Improved decoding of staircase codes

Improved Decoding of Staircase Codes: The Soft-Aided Bit-Marking (SABM) Algorithm

Yi Lei, Bin Chen, *Member, IEEE*, Gabriele Liga, *Member, IEEE*, Xiong Deng, *Member, IEEE*, Zizeng Cao, Jianqiang Li, *Senior Member, IEEE*, Kun Xu, *Member, IEEE*, and Alex Alvarado, *Senior Member, IEEE*

***Abstract*— Staircase codes (SCCs) are typically decoded using iterative bounded-distance decoding (BDD) and hard decisions. In this paper, a novel decoding algorithm is proposed, which partially uses soft information from the channel. The proposed algorithm is based on marking certain number of highly reliable and highly unreliable bits. These marked bits are used to improve the miscorrection-detection capability of the SCC decoder and the error-correcting capability of BDD. For SCCs with 2-error-correcting Bose-Chaudhuri-Hocquenghem component codes, our algorithm improves upon standard SCC decoding by up to 0.30 dB at a bit-error rate (BER) of 10^{-7}. The proposed algorithm is shown to achieve almost half of the gain achievable by a genie decoder with this structure. The increased complexity caused by bit marking and additional calls to the component BDD decoder is discussed as well. Our algorithm is also extended (with minor modifications) to product codes. The simulation results show that in this case, the algorithm offers gains of up to 0.5 dB at a BER of 10^{-7}.**

Manuscript received February 18, 2019; revised July 25, 2019; accepted September 24, 2019. Date of publication October 3, 2019; date of current version December 17, 2019. This work is in part supported by the NSFC Program (No. 61431003, 61821001, 61625104, and 61701155), and the Fundamental Research Funds for the Central Universities (JZ2019HGBZ0126, JZ2019HGBZ0130, PA2019GDZC0098). The work of A. Alvarado and G. Liga is supported by the Netherlands Organisation for Scientific Research (NWO) via the VIDI Grant ICONIC (project number 15685) and has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 57791). This article was presented in part at the International Symposium on Turbo Codes & Iterative Information Processing 2018, Hong Kong, China, December 2018. The associate editor coordinating the review of this article and approving it for publication was A. Graell I Amat. *(Corresponding author: Yi Lei.)*

Y. Lei was with the State Key Laboratory of Information of Photonics and Optical Communications, Beijing University of Posts and Telecommunications (BUPT), Beijing 100876, China. She is now with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China (e-mail: leiyi@hfut.edu.cn).

B. Chen is with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China (e-mail: bin.chen@hfut.edu.cn).

G. Liga, X. Deng, and A. Alvarado are with the Signal Processing Systems (SPS) Group, Department of Electrical Engineering, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands (e-mail: g.liga@tue.nl; x.deng@tue.nl; a.alvarado@tue.nl).

Z. Cao is with the Electro-Optical Communications (ECO) Group, Department of Electrical Engineering, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands (e-mail: z.cao@tue.nl).

J. Li was with the State Key Laboratory of Information of Photonics and Optical Communications, Beijing University of Posts and Telecommunications (BUPT), Beijing 100876, China. He is now with the Alibaba Group U.S. Inc., Bellevue, WA 98004 USA (e-mail: jqjqlee@gmail.com).

K. Xu is with the State Key Laboratory of Information of Photonics and Optical Communications, Beijing University of Posts and Telecommunications (BUPT), Beijing 100876, China (e-mail: xukun@bupt.edu.cn).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCOMM.2019.2945317

***Index Terms*— Optical communication systems, staircase codes, product codes, iterative bounded distance decoding, marked bits.**

I. Introduction and Motivation

FORWARD error correction (FEC) is required in optical communication systems to meet the ever increasing data demands in optical transport networks (OTNs). FEC codes that can boost the net coding gain (NCG) are of key importance. A Reed-Solomon (RS) code with parameters of $(255, 239)$ was the first standardized FEC code for OTNs in the ITU-T Recommendation G.975 [1]. For an output bit error ratio (BER) of 10^{-15}, the NCG of RS$(255, 239)$ is 6.2 dB. In order to increase transmission data rate and distance, several super FEC codes were considered in the ITU-T Recommendation G.975.1 [2]. Most of these super FECs utilize two concatenated FEC codes, such as: BCH$(3860, 3824, 3)$ + BCH$(2040, 1930, 10)$ codes,[1] RS$(1023, 1007)$ + BCH$(2047, 1952, 8)$ codes, etc. The achieved NCG can be up to 8.99 dB at a BER of 10^{-15}.

OTNs are currently targeting data rates of 400 Gb/s and beyond [3], [4]. In this scenario, FEC codes with higher NCG are highly desired. Soft-decision (SD) FEC codes provide large NCGs, however, they are not the best candidates for very high data rate applications due to their high power consumption and decoding latency. For applications with strict latency and complexity requirements (e.g., short reach), simple but powerful hard-decision (HD) FEC codes, e.g., product codes (PCs) [5] and staircase codes (SCCs) [6], [7], have received considerable attention: PC has been adopted (as an inner code) in the subclass I.5 of G.975.1 [2], while SCC is part of the 400ZR Implementation Agreement (as an outer code) in the Optical Internetworking Forum [8]. SCC is also recommended for 100G optical transport unit (OTU) order 4 for long-reach applications in the ITU-T Recommendation G.709.2/Y.1331.2 [9]. In [10], product and staircase codes are implemented in very-large-scale integration system, which reach more than 1 Tb/s information throughputs with only energy efficiencies of around 2 pJ/bit. Recently, low-complexity concatenated FEC and adaptive coded modulation schemes have also been studied to combine the advantages of soft- and hard-decision decoders [11], [12].

[1]Throughout this paper we use n_c, k_c, and t to denote the codeword length, information length, and error-correcting capability, resp. BCH codes are denoted as BCH(n_c, k_c, t).

Both SCCs and PCs are based on simple component codes, Bose-Chaudhuri-Hocquenghem (BCH) codes being the most popular ones. The decoding is done iteratively based on bounded-distance decoding (BDD) for the component codes. Although very simple, one drawback of BDD is that its error-correcting capability is limited to $t = \lfloor (d_0 - 1)/2 \rfloor$, where d_0 is the minimum Hamming distance (MHD) of the component code [13]. BDD can detect more than t errors, but cannot correct them. In some cases, BDD may also return a codeword (but not the transmitted one), even if the received sequence is with more than t errors. This situation is known as a *miscorrection*. Miscorrections are known to degrade the performance of iterative BDD. To prevent miscorrections and/or extend the error correcting capability, several methods have been studied in the literature. In what follows we review those methods.

To prevent miscorrections in SCCs, the authors of [14] proposed rejecting bit-flips from the decoding of bit sequences associated with the last SCC block if they conflict with a zero-syndrome codeword from the previous block. As pointed out in [15, Sec. I], the obtained gains of [14] are expected to be limited. An anchor-based decoding algorithm has been proposed in [15], [16], where some bit sequences are labeled as anchor codewords. These sequences are thought to have been decoded without miscorrections. Decoding results that are inconsistent with anchor codewords are discarded. It has been demonstrated that this algorithm works well with both SCCs and PCs. The algorithm of [16] outperforms [14], but it suffers from an increased complexity as anchor codewords need to be tracked during iterative BDD. Very recently, iterative decoding algorithms for PCs [17]–[21] and SCCs [18] were proposed. A decoding algorithm called iBDD-SR [18] exploits channel reliabilities to modify the BDD outbound messages, yielding additional gains up to 0.31 dB. We remark that iBDD-SR requires storing the channel reliabilities, which in turn requires additional memory (and processing). In [19], a novel implementation for iBDD-SR for PCs is proposed which only requires 1-bit additional reliability memory with minor performance degradation compared to the original iBDD-SR proposal [18]. Later on, the component BDD decoder is replaced by generalized minimum distance decoding (GMD), which treats bits with least reliability as erasures, to obtain more gains [20]. In [21], Hamming distance has taken the place of generalized distance defined in [20, Eq. (3)] to avoid the exchange of soft information in the decoding process and reduce the internal decoder data flow.

To extend the error correcting capability, Chase proposed three kinds of algorithms to decode block codes with channel soft information [22]. In this class of algorithms, each bit is accompanied with a stored reliability, according to the soft information. During the decoding, the algorithms will first generate a set of test patterns, and add each of them, modulo-2, to the received sequence, respectively. The obtained new sequences are all decoded by a binary decoder, and the algorithms choose an error pattern that has minimum analog weight defined in [22, Eq. (6)] as the final output. Through this way, the error correcting capability can be extended from $\lfloor (d_0 - 1)/2 \rfloor$ to $d_0 - 1$. The main drawback of these three algorithms is that the decoder needs to decode at least $\lfloor (d_0/2) + 1 \rfloor$ test patterns (while in fact, not all of them are necessary). This significantly increases the decoding complexity and latency. Specifically, the test patterns in algorithm 3 contains such two types: one is all with zeros and the other is all with 1's at certain number of positions with lowest reliability. This behaves similar to erasure decoding [23, Sec. 6.6] as well as GMD [20], where the sequence of test patterns is equivalent to the sequence of erasures. However, they were not designed to take miscorrections into account.

In addition, the authors of [24] have considered to use soft-input/soft-output decoder to decode each component code within all iterations. However, the achieved additional gain is only 0.30 dB for large block size, at the expense of greatly increased complexity. In [25], the authors proposed a fully HD-based bit flipping (BF) method to lower the error floor of SCCs. It is based on the fact that each row and column within a stall pattern contains at least $t + 1$ errors that results in a non-zero syndrome. Therefore, the stall patterns can be found by positioning the intersections of those non-zero syndromes' rows and columns. After that, BF is applied to the positioned intersections to solve the stall patterns.

In this paper, we propose the soft-aided bit-marking (SABM) algorithm to improve the decoding of SCCs as well as PCs. It is achieved by marking highly reliable bits (HRBs) and highly unreliable bits (HUBs), according to the soft information. HRBs is used to enhance the miscorrection-detection (MD) method proposed in [14], while HUBs is used to help to decode those originally unsolvable component codes via BF. This idea was first proposed in [26] and also experimentally validated in a multi-span hybrid-amplified system in [27]. As high order modulation formats are often used in modern optical communication systems, the performance of the proposed SABM algorithm under different modulation formats was investigated. The main feature of the proposed SABM algorithm is its low complexity, as explained in what follows. For SCCs, the SABM algorithm only requires modifications to the decoding structure of the last block of each decoding window. Furthermore, in the SABM algorithm each component code needs to be decoded at most twice. Also, the algorithm is based on marking bits only, which simply requires to store very small part of the soft bits (log-likelihood ratios, LLRs). Finally, marked bits do not need to be tracked and updated during the iterative process either. The applicability of the proposed SABM algorithm to product codes (PCs) was also investigated.

The remainder of the paper is organized as follows. In Sec. II, we present the system model we consider and introduce the principles of SCCs and BDD. In Sec. III, we describe the proposed SABM algorithm. Some examples are also given to explain how it works. In Sec. IV, we present the simulation results for SCCs, and discuss the complexity of the SABM algorithm. In Sec. V, we extend this algorithm to PCs and investigate the performance. Finally, we conclude this paper in Sec. VI.

II. System Model, SCCs, and BDD

A. System Model

As shown in Fig. 1 (a), information bits are encoded by a staircase encoder and then mapped to symbols x_l labeled by every m coded bits $b_{l,1}, \ldots, b_{l,m}$ at each time instant l. The symbols x_l are taken from an equally-spaced bipolar M-ary Pulse Amplitude Modulation (PAM) constellation $\mathcal{S} = \{s_1, s_2, \ldots, s_M\}$ with $M = 2^m$ points. The bit-to-symbol mapping is the binary reflected Gray code. The received signal is $y_l = \sqrt{\rho}x_l + z_l$, where z_l is zero-mean unit-variance additive white Gaussian noise (AWGN) and $\sqrt{\rho}$ is the channel gain.

The standard HD receiver structure uses an HD-based demapper ($\mathbb{R} \to \{0, 1\}$) to estimate the coded bits and feeds them to the staircase decoder (green area in Fig. 1 (a)), where $\mathbb{R}$ denotes the real number. In this paper, we consider a receiver architecture where the HD-FEC decoder uses soft information from the channel. This soft information is typically represented using LLRs, calculated as [28, Eq. (3.50)]

$$\lambda_{l,k} = \sum_{b \in \{0,1\}} (-1)^{\bar{b}} \log \sum_{i \in \mathcal{I}_{k,b}} \exp\left(-\frac{(y_l - \sqrt{\rho}s_i)^2}{2}\right), \quad (1)$$

with $k = 1, \ldots, m$, and where $\bar{b}$ denotes bit negation. In (1), the set $\mathcal{I}_{k,b}$ enumerates all the constellation points in $\mathcal{S}$ whose kth bit $c_{i,k}$ is b, i.e., $\mathcal{I}_{k,b} \triangleq \{i = 1, 2, \ldots, M : c_{i,k} = b\}$.

The proposed structure is shown in Fig. 1 (a) (red area). In this structure, apart from the HD-estimated bits $\hat{b}_{l,1}, \ldots, \hat{b}_{l,m}$, a sequence of *marked* information will also be made available to the decoder. We call this architecture soft-aided (SA) HD-FEC decoding. These marked information is denoted by $q_{l,k}$ and can be: HRBs, HUBs, or unmarked bits. The marking is made based on the absolute value of the LLRs $|\lambda_{l,k}|$. More details about the marking procedure and how this can be exploited by the decoder will be given in Sec. III.

B. Staircase Codes

Fig. 1 (b) shows the staircase structure of SCCs we consider in this paper, where block $\boldsymbol{B}_0$ is initialized to all zeros. Each subsequent SCC block $\boldsymbol{B}_i, i = 1, 2, \ldots$, is composed of $w(w-p)$ information bits (white areas) and wp parity bits (gray areas). Each row of the matrix $[\boldsymbol{B}_{i-1}^T \boldsymbol{B}_i]$ $\forall i > 1$ is a valid codeword in a component code $\mathcal{C}$. We consider the component code $\mathcal{C}$ to be a binary code with parameters (n_c, k_c, t). Then, w and p are given by: $w = n_c/2$ and $p = n_c - k_c$. The code rate R of the SCC is $R = 1 - p/w = 2k_c/n_c - 1$. Throughout this paper, the component codes $\mathcal{C}$ considered are extended (by 1 additional parity bit) BCH codes. The mapping between code bits and symbols is done by reading row-by-row the SCC blocks $\boldsymbol{B}_i, i = 1, 2, \ldots$.

At the receiver side, SCCs are decoded iteratively using a sliding window decoder covering L blocks. We use $\boldsymbol{Y}_i$ to indicate the received SCC block after HD-demapper corresponding to the transmitted block $\boldsymbol{B}_i$. The decoder first iteratively decodes the blocks $\{\boldsymbol{Y}_0, \boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_{L-1}\}$. When a maximum number of iterations ℓ is reached, the decoding window outputs the block $\boldsymbol{Y}_0$ and moves to decode the blocks $\{\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_L\}$. The block $\boldsymbol{Y}_1$ is then delivered and operation continues on $\{\boldsymbol{Y}_2, \boldsymbol{Y}_3, \ldots, \boldsymbol{Y}_{L+1}\}$. This process continues indefinitely. Multiple decoding scheduling alternatives exist (see, e.g., [6, Sec. IV] [7, Sec. II]). We chose the most popular one, namely, alternated decoding of pairs of SCC blocks within a window, from the bottom right to the top left of the SCC window.

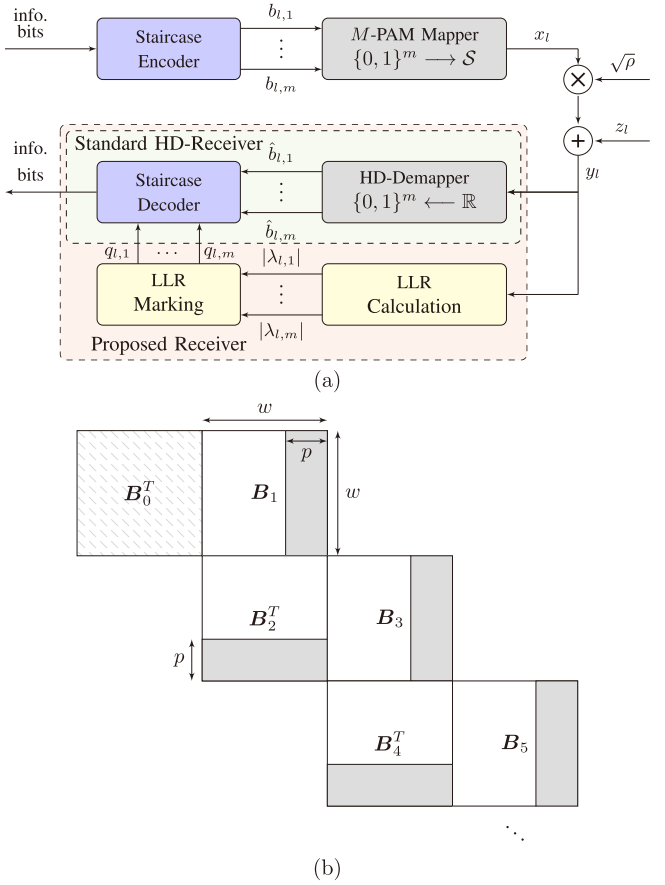


Fig. 1. (a) System model under consideration. (b) Staircase structure of SCCs considered in this paper.

C. Bounded-Distance Decoding

BDD is used to decode (in Hamming space) the received bit sequence for the component code $\mathcal{C}$. To correct up to t errors, the MHD d_0 of $\mathcal{C}$ must satisfy $d_0 \geq 2t + 1$ ($d_0 \geq 2t + 2$ for extended BCH codes with 1 additional parity bit). Thus, every codeword in the code $\mathcal{C}$ can be associated to a sphere of radius t. Within such a sphere, no other codewords exist. If the received sequence r falls inside one of these spheres, BDD will decode r to the corresponding codeword. Otherwise, BDD will declare a failure. For a given transmitted codeword c and a received sequence r, the BDD output $\hat{c}$ is thus given by

$$\hat{c} = \begin{cases} c, & \text{if } d_{\mathrm{H}}(r, c) \leq t \\ \tilde{c} \in \mathcal{C}, & \text{if } d_{\mathrm{H}}(r, c) > t \text{ and } d_{\mathrm{H}}(r, \tilde{c}) \leq t \\ r, & d_{\mathrm{H}}(r, \tilde{c}) > t \ \forall \tilde{c} \in \mathcal{C}, \end{cases} \quad (2)$$

where $d_{\mathrm{H}}(\cdot, \cdot)$ represents the Hamming distance. In practice, BDD is often a syndrome-based decoder that uses syndromes

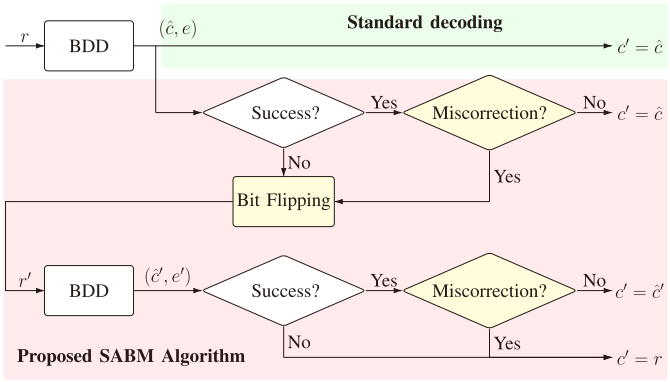


Fig. 2. Flow chart of the proposed SABM algorithm: r is the received row sequence taken from two neighbor SCC blocks and c' is the output of the staircase decoder. BDD returns a decoded codeword $\hat{c}$ based on (2) and an error pattern e. The three highlighted blocks use marked bits for operation.

to estimate the error pattern e. If the syndromes are all zeros, no errors are present. For the first two cases in (2), BDD will both declare decoding success and $\hat{c} = r \oplus e$. In the second case, although BDD will still return an error pattern e, this case corresponds to a miscorrection. In the next section, we will show how to improve miscorrection detection (MD) using the underlying structure of SCCs and the marked HRBs.

III. The SABM Algorithm

The flow chart of the proposed SABM algorithm is shown in Fig. 2 (red area). Assume that decoding is being performed over the blocks $\{\boldsymbol{Y}_i, \boldsymbol{Y}_{i+1}, \ldots, \boldsymbol{Y}_{i+L-1}\}$, then r is given by a row sequence taken from two neighbor blocks $[\boldsymbol{Y}^T_{i+j-1}\boldsymbol{Y}_{i+j}]$, where $j \in \{1, 2, \ldots, L-1\}$. Compared to standard SCC decoding (green area in Fig. 2), which always accepts the decoding result $\hat{c}$ of BDD, the SABM algorithm further checks the decoding status of BDD. If BDD successfully decodes r, miscorrection detection is performed. Furthermore, BF is proposed as a way to handle decoding failures and miscorrections. In this section, we will explain the steps in the SABM algorithm.

The SABM algorithm can in principle be applied to all received sequences r within L SCC blocks. However, due to the iterative sliding window decoding structure applied to SCCs, most of the errors are expected to be located in the last two blocks. To keep the complexity and latency low, we will therefore only use this algorithm on the received sequences from the last two blocks of the window. Therefore, from now on we only consider rows of the matrix $[\boldsymbol{Y}^T_{i+L-2}\boldsymbol{Y}_{i+L-1}]$.

A. Decoding Success: Improved Miscorrection Detection

To avoid miscorrections, it was suggested in [14] to reject the decoding result of BDD applied to $[\boldsymbol{Y}^T_{i+L-2}\boldsymbol{Y}_{i+L-1}]$ if the decoded codeword would cause conflicts with zero-syndrome codewords in $[\boldsymbol{Y}^T_{i+L-3}\boldsymbol{Y}_{i+L-2}]$. This method protects bits in $\boldsymbol{Y}_{i+L-2}$ but cannot handle bits in the last block $\boldsymbol{Y}_{i+L-1}$. We propose to enhance this method by using marked bits in $\boldsymbol{Y}_{i+L-1}$. In particular, we add one additional constraint to the algorithm in [14]: no HRBs in $\boldsymbol{Y}_{i+L-1}$ shall ever be flipped.

The reliability of a bit is given by the absolute value of its LLR, a high value indicating a more reliable bit. Therefore, a threshold δ is set to decide if the bit is highly reliable. If $|\lambda_{l,k}| > \delta$, the corresponding bit is marked as an HRB. The decision of the staircase decoder will therefore be marked as a miscorrection if the decoded codeword causes conflicts with zero-syndrome codewords in $[\boldsymbol{Y}^T_{i+L-3}\boldsymbol{Y}_{i+L-2}]$, *or* if the decoded codeword flips a bit whose LLR satisfies $|\lambda_{l,k}| > \delta$.

Example 1: Fig. 3 (a) shows a decoding window with $w = 6$ and $L = 5$ and a component code $\mathcal{C}$ with $t = 2$ ($d_0 = 6$). Following the notation of [16], a pair (i, j) is used to specify the location of a component codeword in each window, where $i \in \{1, 2, \ldots, L-1\}$ indicates the position relative to the current window and $j \in \{1, 2, \ldots, w\}$ indicates the corresponding row or column index in the matrix of two neighbor blocks. A triple (i, j, k) is used to indicate the kth bit in the component codeword (i, j), where $k \in \{1, 2, \ldots, 2w\}$. For example, the component codewords $(1, 2)$ and $(3, 1)$ are highlighted with light magenta, while bits $(1, 2, 11)$ and $(3, 1, 4)$ are highlighted with dark magenta. The bit sequence $(3, 1)$ is a codeword in $[\boldsymbol{Y}^T_{i+2}\boldsymbol{Y}_{i+3}]$ whose syndrome is equal to zero. The cells filled with dark yellow are the ones marked as HRBs whose $|\lambda_{l,k}|$ is more than δ, while $\delta = 10$.

After transmission, the received bit sequences for $(4, 1)$ and $(4, 3)$ have 5 and 4 errors (black crosses), respectively. When applying BDD, miscorrections (red crosses) occur. For the received bits in $(4, 1)$, BDD mistakenly detects bit $(4, 1, 1)$ as an error and suggests to flip it. However, because it is involved in the zero-syndrome codeword $(3, 1)$, it will be identified as a miscorrection by both our MD algorithm and by the one in [14]. For the received bits in $(4, 3)$, however, the suggested flipping bit $(4, 3, 5)$ in $\boldsymbol{Y}_{i+L-2}$ is not involved in any zero-syndrome codewords, and thus, [14] would fail to detect this miscorrection. The bit $(4, 3, 9)$ is a HRB, and thus, our MD algorithm will successfully identify it as a miscorrection.

The MD algorithm in [14] does not always detect the miscorrections. The new rule we introduced (never flip HRBs in $\boldsymbol{Y}_{i+L-1}$) is only heuristic and does not guarantee perfect MD either. For example, our MD algorithm fails when no bits are flipped by BDD because $r = \tilde{c} \in \mathcal{C}$. Nevertheless, as we will see later, our MD algorithm combined with bit flipping (see next Sec.) gives remarkably good results with very small added complexity.

B. Decoding Failures and Miscorrections: Bit Flipping

To deal with decoding failures and miscorrections, we propose to flip bits (see BF block in Fig. 2). The main idea is to flip certain bits in r and make the resulting sequence r' (after BF) closer to c in Hamming space. In particular, the proposed BF aims at making the Hamming distance between r' and c equal to t, so that BDD can correct r' to the transmitted codeword c. Two cases are considered by our proposed algorithm: (1) decoding failures, and (2) miscorrections.

1) Case 1 (Decoding Failures): We target received sequences with $t + 1$ errors. In this case, we flip a HUB with the lowest absolute LLR. The intuition here is that this marked

bit was indeed one flipped by the channel. In the cases where the marked HUB corresponds to a channel error, the error correction capability of the code $\mathcal{C}$ is effectively increased by 1 bit.

2) Case 2 (Miscorrections): We target miscorrections where BDD chooses a codeword $\tilde{c} \in \mathcal{C}$ at MHD of c. The intuition here is that most of the miscorrections caused by BDD will result in codewords at MHD from the transmitted codeword. When a miscorrection has been detected, our algorithm calculates the number of errors detected by BDD. This is equal to $d_{\mathrm{H}}(r, \tilde{c}) = w_{\mathrm{H}}(e)$. Then, our algorithm flips $d_0 - w_{\mathrm{H}}(e) - t$ bits, which in *some cases* will result in r' that satisfy $d_{\mathrm{H}}(c, r') = t$. This will lead BDD to find the correct codeword. More details are given in Examples 2 and 3. Again using the intuition that bits with the lowest reliability are the most likely channel errors, our BF algorithm flips $d_0 - w_{\mathrm{H}}(e) - t$ HUBs with the lowest absolute LLR.

Taking into account all the cases, the maximum number of flipped bits is $d_0 - t - 1$ per component code (when miscorrection happens and the Hamming weight of the detected error pattern is 1). In practice, this means only $d_0 - t - 1$ bits out of w bits in each row of $\boldsymbol{Y}_{i+L-1}$ need to be marked (and sorted). To position the HUBs, we partially sort the reliability list in each row that returns the $d_0 - t - 1$ smallest values with the indices. The corresponding HD-estimated bits in $\boldsymbol{Y}_{i+L-1}$ are then marked as HUBs. The BF block (see Fig. 2) chooses the number of marked HUBs to flip based on this sorted list and the Hamming weight of the detected error pattern.

Example 2: Fig. 3 (b) shows a representation of BDD ($t = 2$ and $d_0 = 6$), where the black dots represent the transmitted codeword c and another codeword $\tilde{c} \in \mathcal{C}$ with $d_{\mathrm{H}}(c, \tilde{c}) = d_0$. The red dashed circle and solid blue circles correspond to locations of r for Cases 1 and 2, respectively. The bit sequence $(4, 5)$ in Fig. 3 (a) (3 errors) would lie on the red dashed circle, while sequences $(4, 1)$ and $(4, 3)$ correspond to red diamonds (5 and 4 errors, respectively). For the latter two bit sequences, provided that we flip the correct bits (flipping 3 and 2 marked bits, respectively), will give a r' with $d_{\mathrm{H}}(c, r') = t$ which can be correctly decoded.

Example 3: Light yellow cells in Fig. 3 (a) show the marked 3 HUBs with the lowest reliability within that codeword. The lighter yellow color indicates a smaller value of $|\lambda_{l,k}|$. In this example, BDD fails to decode bit sequence $(4, 5)$. Fortunately, $(4, 5, 8)$ corresponds to the marked HUB with smallest $|\lambda_{l,k}|$. Thus, it will be flipped after BF, and then the remaining 2 errors $(4, 5, 3)$ and $(4, 5, 10)$ will be fully corrected by applying BDD again. This corresponds to Case 1.

For bit sequences (4,1) and (4,3), the decoding results of BDD are identified as miscorrections (as explained in Example 1) with $w_{\mathrm{H}}(e) = 1$ and $w_{\mathrm{H}}(e) = 2$, respectively. According to the BF rule for miscorrections, 3 and 2 bits with smallest $|\lambda_{l,k}|$ among the marked HUBs, i.e., (4,1,8), (4,1,10), (4,1,11) in (4,1), and (4,3,7), (4,3,10) in (4,3), will all be flipped. As a result, only 2 errors are left in (4,1) and (4,3), which are within the error correcting capability of BDD. This corresponds to Case 2.

BF will not always result in the correct decision. As shown in Example 2, this is the case for certain miscorrections

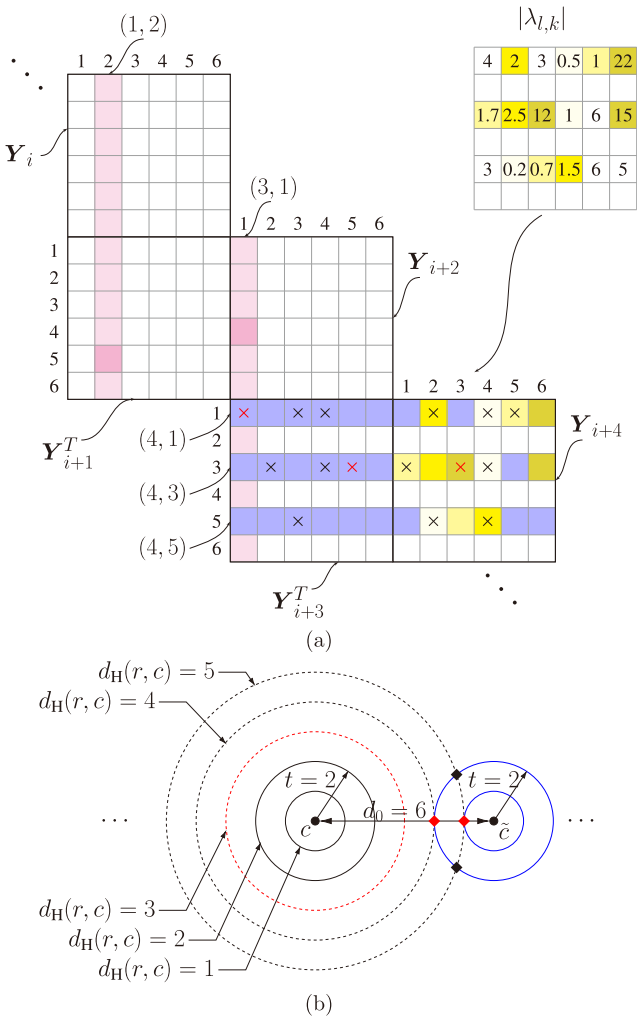


Fig. 3. (a) SCC decoding example ($w = 6$, $L = 5$, $t = 2$): black crosses are received errors after channel transmission and red crosses indicate miscorrections after BDD. Dark yellow cells are marked HRBs ($\delta = 10$), while light yellow cells are marked HUBs. Lighter yellow colors indicate a smaller value of $|\lambda_{l,k}|$. (b) Schematic representation of BDD: c is the transmitted codeword and $\tilde{c} \in \mathcal{C}$ is another codeword at MHD $d_0 = 6$. The circles around c show the possible locations of r with 1, 2, ... errors from inside to outside in turn, while black solid circles indicate cases that BDD will decode successfully. Diamonds indicate four possible locations, where miscorrection happens.

(black diamonds in Fig. 3 (b)). Additionally, miscorrections for codewords at distances larger than d_0 are not considered either. Finally, marked LLRs might not correspond to channel errors. In all these cases, either decoding failures or miscorrections will happen. To avoid these cases, the SABM algorithm includes two final checks after BF and BDD (see lowest part of Fig. 2): successful decoding and MD.

Note that BF in the binary case commonly corresponds to erasure decoding, which is also used in [20], [21] as a component decoder of GMD. But the difference is that the component decoder in the SABM algorithm will first decode the received sequence itself as usual, while BF occurs only when the decoding fails or is a miscorrection. This can effectively reduce the unnecessary trials. The number of flipped bits, in turn, depends on the feedback of the first

TABLE I
SCC PARAMETERS WE USED IN THIS PAPER

R	v	t	n_c	k_c	u	w	p
0.87	8	2	256	239	0	128	17
0.83	9	2	228	209	284	114	19
0.92	9	2	504	485	8	252	19
0.80	8	3	256	231	0	128	25
0.74	8	4	256	223	0	128	33

decoding trial. Additionally, BF in the SABM algorithm only tries to correct the most likely errors (not all errors) for those original unsolvable cases, while the left is handed over to BDD to solve. This is distinct from the BF in [25].

IV. ALGORITHM OPTIMIZATION AND SIMULATION RESULTS

In this section, we focus on SCCs that the component codes are BCH codes with $t = 2$. Three kinds of code rates, i.e., $R \in \{0.87, 0.83, 0.92\}$, are considered. SCCs with $t = 3$ and 4 are investigated as well, but only the BER performance. Table I shows the parameters we used. They are listed in the order they appear in the paper. The component codes are extended (with 1 additional parity bit) and shortened (with u information bits) BCH codes. The component codeword length is $n_c = 2^v - u$, and the information bit length is $k_c = n_c - vt - 1$, where v is an integer. Throughout this section, the decoding window size is $L = 9$, and the maximum number of iterations is $\ell = 7$.

A. LLR Threshold Choice

One key aspect of the proposed SABM algorithm is the selection of the bits to be marked as HRBs. This selection is based on the channel reliabilities, in particular, by using an LLR threshold δ. In order to optimize the process of marking bits as highly reliable, the optimum LLR threshold need to be investigated. We do this in the following.

To directly compare our algorithm with the results presented in [16], we firstly consider an SCC with $R = 0.87$, whose component code is BCH$(256, 239, 2)$ [2] ($w = 128$). Fig. 4 shows the post BER performance under different threshold δ. The modulation format is 2-PAM. The three curves are obtained for SNRs of 6.98 dB (triangles), 7.02 dB (circles) and 7.05 dB (stars), respectively. These SNRs are chosen so that the achieved BERs are 10^{-4}, 10^{-5} and 10^{-6}, resp. Fig. 4 shows that, to obtain the best performance, the corresponding optimum threshold δ^* is 10, 11 and 11. However, the difference between these values is small, and the resulting performance difference is negligible as long as $\delta \approx 10$.[3]

The U-type trend results in Fig. 4 can be intuitively understood as follows. If $\delta < \delta^*$ is used, the performance degrades because some of the bits that are not reliable enough

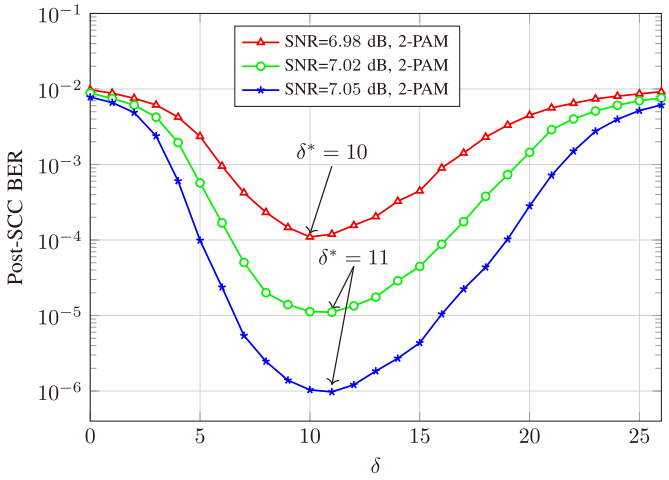


Fig. 4. Post-SCC BER vs. LLR threshold δ for code rate $R = 0.87$ and 2-PAM.

are marked as HRBs. This will lead to some correct BDD decisions being mistakenly marked as miscorrections, which are then rejected by the SABM-based staircase decoder. On the other hand, the performance degradation for $\delta > \delta^*$ is due to the fact that some of the bits that should probably be trusted, are not marked as HRBs. This weakens the ability of the SABM algorithm to identify miscorrections.

Fig. 5 shows the post BER performance vs. δ for SCC code rates of $R = 0.83$ and 0.92. The corresponding component codes we used are BCH$(228, 209, 2)$ and BCH$(504, 485, 2)$.[4] These parameters are obtained by shortening the extended BCH$(512, 493, 2)$ by 284 and 8 bits, respectively. We investigate the BER performance under two SNRs for each code rate. Furthermore, we investigate two modulation formats: 2-PAM (solid lines) and 8-PAM (dashed lines). The results in Fig. 5 show that for both code rates and modulation formats, the optimum threshold is $\delta^* = 12$, which is slightly larger than the one in Fig. 4. Fig. 5 also shows that SCCs with 8-PAM are less sensitive to an overestimation of the optimum value of δ^* than SCC with 2-PAM. This can observed by the relatively flat BER curves for 8-PAM when $\delta > \delta^*$.

B. Post-BER Performance Analysis

Fig. 6 shows the BER performance vs. SNR for $R = 0.87$ and 2-PAM. As suggested by the results in Fig. 4, the LLR threshold to mark HRBs is set to $\delta = 10$, which is the optimum value δ^* at the point of SNR$= 6.98$ dB. Two baselines are: standard decoding where miscorrections are not dealt with (circles), and miscorrection-free (MF) decoding (stars). The latter is obtained via a genie BDD decoder which corrects the received sequence only when the number of errors is not more than t. The black dotted curve shows the estimated error floor of standard SCC decoding.[5] It only considers the main

[2]The generator polynomial is $g(x) = (x^8 + x^6 + x^5 + x^4 + 1)(x^8 + x^7 + x^6 + x^4 + x^3 + x^2 + 1)(x + 1)$. The last factor $(x + 1)$ is because the BCH code considered is extended with 1 additional parity bit.

[3]It is important to note that this difference becomes important for optical transmission experiments. This was recently shown in [27, Fig. 3], where the optimum value δ^* for a long-haul system was found to be as low as $\delta^* = 4$.

[4]The generator polynomials of the two BCH codes are the same that is $g(x) = (x^9 + x^4 + 1)(x^9 + x^6 + x^4 + x^3 + 1)(x + 1)$.

[5]The reason why there is a quite high error floor in Fig. 6 is the relatively short SCC and low error-correcting capability we used. Longer SCC codes or higher error-correcting capability (like a G.709-compatible one Yi Cai studied [29], [30]) do not have an error floor above BER $= 10^{-15}$.

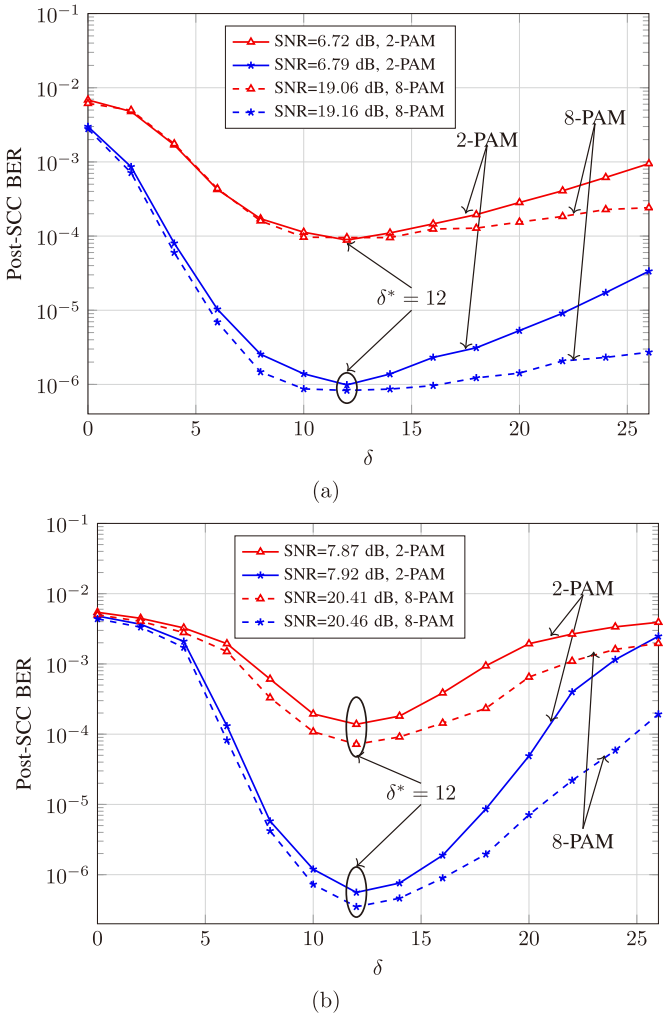


Fig. 5. Post-SCC BER vs. LLR threshold δ for code rates $R = 0.83$ (a) and $R = 0.92$ (b). The modulation formats include 2-PAM (solid lines) and 8-PAM (dashed lines).

contributor of minimal stall patterns, estimated as [6, Sec. V]

$$\text{BER}_{\text{post}} \approx \frac{(t+1)^2}{w^2} M_{\min} \text{BER}_{\text{pre}}^{(t+1)^2} \tag{3}$$

where

$$M_{\min} = \binom{w}{t+1} \sum_{m=1}^{t+1} \binom{w}{m} \binom{w}{t+1-m}.$$

and BER_{pre} is the channel error probability. This figure also shows the performance of previously proposed methods: [14] (diamonds) and [16] (crossed circles).

As shown in Fig. 6, the SABM algorithm (squares) outperforms standard decoding by 0.3 dB, while almost half of this is achieved by the proposed miscorrection detection mechanism (crosses). It can be seen that, the SABM algorithm also outperforms both [14] and [16]. These two methods in [14] and [16] only prevent miscorrections, and thus, their performance is bounded by the MF case. Although the SABM algorithm only deals with miscorrections related to the last block of each window, it outperforms the MF case. This is due to its additional ability to better deal with miscorrections and decode even when BDD initially fails. In terms of the

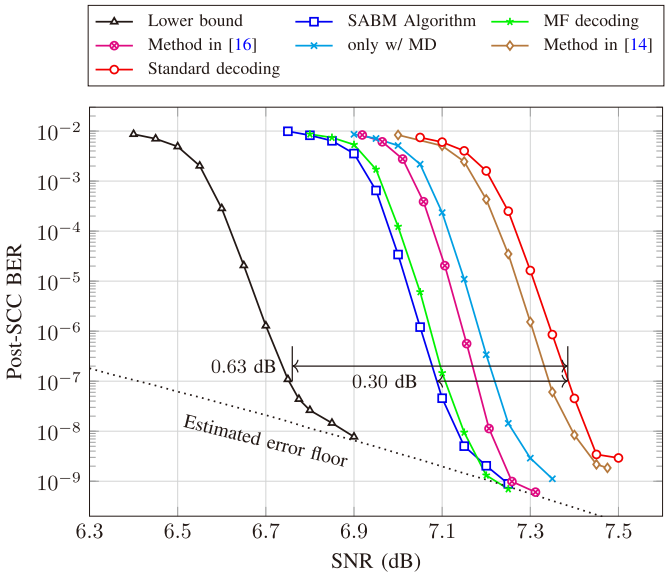


Fig. 6. Post-SCC BER vs. SNR for code rate $R = 0.87$ and 2-PAM. Black dotted line is the estimated error floor of standard SCC decoding based on (3).

error floor, it can be found that the performance of the SABM algorithm is lower than standard decoding, and close to the MF case.

Fig. 6 also shows a lower bound for the SABM algorithm (triangles). This bound is obtained by a genie decoder which emulates a best-case scenario for the SABM algorithm. This genie decoder is assumed to be able to ideally identify all miscorrections in the last two blocks of the window. This corresponds to have an idealized MD block in the top part of Fig. 2. The genie decoder also emulates an idealized assumption on what the BF block in Fig. 2 can do. For this, we assume that the decoder knows exactly which bits in the last two blocks are errors. If a given sequence has $t+j$ errors ($j = 1$ for Case 1, or $j = d_0 - w_{\text{H}}(e) - t$ for Case 2), and at least j errors are located in the last block, the genie decoder flips j errors in the last block, and then the received sequence is correctly decoded. If less than j errors are located in the last block, the genie decoder declares a failure. The results in Fig. 6 show that the maximum potential gain for our receiver structure (for 2-PAM, $R = 0.87$, and $t = 2$) is 0.63 dB. The SABM algorithm almost achieved half of this gain with very small added complexity (see Sec. IV-C for details).

Fig. 7 shows the simulation results of the SABM algorithm for $R = 0.83$ and 0.92. For each code rate, three modulation formats are considered: 2-PAM, 4-PAM and 8-PAM. As shown in Figs. 4 and 5, using an optimized δ^* for each code rate and modulation format gives the best BER. However, for simplicity, the LLR threshold we use here is set to $\delta = 10$. For SCCs with $R = 0.92$, it is difficult to obtain very low BER, thus only the waterfall region are shown. It can be seen from Fig. 7 that for different modulation formats and code rates, the SABM algorithm always outperforms the miscorrection-free case, also on the error floor region for $R = 0.83$. When compared to standard staircase decoding, the achieved gains are between 0.20 dB and 0.29 dB, while the obtained maximum potential gains are between 0.46 dB

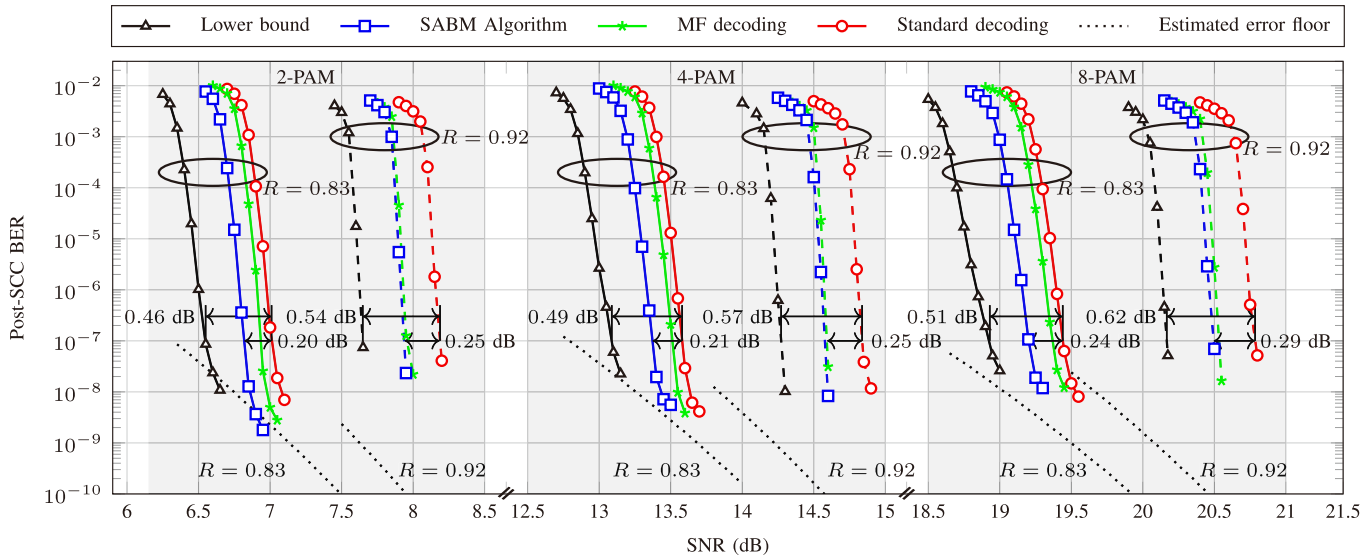


Fig. 7. Post-SCC BER vs. SNR for code rates $R = 0.83$ (solid lines) and $R = 0.92$ (dashed lines) with 2-PAM, 4-PAM, and 8-PAM modulation formats. Black dotted lines are the estimated error floors of standard SCC decoding based on (3).

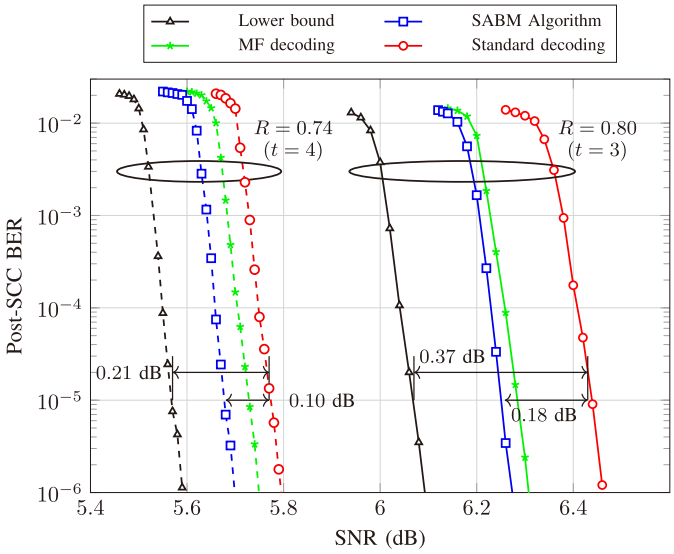


Fig. 8. Post-SCC BER vs. SNR for code rates $R = 0.80$ ($t = 3$) and $R = 0.74$ ($t = 4$) with modulation format of 2-PAM.

and 0.62 dB at the BER of 10^{-7}. The results in Fig. 7 also show that the gains increase as the modulation size increases. The reason for this is that an interleaver is not considered in our simulations. Due to the non-uniform distribution of the reliabilities in the large modulation size, the probability of burst errors becomes higher with respect to 2-PAM. As a result, miscorrections are more likely to happen, which makes the SABM algorithm work more effectively.

Fig. 8 shows the results for SCCs with component codes with more than 2-error-correcting capability. The BCH parameters we considered are $(256, 231, 3)$[6] and $(256, 223, 4)$.[7] The resulted SCC code rates are $R = 0.80$ and $R = 0.74$, respectively. The LLR threshold is still set to $\delta = 10$. The modulation format is 2-PAM. Compared to standard SCC decoding, the achieved additional gains for $R = 0.80$ and $R = 0.74$ are 0.18 dB and 0.10 dB at the BER of 10^{-5}, respectively. The reason why the gains become less (compared to Fig. 6 in which $t = 2$) is the reduction of the probability of miscorrection, as the error correcting capability t increases. This can be verified by the much closer two curves of MF and standard decoding. When compared to [18], the achieved additional gain is smaller. Please note that, the parameters we used are slightly different from [18] in which the SCC code rate is $R = 0.81$ with component code of BCH$(254, 230, 3)$. Nevertheless, the proposed SABM algorithm can still achieve almost half of the gain achievable by the genie decoder with this structure.

C. Complexity Analysis

In general, since the methods proposed in [14]–[16], [25] are fully HD-based and [24] is fully SD-based, the corresponding complexities are thought to be quite low and high, respectively. The methods proposed in [17]–[21] as well as our SABM algorithm are the combination of hard and soft decoding, their complexity is typically higher than that in [14]–[16], [25], and lower than that in [24]. Both of the methods proposed in [20], [21] and the SABM algorithm need to partially sort the bits according to the reliability, and call the BDD decoder multiple times to decode each of the component code. These two factors are also key contributors to the complexity of the proposed SABM algorithm. With respect to these two aspects, a rough discussion on the complexity of the SABM algorithm is made in the following.

To mark HUBs, we need to partially sort the reliability list with the indices corresponding to the bits in each row of the SCC block. A usual solution to this problem is heapsort [31, Sec. 9.2]. The complexity is $O(w\log(d_0 - t - 1))$, which

[6]The generator polynomial is $g(x) = (x^8 + x^6 + x^5 + x^4 + 1)(x^8 + x^7 + x^6 + x^4 + x^3 + x^2 + 1)(x^8 + x^7 + x^4 + x^3 + x^2 + x^1 + 1)(x + 1)$.

[7]The generator polynomial is $g(x) = (x^8 + x^6 + x^5 + x^4 + 1)(x^8 + x^7 + x^6 + x^4 + x^3 + x^2 + 1)(x^8 + x^7 + x^4 + x^3 + x^2 + x^1 + 1)(x^8 + x^5 + x^3 + x^2 + 1)(x + 1)$.

returns the sorted $d_0 - t - 1$ smallest elements out of w absolute LLRs. The occupied memory is minimal because only $d_0 - t - 1$ absolute LLRs in each row need to be stored. In fact, only the HUBs in the last block are utilized within the decoding window, and thus, the required additional memory to make the BF work is the storage of $w(d_0 - t - 1)$ absolute values of LLRs in total. Moreover, the SABM decoder only utilizes the relatively reliable relationship among the bits. This relaxes the requirement on the precision of the LLRs. Thus, it to some extent allows to reduce the LLR resolution without significant performance degradation, as long as the relative reliable relationship among the bits still stands. The relation between the achieved gain and the LLR resolution is another topic of future work. With regard to HRBs, it only needs 1-bit additional memory to record if a bit has a higher reliability than the threshold δ. Note that the marked information is not updated within all iterations. Some aspects of the implementation of the SABM algorithm have also been discussed in [32].

The number of calls to the component BDD decoder is also a key factor defining the complexity and latency for iterative decoding of SCCs. In order to deal with BDD decoding failures and miscorrections, the SABM algorithm needs to call the component BDD decoder multiple times (once after every BF operation). These additional calls will increase the SCC decoding complexity and latency. To quantify this, we estimate the average number of calls to the component BDD decoder within one decoding window. The relative complexity increase caused by the SABM algorithm with respect to standard SCC decoding is thus given by

$$\eta \triangleq \frac{\overline{N} - N_{\text{sd}}}{N_{\text{sd}}} = \frac{\overline{N} - w(L-1)\ell}{w(L-1)\ell}, \tag{4}$$

where $\overline{N}$ and N_{sd} are the number of BDD calls for the SABM algorithm and for the standard SCC decoding, respectively. In what follows we estimate the value of η in (4) by estimating the average $\overline{N}$ using the first $10,000$ decoding windows.

Fig. 9 (a) shows the relative complexity increase η under different LLR threshold δ. The SNRs are 6.98 dB, 6.72 dB and 7.87 dB, which result in a post-SCC BER of 10^{-4} under $\delta = 10$, 2-PAM, and code rates $R = 0.87$, 0.83 and 0.92, respectively. The number of calls to BDD for the standard SCC decoding are $N_{\text{sd}} = 7168$, 6384 and 14112 for code rates $R = 0.87$, 0.83 and 0.92, respectively. The black fitted curve in Fig. 9 (a) is used to better show the trend of the increased complexity of SCC with $R = 0.83$. The other two code rates show a similar trend (not shown in this figure). The results in Fig. 9 (a) show that the relative complexity increase around the optimum LLR threshold δ^* ($\delta^* \approx 10$ for $R = 0.87$, $\delta^* = 12$ for $R = 0.83$ and 0.92) is the least, and is only around 4%. As explained in Sec. IV-A, if δ is too small, more outputs of BDD will be mistakenly identified as miscorrections. Consequently, the SABM-based staircase decoder will recall BDD for each marked miscorrection to try to decode it, thus lead to an increased additional calls to BDD. On the other hand, if δ is higher than the optimum threshold δ^*, there are less bits marked as HRB, and thus, miscorrections cannot be identified effectively. More errors

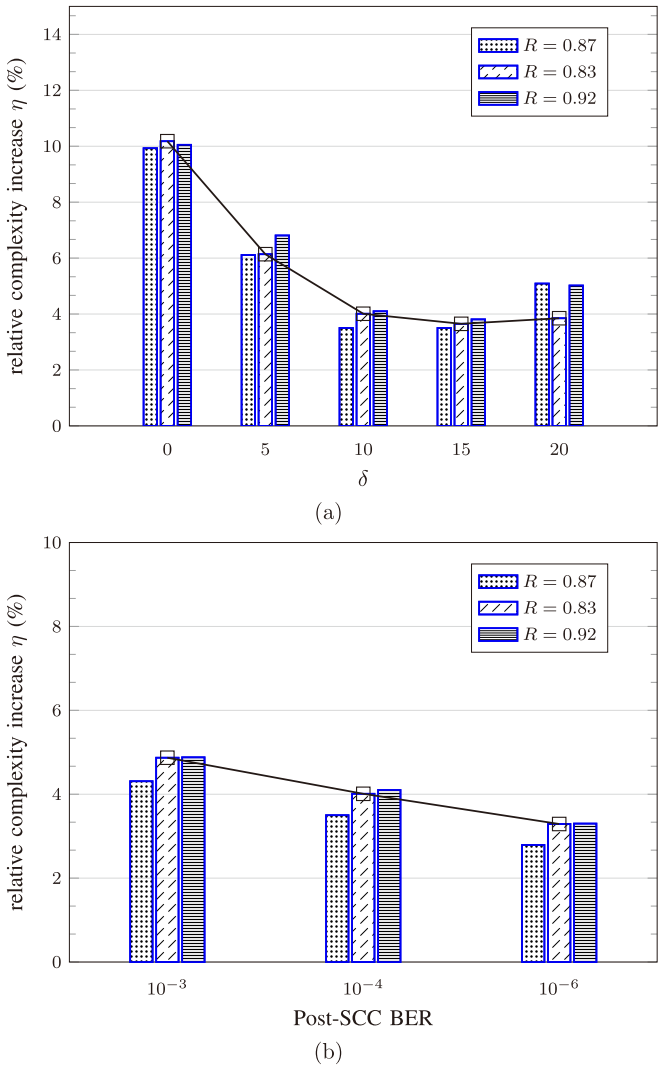


Fig. 9. (a) The relative complexity increase η vs. δ with $L = 9$, $\ell = 7$ and 2-PAM. The SNRs are 6.98 dB, 6.72 dB and 7.87 dB for $R = 0.87$, 0.83 and 0.92, respectively. (b) The relative complexity increase η vs. post-BER with $\delta = 10$, $L = 9$, $\ell = 7$ and 2-PAM.

(caused by miscorrections) will then be added to the received sequences. As a consequence, decoding failure happens more often in the following iterations. Similarly, the SABM-based staircase decoder will recall BDD to try to decode each BDD decoding failure. Therefore, the complexity increases slightly in this case too.

Fig. 9 (b) shows the relative complexity increase η of the SABM algorithm under different post-SCC BER. Similarly to Fig. 9 (a), the black fitted curve is used to better show the trend of the increased complexity of SCC with $R = 0.83$. The LLR threshold used was $\delta = 10$. When the SNR increases, there are less errors in the received sequence and most of the time BDD can deal with them successfully. Therefore, the case of decoding failure or miscorrection happens less frequently, leading to a decreased number of additional calls to BDD in the SABM algorithm. This effect is shown in Fig. 9 (b), where the relative complexity increase reduces as the channel condition improves. In the asymptotic case (SNR tending to infinity), the total number of BDD calls in the SABM

algorithm will approach that of the standard SCC decoding, and thus, $\eta \to 0$.

V. Extension to Product Codes

A product code is a set of square arrays of size $n_c \times n_c$, constructed in such a way that every row or column is an allowed codeword in some component (block) code (n_c, k_c, t) [33]. Multiple algorithms have been recently proposed to improve the decoding performance of PCs while keeping a manageable decoding complexity, as in e.g. [5], [15], [20], [34]–[37]. The algorithm we introduced in this paper can also be used, with slight modifications, to improve conventional decoding of PCs. In this section, first we show how to modify the SABM algorithm presented in Sec. III to suit PCs, and second we illustrate the gains achieved with this improved approach.

A. Modification to the SABM Algorithm for PC Decoding

In the SCC case, both MD and BF are applied only to the last block in the decoding window exploiting the channel reliabilities (LLRs). This is justified by the fact that the last block contains less reliable bits as no previous decoding iterations were perfomed on it. Differently from SCCs, in the PC case, row and column decoding are performed iteratively within the same block. As a result, no bits within each block can be regarded as more or less reliable than others, and conflicts between column and row decoding are likely to arise. Thus, one may expect to obtain gains only when MD and BF is performed within the first decoding iterations.

In particular, we have analyzed the performance of our algorithm and found it needs to be modified as follows. MD and BF operations should only be performed within the first decoding iteration and the first half of the second iteration (row decoding). Extending beyond the second iteration was observed to degrade the decoding performance, hypothetically due to conflicts between row and column decodings. Furthemore, the BF is only adopted in case of decoding failure (HUB flipping) and not in the case of miscorrection. As for the row decoding operated in the first iteration, MD is only operated based on the marked HRBs, since no previous information on the codeword syndromes is available from the decoder. From the first column decoding onwards MD is based on both bit marking or syndrome information. The reliability threshold to mark the bits was also optimized for the PC case and the optimal value was found to be identical to the case of SCC, $\delta = \delta^* = 10$.

B. Post-BER Performance Analysis

We consider 3 different PCs based on 1-bit extended BCH codes as component codes with the following parameters $(128, 113, 2)$, $(256, 239, 2)$, and $(512, 493, 2)$.[8] These parameters result in a 128×128, 256×256, and 512×512 PC code arrays with overall code rate $R = 0.78$, 0.87 and 0.93, respectively. In order to compare with the existing algorithms, the parameters of PCs with $R = 0.78$ and $R = 0.87$ are the same as that in [15, Fig. 2] and in [21, Fig. 3], respectively.

[8]The generator polynomial of BCH (128, 113, 2) is $g(x) = (x^7 + x + 1)(x^7 + x^5 + x^3 + x + 1)(x + 1)$, while that of BCH $(256, 239, 2)$ and BCH $(512, 493, 2)$ are the same as that in footnote 2 and 4, respectively.

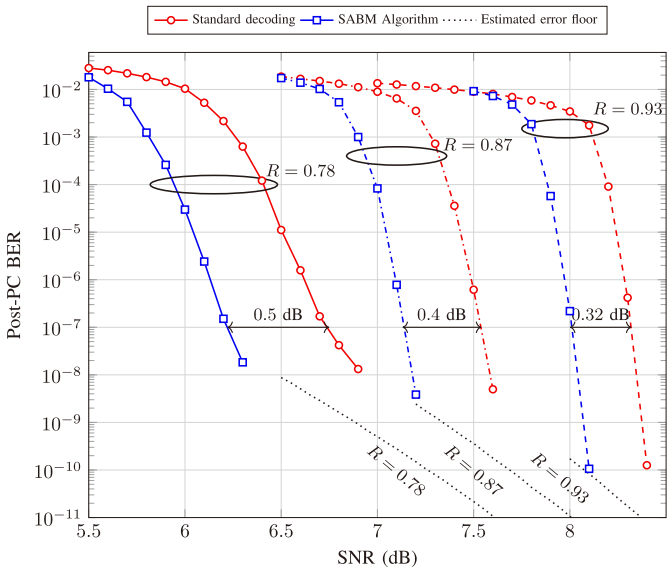


Fig. 10. Post-PC BER vs. SNR for code rates $R = 0.78$, 0.87 and 0.93 and with 2-PAM. Black dotted lines are the estimated error floors of standard PC decoding.

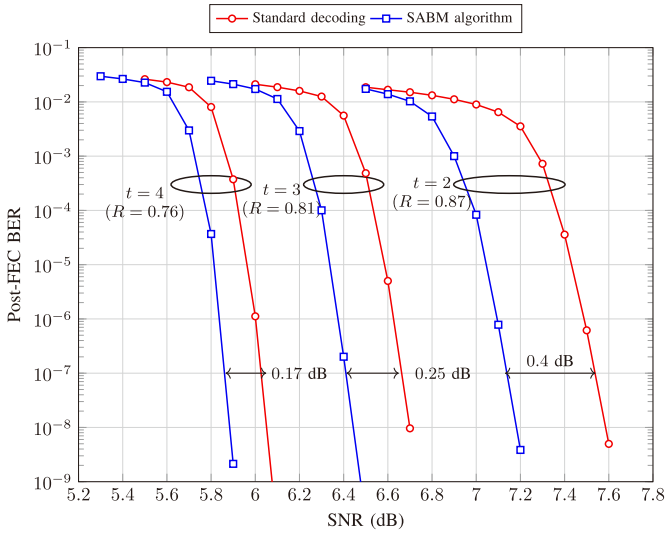


Fig. 11. Post-PC BER vs. SNR for variable error correcting capability t and fixed block length $n_c = 256$.

The results are shown in Fig. 10 for an AWGN channel and for a 2-PAM modulation format. The black dotted curves show the estimated error floor calculated by using (3) but with $M_{\min} = \binom{w}{t+1}^2$ [15, Eq. (7)]. For PCs with code rates of $R = 0.78$, 0.87 and 0.93, $w = 128$, 256, and 512, respectively. When compared to standard PC decoding, the achieved gains at BER of 10^{-7} are 0.5 dB, 0.4 dB, and 0.32 dB for $R = 0.78$, 0.87 and 0.93, respectively. In particular, the 0.5 dB gain of SABM on PC with $R = 0.78$ is 0.1 dB larger than that in [15, Fig. 2] (0.40 dB).

The 0.4 dB gain of SABM for PC with $R = 0.87$ also outperforms that in [17], but is smaller than the generalized minimum distance-based decoders in [20], [21] (the gains in [17], [20], [21] are respectively 0.27 dB, 0.58 dB and 0.51 dB as given in [21, Table I]). An efficient way to update the reliabilities through different BDD iterations can further improve the performance of the SABM algorithm. This has been demonstrated in our latest work in [38].

Finally, we evaluate the performance of SABM for more pragmatic BCH component codes, such as BCH with error correcting capabilities $t = 3$ and $t = 4$. These component codes are of particular interest as they show, within a PC structure, a much lower error floor than the case $t = 2$. The PC post-FEC BER for BCH(256,231,3) ($R = 0.81$) and BCH(256,223,4) ($R = 0.76$), as well as BCH(256,239,2), is shown in Fig. 11. The results show a diminishing SABM coding gain, which decreases from 0.4 dB in the case of $t = 2$ to 0.17 dB in the case $t = 4$. This vanishing gain is to be attributed to the structure of the component code which substantially reduces the probability of miscorrection as t increases. However, even for large error correcting powers ($t = 4$) the SABM gain is still appreciable, due to its ability to correct some of the decoding failures.

VI. Conclusion

In this paper, a novel decoding algorithm for staircase codes was proposed. This algorithm is based on simple modification of the standard hard-decision-based staircase decoder and relies on the idea of marking bits. The algorithm consists of an improved miscorrection-detection mechanism and a bit-flipping operation to effectively prevent miscorrections and increase the error correcting performance of bounded-distance decoding. Large gains compared to standard SCC decoding were obtained with a very low added complexity. The proposed algorithm was also extended to product codes with a similar performance improvement. Future works include a detailed implementation analysis, a detailed experimental verification as well as a thorough theoretical analysis.

Acknowledgment

The author Yi Lei would like to thank China Scholarship Council (CSC) for supporting her study in The Netherlands, and Alireza Sheikh (TU/e) for the kind discussions on the iBDD-SR.

References

[1] *Forward Error Correction for Submarine Systems*, document ITU-T Rec. G.975, ITU, Oct. 2000.

[2] *Forward Error Correction for High Bit-Rate DWDM Submarine Systems*, document ITU-T Rec. G.975.1, Feb. 2004.

[3] M. Camera, B.-E. Olsson, and G. Bruno, "Beyond 100 Gbit/s: System implications towards 400 G and 1T," in *Proc. Eur. Conf. Opt. Commun. (ECOC)*, Turin, Italy, Sep. 2010, pp. 1–15.

[4] X. Zhou and L. Nelson, "Advanced DSP for 400 Gb/s and beyond optical networks," *J. Lightw. Technol.*, vol. 32, no. 16, pp. 2716–2725, Aug. 15, 2014.

[5] J. Justesen, "Performance of product codes and related structures with iterated decoding," *IEEE Trans. Commun.*, vol. 59, no. 2, pp. 407–415, Feb. 2011.

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

[7] L. M. Zhang and F. R. Kschischang, "Staircase codes with 6% to 33% overhead," *J. Lightw. Technol.*, vol. 32, no. 10, pp. 1999–2002, May 15, 2014.

[8] *Implementation Agreement 400 ZR*, Opt. Internetworking Forum, Washington, DC, USA, Jan. 2018.

[9] *OTU4 Long-Reach Interface*, document ITU-T Rec. G.709.2/Y.1331.2, ITU, Jul. 2018.

[10] C. Fougstedt and P. Larsson-Edefors, "Energy-efficient high-throughput VLSI architectures for product-like codes," *J. Lightw. Technol.*, vol. 37, no. 2, pp. 477–485, Jan. 15, 2019.

[11] M. Barakatain and F. R. Kschischang, "Low-complexity concatenated LDPC-staircase codes," *J. Lightw. Technol.*, vol. 36, no. 12, pp. 2443–2449, Jun. 15, 2018.

[12] B. Chen, Y. Lei, D. Lavery, C. Okonkwo, and A. Alvarado, "Rate-adaptive coded modulation with geometrically-shaped constellations," in *Proc. Asia Commun. Photon. Conf. (ACP)*, Hangzhou, China, Oct. 2018, pp. 1–3.

[13] I. Reed and X. Chen, *Error-Control Coding for Data Networks*, 1st ed. New York, NY, USA: Academic, 1999.

[14] B. P. Smith, "Error-correcting codes for fibre-optic communication systems," Ph.D. dissertation, Dept. Elect. Comput. Eng., Univ. Toronto, Toronto, ON, Canada, 2011.

[15] 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.

[16] C. Häger and H. D. Pfister, "Miscorrection-free decoding of staircase codes," in *Proc. Eur. Conf. Opt. Commun. (ECOC)*, Gothenburg, Sweden, Sep. 2017, pp. 1–3.

[17] 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)*, Sep. 2018, pp. 1–3.

[18] A. Sheikh, A. Graell i Amat, and G. Liva, "Binary message passing decoding of product-like codes," *IEEE Trans. Commun.*, to be published.

[19] C. Fougstedt, A. Sheikh, A. Graell i Amat, G. Liva, and P. Larsson-Edefors, "Energy-efficient soft-assisted product decoders," in *Proc. Opt. Fiber Commun. Conf. Expo. (OFC)*, San Diego, CA, USA, Mar. 2018, pp. 1–3.

[20] 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 10th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Hong Kong, Dec. 2018, pp. 1–5.

[21] A. Sheikh, A. Graell i Amat, and G. Liva, "Binary message passing decoding of product codes based on generalized minimum distance decoding: (Invited paper)," in *Proc. 53rd Annu. Conf. Inf. Sci. Syst. (CISS)*, Baltimore, MD, USA, Mar. 2019, pp. 1–5.

[22] D. Chase, "Class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 1, pp. 170–172, Jan. 1972.

[23] S. Lin and D. J. Costello, *Error Control Coding*. Hoboken, NJ, USA: Pearson, 2004.

[24] X. Dou, M. Zhu, J. Zhang, and B. Bai, "Soft-decision based sliding-window decoding of staircase codes," in *Proc. IEEE 10th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Hong Kong, Dec. 2018, pp. 1–5.

[25] L. Holzbaur, H. Bartz, and A. Wachter-Zeh, "Improved decoding and error floor analysis of staircase codes," *Des. Codes Cryptogr.*, vol. 87, nos. 2–3, pp. 647–664, 2018.

[26] Y. Lei *et al.*, "Decoding staircase codes with marked bits," in *Proc. IEEE 10th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Hong Kong, Dec. 2018, pp. 1–5.

[27] B. Chen, Y. Lei, S. van der Heide, J. van Weerdenburg, A. Alvarado, and C. Okonkwo, "First experimental verification of improved decoding of staircase codes using marked bits," in *Proc. Opt. Fiber Commun. Conf. Expo. (OFC)*, San Diego, CA, USA, Mar. 2019, pp. 1–3.

[28] L. Szczecinski and A. Alvarado, *Bit-Interleaved Coded Modulation: Fundamentals, Analysis and Design*. Chichester, U.K.: Wiley, 2015.

[29] Y. Cai *et al.*, "FPGA investigation on error-floor performance of a concatenated staircase and Hamming code for 400 G-ZR forward error correction," in *Proc. Opt. Fiber Commun. Conf. Expo. (OFC)*, San Diego, CA, USA, Mar. 2018, pp. 1–3.

[30] Y. Cai *et al.*, "FPGA investigation on error-flare performance of a concatenated staircase and Hamming FEC code for 400 G inter-data center interconnect," *J. Lightw. Technol.*, vol. 37, no. 1, pp. 188–195, Jan. 1, 2019.
[31] N. M. Josuttis, *The C++ Standard Library: A Tutorial and Reference*. Reading, MA, USA: Addison Wesley, 1999.
[32] A. Alvarado, G. Liga, Y. Lei, B. Chen, and A. Balatsoukas-Stimming, "Improving HD-FEC decoding via bit marking," in *Proc. 24th Optoelectron. Commun. Conf. (OECC)*, Fukuoka, Japan, Jul. 2019, pp. 1–3.
[33] P. Elias, "Error-free Coding," *Trans. IRE Prof. Group Inf. Theory*, vol. 4, no. 4, pp. 29–37, Sep. 1954.
[34] T. Mittelholzer, T. Parnell, N. Papandreou, and H. Pozidis, "Improving the error-floor performance of binary half-product codes," in *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Nov. 2016, pp. 295–299.
[35] C. Condo, F. Leduc-Primeau, G. Sarkis, P. Giard, and W. J. Gross, "Stall pattern avoidance in polynomial product codes," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Dec. 2016, pp. 699–702.
[36] S. Sridharan, M. Jarchi, and T. Coe, "Product code based forward error correction system," U.S. Patent 6 810 499 B2, Oct. 26, 2004. [Online]. Available: https://www.lens.org/lens/patent/059-746-112-729-392
[37] J. Justesen and T. Høholdt, "Analysis of iterated hard decision decoding of product codes with Reed–Solomon component codes," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Tahoe City, CA, USA, Sep. 2007, pp. 174–177.
[38] G. Liga, A. Sheikh, and A. Alvarado, "A novel soft-aided bit-marking decoder for product codes," presented at the Proc. 45th Eur. Conf. Opt. Commun. (ECOC), Dublin, Ireland, Sep. 2019.

Yi Lei was born in Chongqing, China. She received the B.E. degree in electronic information and technology program from Beijing Forestry University, Beijing, China, in 2013, and the Ph.D. degree in electronic science and technology from the Beijing University of Posts and Telecommunications, Beijing, in 2019.

She held a visiting Ph.D. position at Signal Processing Systems (SPS) Group, Department of Electrical Engineering, Eindhoven University of Technology (TU/e), Eindhoven, The Netherlands, from August 2017 and August 2018. Since August 2018, she has been a Guest Researcher with TU/e. Since March 2019, she has also been with the Hefei University of Technology, as a Lecturer. Her current research interests include optical communication systems, fiber-wireless integration, MIMO, channel coding, and signal processing.

Bin Chen (S'14–M'17) was born in Hefei, China, in 1989. He received the B.Sc. degree in electronic information science and technology from the Hefei University of Technology, Hefei, in 2010, and the Ph.D. degree in electronics and communications engineering from University College Dublin, Dublin, Ireland, in 2015, funded by the China Scholarship Council (CSC) and Science Foundation Ireland (SFI).

He is a Lecturer with the Hefei University of Technology. From 2016 and 2019, he held a post-doctoral position with the Signal Processing Systems (SPS) Group and the Electro-Optical Communication Systems (ECO) Group, Department of Electrical Engineering, Eindhoven University of Technology (TU/e), Eindhoven, The Netherlands. Since 2019, he has also been a Guest Researcher with TU/e. His research interests include developing practical and novel schemes for digital communication systems related to coded modulation, information theory, channel coding, network coding, and signal processing.

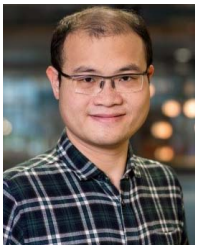
Dr. Chen was a recipient of the multiple awards, including the CSC Scholarship, the 2018 Asia Communications and Photonics Conference (ACP2018) Best Paper Award, and the 2019 Optoelectronics and Communications Conference (OECC2019) Best Paper Award.

Gabriele Liga (S'12–M'17) was born in Palermo, Sicily, Italy, in 1983. He received the B.Sc. degree (Laurea triennale) in telecommunications engineering from the Università degli Studi di Palermo in 2005, the M.Sc. degree in telecommunications engineering (Laurea specialistica) from the Politecnico di Milano in 2011, and the Ph.D. degree in optical communications from Optical Networks Group, University College London, U.K., in 2017.

From 2017 to 2018, he was a Post-Doctoral Research Associate with the Optical Networks Group, University College London, U.K. Since 2018, he has been a Research Fellow, under a Marie Curie Eurotech Fellowship, with the Signal Processing Systems (SPS) Group, Department of Electrical Engineering, Eindhoven University of Technology (TU/e), Eindhoven, The Netherlands. His research interests embrace the areas of digital communications, mathematical modeling, and information theory applied to optical fibre transmission. He currently serves as a Reviewer for several scientific journals in the area of optics, such as IEEE JOURNAL OF LIGHTWAVE TECHNOLOGY, IEEE TRANSACTIONS ON COMMUNICATIONS, *OSA Optics Express*, and IEEE PHOTONICS TECHNOLOGY LETTERS.

Xiong Deng (S'13–M'18) received the M.Eng. degree in communication and information engineering from the University of Electronic Science and Technology of China in 2013 and the Ph.D. degree in optical wireless communications from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 2018.

In 2013, he was a Researcher with the Terahertz Science and Technology Research Center, China Academy of Engineering Physics, where he was involved in the integrated terahertz communication and imaging system. He was a Guest Researcher with Signify (Philips Lighting) Research, where he was involved in light fidelity. He is currently a Post-Doctoral Researcher with the Eindhoven University of Technology. His research interests include the system modeling, digital signal processing, and circuits for intelligent lighting, millimeter wave, radio over fiber, and optical wireless communications.



Zizeng Cao received the Ph.D. degree *(cum laude)* from the Eindhoven University of Technology in 2015. He was a Post-Doctoral Researcher with Eindhoven University of Technology (TU/e), where he has been appointed as an Assistant Professor. He is currently a tenured Assistant Professor with the ECO Group, Institute for Photonic Integration (IPI), Eindhoven University of Technology (TU/e), The Netherlands. He has more than ten years of research experience on optical communication system design, high speed digital signal processing, and the design, fabrication, and characterization of photonics integrated circuit in multiple platform, including SOI, SiN, and InP. His current research interests include indoor optical communications, microwave photonics, and photonic integration. He serves as a member of the Technical Program Committee (TPC) for the European Conference on Optical Communication (ECOC). He was a recipient of the IEEE Photonics Society Graduate Student Fellowship 2014. He serves as an active reviewer of the IEEE/OSA journals.

Jianqiang Li (M'09–SM'15) received the B.E. and Ph.D. degrees from the Beijing University of Posts and Telecommunications in 2005 and 2009, respectively. In July 2009, he joined Fujitsu Research and Development Center, as a Researcher, working on coherent optical transmission systems. He was with the Chalmers University of Technology, Gothenburg, Sweden, as a Post-Doctoral Researcher from June 2011 and April 2012. Since May 2012, he has been an Associate Professor with the Beijing University of Posts and Telecommunications. In July 2018, he joined Alibaba Group U.S. Inc., where he is working on intelligent optical networks for data centers. He has authored or coauthored more than 100 conference and journal articles and one book and holds two U.S. issued patents.

Kun Xu received the B.Sc. degree in applied physics from the Central South University of Technology (currently Central South University), China, in 1996, the M.Sc. degree in optical engineering from the University of Electronic Science and Technology, China, in 1999, and the Ph.D. degree in physical electronics from Tsinghua University, China, in 2002.

Then, he joined the Beijing University of Posts and Telecommunications, China. He was a Visiting Scholar with Nanyang Technological University, Singapore, in 2004. He is currently a Professor with the Institute of Information Photonics and Optical Communications, Beijing University of Posts and Telecommunications. He is an author or coauthor of 339 publications and conference presentations, one book, and one book chapter, and holds more than 50 patents. His research interests include fiber-wireless integrated networks, distributed antenna systems, ubiquitous wireless sensing and access, RF photonic integrated systems, artificial intelligence and photonics neural networks, machine learning, and ultrafast optics.

Dr. Xu has served for technical program committees (TPC) and workshop/session co-chairs of several international conferences, including the IEEE Microwave Photonics/Asia–Pacific Microwave Photonics Conference (MWP/APMP), the IEEE Global Symposium on Millimeter Waves (GSMM), the IEEE International Conference on Communications (ICC), and Progress in Electromagnetics Research Symposium (PIERS). He was also the TPC Co-Chair of 2015 IEEE/OSA/SPIE Asia Communications and Photonics Conference (ACP 2015 and ACP 2018) and the LOC Chair of ACP 2013 and APMP 2009. He was a Guest Editor of the Special Issue on Microwave Photonics in *Photonics Research* (OSA/CLP) in 2014.

Alex Alvarado (S'06–M'11–SM'15) was born in Quellón, Chiloé, Chile. He received the Electronics Engineer degree (Ingeniero Civil Electrónico) and the M.Sc. degree (Magíster en Ciencias de la Ingeniería Electrónica) from Universidad Técnica Federico Santa María, Valparaíso, Chile, in 2003 and 2005, respectively, and the degree of Licentiate of Engineering (Teknologie Licentiatexamen) and the Ph.D. degree from the Chalmers University of Technology, Gothenburg, Sweden, in 2008 and 2011, respectively.

From 2014 to 2016, he was a Senior Research Associate with the Optical Networks Group, University College London, U.K. From 2012 to 2014, he was a Marie Curie Intra—European Fellow, and from 2011 to 2012, he was a Newton International Fellow with the University of Cambridge, U.K. He is an Associate Professor with the Signal Processing Systems (SPS) Group, Department of Electrical Engineering, Eindhoven University of Technology (TU/e), The Netherlands. Since 2018, he has been a member of the TU/e Young Academy of Engineering. His current research is funded in part by the Netherlands Organization for Scientific Research (NWO) via a VIDI grant, as well as by the European Research Council (ERC) via an ERC Starting grant. His general research interests are in the areas of digital communications, coding, and information theory.

Dr. Alvarado's research has received multiple awards, including the best paper awards at the 2018 Asia Communications and Photonics Conference and at the 2019 OptoElectronics and Communications Conference, and the best poster awards at the 2009 IEEE Information Theory Workshop and at the 2013 IEEE Communication Theory Workshop. He was also a recipient of the 2015 IEEE TRANSACTIONS ON COMMUNICATIONS Exemplary Reviewer Award, and the 2015 *Journal of Lightwave Technology* Best Paper Award, honoring the most influential, highest-cited original article published in the journal in 2015. He also served for the ECOC 2019 Subcommittee Theory of Optical Communications (SC5). Since 2018, he has been serving in the OFC Subcommittee Digital and Electronic Subsystems (S4). He served as an Associate Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS (Optical Coded Modulation and Information Theory) from 2016 to 2018.