and $\mathbf{x}_a^{c,(\ell)}$ can be calculated as

$$\mathbf{x}_a^{c,(\ell)} = \frac{1}{u} \sum_{g=0}^{u-1} \mathbf{x}_{a-g}^{(\ell)}, \quad (23)$$

$$\begin{aligned} \mathbf{x}_a^{(\ell+1)} = & \frac{1}{u} \sum_{g=0}^{u-1} \left(Q\left(\frac{1}{\sigma} - \frac{\sigma w_{a+g}^{(\ell)}}{2}\right) - p_{\text{ch}} \right) f^{Q^e}(\mathbf{x}_{a+g}^{c,(\ell)}) + \\ & Q\left(\frac{1}{\sigma} + \frac{\sigma w_{a+g}^{(\ell)}}{2}\right) f^{P^c}(\mathbf{x}_{a+g}^{c,(\ell)}) + \\ & (1 - f^{P^c}(\mathbf{x}_{a+g}^{c,(\ell)}))p_{\text{ch}}. \end{aligned} \quad (24)$$

We consider $u = 2$ to account for the ensemble containing staircase codes. Thus, combining (23) and (24), we obtain

$$\mathbf{x}_a^{(\ell+1)} = \frac{\mathbf{x}_{a,0}^{(\ell+1)} + \mathbf{x}_{a,1}^{(\ell+1)}}{2}, \quad (25)$$

where $\mathbf{x}_{a,0}^{(\ell+1)}$ and $\mathbf{x}_{a,1}^{(\ell+1)}$ are given as

$$\begin{aligned} \mathbf{x}_{a,0}^{(\ell+1)} = & \left(Q\left(\frac{1}{\sigma} - \frac{\sigma w_a^{(\ell)}}{2}\right) - p_{\text{ch}} \right) f^{Q^e}\left(\frac{\mathbf{x}_a^{(\ell)} + \mathbf{x}_{a-1}^{(\ell)}}{2}\right) + \\ & Q\left(\frac{1}{\sigma} + \frac{\sigma w_a^{(\ell)}}{2}\right) f^{P^c}\left(\frac{\mathbf{x}_a^{(\ell)} + \mathbf{x}_{a-1}^{(\ell)}}{2}\right) + \\ & (1 - f^{P^c}\left(\frac{\mathbf{x}_a^{(\ell)} + \mathbf{x}_{a-1}^{(\ell)}}{2}\right))p_{\text{ch}}, \end{aligned} \quad (26)$$

$$\begin{aligned} \mathbf{x}_{a,1}^{(\ell+1)} = & \left(Q\left(\frac{1}{\sigma} - \frac{\sigma w_{a+1}^{(\ell)}}{2}\right) - p_{\text{ch}} \right) f^{Q^e}\left(\frac{\mathbf{x}_a^{(\ell)} + \mathbf{x}_{a+1}^{(\ell)}}{2}\right) + \\ & Q\left(\frac{1}{\sigma} + \frac{\sigma w_{a+1}^{(\ell)}}{2}\right) f^{P^c}\left(\frac{\mathbf{x}_a^{(\ell)} + \mathbf{x}_{a+1}^{(\ell)}}{2}\right) + \\ & (1 - f^{P^c}\left(\frac{\mathbf{x}_a^{(\ell)} + \mathbf{x}_{a+1}^{(\ell)}}{2}\right))p_{\text{ch}}. \end{aligned} \quad (27)$$

Similar to the GLDPC code ensemble, $w_a^{(\ell)}$ and $w_{a+1}^{(\ell)}$ can be obtained as

$$w_a^{(\ell)} = \log\left(\frac{f_n^c\left(\frac{\mathbf{x}_a^{(\ell)} + \mathbf{x}_{a-1}^{(\ell)}}{2}; p_{\text{ch}}\right)}{f_n^e\left(\frac{\mathbf{x}_a^{(\ell)} + \mathbf{x}_{a-1}^{(\ell)}}{2}; p_{\text{ch}}\right)}\right), \quad (28)$$

$$w_{a+1}^{(\ell)} = \log \left(\frac{f_n^c \left(\frac{x_a^{(\ell)} + x_{a+1}^{(\ell)}}{2}; p_{\text{ch}} \right)}{f_n^e \left(\frac{x_a^{(\ell)} + x_{a+1}^{(\ell)}}{2}; p_{\text{ch}} \right)} \right). \quad (29)$$

We consider the decoding of staircase codes based on the sliding-window operation. To account for the effect of window decoding, we assume that the width of the window is U , i.e., the window contains U spatial positions. Let \mathcal{W}_U be the set containing the indices of the spatial positions in the current window. In window decoding, the decoder freezes the messages coming from the VNs and CNs outside the window, i.e., the VNs and CNs inside the window are updated based on the information exchanged inside the window and no information comes from the positions outside it. In the particular case of staircase codes, which are contained in the ensemble of SC-GLDPC codes with $u = 2$, the first and last positions inside the window do not get any information from the positions outside it. Therefore, one can define $\tilde{x}_a^{(\ell)}$ as

$$\tilde{x}_a^{(\ell)} = \begin{cases} 0 & \text{if } a \notin \mathcal{W}_U \\ x_a^{(\ell)} & \text{if } a \in \mathcal{W}_U \end{cases}, \quad (30)$$

and use it in (25)–(29) to find the average bit error probability for the spatial positions within the window.

V. COMPLEXITY CONSIDERATIONS

A thorough complexity analysis of the iBDD-SR algorithm requires delving into the hardware implementation in order to address aspects such as data bus requirements, impact on the degree of parallelism, etc., that go beyond the scope of this paper. We refer the interested reader to [35], where an efficient architecture for iBDD-SR of PCs is presented. In the following, we provide a high-level discussion of the decoding complexity. In particular, we estimate the additional resources required in terms of memory with respect to iBDD for both PCs and staircase codes. We also compare the complexity of iBDD-SR with that of AD [22] in terms of memory requirements.

Consider a PC with BCH component code (n, k, d_{\min}) of error correcting capability t decoded over ℓ_{\max} iterations. iBDD-SR requires some extra memory compared to iBDD. With reference to (3), the decision on code bit $c_{i,j}$ at iteration ℓ can be divided into two cases: i) $\psi_{i,j}^{r,(\ell)} = B(L_{i,j})$ if $w_i^{r,(\ell)} < |L_{i,j}|$ or BDD fails (i.e., $\bar{\mu}_{i,j}^{r,(\ell)} = 0$); and ii) $\psi_{i,j}^{r,(\ell)} = B(\bar{\mu}_{i,j}^{r,(\ell)})$ if BDD is successful and $w_i^{r,(\ell)} > |L_{i,j}|$. By storing the channel LLRs $(L_{i,j})$ and $w_i^{r,(\ell)}$, one can implement (3) with a simple logic comparison. Using n_q bits for quantization of $L_{i,j}$ and $w_i^{r,(\ell)}$, the additional memory required for iBDD-SR compared to iBDD is $(\ell_{\max} + n^2)n_q \approx n^2 n_q$ bits.²

Consider now a staircase code with an (even length) BCH component code decoded using a sliding window of size U . We remark that the corresponding SC-GLDPC code ensemble comprises $U - 1$ spatial positions, i.e., the window size for the corresponding SC-GLDPC ensemble is $U - 1$. In general, the scaling factors may be different for each spatial position

(and vary also with the number of iterations). This could be impractical, as many scaling factors would be required to be stored. Interestingly, as discussed in Section VI, the density evolution for SC-GLDPC code ensembles shows that the scaling factors converge to certain values after few $(U - 2)$ window slides, after which the scaling factors are identical for each window. Thus few scaling factors need to be stored. Therefore, the additional memory required for the decoding of one staircase block in iBDD-SR compared to iBDD is $((U - 2)(U - 1)\ell_{\max} + \frac{n^2}{4})n_q \approx \frac{n^2 n_q}{4}$.

We remark that the required memory for both PCs and staircase codes is static, i.e., no switching activities are involved, as the LLRs and scaling factors are not updated in the iterative decoding. This means that the additional memory has a limited cost in terms of energy consumption for the decoder [35], [36].

The direct complexity comparison with AD algorithm is a non trivial task, as AD can be implemented in various ways (see [22, Remark 7]). In a high level view, AD assigns a flag to each component code, which indicates the corresponding status in decoding. Furthermore, AD reduces the decoding conflicts between component decoders by keeping track of the conflict locations and preventing the bit flips on the most trusted component codes, called anchors. As the anchors can also be miscorrected, AD allows to backtrack the decisions on anchors. Overall, a memory in the order of $4n$ bits is required for storing the status of the component codes. Also $4nt(\lceil \log_2 n \rceil + 1)$ bits are required for storing the location of conflicts and backtracking procedure (see [22, Sec. VI. B] for more details). Unlike the static memory of iBDD-SR, the memory required for AD is dynamic, as the status of component codewords and the location of errors changes during the iterative decoding. The dynamic memory is usually significantly more costly in hardware implementation compared to the static counterpart [36].

Note that in iBDD-SR only binary messages are exchanged between component decoders, hence the internal decoder data flow is the same as that of iBDD. Thus, iBDD-SR is particularly interesting for very high-throughput applications, for which the internal decoder data flow is a limiting factor.

VI. NUMERICAL RESULTS

To evaluate the performance of iBDD-SR, we simulate the transmission of two PCs with (255,231,3) BCH component codes and (511,484,3) BCH component codes and two staircase codes with the same component codes shortened by one bit (i.e., (254,230,3) and (510,483,3)), over the bi-AWGN channel.⁴ The code rate of the resulting product codes is $R = 0.820$ and 0.897 for $n = 255$ and $n = 511$, respectively. The code rate of the staircase codes is $R = 0.811$ and 0.894 for $n = 254$ and $n = 510$, respectively. For the sake of comparison, we also simulate the performance of the codes under iBDD, ideal iBDD (i.e., a genie-aided

³Note that the typical window size is in the range of 4–7 blocks, hence $(U - 2)(U - 1)\ell_{\max} \ll \frac{n^2}{4}$.

⁴We are particularly interested in the performance of codes based on BCH component codes with $t = 3$, since their decoders can be efficiently implemented via lookup tables [8, Appendix I].

²Note that $\ell_{\max} \ll n^2$.

Table I: Comparison of different decoding algorithms for PCs and staircase codes with (255,231,3) and (254,230,3) BCH component codes, respectively. The code rate of the PC and the staircase code is 0.820 and 0.811, respectively. The E_b/N_0 , coding gains, and corresponding capacity gaps are measured at $\text{BER} = 10^{-6}$ from the simulations. The values for staircase codes are provided within parenthesis.

decoding algorithm	channel reliabilities	exchanged messages	E_b/N_0 [dB]	gain over iBDD [dB]	capacity [dB]	gap from capacity [dB]
iBDD	no	hard	4.62 (4.52)	-	3.54 (3.46) (HD)	1.08 (1.06)
AD [22]	no	hard	4.43 (4.25)	0.21 (0.27)	3.54 (3.46) (HD)	0.89 (0.79)
iBDD-SR	yes	hard	4.34 (4.21)	0.29 (0.31)	2.23 (2.14) (SD)	2.11 (2.07)
ideal iBDD	no	hard	4.31 (4.19)	0.33 (0.33)	3.54 (3.46) (HD)	0.77 (0.73)

Table II: Comparison of different decoding algorithms for PCs and staircase codes with (511,484,3) and (510,483,3) BCH component codes, respectively. The code rate of the PC and the staircase code is 0.897 and 0.894, respectively. The E_b/N_0 , coding gains, and corresponding capacity gaps are measured at $\text{BER} = 10^{-6}$ from the simulations. The values for staircase codes are provided within parenthesis.

decoding algorithm	channel reliabilities	exchanged messages	E_b/N_0 [dB]	gain over iBDD [dB]	capacity [dB]	gap from capacity [dB]
iBDD	no	hard	5.18 (5.06)	-	4.36 (4.32) (HD)	0.82 (0.74)
AD [22]	no	hard	5.01 (4.86)	0.17 (0.21)	4.36 (4.32) (HD)	0.65 (0.54)
iBDD-SR	yes	hard	4.93 (4.80)	0.24 (0.265)	3.15 (3.11) (SD)	1.78 (1.69)
ideal iBDD	no	hard	4.92 (4.79)	0.26 (0.27)	4.36 (4.32) (HD)	0.56 (0.47)

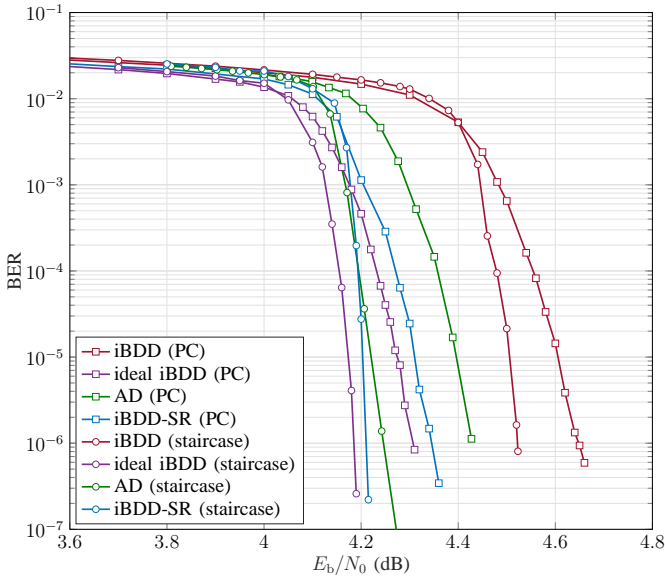


Fig. 5: Performance of iBDD, ideal iBDD, AD, and iBDD-SR for a PC with component code (255,231,3) and a staircase code with component code (254,230,3).

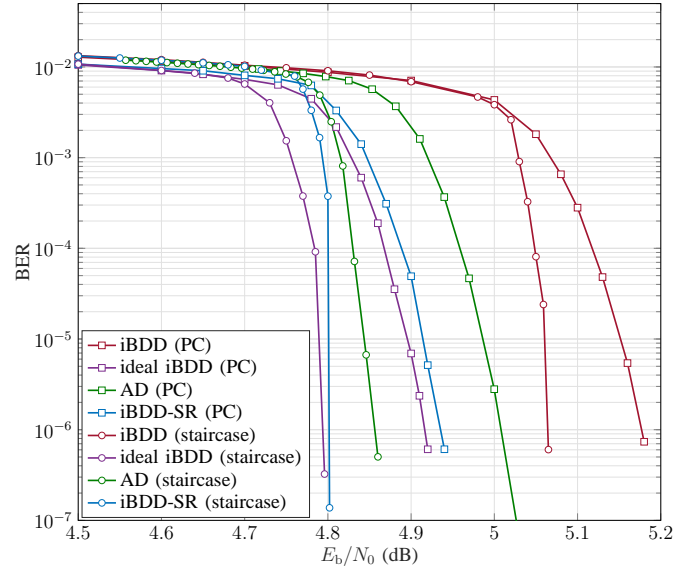


Fig. 6: Performance of iBDD, ideal iBDD, AD, and iBDD-SR for a PC with component code (511,484,3) and a staircase code with component code (510,483,3).

iBDD where misscorrections are avoided by providing the component decoder input at its output whenever the error correction capability of the component code is exceeded), and AD. The BER performance of the considered codes is shown in Figs. 5-6.

It is important to remark that if the channel LLRs are highly reliable but with wrong sign, one can expect that the decoding rule in (3)–(4) will be unable to recover from these errors. In this situation, although $\bar{\mu}_{i,j}^r$ may correspond to a correct decision, it is overridden by the channel, i.e., the hard decision on code bit $c_{i,j}$ made by the i -th row decoder, $\psi_{i,j}^{r,(\ell)}$, becomes $\psi_{i,j}^{r,(\ell)} = B(w_i^{r,(\ell)} \cdot \bar{\mu}_{i,j}^{r,(\ell)} + L_{i,j}) = B(L_{i,j})$ (cf. (3) and (4)), which leads to an erroneously decoded bit. Therefore, one needs to be careful when applying iBDD-SR to avoid the appearance of an error floor. In particular, to avoid such errors and the presence of a high error floor, we run iBDD-SR 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. For PCs, we consider a maximum of 10 iBDD-SR iterations followed by 2 conventional iBDD iterations. For a fair comparison, for iBDD, ideal iBDD, and AD, we use a maximum of 12 iterations. For staircase codes, we use a window decoder with window size of 7 blocks and a maximum of 10 iBDD-SR iterations followed by 2 conventional iBDD iterations.

Table I summarizes the gains of iBDD-SR, ideal iBDD, and AD over conventional iBDD (fifth column) and the gap to the corresponding capacity (sixth column) at the BER of 10^{-6} for both PCs and staircase codes with BCH component codes of

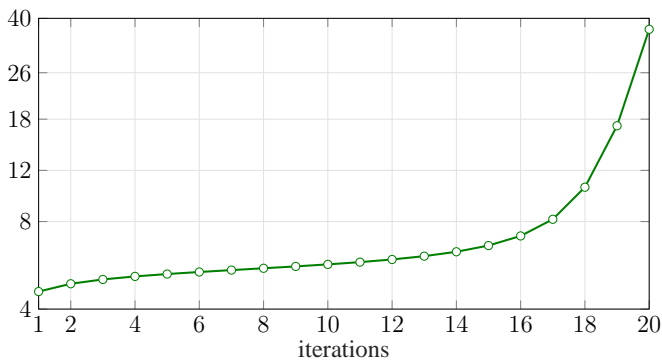


Fig. 7: The evolution of scaling factors for the GLDPC code ensemble with BCH component code (255,231,3) over 20 iterations.

parameters (255,231,3) and (254,230,3), respectively. Whether the decoder exploits the channel reliabilities and the nature of the messages exchanged in the iterative decoding (hard or soft) is indicated in the third and fourth column, respectively. iBDD-SR yields a performance gain of 0.29 dB and 0.31 dB with respect to iBDD for the PC and the staircase code, respectively. For very high-throughput applications such as fiber-optic communications such gains are significant. Furthermore, one can see that iBDD-SR performs close to the ideal iBDD for both PCs and staircase codes. Interestingly, the staircase code with iBDD-SR outperforms the PC with ideal iBDD by 0.1 dB at a BER of 10^{-6} .

In Table II, we give the gains of iBDD-SR, ideal iBDD, and AD over conventional iBDD and the gap to the corresponding capacity at the BER of 10^{-6} for both PCs and staircase codes with BCH component codes of parameters (511,484,3) and (510,483,3), respectively. Compared to the shorter block length, one can see that iBDD-SR yields similar gains with respect to iBDD, albeit slightly smaller. Also, the gap between iBDD-SR and ideal iBDD reduces even further. In particular, for the staircase code iBDD-SR performs almost the same as mis-correction-free decoder. Furthermore, the gap to capacity for all decoders is also reduced.

From the GN model, one can see that an optical link SNR improvement of a dB yields a dB of optical reach enhancement [37, Eq. 59]. Thus, the 0.3 dB performance improvement of iBDD-SR over iBDD yields about 7.2% of reach enhancement. We also remark that for extended BCH component codes with $t = 2$, not reported here, the gain of iBDD-SR with respect to iBDD is slightly larger and iBDD-SR performs almost identical as conventional iBDD (see Fig. 3 in [30] for a curve).

In Fig. 7, we show the evolution of the scaling factors resulting from the density evolution for the GLDPC code ensemble with BCH component code (255,231,3) as a function of the number of *half* iterations, where each iteration corresponds to one row or column decoding step. The scaling factors in Fig. 7 are obtained at the decoding threshold, which is 4.18 dB. As can be seen, the scaling factors are monotonically increasing over iterations. This is expected, as the reliability of the BDD output increases with the number of iterations. In Fig. 8, we show the evolution of the scaling factors for the SC-GLDPC code ensemble with the same component code. In the figure,

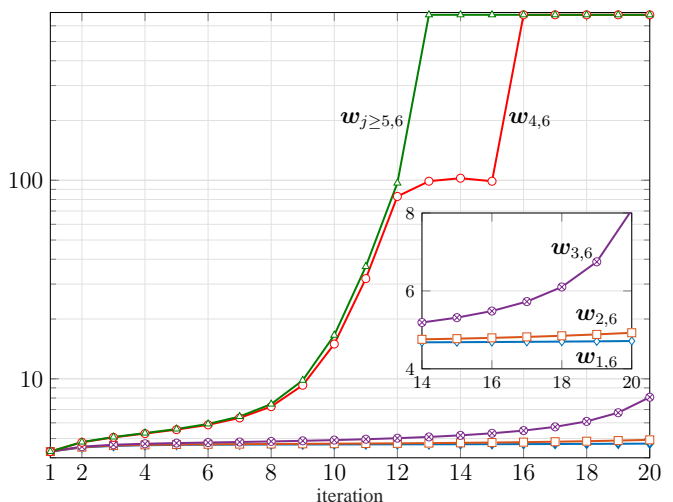


Fig. 8: The evolution of scaling factors for the SC-GLDPC code ensemble with BCH component code (255,231,3) over 20 iterations, using a window decoder of window size 6, for the 6-th spatial position within the window.

$w_{a,j}$ corresponds to the vector of scaling factors corresponding to the j -th position inside the decoding window (of size U) containing spatial positions $[a, \dots, a + U - 1]$. In particular, the figure plots the scaling factors obtained by the density evolution (at the code threshold, i.e., 4.05 dB) for the 6-th spatial position inside the windows. Note that after 5 ($U - 1$) window slides, the scaling factors converge to a given value. The same phenomenon has been observed for other spatial positions within the window and other ensembles. Thus, from a practical viewpoint, it is not necessary to store the scaling factors for every spatial position, but only for a limited number of positions.

It is important to remark that the density evolution in Section IV-A assumes the exchange of extrinsic information between component decoders. However, *extrinsic* message passing decoding of product-like codes (as explained in Section IV-A and [27] for iBDD-SR and iBDD, respectively) is complex, as it requires that the component codes are decoded n times in each iteration. In practice, when very high throughputs are required, product-like codes are hence decoded using conventional iterative row/column decoding of the component codes, i.e., *intrinsic* message passing, and this is the algorithm that we used for the simulations. Thus, the scaling factors are obtained for a slightly different decoder than the one used in practice. One may then wonder how good the scaling factors derived for the extrinsic message passing are for the conventional decoder. To verify the impact of using the scaling factors found from the density evolution for finite-length codes with conventional row/column decoding, we also performed a search based on Monte-Carlo simulations to optimize the scaling factors for PCs, with a grid search of step size of 0.01. Interestingly, the performance of iBDD-SR with scaling factors from the density evolution yields slightly better results (probably due to the non-exhaustive limited Monte-Carlo search), supporting that their derivation using the density evolution is very useful in practice. Also, we would like to stress that the Monte-Carlo search is not feasible for staircase codes, thus the derived density evolution is crucial

in the design of iBDD-SR in this case.

VII. CONCLUSION

We proposed iterative bounded distance decoding with scaled reliability, a new decoding algorithm for the decoding of product-like codes. The proposed algorithm, based on BDD of the component codes, exploits the channel reliabilities but, notably, is a binary message passing algorithm, i.e., the component decoders exchange only hard decisions. The proposed algorithm improves the performance of conventional iBDD, with the same decoder data flow, at the expense of a minor increase in complexity. For two particular PCs and staircase codes built from BCH component code codes (255, 231, 3) and (511, 484, 3) (shortened by one bit for staircase codes), the proposed algorithm outperforms conventional iBDD by 0.24–0.31 dB and yields performance very close to that of ideal iBDD without miscorrections. For a PC with (255, 231, 3) BCH component codes, in [35] we implemented iBDD-SR in a 28nm process technology, achieving 1 Tb/s with an area and energy dissipation less than half of that of staircase decoders, at similar estimated net coding gains.

The proposed algorithm is appealing for applications requiring very high throughputs such as fiber-optic communications.

APPENDIX

We clarify the derivation of $P^e(i)$. Recall that $P^e(i)$ corresponds to the probability that BDD results in a codeword (within Hamming distance t of the input vector, see (2)) that has an error in the randomly selected position, given that the input vector has an error in that position and contains i errors in the other $n-1$ positions. One can easily see that if $0 \leq i \leq t-1$, the total number of errors is less than or equal to t , meaning that the decoder is able to correct the codeword, therefore $P^e(i) = 0$. On the other hand, if $n-t-1 \leq i \leq n-1$, the Hamming distance between the input vector and the all-one codeword is less than or equal to t , therefore BDD decodes onto the all-one codeword and the randomly selected position will always be in error, i.e., $P^e(i) = 1$.

The nontrivial case corresponds to $t \leq i \leq n-t-2$. In this case, we need to compute the probability that there exists a codeword \mathbf{c} that has a one in the randomly-chosen position and is within Hamming distance t of the weight- $(i+1)$ vector at the input of the bounded distance decoder, which we denote by \mathbf{r} . To do so, we can exploit the weight enumerator of the BCH code, A_h , and consider the equivalent problem of computing the probability that, given a codeword \mathbf{c} of a given weight, the randomly selected bit for the input vector corresponds to an entry where \mathbf{c} is one and the i other ones of the input vector are placed such that $d_H(\mathbf{c}, \mathbf{r}) \leq t$.

We proceed as follows. Consider codewords of weight $h+1$, the total number of which is A_{h+1} . For a given codeword of weight $h+1$, the probability that the randomly selected erroneous bit is chosen among the codeword bit positions that are one is $\frac{h+1}{n}$. Now, assume that the input vector has $h-j$ ones in $h-j$ out of the h entries (one entry is already fixed) where the given codeword has ones. Thus, the input vector has $i-(h-j)$ ones in $i-(h-j)$ out of the $n-h-1$ entries

where the given codeword is zero. The number of possibilities is

$$\binom{h}{h-j}$$

and

$$\binom{n-h-1}{i-(h-j)} = \binom{n-h-1}{\delta-j},$$

respectively, where we defined $\delta \triangleq i-h+2j$ for convenience. Thus, the probability that this occurs is

$$\frac{\binom{h}{h-j} \binom{n-h-1}{\delta-j}}{\binom{n-1}{i}},$$

which we defined as $F_h^{(1)}$ in (7).

Finally, we need to sum over all cases such that \mathbf{r} and the candidate codeword \mathbf{c} are within Hamming distance t . Note that $d_H(\mathbf{c}, \mathbf{r}) = j + (i - (h-j)) = i - h + 2j = \delta$ and we need that $\delta \leq t$. Thus, we need to sum over $\delta = 1, \dots, t$ and subsequently $j = 0, \dots, \delta$, which results in the expression in (5).

ACKNOWLEDGMENT

The authors would like to thank Dr. Christian Häger and Prof. Henry Pfister for fruitful discussions and providing the simulation results of AD in Figs. 5–6.

REFERENCES

- [1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Geneva, Switzerland, May 1993, pp. 1064–1070.
- [2] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.
- [3] J. Justesen, K. J. Larsen, and L. A. Pedersen, "Error correcting coding for OTN," *IEEE Commun. Magazine*, vol. 48, no. 9, pp. 70–75, Sep. 2010.
- [4] 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.
- [5] 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.
- [6] 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.
- [7] 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.
- [8] 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.
- [9] A. Sheikh, A. Graell i Amat, and G. Liva, "Achievable information rates for coded modulation with hard decision decoding for coherent fiber-optic systems," *IEEE/OSA J. Lightw. Technol.*, vol. 35, no. 23, pp. 5069–5078, Dec 2017.
- [10] P. Elias, "Error-free coding," *Trans. IRE Professional Group on Inf. Theory*, vol. 4, no. 4, pp. 29–37, Sep. 1954.
- [11] N. Abramson, "Cascade decoding of cyclic product codes," *IEEE Trans. Commun. Tech.*, vol. 16, no. 3, pp. 398–402, Jun. 1968.

- [12] R. M. Pyndiah, "Near-optimum decoding of product codes: block turbo codes," *IEEE Trans. Commun.*, vol. 46, no. 8, pp. 1003–1010, Aug. 1998.
- [13] A. Al-Dweik, S. L. Goff, and B. Sharif, "A hybrid decoder for block turbo codes," *IEEE Trans. Commun.*, vol. 57, no. 5, May 2009.
- [14] P. Lu, E. Lu, and T. Chen, "An efficient hybrid decoder for block turbo codes," *IEEE Commun. Lett.*, vol. 18, no. 12, pp. 2077–2080, Dec. 2014.
- [15] H. Mukhtar, A. Al-Dweik, and A. Shami, "Turbo product codes: Applications, challenges, and future directions," *IEEE Commun. Surveys Tutorials*, vol. 18, no. 4, pp. 3052–3069, 2016.
- [16] A. Al-Dweik, H. Mukhtar, E. Alsusa, and J. Dias, "Ultra-Light decoder for turbo product codes," *IEEE Commun. Lett.*, vol. 22, no. 3, pp. 446–449, Mar. 2018.
- [17] B. Ahn, S. Yoon, and J. Heo, "Low complexity syndrome-based decoding algorithm applied to block turbo codes," *IEEE Access*, vol. 6, pp. 26 693–26 706, 2018.
- [18] B. Li, K. J. Larsen, D. Zibar, and I. T. Monroy, "Over 10 dB net coding gain based on 20% overhead hard decision forward error correction in 100G optical communication systems," in *Proc. Eur. Conf. Opt. Commun. (ECOC)*, Geneva, Switzerland, Sep. 2011.
- [19] 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)*, Atlanta, GA, USA, Dec. 2013, pp. 2376–2381.
- [20] 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.
- [21] "Forward error correction for high bit-rate DWDM submarine systems," ITU-T Recommendation G.975.1, 2004.
- [22] 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.
- [23] 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.
- [24] 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.
- [25] 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.
- [26] 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.
- [27] 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.
- [28] 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 Inf. Proc. (ISTC)*, Hong Kong, Dec. 2018.
- [29] Y. Lei, A. Alvarado, B. Chen, X. Deng, Z. Cao, J. Li, and K. Xu, "Decoding staircase codes with marked bits," in *Proc. Int. Symp. Turbo Codes & Iterative Inf. Proc. (ISTC)*, Hong Kong, Dec. 2018.
- [30] A. Sheikh, A. Graell i Amat, and G. Liva, "Binary message passing decoding of product codes based on generalized minimum distance decoding," in *Proc. 53rd Annu. Conf. Inf. Sciences and Systems (CISS)*, Baltimore, MD, USA, Mar. 2019.
- [31] P. Poggiolini, "The GN model of non-linear propagation in uncompensated coherent optical systems," *IEEE/OSA J. Lightw. Technol.*, vol. 30, no. 24, pp. 3857–3879, Dec. 2012.
- [32] J. Justesen, "Performance of Product Codes and Related Structures with Iterated Decoding," *IEEE Trans. Commun.*, vol. 59, no. 2, pp. 407–415, Feb. 2011.
- [33] 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.
- [34] C. Häger, A. Graell i Amat, H. D. Pfister, A. Alvarado, F. Brännström, and E. Agrell, "On parameter optimization for staircase codes," in *Proc. Optical Fiber Commun. Conf. (OFC)*, Los Angeles, CA, USA, 2015.
- [35] C. Fougstedt, A. Sheikh, A. Graell i Amat, G. Liva, and P. Larsson-Edefors, "Energy-efficient soft-assisted product decoders," in *Proc. Optical Fiber Commun. Conf. (OFC)*, San Diego, CA, USA, Mar. 2019.
- [36] C. Fougstedt and P. Larsson-Edefors, "Energy-efficient high-throughput staircase decoders," in *Proc. Optical Fiber Commun. Conf. (OFC)*, San Diego, CA, USA, Mar. 2018.
- [37] P. Poggiolini, G. Bosco, A. Carena, V. Curri, Y. Jiang, and F. Forghieri, "The GN-model of fiber non-linear propagation and its applications," *IEEE/OSA J. Lightw. Technol.*, vol. 32, no. 4, pp. 694–721, Feb. 2014.