Error-Aware SCFlip Decoding of Polar Codes

DAEYOEOL YANG, (Associate Member, IEEE), AND KYEONGCHEOL YANG, (Senior Member, IEEE)
Department of Electrical Engineering, Pohang University of Science and Technology (POSTECH), Pohang 37673, South Korea
Corresponding author: Kyeongcheol Yang (kcyang@postech.ac.kr)
This work was supported in part by the National Research Foundation (NRF) of Korea funded by the Ministry of Science and Information and Communication Technology (MSIT) of the Korea Government under Grant 2019R1A2B5B03069970; and in part by the Institute for Information and communications Technology Promotion (IITP) funded by the MSIT of the Korea Government under Grant 2020(2016-0-00123).

ABSTRACT
The successive-cancellation flip (SCFlip) decoder and its variants provide a significant coding gain with the average complexity practically identical to that of the successive cancellation (SC) decoder in a wide range of signal-to-noise ratios (SNRs). But, they suffer from high complexity and long latency when the SNR decreases, since the average number of extra decoding attempts becomes inevitably large. To mitigate this problem, we propose a novel SCFlip decoder, called an error-aware SCFlip (EA-SCFlip) decoder, for distributed cyclic-redundancy-check (CRC) polar codes. Based on the distributed CRC bits, it employs early termination at each extra decoding attempt so that it can reduce the decoding complexity and latency on the average. It also reduces the search space of candidate bit-flips in the dynamic building of the bit-flip list by exploiting the parity-check relationship (PCR) of the first error-detected CRC bit at each extra decoding attempt. Furthermore, we propose a greedy algorithm to design a distributed CRC code such that the obtained PCRs make the early-error-detection capability of the EA-SCFlip decoder as high as possible. Numerical results demonstrate that the EA-SCFlip decoder can indeed achieve an early termination gain as well as a complexity reduction, when a polar code is concatenated with the distributed CRC code designed by the proposed algorithm.

INDEX TERMS
Polar codes, convolutional polar codes, distributed CRC, early termination, SCFlip decoding.

I. INTRODUCTION
Polar codes, introduced by Arikan [1], are the first class of structured error-correcting codes that are proved to achieve the capacity of an arbitrary binary-input discrete memoryless channel asymptotically with an exponent of \(1/2\) under successive cancellation (SC) decoding. However, they have quite poor finite-length performance due to the presence of imperfectly polarized bit-channels and the suboptimality of SC decoding. To improve their performance, several decoding methods including SC list (SCL) decoding [2], [3], SC stack (SCS) decoding [4], and SC flip (SCFlip) decoding [5] have been developed as well as various code constructions [6], [7] have been suggested. As a result, polar codes were recently adopted as a channel coding scheme for the control channel in 5G New Radio (NR) [8].

Cyclic redundancy check (CRC) codes were introduced by Tal and Vardy [2], [3] in order to improve the performance of polar codes under SCL decoding. Conventionally, the CRC bits are placed at the end of the information block. On the other hand, Chen et al. [9] proposed distributed CRC codes by introducing an interleaver between the CRC encoder and the polar encoder in order to make CRC bits scattered over the information block. These distributed CRC bits can be used in the early termination process to reduce the decoding latency or in the tree pruning process to improve the performance of SCL decoding. Later, some greedy algorithms to design an interleaver for distributed CRC codes were presented in [10], [11] so that the CRC bits are placed at the earlier positions of the information block. For short notation, a polar code concatenated with a (distributed) CRC code will be referred to as a (distributed) CRC-polar code in this article.

Although the SCL decoder provides outstanding performance of a CRC-polar code, it suffers from high complexity and long latency. As an alternative, the SC flip (SCFlip) decoder [5] performs standard SC decoding and possibly extra decoding at most \(T\) times until the CRC constraints are satisfied. Later, several methods to achieve a complexity reduction or a performance improvement have been investigated [12]–[16]. In particular, dynamic SCFlip (D-SCFlip)
decoding [17] is a novel method to boost the code performance by introducing a new metric to a bit-flip, allowing multi-bit flipping and updating a bit-flip list dynamically. Recently, some modifications of the D-SCFlip decoder were proposed. A simplified D-SCFlip decoder was proposed in [18] to improve the throughput by exploiting easily-decoded subcodes such as rate-0, rate-1 and repetition nodes. In [19], the bit error rates estimated via Gaussian approximation were used to reduce the search space for updating the bit-flip list.

The SCFlip decoder and its variants only perform standard SC decoding for most of the received frames at a middle-to-high signal-to-noise ratio (SNR) region. That is, they provide a significant coding gain with the average complexity practically identical to that of the SC decoder in a wide range of SNRs. But, they require a number of extra decoding attempts at a low SNR region. This phenomenon gets worse when the maximum number of extra decoding attempts is allowed to be sufficiently large for a performance improvement. Finally, they suffer from high complexity and long latency as the SNR decreases.

In order to mitigate the complexity and latency problem, we propose a novel SCFlip decoder, called an error-aware SCFlip (EA-SCFlip) decoder, for a distributed CRC-polar code. Whenever an error is detected by one of the distributed CRC bits at each extra decoding attempt, the proposed EA-SCFlip decoder early terminates the corresponding decoding procedure so that it can reduce the decoding complexity and latency on the average. Given an error-detected CRC bit at a decoding attempt, it also makes use of the corresponding parity-check relationship (PCR) to limit the search space of candidate bit-flips. This leads to reducing the complexity of updating a bit-flip list at each extra decoding attempt. Furthermore, in order to maximize the effect of early error detection in the EA-SCFlip decoder, we propose a greedy algorithm to design a distributed CRC code having a high early-error-detection capability. The proposed algorithm tries to find the parity-check matrix (PCM) for a new equivalent CRC code by applying a number of elementary row operations and column permutations to the PCM for a CRC code with a given generator polynomial. Numerical results demonstrate that the EA-SCFlip decoder can indeed achieve an early termination gain as well as a complexity reduction, when a polar code is concatenated with the distributed CRC code designed by the proposed algorithm.

The rest of paper is organized as follows. Section II provides a background on polar codes, the SC decoder and the SCFlip decoder. In Section III, we propose a greedy algorithm to design a distributed CRC code having a high early-error-detection capability. The proposed EA-SCFlip decoder is presented in Section IV. In Section V, we evaluate the early termination gain of the proposed EA-SCFlip decoder and analyze its computational complexity. Finally, we give concluding remarks in Section VI.

Notation: We use calligraphic letters (e.g. \( \mathcal{A} \)) to denote sets. Conventionally, we use \( \mathcal{A}^{c} \) to denote the complementary set of \( \mathcal{A} \). We write boldface lowercase letters (e.g. \( \mathbf{a} \)) to stand for vectors, and write boldface uppercase letters (e.g. \( \mathbf{A} \)) to denote matrices.

II. PRELIMINARIES

A polar code is characterized by a 2-tuple \((N, \mathcal{I})\), where \( N = 2^n \) is the code length and \( \mathcal{I} = \{i_1, \ldots, i_K\} \subset \{1, \ldots, N\} \) is an information set of cardinality \( K_I = |\mathcal{I}| \). A polar codeword \( \mathbf{c} \) is obtained by

\[
\mathbf{c} = \mathbf{uG}_N, \tag{1}
\]

where \( \mathbf{G}_N \) is the generator matrix and \( \mathbf{u} \) is an encoder input vector which has \( K_I \) information bits at the indices in \( \mathcal{I} \) and \( N - K_I \) frozen bits at the indices in \( \mathcal{I}^c \). For classical polar codes [1], \( \mathbf{G}_N \) is given by the \( n \)-th Kronecker product of \( \mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \), that is, \( \mathbf{G}_N = \mathbf{F}^\otimes n \).

Given the received signals \( \mathbf{y}^N \) and the previous estimates \( \hat{u}^{i-1} \), the SC decoder for a polar code computes the estimate of \( u_i \) as

\[
\hat{u}_i = \begin{cases} 0, & \text{if } i \in \mathcal{I}^c \\ \frac{1}{2}(1 - \text{sign}(L_i)), & \text{if } i \in \mathcal{I}, \end{cases}
\]

where \( L_i \) is the decision log-likelihood ratio (LLR) of \( u_i \), defined as

\[
L_i \triangleq \log \left( \frac{\Pr \left( y_1^N, \hat{u}_i^{-1} | u_i = 0 \right)}{\Pr \left( y_1^N, \hat{u}_i^{-1} | u_i = 1 \right)} \right). \tag{2}
\]

The decision LLRs in (2) can be efficiently calculated through a polar encoding graph. Recursive formulas for computing them are available in [20].

In the SC decoder, an estimation error in \( \hat{u}_i \) results from channel noise or error propagation due to the previous estimation errors. It was reported in [5], [15] that the errors induced by channel noise are mainly concentrated within three bits, and even within two bits when the SNR increases. This tendency is observed over a wide range of rates.

The SCFlip decoder and its variants are a class of decoders which perform SC decoding and bit flipping. They aim to correct channel-induced errors, and eliminate the errors caused by error propagation in turn. Typically, they perform standard SC decoding, possibly followed by at most \( T \) extra decoding attempts until the CRC constraints are satisfied. In the SCFlip decoder [5], the absolute values of the decision LLRs obtained by the initial SC decoding are used for selecting a set of \( T \) unfrozen bit indices to be flipped. Each extra decoding attempt is done with flipping only one bit in this set.

The D-SCFlip decoder, proposed by Chandesris et al. [17], is a generalized SCFlip decoder which allows multi-bit flipping. Here, the notion of a bit-flip of order \( \omega \) was introduced. It represents a set of \( \omega \) indices, \( \mathcal{E} = \{i_1, \ldots, i_{\omega}\} \subset \mathcal{I} \) such that \( i_1 < \cdots < i_{\omega} \), to be flipped. The new metric associated with a bit-flip \( \mathcal{E} = \{i_1, \ldots, i_{\omega}\} \) of order \( \omega \), defined by

\[
M_{\alpha}(\mathcal{E}) = \sum_{i \in \mathcal{E}} |L_i| + \frac{1}{\alpha} \sum_{i \leq i', i' \in \mathcal{I}} \log(1 + \exp(-\alpha|L_i|)). \tag{3}
\]

was also proposed, where \( \alpha \) is a perturbation parameter.
The overall procedure of D-SCFlip decoding is summarized as follows. Whenever the initial standard SC decoding fails, the decoder builds a list of $T$ candidate bit-flips, $\mathcal{L}_{\text{flip}} = \{\mathcal{E}_1, \ldots, \mathcal{E}_T\}$, referred to as a ‘bit-flip list’, based on the metrics defined in (3). At the initial step, each bit-flip consists of only one unfrozen bit index, i.e., $[\mathcal{E}_i] = \omega_i = 1, \forall i$. The metric set corresponding to $\mathcal{L}_{\text{flip}}$ is denoted by $M_{\text{flip}} = \{M_\omega(\mathcal{E}_1), \ldots, M_\omega(\mathcal{E}_T)\}$. If the $t$-th decoding attempt with a bit-flip $\mathcal{E}_t = \{i_1, \ldots, i_0\}$ still fails and $\omega_t$ is less than a predetermined maximum bit-flip order $\omega$, the decoder computes $M_\omega(\mathcal{E})$ for all possible bit-flips $\mathcal{E} = \mathcal{E}_t \cup \{i\}$ such that $i \in \mathcal{I}$ and $i > i_0$, and updates both $\mathcal{L}_{\text{flip}}$ and $M_{\text{flip}}$ by selecting more likely bit-flips.

### III. DESIGN OF DISTRIBUTED CRC CODES FOR EARLY ERROR DETECTION

CRC bits are commonly placed at the end of the information block before polar encoding, but can be arranged within the information block by properly employing an interleaver [9]. This kind of CRC bits will be referred to as ‘distributed CRC bits’ in this article. Since they can be used for both early detection of decision errors and early termination of decoding, the decoding latency can be reduced. In this section, we consider how to design a distributed CRC code minimizing the decoding latency. More specifically, we propose an algorithm to construct its parity-check matrix (PCM) maximizing the latency-reduction effect for a given CRC generator polynomial. We also evaluate the early-error-detection capability of the designed code.

#### A. DESIGN PROBLEM OF A DISTRIBUTED CRC CODE

Given an $M$-bit CRC generator polynomial $g(x)$, the systematic generator matrix and PCM of a $[K + M, K]$ CRC code $C$ can be written as

$$G = \begin{bmatrix} I_K & | & P \end{bmatrix} \quad \text{and} \quad H = \begin{bmatrix} P^T & | & I_M \end{bmatrix},$$

respectively, where $I_K$ is the $K \times K$ identity matrix and $P$ is the $K \times M$ matrix determined by $g(x)$. The non-zero entries in each column of $P$ represent the information bits participating in the parity-check equation (PCE) of the corresponding CRC bit. Each row of $H$ corresponds to the PCE associated with one single CRC bit. Note that $C$ has a PCM of a cyclic form, denoted by $H^\dagger$.

Applying column/row permutations to $G$ [9], [10] or $H^\dagger$ [11], it is possible to get an equivalent CRC code whose parity bits are distributed such that their indices are higher than all the associated information bit indices. This property makes this CRC code have an early-error-detection capability. Since there are many equivalent CRC codes, it is very valuable to construct a CRC code maximizing an early-error-detection capability.

In order to do so, we first make a problem formulation. Consider a PCM $H_{\text{new}}$ of a $[K + M, K]$ distributed CRC code. Let $h_j$ be the index of the last non-zero component of the $j$-th row in $H_{\text{new}}$. We assume that $h_1 < h_2 < \ldots < h_M = K + M$, that is, $H_{\text{new}}$ is of the staircase form. Intuitively, lowering $h_j$ for all $j$ can be expected to improve the early-error-detection capability. Therefore, our problem to design a good CRC code is to construct $H_{\text{new}}$ with $h_j$ as low as possible for all $j$. As a measure of early-error-detection capability, the last index sum given by $h(H_{\text{new}}) = \sum_{j=1}^{M} h_j$ can be considered.

#### B. PROPOSED DESIGN ALGORITHM

To solve our problem, we start with a PCM $H$ such as the PCM in (4) and apply a number of elementary row operations and column permutations to it. The main difference between our method and conventional methods in [9]–[11] is whether the addition of one row to another row is allowed or not. This makes high the possibility that $h_j$ of a new PCM $H_{\text{new}}$ is smaller for all $j$.

The proposed design algorithm can be inductively described as follows. In order to determine the first row of $H_{\text{new}}$, we randomly generate linear combinations of the rows of $H$ in (4). More specifically, $L$ nonsingular matrices $U_1$ of size $M \times M$ are randomly generated and are multiplied by $H$. We then choose $U_1^\dagger$ such that the minimum row weight of $U_1 H$ is minimized over $L$ nonsingular matrices. As a next step, we permute the rows of $U_1^\dagger H$ such that the row of minimum weight is placed on the first row and then permute the columns such that the nonzero components of the first row are located to the left. The resultant matrix is denoted by $H^{(1)}$.

Now we assume that $H^{(1)}$ is fixed, that is, the first $j - 1$ rows of $H_{\text{new}}$ have already been determined. The next step is to determine the $j$-th row of $H_{\text{new}}$. Note that $h_{j-1}$ is the column index of the rightmost non-zero component in the $(j-1)$-th row of $H^{(j-1)}$. Similarly to the previous steps, we again randomly generate $L$ nonsingular matrices $U_j$ of size $M \times M$, given by

$$U_j = \begin{bmatrix} I_{j-1} & 0 & 0 \\ 0 & V \end{bmatrix},$$

where $V$ is an $(M-j+1) \times (M-j+1)$ nonsingular matrix. We then choose $U_j^\dagger$ such that the minimum nonzero row weight of $U_j^\dagger H^{(j-1)}$ is minimized over $L$ nonsingular matrices, where $H^{(j-1)} = \begin{bmatrix} H^{(j-1)}_{j-1} & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$ is the $M \times (K + M - h_{j-1})$ submatrix of $H^{(j-1)}$ formed by the columns with the indices in $\{h_{j-1} + 1, h_{j-1} + 2, \ldots, K + M\}$. Next, we permute the rows of $U_j^\dagger H^{(j-1)}$ such that the first $j-1$ rows are fixed and the newly obtained row is placed on the $j$-th row, and then permute the columns such that the nonzero components within the last part with $K + M - h_{j-1}$ components in the $j$-th row are located to the left of this part. The resultant matrix is denoted by $H^{(j)}$. We repeat this step until $j = M$ and get $H_{\text{new}}$.

The procedure of the proposed algorithm is summarized in Algorithm 1. In order to get an optimal matrix $H_{\text{new}}$, an exhaustive search needs $(2^M - 1)(2^{M-2}) \cdots (2^{M-2^{M-1}})$ trials. This number of trials increases too fast as $M$ increases. For this reason, $L$ is introduced as the parameter to control the tradeoff between complexity and goodness in our algorithm. Algorithm 1 is universal in the sense that for any given initial PCM $H$, it always outputs a new PCM $H_{\text{new}}$ which has
Algorithm 1 Construction of a PCM $\mathbf{H}_{\text{new}}$

**Input**: an input PCM $\mathbf{H}$

**Output**: an output PCM $\mathbf{H}_{\text{new}}$

1. $\mathbf{H}^{(0)} \leftarrow \mathbf{H}$;
2. $h_0 \leftarrow 0$;
3. for $j = 1, \ldots, M$ do
4.   $\Delta \leftarrow K + M - h_{j-1}$;
5.   for $s = 1, \ldots, L$ do
6.     Randomly generate a $M \times M$ nonsingular matrix $\mathbf{U}_j$ in (5).
7.     $\mathbf{H} \leftarrow \mathbf{U}_j\mathbf{H}^{(j-1)}$;
8.     for $k = j, \ldots, M$ do
9.       $\Delta_k \leftarrow \sum_{p=s+1}^{K+M} h_{k,p}$;
10.      end for
11.     $\bar{\Delta} \leftarrow \min_{k \in \{j, \ldots, M\}} \Delta_k$;
12.     if $\Delta > \bar{\Delta}$ then
13.       $\Delta \leftarrow \bar{\Delta}$;
14.       $\mathbf{U}^* \leftarrow \mathbf{U}_j$;
15.       $k^* \leftarrow \arg \min_{k \in \{j, \ldots, M\}} \Delta_k$;
16.     end if
17.   end for
18. $\mathbf{H}^{(0)} \leftarrow \mathbf{U}_j^*\mathbf{H}^{(j-1)}$;
19. Swap the $j$-th row and the $k^*$-th row of $\mathbf{H}^{(0)}$.
20. Perform proper column permutations to obtain a PCM $\mathbf{H}^{(0)}$ in staircase form.
21. $h_j \leftarrow h_{j-1} + \Delta$;
22. end for
23. $\mathbf{H}_{\text{new}} \leftarrow \mathbf{H}^{(M)}$;
24. return $\mathbf{H}_{\text{new}}$.

better performance with respect to the early-error-detection capability.

As a design example, consider the $[30 + 10, 30]$ CRC code with generator polynomial $g(x) = x^{10} + x^9 + x^5 + x^4 + x + 1$. The PCM $\mathbf{H}$ of the standard form, the PCM obtained by swapping the rows and columns of $\mathbf{H}$, and the PCM $\mathbf{H}_{\text{new}}$ obtained by applying Algorithm 1 to $\mathbf{H}$ are graphically shown in Fig. 1. The indices of the distributed CRC bits obtained from $\mathbf{H}_{\text{new}}$ are lower than those from the matrix in Fig. 1 (b). Therefore, a good distributed CRC code to detect decision errors earlier can be efficiently constructed by Algorithm 1.

Before verifying the validity of Algorithm 1, we first discuss some theoretical backgrounds on the generalized Hamming weights (GHWs) of a code $[21]–[23]$ and their relation to $\{h_1, \ldots, h_M\}$. Given a $[K+M, K]$ CRC code $\mathbf{C}$ with a PCM $\mathbf{H}$, let $\mathbf{C}^\perp$ be the dual code of $\mathbf{C}$, given by

$$\mathbf{C}^\perp \triangleq \{\mathbf{x}| \mathbf{c}^T \mathbf{x} = 0 \text{ for all } \mathbf{c} \in \mathbf{C}\}.$$

Clearly, $\mathbf{H}$ is a generator matrix for $\mathbf{C}^\perp$. For $1 \leq j \leq M$, the $j$-th GHW of $\mathbf{C}^\perp$ is defined as the minimum support size of $j$-dimensional subcodes of $\mathbf{C}^\perp$, that is,

$$d_j(\mathbf{C}^\perp) = \min(|\text{supp}(\mathbf{D})| : \mathbf{D} \text{ is a } j\text{-dimensional subcode of } \mathbf{C}^\perp).$$

where

$$\text{supp}(\mathbf{D}) = \{i | \exists (x_1, \ldots, x_{K+M}) \in \mathbf{D} \text{ such that } x_i \neq 0\}.$$  

In particular, $d_1(\mathbf{C}^\perp)$ is the minimum distance of $\mathbf{C}^\perp$. Note that the GHWs of $\mathbf{C}^\perp$ are invariant under elementary row operations and column permutations on $\mathbf{H}$. This means that the code with $\mathbf{H}_{\text{new}}$ as a generator matrix has exactly the same GHWs as those of $\mathbf{C}^\perp$. We simply write $d_j$ instead of $d_j(\mathbf{C}^\perp)$, if there is no confusion.

From the definition of the GHW, $h_j$ is lower bounded by

$$h_j \geq d_j$$

for any $j$, since the first $j$ rows of $\mathbf{H}_{\text{new}}$ form a $j$-dimensional subcode of the code with a generator matrix $\mathbf{H}_{\text{new}}$, which has support size $h_j$. When the minimum distance $d$ of $\mathbf{C}$ is known, the duality of GHWs [21] tells us that

$$d_{M-j} = M + K - j, \quad j = 0, 1, \ldots, d - 2.$$  

However, it is generally a very difficult problem to completely determine the GHWs of a code, although the GHWs of several special codes are completely known [21]–[23]. In the case of a $[K+M, K]$ CRC code or its dual code, it may be an interesting open problem to completely determine their GHWs for any $K \leq 2^M - 1 - M$. For this reason, the estimated GHWs of $\mathbf{C}^\perp$, denoted by $\hat{d}_j(1 \leq j \leq M)$, are numerically obtained for $K \in \{64, 128, 256\}$ and $M \in \{10, 16\}$. In this case, $d_1$ is exactly determined by an search, while we have $d = 4$ by an analysis. Hence, we

$$\hat{d}_1 = d_1,$$

$$\hat{d}_j = d_j = K + j, \quad j = M - 2, M - 1, M.$$  

For other values of $j$, we have $\hat{d}_j \geq d_j$, but the difference between them is made as small as possible by an extensive search.

In order to verify the validity of Algorithm 1, we show the differences between $E(h_j)$ and $\hat{d}_j$ for various values of $L$ in Fig. 2. Here, $E(h_j)$ is the expected value of $h_j$ taken over the PCMs designed by Algorithm 1, which depends on the parameter $L$ controlling the tradeoff between complexity and goodness in the algorithm. The cumulative sum of these differences decreases as $L$ increases. When this sum gets smaller, the early-error-detection capability of the designed PCM for the distributed CRC code becomes better on the average. The parameter $L$ is properly chosen so that the sum converges sufficiently, say $L = 10^2$ for $M = 10$ and $L = 10^4$ for $M = 16$, as shown in Fig. 3.

C. EARLY-ERROR-DETECTION CAPABILITY

When a distributed CRC code is concatenated with a polar code, its PCM may be designed by Algorithm 1. Based on it, the decoder for this distributed CRC-polar code can be aware of the presence of decision errors before estimating all the unfrozen bits. It is worthy of noting that the positions of the first few distributed CRC bits may be more crucial to the effect of early error detection.

The distributed CRC codes designed by the proposed algorithm are compared with those constructed by the methods in [10] and [11], when their generator polynomials are given by $g_1(x) = x^{10} + x^9 + x^5 + x^4 + x + 1$ for CRC-10 and $g_2(x) = x^{16} + x^{12} + x^5 + 1$ for CRC-16. These comparisons are shown in Fig. 4 in terms of the early-error-detection capability. Here, the $x$-axis represents the information length of the considered CRC code, while the $y$-axis represents the ratio of the number of information bits appearing before the $j$-th CRC bit to the information length, where $j \in \{1, 2, 3\}$. Numerical results show that the CRC codes designed by Algorithm 1 have higher early-error-detection capability than those constructed by the conventional methods [10], [11] in both the CRC-10 and the CRC-16 cases. From a practical viewpoint, the early termination gain of a decoder for a CRC-polar code will be defined and extensively discussed in Section V.

IV. ERROR-AWARE SCFlip DECODER

In this section, we propose a novel SCFlip decoder, called an EA-SCFlip decoder, for distributed CRC-polar codes. Based on the distributed CRC bits, it employs early termination at each extra decoding attempt so that it can reduce the average decoding complexity and latency. It also reduces the search space of candidate bit-flips in the dynamic building of the bit-flip list by exploiting the PCR of the error-detected CRC bit.

A. UTILIZATION OF DISTRIBUTED CRC BITS

In conventional SCFlip decoding of a CRC-polar code, each extra decoding attempt is performed if the CRC test fails. As the CRC bits are appended at the end of the information block, all the decision LLRs and the corresponding estimates...
need to be computed before the CRC test. This causes a decoding complexity and latency problem in a low-to-middle SNR region, together with the sequential nature of SCFlip decoding.

To mitigate this problem, the proposed EA-SCFlip decoder for distributed CRC-polar codes reduces the search space of candidate bit-flips in the dynamic building of the bit-flip list, in addition to early termination. This is done by exploiting the PCR of the first error-detected CRC bit at each extra decoding attempt. For a given information set $\mathcal{I} \triangleq \{i_1, \ldots, i_{K+M}\} \subset \{1, 2, \ldots, N\}$ with $i_1 < \ldots < i_{K+M}$, a polar codeword is obtained by the encoding process in (1), where the CRC bits are assigned to the components of $\mathbf{u}$, whose indices are in $\mathcal{I}$. For $1 \leq j \leq M$, we denote by $\mathcal{P}_j$ the $j$-th PCR set corresponding to the $j$-th CRC bit, defined by

$$\mathcal{P}_j \triangleq \{i_k | h_{j,k} = 1, 1 \leq k \leq K + M \},$$

where $i_k$ is the $k$-th information bit index in $\mathcal{I}$ and $h_{j,k}$ is the $(j, k)$-entry of the PCM for the employed CRC code.

![FIGURE 4. Early-error-detection capability of a \([K + M, K]\) distributed CRC code.](image)

In addition, we let $r_j$ be the $j$-th CRC bit index in $\mathcal{I}$, given by

$$r_j \triangleq \max \{i | i \in \mathcal{P}_j \}$$

for $1 \leq j \leq M$.

The structure of a distributed CRC code is shown in Fig. 5. Assume that a decision error is detected by the $j$-th CRC bit for the first time during the $t$-th extra decoding attempt with a bit-flip $\mathcal{E}_t = \{i_1, \ldots, i_{\omega_t}\}$ of order $\omega_t$. Then, we conclude that an odd number of decision errors have occurred among the unfrozen bits with the indices in $\mathcal{P}_j$, and need to flip at least one of them to make the $j$-th PCE satisfied. Therefore, it suffices to search for candidate bit-flips over the reduced set given by

$$\mathcal{E}_t \cup \{i \in \mathcal{I} | i_{\omega_t} + 1 \leq i \leq r_j\}, \quad (6)$$

instead of

$$\mathcal{E}_t \cup \{i \in \mathcal{I} | i_{\omega_t} + 1 \leq i \leq i_{K+M}\},$$

when we update the bit-flip list $\mathcal{L}_{\text{flip}}$ and the corresponding metric set $\mathcal{M}_{\text{flip}}$. This reduces the complexity of updating $\mathcal{L}_{\text{flip}}$ and $\mathcal{M}_{\text{flip}}$ in the decoding procedure.

Conventional flipping-set reduction techniques including the method in [12] are usually based on the selection of a subset of the information set for a polar code. Note that this subset is predetermined off-line and is chosen to include almost all the error-prone unfrozen bit indices so as to avoid a performance loss. On the other hand, the proposed search-space reduction in (6) is adaptively applied on the basis of the distributed CRC bits so that the set of candidate bit-flips is dependent on the received signal vector. Since it excludes only the candidate bit-flips that would definitely induce a decoding failure, it has no performance loss in the process of updating the bit-flip list.

### B. PROPOSED DECODING ALGORITHM

A high-level description of the proposed EA-SCFlip decoder is shown in Algorithm 2. Based on the structure of a distributed CRC code, the main operations of the EA-SCFlip decoder are described as follows:

- Standard SC decoding
- SC flip decoding with early termination (Algorithm 3)
- Update of the bit-flip list over a reduced search space (Algorithm 4).
Algorithm 2 A High-Level Description of the EA-SCFlip Decoder

| Input   | received signals $y^N_1$, information set $\mathcal{I}$, PCR sets $\{P\}_{j=1}^M$, maximum bit-flip order $\omega$, maximum number of extra decoding attempts $T$ |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Output  | estimated unfrozen bits $\{\hat{u}_i\}_{i\in \mathcal{I}}$; $\{L_i\}_{i \in \mathcal{I}}$, PCR sets $\{\mathcal{P}\}_{j=1}^M$, bit-flip set $\mathcal{E}$ |

1: $\{(\hat{u}_i)_{i\in \mathcal{I}}, \{L_i\}_{i \in \mathcal{I}}\} \leftarrow \text{SC}(y^N_1, \mathcal{I})$;
2: for $j = 1, \ldots, M$ do
3: $s_j \leftarrow \oplus \hat{u}_k$;
4: end for
5: if $s^M_1 = 0_M$ then
6: return $\{\hat{u}_i\}_{i \in \mathcal{I}}$;
7: else
8: Initial($\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{L_i\}_{i \in \mathcal{I}}$);
9: for $t = 1, \ldots, T$ do
10: $(\hat{u}_i)_{i\in \mathcal{I}}, \{L_i\}_{i \in \mathcal{I}}, s^M_1) \leftarrow \text{ET-SCFlip}(y^N_1, \mathcal{I}, \{P\}_{j=1}^M, \mathcal{E}_t)$;
11: if $s^M_1 = 0_M$ then
12: return $\{\hat{u}_i\}_{i\in \mathcal{I}}$;
13: else if $|\mathcal{E}_t| < \omega$ then
14: $l \leftarrow \text{MinIndex}(s^M_1)$;
15: Update($\mathcal{I}, \mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{L_i\}_{i \in \mathcal{I}}, \mathcal{E}_t, l$);
16: end if
17: end for
18: end if

The EA-SCFlip decoder first do standard SC decoding. When standard SC decoding fails, it builds a list of $T$ candidate bit-flips, and then performs extra SC decoding with early termination at most $T$ times until all the PCEs are satisfied. Like the D-SCFlip decoder in [17], it updates the bit-flip list by evaluating candidate bit-flips, whenever it fails.

The proposed decoder is similar to the D-SCFlip decoder, except for the CRC test and the utilization of the information on the PCR sets of the error-detected CRC bits. In Algorithm 2, we denote by $\oplus$ the modulo-2 addition operation. The function Initial() in Line 8 computes the initial bit-flip list $\mathcal{L}_{\text{flip}}$ and the corresponding metric set $\mathcal{M}_{\text{flip}}$ for the following decoding attempts. The function MinIndex() in Line 14 outputs the minimum index among the indices of the CRC bits not satisfying their corresponding PCEs.

The function ET_SCFlip() as a subroutine of Algorithm 2 is described in Algorithm 3, which runs a SCFlip decoding attempt with early termination. Here, the function $h_{\mathcal{E}}(i, L)$ is given by

$$h_{\mathcal{E}}(i, L) = \begin{cases} 0, & \text{if } i \in \mathcal{I}^c \\ 1 - \frac{1}{2}(1 - \text{sign}(L)), & \text{if } i \in \mathcal{E} \\ \frac{1}{2}(1 - \text{sign}(L)), & \text{if } i \in \mathcal{I} \setminus \mathcal{E}. \end{cases}$$

The current decoding attempt is terminated as soon as the decoder checks the failure of a PCE. Finally, Algorithm 3 returns estimated unfrozen bits $\{\hat{u}_i\}_{i \in \mathcal{I}}$, decision LLRs $\{L_i\}_{i \in \mathcal{I}}$ and syndromes $s^M_1$ as its output.

Algorithm 3 ET_SCFlip($y^N_1$, $\mathcal{I}$, $\{P\}_{j=1}^M$, $\mathcal{E}$)

| Input   | received signals $y^N_1$, information set $\mathcal{I}$, PCR sets $\{P\}_{j=1}^M$, bit-flip set $\mathcal{E}$ |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Output  | estimated unfrozen bits $\{\hat{u}_i\}_{i \in \mathcal{I}}$, decision LLRs $\{L_i\}_{i \in \mathcal{I}}$, syndromes $s^M_1$ |

1: $j \leftarrow 1$;
2: for $i = 1, \ldots, N$ do
3: For given $y^N_1$ and $\hat{u}^N_1$, calculate $L_i$ in a recursive manner.
4: $\hat{u}_i \leftarrow h_{\mathcal{E}}(i, L_i)$;
5: if $i = r_j$ then
6: $s_j \leftarrow \oplus \hat{u}_k$;
7: if $s_j = 1$ then
8: break;
9: else
10: $j \leftarrow j + 1$;
11: end if
12: end if
13: end for
14: return $\{\hat{u}_i\}_{i \in \mathcal{I}}, \{L_i\}_{i \in \mathcal{I}}, s^M_1$;

The function Update() as a subroutine of Algorithm 2 is presented in Algorithm 4, which updates the bit-flip list and the corresponding metric set by expanding the just failed bit-flip. The candidate bit-flips in (6), together with the unchecked bit-flips in the bit-flip list, are compared and only a part of them comprises the newly updated list. Since the proposed EA-SCFlip decoder has a smaller search space for candidate bit-flips than the D-SCFlip decoder, it has lower computational complexity in the update of a bit-flip list. A more detailed comparison will be made in Section V.

Algorithm 4 Update($\mathcal{I}$, $\mathcal{L}_{\text{flip}}$, $\mathcal{M}_{\text{flip}}$, $\{L_i\}_{i \in \mathcal{I}}$, $\mathcal{E}_t$, $l$)

| Input   | information set $\mathcal{I}$, bit-flip list $L_{\text{flip}}$, metric set $\mathcal{M}_{\text{flip}}$, decision LLRs $\{L_i\}_{i \in \mathcal{I}}$, bit-flip $\mathcal{E}_t$, CRC bit index $l$ |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Output  | bit-flip list $L_{\text{flip}}$, metric set $\mathcal{M}_{\text{flip}}$ |

1: for $i = \text{last}(\mathcal{E}) + 1, \ldots, l$ and $i \in \mathcal{I}$ do
2: $\mathcal{E}' \leftarrow \mathcal{E}_t \cup \{i\}$;
3: $m \leftarrow M_{\mathcal{E}}(\mathcal{E}')$;
4: if $m < \mathcal{M}_{\text{flip}}(T)$ then
5: Insert($L_{\text{flip}}$, $\mathcal{M}_{\text{flip}}$, $\mathcal{E}'$, $m$);
6: end if
7: end for
8: return ($L_{\text{flip}}$, $\mathcal{M}_{\text{flip}}$);

V. NUMERICAL RESULTS

The early termination gain and the computational complexity of the proposed coding scheme over the binary-input additive white Gaussian noise (BI-AWGN) channel are numerically discussed in this section. For this purpose, we consider polar codes concatenated with a distributed CRC code with the generator polynomial $g(x) = x^{10} + x^9 + x^5 + x^4 + x + 1$. Here, two types of polar codes are considered: one is an Arikan’s polar...
code (APC) and the other is a convolutional polar code (CPC) regraded as a natural extension of APCs [24]–[27]. The information set for the APC is designed according to the density evolution [28] at a specific SNR, while that for the CPC is chosen by using the Monte Carlo method at the same SNR. For example, the design SNR is set to 2.5 dB for the codes of rate 1/2 and length 512.

A. EVALUATION OF THE EA-SCFlip DECODER

Given a bit-flip at a decoding attempt, the proposed EA-SCFlip decoder and the D-SCFlip decoder for a distributed CRC-polar code output the same result (a success or a failure). When the output does not satisfy the CRC constraints, these two decoders update the bit-flip list by taking more likely bit-flips in the search space and inserting them into the bit-flip list. If newly inserted bit-flips for the D-SCFlip decoder are not included in the reduced search space given by (6), the corresponding decoding attempts are definitely unnecessary since they would fail eventually. Hence, when the maximum number $T$ of extra decoding attempts is limited, the EA-SCFlip decoder performs better than the D-SCFlip decoder, although the frame error rate (FER) performance gain is negligible.

In Fig. 6, the FER performance of the proposed EA-SCFlip decoder for both the APC and the CPC of rate 1/2 and length 512 is shown with respect to $T$. Here, the bit-flip order $\omega$ is chosen as 1, 2 or 3, and the perturbation parameter $\alpha$ is set to 0.4 for the APC and 0.3 for the CPC. The CPC has better performance than the APC, when we fix $\omega$ and $T$. Furthermore, for each $\omega$, their FER performances are saturated as $T$ increases. Based on this observation, we choose $T = 8$ for $\omega = 1$, $T = 64$ for $\omega = 2$, and $T = 256$ for $\omega = 3$ for these codes.

The average number of decoding attempts for the EA-SCFlip decoder, denoted by $T_{\text{avg}}$, is shown in Fig. 7. Note that $T_{\text{avg}} \geq 1$, since the initial decoding attempt is counted. The CPC has smaller $T_{\text{avg}}$ than the APC in a low SNR region, say, for SNR $\leq$ 2.0 dB. As the SNR increases, $T_{\text{avg}}$ for both codes converges to one, which corresponds to the initial decoding attempt by the standard SC decoder. In particular, $T_{\text{avg}}$ increases when the SNR decreases or $T$ gets large. In this case, we are faced with the computational complexity and latency problem.

The relative frequency of the first error detection at a CRC bit is given in Fig. 8, when the SNR is set to 1.5 dB. Here, $10^7$ frame errors are detected in total by the Monte Carlo simulation. We observe that most of them can be detected by the first few CRC bits, regardless of the choice of $\omega$ and $T$. This suggests to employ early error detection whose effect will be in the following subsection.

B. EARLY TERMINATION GAIN

The proposed EA-SCFlip decoder terminates each extra decoding attempt whenever a decision error is detected by any
FIGURE 9. Early termination gain of the EA-SCFlip decoder for \([512, 256 + 10]\) distributed CRC-polar codes. (a) APC and (b) CPC.

FIGURE 10. FER performance of the EA-SCFlip decoder for rate-1/3 distributed CRC-polar codes.

FIGURE 11. Early termination gain of the EA-SCFlip decoder for rate-1/3 distributed CRC-polar codes.

one distributed CRC bit, while conventional SCFlip decoders complete every decoding attempt until all the unfrozen bits are estimated. To evaluate the effect of early termination, we define the early termination gain \(\eta \triangleq N_u/N_t\) as a complexity measure. Here, \(N_t\) is the number of estimated unfrozen bits at one extra decoding attempt of a conventional SCFlip decoder and \(N_u\) is the average number of unfrozen bits which are not estimated due to early termination. Note that \(N_t = K + M\) and the average is taken over all extra decoding attempts.

The early termination gain of the proposed EA-SCFlip decoder for the APC is shown in Fig. 9 (a), where distributed CRC codes are constructed by Algorithm 1 and the methods in [10], [11], respectively. Numerical results demonstrate that the proposed EA-SCFlip decoder terminates unnecessary decoding attempts very early, especially at a low SNR region. As expected in Subsection III.C, it achieves a larger gain when combined with the distributed CRC code designed by Algorithm 1. The early termination gain for the CPC has a similar tendency, as shown in Fig. 9 (b). The gain is almost 0.4 at a low SNR region, when \(\omega = 3\) and the distributed CRC code designed by Algorithm 1 is employed.

In order to further evaluate the early termination gain of the proposed EA-SCFlip decoder for other APCs and CPCs, we consider rate-1/3 APCs and CPCs of length 256, 512, and 1024. They are concatenated with the distributed CRC codes with 10 parity bits, constructed by Algorithm 1. The EA-SCFlip decoder has \(\omega = 3\) as the bit-flip order, while it has \(\alpha = 0.4\) for the APCs and \(\alpha = 0.3\) for the CPCs as the perturbation parameter. The maximum number of extra decoding attempts is set to 64, 128, and 256 for the code lengths 256, 512, and 1024, respectively. The FER performance and the early termination gain of the EA-SCFlip decoder for these codes are shown in Fig. 10 and Fig. 11, respectively. The EA-SCFlip decoder achieves a large early termination gain at a low-to-middle SNR region, similarly to that for half-rate codes. It is also noteworthy that a shorter...
distributed CRC-polar code has a larger gain. This is because a distributed CRC code with a shorter information length has a higher early-error-detection capability, as suggested in Fig. 4.

C. COMPUTATIONAL COMPLEXITY

Like the D-SCFlip decoder, the proposed EA-SCFlip decoder requires a number of SCFlip decoding attempts and dynamically builds a bit-flip list. The average number of decoding attempts for the latter is almost the same as that for the former. For each extra decoding attempt, the latter reduces approximately $\eta$ of the amount of required computations for the former by employing early termination. This is because the former estimates all the unfrozen bits at every decoding attempt.

Whenever the current decoding attempt fails and the corresponding bit-flip order is less than the maximum bit-flip order $\omega$, the D-SCFlip decoder updates the bit-flip list by searching for all the unfrozen bits with indices higher than the maximum index belonging to the just failed bit-flip. On the other hand, the proposed EA-SCFlip decoder investigates all the unfrozen bits whose indices are higher than the maximum index in the currently failed bit-flip and are less than or equal to the index of the just error-detected CRC bit.

The average number of unfrozen bits to be considered in the update of the bit-flip list for these two decoders is shown in Fig. 12. Numerical results demonstrate that the proposed EA-SCFlip decoder has about 40% less unfrozen bits than the D-SCFlip decoder. This leads to approximately 40% reduction in the complexity of updating the bit-flip list over a wide range of SNRs, compared with the D-SCFlip decoder, as shown in Fig. 13.

A similar phenomenon is observed for other APCs and CPCs. As an example, Fig. 14 shows the average number of unfrozen bits to be considered in the update of the bit-flip list for rate-1/3 distributed CRC-polar codes, compared with the D-SCFlip decoder.
1024, considered in Subsection V.B. It is shown in Fig. 15 that the EA-SCFlip decoder has about 40% reduction in the complexity of updating the bit-flip list for the codes of length 1024 over a wide range of SNRs, compared with the D-SCFlip decoder, while it has about 50% reduction for the codes of length 256. In summary, the EA-SCFlip decoder is very effective in terms of complexity as well as latency, especially for distributed CRC-polar codes of short length.

VI. CONCLUSION

The EA-SCFlip decoder for a distributed CRC-polar code was proposed to mitigate the decoding complexity and latency problem in conventional SCFlip decoders. A greedy algorithm to design a distributed CRC code was also proposed to further maximize the effect of early error detection in the EA-SCFlip decoder. Numerical results show that the early termination gain of the EA-SCFlip decoder over the D-SCFlip decoder is significant when an APC or a CPC is concatenated with the distributed CRC code designed by the proposed algorithm. As a further research, it is an interesting subject to develop a design method to systematically construct a single distributed CRC code supporting various parameters.

REFERENCES

[1] E. Arikian, “Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels,” IEEE Trans. Inf. Theory, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.
[2] I. Tal and A. Vardy, “List decoding of polar codes,” in Proc. IEEE Int. Symp. Inf. Theory Proc., St. Petersburg, Russia, Jul. 2011, pp. 1–5.
[3] I. Tal and A. Vardy, “List decoding of polar codes,” IEEE Trans. Inf. Theory, vol. 61, no. 5, pp. 2213–2226, May 2015.
[4] K. Niu and K. Chen, “Stack decoding of polar codes,” Electron. Lett., vol. 48, no. 12, pp. 695–697, Jun. 2012.
[5] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, “A low-complexity improved successive cancellation decoder for polar codes,” in Proc. 48th Asilomar Conf. Signals, Syst. Comput., Pacific Grove, CA, USA, Nov. 2014, pp. 2116–2120.
[6] P. Ercan, A. Bioglio, and C. Condo, “Polar subcodes,” IEEE J. Sel. Areas Commun., vol. 34, no. 2, pp. 254–266, Feb. 2016.
[7] T. Wang, D. Qu, and T. Jiang, “Parity-Check-Concatenated polar codes,” IEEE Commun. Lett., vol. 20, no. 12, pp. 2342–2345, Dec. 2016.
[8] G. Trifonov, “Efficient SC decoding of convolutional polar codes,” in Proc. IEEE Int. Symp. Inf. Theory Proc. (ISIT), Singapore, Jun. 2018, pp. 1480–1484.
[9] R. Morozov and P. Trifonov, “Efficient SC decoding of convolutional polar codes,” in Proc. IEEE Int. Symp. Inf. Theory Proc. (ISIT), Singapore, Jun. 2018, pp. 442–446.
[10] H. Saber, Y. Ge, R. Zhang, W. Shi, and W. Tong, “Convolutional polar codes: LLR-based successive cancellation decoder and list decoding performance,” in Proc. IEEE Int. Symp. Inf. Theory Proc. (ISIT), Vail, CO, USA, Jun. 2018, pp. 1480–1484.

DAEYEOL YANG

received the B.S. degree in electronics and electrical engineering from Pusan National University, Busan, South Korea, in 2014, and the M.S. degree in electrical engineering from the Pohang University of Science and Technology (POSTECH), Pohang, South Korea, in 2016, where he is currently pursuing the Ph.D. degree in electrical engineering. His current research interests include coding theory and communication systems.

KYEONGCHEOL YANG

received the B.S. and M.S. degrees in electronics engineering from Seoul National University, Seoul, South Korea, in 1986 and 1988, respectively, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, CA, in 1992.

In summer 1990, he was an Intern with Bellcore, Morristown, NJ. From March 1993 to January 1999, he was an Assistant Professor with the Department of Electronic Communication Engineering, Hanyang University, Seoul. He joined the faculty of the Pohang University of Science and Technology (POSTECH), in February 2019, where he is currently a Professor with the Department of Electrical Engineering. During his sabbatical years in 2006 and 2013, he was a Technical Consultant with Samsung Electronics Company, Suwon, South Korea. His research interests include coding theory, signal design, iterative information processing, and communication systems.

Dr. Yang was the Program Co-Chair at the International Conference on Sequences and Their applications (SETA 2001) and the Fifth International Workshop on Signal Design and Its Applications in Communications (IWSDA 2011). He was an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION THEORY, from 2012 to 2015. He has served as an Associate Editor for Cryptography and Communications, from 2008 to 2018.