A Balanced Tree Approach to Construction of Length-Flexible Polar Codes

Xinyuanmeng Yao and Xiao Ma, Member, IEEE

Abstract—We propose a length-flexible coding scheme by defining a balanced tree. For an arbitrary code length, we first construct a balanced binary tree (BBT) where the root node represents a transmitted codeword, the leaf nodes represent either active bits or frozen bits, and a parent node is related to its child nodes by a length-adaptive \((U + V | V)\) operation. Both the encoding and the successive cancellation (SC)-based decoding can be implemented over the constructed coding tree. For code construction, we propose a signal-to-noise ratio (SNR)-dependent method and two SNR-independent methods, all of which evaluate the reliabilities of leaf nodes and then select the most reliable leaf nodes as the active nodes. Numerical results demonstrate that our proposed codes can have comparable performance to the 5G polar codes. To reduce the decoding latency, we propose a partitioned successive cancellation (PSC)-based decoding algorithm, which can be implemented over a sub-tree obtained by pruning the coding tree. Numerical results show that the PSC-based decoding can achieve similar performance to the conventional SC-based decoding.

Index Terms—Code construction, coding tree, length-flexible coding, partitioned successive cancellation-based decoding, polar coding.

I. INTRODUCTION

As a class of error-correcting codes which can provably achieve capacity for binary-input discrete memoryless channels (B-DMCs) with low encoding and decoding complexity [1], polar codes have attracted a lot of attention in the past decade. Recently, they have been adopted as the channel coding scheme for the control channel in the Enhanced Mobile Broadband scenario in the fifth generation (5G) wireless communications standard [2]. For polar coding, one of the most important techniques is channel polarization. Based on a mapping from two independent and identical B-DMCs to a “worse” channel and a “better” channel, \(2^n\) copies of an original B-DMC are transformed into \(2^n\) bit-channels with different reliabilities. Then, the most reliable polarized channels are chosen to transmit data bits. In the finite-length regime, polar codes with the successive cancellation list (SCL) decoder or the cyclic redundancy check (CRC)-aided SCL (CA-SCL) decoder can have satisfactory performance.

From the definition of polar codes, we see that the code rates of polar codes are flexible. However, the lengths of polar codes are limited to powers of two, which are not convenient for practical applications. To construct length-flexible polar codes, one can use puncturing, shortening or extending. The punctured and shortened (P/S) polar codes [3], [4], [5], [6], [7], [8], [9], [10] are obtained by deleting some coded bits of their mother codes, while the extended polar codes [11], [12], [13], [14], [15] are obtained by padding some bits to slightly shorter polar codes, which are usually designed for hybrid automatic repeat-and-request. It has been shown in the existing literature that the P/S polar codes can have good performance. However, the decoding of the P/S polar codes is based on their mother codes, which results in an increased decoding latency. For some parameter configurations (for example, the cases when the code length is slightly larger than a power of two and the code rate is low), the extended polar codes might be a better choice in terms of performance and complexity. Recently, for 5G applications, a well designed low-complexity rate matching scheme has been proposed in [2], which incorporates puncturing, shortening and repetition (the simplest way of extending) techniques. In fact, one can also construct length-flexible polar codes with decoding latency depending only on the real code lengths (in contrast to the P/S polar codes depending on their mother codes’ lengths). Multi-kernel polar coding [16] is one of such techniques, which mixes kernels of different sizes. Notice that any positive integer can be decomposed into a sum of powers of two, chained polar subcodes have been proposed in [17]. Another technique is asymmetric polar coding [18], which links polar codes of unequal lengths.

Notice that the conventional polar coding [1] can be implemented over a full binary tree where the root node represents a transmitted codeword, each of the leaf nodes represents either an active bit or a frozen bit and the relationship between a parent node and its child nodes is specified by the \((U + V | V)\) operation. Motivated by this idea, we propose to construct a balanced binary tree (BBT) for any code length, referred to as a coding tree, and design a length-flexible coding scheme by implementing a length-adaptive \((U + V | V)\) operation over the coding tree. For BBT polar code construction, we turn to density evolution with Gaussian approximation (GA), termed as GA construction. We also present two signal-to-noise ratio (SNR)-independent methods for practical use. Similar to [19], [20], and [21],
we assign to each node a polar subcode and present a recursive algorithm to approximately calculate the minimum Hamming weight (MHW) and the multiplicity of the MHW codewords for a polar subcode. Then, a theoretically explainable MHW construction is proposed. In addition, based on [22], we define a polarization weight (PW) for the leaf nodes in the coding tree and propose a heuristic PW construction.

From the viewpoint of coding tree, the SC decoder for (BBT) polar codes makes hard decisions at leaf nodes in a serial manner, which results in high decoding latency when the code length is long. For the conventional polar codes, the existing works [23], [24], [25], [26], [27], [28], [29], [30] identified different types of inner nodes according to the pattern of active and frozen bits and then designed the corresponding multi-bits parallel decoders. For example, Alamdar-Yazdi et al. proposed to make hard decisions at rate-0 and rate-1 nodes [23], Sarkis et al. further proposed to make hard decisions at single-parity-check and repetition nodes [24], and Hanif et al. proposed to make hard decisions at five types of nodes (Type-I, Type-II, Type-III, Type-IV and Type-V nodes) [25]. Recently, for the 5G polar codes involving parity-check (PC) bits, Zhou et al. identified various types of special nodes which contain PC bits and proposed the corresponding fast decoders [31]. Actually, any inner node satisfying computational resource constraints can be decoded by maximum likelihood (ML) decoding to reduce latency [32], [33]. In this work, we present a partitioned successive cancellation (PSC) decoder. First, a decoding sub-tree is extracted from the coding tree according to a preset threshold on the dimensions of the nodes and then the PSC decoder is implemented over the decoding sub-tree. To analyze the error performance of the BBT polar codes with the PSC decoding, we derive three bounds on the frame error rate (FER). Furthermore, the PSC decoder can be adapted to the partitioned successive cancellation list (PSCL) decoder or CA-PSCL decoder for the BBT polar codes.

The rest of this paper is organized as follows. We introduce the coding tree and present our BBT polar coding scheme in Section II. Three BBT polar code constructions and their performance are presented in Section III. In Section IV, the proposed PSC-based decoding is described and evaluated in terms of error-correction performance and decoding latency. We conclude this paper in Section V.

II. THE LENGTH-FLEXIBLE CODING SCHEME

In this section, we first introduce a coding tree and then present a coding scheme which is applicable to arbitrary code length.

A. Coding Tree

A BBT is a binary tree structure in which the left and right subtrees of every inner node differ in height (the number of edges from the top-most node to the farthest node in a subtree) by no more than one [34]. The dynamic behavior of the encoder and the decoder in our scheme can be graphically represented by a rooted BBT, called a coding tree, as defined below. Given a code length $N$, the root node is labeled by a codeword $c \in \mathbb{F}_2^N$ (the $N$-dimensional vector space over the binary field $\mathbb{F}_2$). Each inner node labeled by a vector $v \in \mathbb{F}_2^r$, where $r \leq N$ is some positive integer (not necessarily a power of two), has a left child node labeled by $v_l \in \mathbb{F}_2^{\lceil r/2 \rceil}$ and a right child node labeled by $v_r \in \mathbb{F}_2^{\lceil r/2 \rceil}$, where $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ denote the ceiling function and the floor function, respectively. For convenience, we say that a node labeled with $v \in \mathbb{F}_2^r$ has a length $\ell$. If the lengths of two children are equal, i.e., $\lceil \ell/2 \rceil = \lfloor \ell/2 \rfloor$, define as usual

$$v_l \oplus v_r \triangleq (v_{l,0}+v_{r,0}, \ldots, v_{l,\lfloor \ell/2 \rfloor-1}+v_{r,\lfloor \ell/2 \rfloor-1}), \quad (1)$$

and if the length of the left child is one greater than that of the right child, i.e., $\lceil \ell/2 \rceil = \lfloor \ell/2 \rfloor + 1$, define

$$v_l \oplus v_r \triangleq (v_{l,0}+v_{r,0}, \ldots, v_{l,\lfloor \ell/2 \rfloor-2}+v_{r,\lfloor \ell/2 \rfloor-1}, v_{l,\lfloor \ell/2 \rfloor-1}), \quad (2)$$

where “$+$” in the right hand sides of (1) and (2) denote the addition over $\mathbb{F}_2$. Then, the relationship of the parent node and its children is given by

$$v = (v_l \oplus v_r, v_r). \quad (3)$$

In words, in the case when $\lceil \ell/2 \rceil = \lfloor \ell/2 \rfloor$, (3) is exactly the same as a polar transformation, while in the case when $\lceil \ell/2 \rceil = \lfloor \ell/2 \rfloor + 1$, (3) is equivalent to extending the right child $v_r$ by one zero for vector addition and then shortening it from $v$ and $v_r$.

With the above definition, for any code length $N$, we can first construct the corresponding coding tree from the root by splitting each node with length $\ell \geq 2$ into two children. The code tree has $\lceil \log_2 N \rceil$ levels and $N$ leaves. Then, given $N$ labels of the leaf nodes, we can calculate the labels of all inner nodes recursively from the leaves to the root. An example of a coding tree for code length $N = 6$ is shown in Fig. 1. The tree has three levels in total. Two leaf nodes are at the second-to-last level and four leaf nodes are at the last level. The labels of the leaf nodes from left to right are 0, 1, 0, 0, 1, 1 and the codeword of the root node is 101011.

B. Encoding and Decoding

Denote by $K$ and $N$ the code dimension and the code length, respectively. The code rate is $R \triangleq K/N$. To encode a data sequence $u \in \mathbb{F}_2^K$, we first construct the unlabeled coding tree for $N$ according to Section II-A and distinguish the $N$ leaf nodes by $K$ active nodes and $N - K$ frozen nodes. Then the $K$ active nodes are labeled by the bits of $u$ and the $N - K$ frozen nodes are labeled by zero. The choice of the
active nodes will be discussed later. The encoder computes the labels of the non-leaf nodes in a reverse level-order traversal of the tree, resulting in a transmitted codeword \( c \in \mathbb{F}_2^N \) at the root node.

The decoding can also be implemented over the coding tree. In this situation, each tree node is associated with a log-likelihood ratio (LLR) vector and a hard bit estimate (HBE) vector. Then, following three basic rules, the decoder computes the LLRs of the nodes in a post-order traversal and computes the HBE vectors of the nodes in a pre-order traversal, as detailed below. For an inner node of length \( \ell \) with LLRs \( \alpha = (\alpha_0, \alpha_1, \ldots, \alpha_{\ell-1}) \) and HBEs \( \beta = (\beta_0, \beta_1, \ldots, \beta_{\ell-1}) \) in \( \mathbb{F}_2^\ell \), denote by \( \alpha_l \triangleq (\alpha_{l0}, \alpha_{l1}, \ldots, \alpha_{l,\ell/2-1}) \in \mathbb{F}_2^{\ell/2} \) and \( \beta_l \triangleq (\beta_{l0}, \beta_{l1}, \ldots, \beta_{l,\ell/2-1}) \in \mathbb{F}_2^{\ell/2} \) the LLRs and HBEs of its left child, respectively, and similarly by \( \alpha_r \triangleq (\alpha_{r0}, \alpha_{r1}, \ldots, \alpha_{r,\ell/2-1}) \in \mathbb{F}_2^{\ell/2} \) and \( \beta_r \triangleq (\beta_{r0}, \beta_{r1}, \ldots, \beta_{r,\ell/2-1}) \in \mathbb{F}_2^{\ell/2} \) the LLRs and HBEs of its right child, respectively. For convenience, define

\[
\begin{align*}
    f(a, b) &\triangleq \ln \frac{1 + e^{a+b}}{e^a + e^b}, \quad \text{for } a, b \in \mathbb{R}, \\
    g(a, b, c) &\triangleq b + (-1)^e \cdot a, \quad \text{for } a, b \in \mathbb{R} \text{ and } c \in \mathbb{F}_2.
\end{align*}
\]

The decoder starts by initializing the LLRs of the root node with a received vector \( y \in \mathbb{R}^N \) and then proceeds according to the following rules.

1) When the LLRs of a parent node are available, the LLRs of its left child are computed. If \( \ell \) is even,

\[
\alpha_{l,i} = f(\alpha_{i}, \alpha_{\ell/2+i}), \quad \text{for } 0 \leq i \leq \lfloor \ell/2 \rfloor - 1,
\]

which is the same as the case in the SC decoding of polar codes. If \( \ell \) is odd,

\[
\alpha_{l,i} = \begin{cases} 
    f(\alpha_{i}, \alpha_{\ell/2+i}), & \text{for } 0 \leq i \leq \lfloor \ell/2 \rfloor - 2, \\
    \alpha_{i}, & \text{for } i = \lfloor \ell/2 \rfloor - 1.
\end{cases}
\]

2) When the LLRs of a parent node and the HBEs of its left child are available, the LLRs of its right child are computed as

\[
\alpha_{r,i} = g(\alpha_{i}, \alpha_{\ell/2+i}, \beta_{l,i}), \quad \text{for } i = 0, 1, \ldots, \lfloor \ell/2 \rfloor - 1.
\]

3) When the HBEs of a pair of sibling nodes are available, the HBEs of their parent are computed as

\[
\beta = (\beta_l \oplus \beta_r, \beta_r),
\]

where the operation \( \oplus \) is defined in (1) and (2).

C. Algebraic Description

Given a coding tree and a choice of the active leaf nodes which is termed as rate-profiling, all possible labels of the root node form a binary linear code. We refer to such a code as a BBT polar code. The encoding process of a BBT polar code with length \( N \) and dimension \( K \) can be described as follows. We index the leaf nodes from left to right by \( 0, 1, \ldots, N-1 \). Then, denote by \( \mathcal{A} \) and \( \mathcal{F} \triangleq \mathcal{A}^C \) the index set of active leaf nodes and that of frozen leaf nodes, respectively, where \( \mathcal{A} \subset \{0, 1, \ldots, N-1\} \) and \( |\mathcal{A}| = K \). Let \( w \in \mathbb{F}_2^N \) be a leaf node sequence such that \( w_\mathcal{A} \) (the components of \( w \) indexed by \( \mathcal{A} \)) is a data sequence and \( w_\mathcal{F} \) is the all-zero sequence. Then, we can obtain a codeword \( \mathbf{c} = wG \), where \( G \in \mathbb{F}_2^{N \times N} \) is the invertible transformation matrix corresponding to the coding tree for length \( N \). The pseudo code for computing \( G \) is provided in Algorithm 1. As an example, the invertible transformation matrix for BBT codes of length \( N = 6 \) can be obtained as

\[
G_6 = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}.
\]

III. CODE CONSTRUCTION

Like the conventional polar codes, the performance of the proposed scheme is determined by the selection of the active leaf nodes. To achieve a good performance, it is critical to rank the reliabilities of the leaf nodes so that we can select the \( K \) most reliable leaf nodes out of the \( N \) leaf nodes.

In this paper, we assume that a codeword is modulated using binary phase-shift keying (BPSK) which maps \( 0 \to +1 \) and \( 1 \to -1 \), and then transmitted through an additive white Gaussian noise (AWGN) with zero mean and variance \( \sigma^2 \). The SNR is defined as \( E_b/N_0 \), where \( E_b \) is the energy per data bit and \( N_0 = 2\sigma^2 \). In this scenario, the GA method can be employed to evaluate the error performance of the leaf nodes at a given SNR. However, SNR-independent ranking is
Algorithm 2 The Gaussian Approximation Construction

Input: $N$: a code length, $K$: a code dimension, $E_b/N_0$: a design SNR
Output: $\mathcal{A}$: an index set of active leaf nodes

1. $\sigma^2 \leftarrow \frac{1}{K/N_0 E_b/N_0}$; $\mu \leftarrow \frac{1}{\sigma^2} \times 1_{1 \times N}$; $n = (N)$;
2. while $\text{length}(n) < N$
   // the value length(n) is the length of the vector $n$
   3. do
   4. $s \leftarrow 0$; $\tilde{n} \leftarrow \emptyset$; $\tilde{\mu} \leftarrow 0_{1 \times N}$;
   5. for $i = 0, 1, \ldots, \text{length}(n) - 1$
      6. if $n(i) = 1$ then
         7. $\tilde{n} \leftarrow (\tilde{n}, 1)$; $\tilde{\mu}(s) = \mu(s)$; $s \leftarrow s + 1$
      8. else
         9. $L_p \leftarrow n(i)$; $L_t \leftarrow \lfloor L_p/2 \rfloor$; $L_r \leftarrow \lceil L_p/2 \rceil$;
         10. $\tilde{n} \leftarrow (\tilde{n}, L_t, L_r)$;
         11. for $t = 0, 1, \ldots, L_r - 1$
            12. $\tilde{\mu}(s(t + t)) = \phi^{-1}(1 - (1 - \phi(\mu(s + t)))(1 - \phi(\mu(s + L_t + t)))$;
            13. $\tilde{\mu}(s + L_t + t) = \mu(s + L_t + t)$
            14. $\tilde{\mu}(s + t) + \mu(s + L_t + t)$
         15. if $L_t = L_r + 1$ then
            16. $s \leftarrow s + L_p$
            17. $\tilde{n} \leftarrow (\tilde{n}, L_t)$
            18. $\tilde{\mu}(s + L_t - 1) = \mu(s)$
         19. end
      20. end
   21. $n \leftarrow \tilde{n}$; $\mu \leftarrow \tilde{\mu}$
   22. end
   23. $\mathcal{A}$ ← the indices of the greatest $K$ elements in $\mu$

preferable for practice use. Therefore, we propose two new SNR-independent methods to rank more conveniently the leaf nodes.

A. Gaussian Approximation Construction

For the BBT polar codes, the reliabilities of the leaf nodes can be evaluated by the GA method [35] in a similar way to the conventional polar codes. We treat approximately the LLRs of the tree nodes in the decoding process as Gaussian variables and then compute the expectation of the LLRs. The GA construction of the BBT polar code is to select $K$ leaf nodes with the largest LLR expectations to transmit data bits.

The pseudo code of the GA construction for the BBT polar codes is provided in Algorithm 2, where $\phi(x)$ is given in [36] as

$$
\phi(x) = \begin{cases} 
\exp(0.0564x^2 - 0.4856x), & \text{if } 0 \leq x \leq 0.867861 \\
\exp(-0.45272x^{0.86} + 0.0218), & \text{otherwise}
\end{cases}
$$

and $\phi^{-1}(x)$ is the inverse function of $\phi(x)$.

B. Minimum Hamming Weight Construction

Motivated by [19], we can evaluate the reliabilities of the leaf nodes by their error performance under genie-aided SC decoding. Given a coding tree with transformation matrix $G \in \mathbb{F}_2^{N \times N}$, for the $i$-th leaf node, define the corresponding polar subcode by

$$
\mathcal{C}(i) \triangleq \{ c = (0^{i-1}, 1, \omega_{i+1}^{N-1})G | \omega_{i+1}^{N-1} \in \mathbb{F}_2^{N-1} \}.
$$

The error rate of the $i$-th leaf node under genie-aided SC decoding, denoted by $\varepsilon(i)$, can be upper bounded by

$$
\varepsilon(i) \leq \sum_{1 \leq d \leq N} A(i)(d) \text{PEP}(N, d),
$$

where $A(i)(d)$ is the number of the codewords with Hamming weight $d$ in $\mathcal{C}(i)$ and PEP($N, d$) is the pairwise error probability between the all-zero codeword and a codeword with Hamming weight $d$. Based on (11), we can give an estimate of $\varepsilon(i)$ as

$$
\varepsilon(i) \leq A(i)(d(i)_{\text{min}}) \text{PEP}(N, d(i)_{\text{min}}),
$$

where $d(i)_{\text{min}}$ and $A(i)(d(i)_{\text{min}})$ denote the MHW of $\mathcal{C}(i)$ and the number of the MHW codewords in $\mathcal{C}(i)$, respectively.

Observe that $\varepsilon(i)$ is relevant to $d(i)_{\text{min}}$ and $A(i)(d(i)_{\text{min}})$. We propose to sort the leaf nodes according to $d(i)_{\text{min}}$ in ascending order and then sort those leaf nodes with the same $d(i)_{\text{min}}$ according to $A(i)(d(i)_{\text{min}})$ in descending order, resulting in an ascending reliability order of $N$ leaf nodes. The MHW construction of the BBT polar code is to select the last $K$ indices in the ordered sequence as the active set $\mathcal{A}$.

Now we show how to evaluate $d(i)_{\text{min}}$ and $A(i)(d(i)_{\text{min}})$. When $N$ is a power of two, $d(i)_{\text{min}}$ and $A(i)(d(i)_{\text{min}})$ can be calculated using the iterative enumeration algorithm presented in [19]. However, when $N$ is not a power of two, it is not easy to calculate $d(i)_{\text{min}}$ and $A(i)(d(i)_{\text{min}})$. Hence we turn to a randomly interleaved coding tree. To be precise, similar to [37], we can insert uniform interleavers between intermediate layers of a coding tree, as described below. Given a code length $N$, the root node is labeled by a codeword $c \in \mathbb{F}_2^N$. Each node labeled by a vector $v \in \mathbb{F}_2^L$, where $2 \leq L \leq N$, has a left child node labeled by $v_L \in \mathbb{F}_2^{[L/2]}$ and a right child node labeled by $v_R \in \mathbb{F}_2^{[L/2]}$. Define that

$$
\begin{align*}
   v_L &= \begin{cases} 
   v_L, & \text{if } [L/2] = [L/2] \\
   0, & \text{if } [L/2] = [L/2] + 1,
   \end{cases} \\
   v_R &= \begin{cases} 
   v_R, & \text{if } [L/2] = [L/2] + 1,
   \end{cases}
\end{align*}
$$

where $\Pi$ is a random permutation matrix, and

$$
\begin{align*}
   v_L + v_R &= \begin{cases} 
   v_L + v_R, & \text{if } [L/2] = [L/2],
   \end{cases}
\end{align*}
$$

where “+” denotes the addition over $\mathbb{F}_2$. The relationship between a parent node and its child nodes in the interleaved coding tree is given by

$$
\begin{align*}
   v &= (v_L \oplus v_R, v_R).
\end{align*}
$$

Under the assumption that all the interleavers are uniform at random, the labels of the parent nodes are random. Then,
for each node with length $\ell$, we define a weight enumerating function (WEF) as

$$\text{WEF} \triangleq \sum_{d=0}^{\ell} B(d) Y^d,$$

where $Y$ is a dummy variable and $B(d)$ is the average number of the labels with Hamming weight $d$. Let $d_{\text{min}}$ be the minimum positive integer with $B(d_{\text{min}}) > 0$. The minimum weight enumerating function (MWEF) of the node is defined as

$$\text{MWEF} \triangleq B(d_{\text{min}}) Y^{d_{\text{min}}}.$$

**Lemma 1:** Consider a pair of sibling nodes in the randomly interleaved coding tree. Let $v_l$ be a sample label of the left child node and $v_r'$ be a sample label of the right child node. The vector $v_l$ is with length $\ell_l$ and Hamming weight $d_l$ and the vector $v_r'$ is with length $\ell_r$ and Hamming weight $d_r$. The weight distribution for $v = (v_l \oplus v_r', v_r)$ is given by

$$E \Pi \left[ Y^{w_H(v)} \right] = \sum_{\Pi} \Pr(\Pi) Y^{w_H(v)} = \sum_{k=\max(0, d_l + d_r - \ell_l)}^{\min(d_l, d_r)} \left( \frac{d_l}{k} \right) \frac{\ell_l - d_l}{d_r - k} Y^{d_l + 2d_r - 2k},$$

where $w_H(v)$ denotes the Hamming weight of $v$.

**Proof:** Let $k$ be the number of positions at which the components in $v_l$ and $v_r'$ are both equal to 1. With the uniform interleaver, we see that $k$ is a random integer, ranging from $\max(0, d_l + d_r - \ell_l)$ to $\min(d_l, d_r)$ with probability

$$\left( \frac{d_l}{k} \right) \frac{\ell_l - d_l}{d_r - k} \left( \frac{\ell_l}{d_r} \right) \frac{d_r}{k}. $$

Given $k$, the Hamming weight of $v_l \oplus v_r'$ is $d_l + d_r - 2k$. Finally, the additional concatenation of $v_r$ gives the Hamming weight of $v = (v_l \oplus v_r', v_r)$ as $d_l + d_r - 2k + d_r = d_l + 2d_r - 2k$.

**Theorem 1:** Consider a parent node with two child nodes in the randomly interleaved coding tree. Let $\sum_{d_l=0}^{\ell_l} B_l(d_l) Y^{d_l}$ and $\sum_{d_r=0}^{\ell_r} B_r(d_r) Y^{d_r}$ be the WEF of the left child node and that of the right child node, respectively. The WEF of the parent node is

$$\sum_{d_l=0}^{\ell_l} B_l(d_l) B_r(d_r) \sum_{k=\max(0, d_l + d_r - \ell_l)}^{\min(d_l, d_r)} \left( \frac{d_l}{k} \right) \frac{\ell_l - d_l}{d_r - k} Y^{d_l + 2d_r - 2k}.$$

**Proof:** It can be directly derived from Lemma 1 and the proof is omitted here.

**Corollary 1:** Consider a parent node with two child nodes in the randomly interleaved coding tree. If the WEF of the left child node is $Y^{d_{\text{min,}l}}$, then the MWEF of the parent node is

$$B_r(d_r) Y^{d_{\text{min,}r}}.$$

**Proof:** Let $\sum_{d_r=0}^{\ell_r} B_r(d_r) Y^{d_r}$ be the WEF of the right child node. From Theorem 1, the WEF of the parent node is

$$\sum_{d_r=0}^{\ell_r} B_r(d_r) Y^{2d_r}. $$

Then, we have that the MWEF of the parent node is $B_r(d_{\text{min,}r}) Y^{2d_{\text{min,}r}}$.

**Corollary 2:** Consider a parent node with two child nodes in the randomly interleaved coding tree. If the MWEF of the left child node is $B_l(d_{\text{min,}l}) Y^{d_{\text{min,}l}}$ and the WEF of the right node is $(Y + 1)^{\ell_r}$, then the MWEF of the parent node is

$$B_l(d_{\text{min,}l}) \sum_{d_r=0}^{\min(d_{\text{min,}l}, \ell_r)} \left( \frac{\ell_r}{d_r} \right) \left( \frac{d_{\text{min,}l}}{d_r} \right) Y^{d_{\text{min,}l}}.$$

**Proof:** Let $\sum_{d_r=0}^{\ell_r} B_r(d_r) Y^{d_r}$ be the WEF of the left child node. From Theorem 1, the WEF of the parent node is

$$\sum_{d_r=0}^{\ell_r} B_r(d_r) \sum_{k=\max(0, d_r + d_r - \ell_r)}^{\min(d_{\text{min,}l} + 2d_r, d_{\text{min,}l} - \ell_r)} \left( \frac{\ell_r}{d_r} \right) \left( \frac{d_{\text{min,}l} + 2d_r}{d_r} \right) Y^{d_{\text{min,}l} + 2d_r}.$$

Since $d_r \geq k$, the Hamming weight of $v = (v_l \oplus v_r', v_r)$ is no less than the Hamming weight of $v_l$, i.e., $d_l + 2d_r - 2k \geq d_l$. Then, the WEF of the parent node is given by

$$B_l(d_{\text{min,}l}) \sum_{d_r=0}^{\ell_r} \left( \frac{\ell_r}{d_r} \right) \left( \frac{d_{\text{min,}l}}{d_r} \right) Y^{d_{\text{min,}l}}.$$

where $\mathbb{I}\{k = d_r\}$ is 1 if $k = d_r$, and 0 otherwise.

**C. Polarization Weight Construction**

In the following, we propose a heuristic PW construction of the BBT polar code.
In a coding tree, we label the branch from a parent node to its left child node by 0 and the branch from a parent node to its right child node by 1. Then, to each leaf node, we assign a sequence $b = (b_0, b_1, \ldots, b_{m-1})$ that records the labels of the path (of length $m$) from the root to the considered leaf node. This defines a one-to-one correspondence between a binary sequence and a leaf node. Notice that a leaf node at the second-to-last level is associated with a sequence of length $m = \lceil \log_2 N \rceil$ and a leaf node at the last level is associated with a sequence of length $m = \lfloor \log_2 N \rfloor$. In Fig. 2, we show two coding trees. One is for $N = 8$ and the other is for $N = 6$. We have labeled the branches of both the coding trees.

For a BBT polar code, we propose to define a PW of the $i$-th leaf node by

$$\text{PW}^{(i)} = \sum_{j=0}^{m-1} b_j^{(i)} \cdot \kappa^{J-j},$$

where $(b_0^{(i)}, b_1^{(i)}, \ldots, b_{m-1}^{(i)})$ is the corresponding sequence of the $i$-th leaf node, $\kappa$ is a design parameter and $J \equiv \lfloor \log_2 N \rfloor - 1$. When $N$ is a power of two, all the leaf nodes are associated with the sequences with length $m = \log_2 N$, which are exactly the binary expressions of the indices of the leaf nodes (see Fig. 2(a) for reference), indicating that the proposed PW for the BBT polar code is consistent with the PW introduced in [22] for the conventional polar code. In this paper, we set the parameter $\kappa = 2^{1/4}$ as proposed in [38] for simplicity.\(^3\) The PW construction of the BBT polar code is to select $K$ leaf nodes with the largest PWs to transmit data bits.

**D. Decoding Performance**

Fig. 3 shows the performance comparisons among the proposed three BBT polar code constructions for $N = 768$ and $K = 384$. The design SNR for the GA construction is 3 dB. The SC decoding, the SCL decoding with list size 8 (denoted by SCL(8)) and the CA-SCL decoding with list size 8 and 11-bit CRC. Here, $N = 768$ and $K = 100$.

\(^3\)Similar to the case for the conventional polar codes [39], it is possible to optimize $\kappa$ for the BBT codes.
is implemented in the simulations. Here, and the 5G polar codes. The CA-SCL decoding with list size 8 and 11-bit CRC is implemented in the simulations. The tested code length is \( N = 768 \) and \( R \in \{1/4, 1/2, 3/4\} \).

code parameters, and a pre-freezing step is performed. Then, an output of the CRC encoder is encoded by a polar encoder, resulting in a polar codeword. Eventually, the codeword is sent to a sub-block interleaver and some bits of the interleaved codeword are chosen by a circular buffer to transmit.

Fig. 5 shows the performance comparisons between the constructed BBT polar codes and the 5G polar codes. The CA-SCL decoding with list size 8 and 11-bit CRC is implemented in the simulations. The tested code length is \( N = 768 \) and code rates are \( R \in \{1/4, 1/2, 3/4\} \). We observe that for the low and moderate rates, the PW-BBT polar codes can achieve similar performance to the 5G polar codes.

IV. PARTITIONED SC-BASED DECODING

A. Decoding Sub-Tree

Recall that for the BBT polar codes, both the encoding and the SC-based decoding can be implemented over the coding tree defined in Section II-A. However, the SC-based decoder makes hard decisions at leaf nodes in a serial manner, which results in high decoding latency when the code length is long.

In the following, we define a decoding sub-tree and then present a low-latency SC-based decoding algorithm.

Now assume that a rate-profiling is given for a coding tree of length \( N \). Each node in the coding tree is associated with a binary linear code, consisting of all the possible labels of the node. Specifically, for a tree node of length \( n \) whose descendants involve \( k \) active leaf nodes and \( n - k \) frozen leaf nodes, the associated code has length \( n \) and dimension \( k \).

For convenience, we say that such a node has a dimension \( k \). When \( k \) is small, it is feasible to compute all the codewords in the code. Let \( \tau \) (a positive integer) be a dimension threshold. We first identify all the nodes which has dimension at most \( \tau \) and whose parent node has dimension greater than \( \tau \) in the coding tree and then extract a decoding sub-tree from the coding tree by removing all the descendants of these identified nodes. We refer to the leaves in the decoding sub-tree as the decoding leaves. An example of a decoding sub-tree for \( N = 6 \) and \( \tau = 1 \) is shown in Fig. 6. We see that, the decoding sub-tree has two levels (less than the original coding tree) and three decoding leaves.

Suppose that there are \( q \) decoding leaves in the decoding sub-tree. Denote by \( N_{\text{dec}}^{(t)} \) the \( t \)-th leaf from left to right in the decoding sub-tree and by \( \mathcal{S}^{(t)} \) the code associated with \( N_{\text{dec}}^{(t)} \), where \( t = 0, 1, \ldots, q-1 \). Then, a juxtaposition of the labels of the \( q \) leaves, written as \( \{v^{(t)}\}_{q=1}^{q-1} \equiv \{v^{(0)}, v^{(1)}, \ldots, v^{(q-1)}\} \), can be transformed recursively into a BBT polar codeword at the root, where \( v^{(t)} \in \mathcal{S}^{(t)} \) has length \( \ell^{(t)} \). Note that the lengths \( \ell^{(t)} (t = 0, 1, \ldots, q-1) \) may be different.

When the LLRs of a decoding leaf \( N_{\text{dec}}^{(t)} \), denoted by \( \alpha^{(t)} \equiv (\alpha_{0}^{(t)}, \alpha_{1}^{(t)}, \ldots, \alpha_{\ell^{(t)}-1}^{(t)}) \), are available, the partitioned SC (PSC) decoder computes the likelihoods of all the codewords in \( \mathcal{S}^{(t)} \) and then directly sets the HBEs of \( N_{\text{dec}}^{(t)} \), denoted by \( \beta^{(t)} \equiv (\beta_{0}^{(t)}, \beta_{1}^{(t)}, \ldots, \beta_{\ell^{(t)}-1}^{(t)}) \), to the most likely codeword. The partitioned SCL (PSCL) decoder performs in a similar way. Define a decoding path at decoding step \( t \) as a juxtaposition of the HBEs of the previous \( t \) decoding leaves, i.e., \( \{\beta^{(t)}(i)\}_{i=0}^{t-1} \equiv \{\beta_{0}^{(t)}, \beta_{1}^{(t)}, \ldots, \beta_{\ell^{(t)}-1}^{(t)}\} \). In the PSC decoding, at most \( L \) decoding paths are retained. At the \((t+1)\)-th decoding step, the LRRs of the leaf \( N_{\text{dec}}^{(t)} \) are available. The PSC decoder splits each path \( \{\beta^{(t)}(i)\}_{i=0}^{t-1} \) into at most \( 2^{\ell^{(t)}} \) paths by setting \( \beta^{(t)}(i) \) to all the codewords in \( \mathcal{S}^{(t)} \), and for each extended path \( \{\beta^{(t)}(i)\}_{i=0}^{t} \), computes its path metric as

\[
PM_t = \sum_{i=0}^{t} \sum_{j=0}^{\ell^{(t)}-1} \ln(1 + \exp(-(1 - 2\beta_{j}^{(t)}(i))\alpha_{j}^{(t)})) \tag{24}
\]

Then, the \( L \) paths with the minimum path metrics are retained, while the other paths are discarded. Eventually, the path with the minimum path metric is selected as the decoder output.

B. Performance Analysis

Without loss of generality, we assume that the all-zero codeword is transmitted over a BPSK-AWGN channel, resulting in a received vector. In this case, all the labels of the decoding leaves are zero vectors.

Similar to [40, Lemma 1], it can be easily verified that the performance of the PSC decoding is identical to that of the genie-aided PSC decoding where the decoder knows the correct previous decoding leaves when the current decoding leaf is visited, and the error event under the genie-aided PSC decoding is a union of the events that the HBEs of the decoding leaves are incorrect. Then, for a decoding sub-tree having \( q \) leaves \( N_{\text{dec}}^{(t)} \), \( t = 0, 1, \ldots, q-1 \), the FER of the PSC decoding is

\[
\frac{1}{L} \sum_{i=0}^{L-1} \prod_{j=0}^{\ell^{(t)}-1} (1 - 2\beta_{j}^{(t)}(i))^{\alpha_{j}^{(t)}}
\]

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
decoding is given by

$$\text{FER} = \Pr \left\{ \bigcup_{t=0}^{q-1} E^{(t)} \right\},$$  

where $E^{(t)}$ denotes the error event for $\mathcal{N}^{(t)}_{\text{dec}}$ under the genie-aided PSC decoding, and which can be upper bounded as

$$\text{FER} \leq \sum_{t=0}^{q-1} \Pr \{ E^{(t)} \},$$  

and lower bounded as

$$\text{FER} \geq \max_{0 \leq t \leq q-1} \Pr \{ E^{(t)} \}.$$  

To analyze the ML decoding error probability $\Pr \{ E^{(t)} \}$ of a decoding leaf $\mathcal{N}^{(t)}_{\text{dec}}$, we first define its “channel” transition probability as

$$P(y, \{ v^{(i)} \}_{i=0}^{t-1}, v^{(t)}) \triangleq \sum_{\{v^{(i)}\}_{i=t+1}^{q-1} \in \mathcal{P}_{2}} \frac{1}{2^{N-t}} \Pr \{ y | \{ v^{(i)} \}_{i=0}^{q-1} \},$$  

where $y \in \mathbb{F}^N$ and $v^{(i)} \in \mathbb{F}^2$, $i = 0, 1, \ldots, q - 1$. Recall that $\mathcal{N}^{(t)}_{\text{dec}}$ is associated with the code $\mathcal{J}^{(t)}$. Let $v^{(0)}_0$ be the all-zero codeword in $\mathcal{J}^{(t)}$ and $v^{(i)}_1$, $i = 1, 2, \ldots, |\mathcal{J}^{(t)}| - 1$, be the non-zero codewords in $\mathcal{J}^{(t)}$. The Hamming weight of $v^{(i)}_i$ is denoted by $w_H(v^{(i)}_i)$. Define by $E^{(0)}_{0-i}$ the event that $v^{(i)}_i$ is not less likely than $v^{(0)}_0$, i.e.,

$$E^{(0)}_{0-i} \triangleq \{ y \in \mathbb{F}^N | P(y, \{ 0^{(i)} \}_{i=0}^{t-1}, v^{(i)}_i) \geq P(y, \{ 0^{(i)} \}_{i=0}^{t-1}, v^{(0)}_0) \}.$$  

Based on the Bonferroni inequalities [41], we have

$$\Pr \{ E^{(t)} \} \leq \sum_{1 \leq i \leq |\mathcal{J}^{(t)}|-1} \Pr \{ E^{(0)}_{0-i} \},$$  

and

$$\Pr \{ E^{(t)} \} \geq \sum_{1 \leq i \leq |\mathcal{J}^{(t)}|-1} \Pr \{ E^{(0)}_{0-i} \} - \sum_{1 < i < j \leq |\mathcal{J}^{(t)}|-1} \Pr \{ E^{(0)}_{0-i} \cap E^{(0)}_{0-j} \}.$$  

Since the BPSK-AWGN channel is memoryless, the transition probability associated with $\mathcal{N}^{(t)}_{\text{dec}}$ can be written as a product of the transition probabilities associated with the components of $\mathcal{N}^{(t)}_{\text{dec}}$. Denote by $P(y, \{ v^{(i)} \}_{i=0}^{t-1}, v^{(t)}_j)$ the transition probability associated with the $j$-th component of $\mathcal{N}^{(t)}_{\text{dec}}$. We have

$$P(y, \{ v^{(i)} \}_{i=0}^{t-1}, v^{(t)}) = \prod_{j=0}^{t-1} P(y, \{ v^{(i)} \}_{i=0}^{t-1}, v^{(t)}_j).$$  

That is, the vector channel associated with $\mathcal{N}^{(t)}_{\text{dec}}$ can be split into $t$ different and independent component channels. Note that the ML decoding performance over the vector channel is worse than that over $t$ uses of the best component channel and better than that over $t$ uses of the worst component channel. Then, we present two upper bounds according to the worst component channel and a lower bound according to the best component channel as below.

By using the GA method [35], we can estimate the noise variance of the worst component channel, denoted by $\sigma_{\text{worst}}^2$, and that of the best component channel, denoted by $\sigma_{\text{best}}^2$. Then, according to [42], an upper bound on the error probability $\Pr \{ E^{(t)} \}$ can be obtained by

$$\Pr \{ E^{(t)} \} \leq \sum_{1 \leq i \leq |\mathcal{J}^{(t)}|-1} Q(\sqrt{w_H(v^{(i)}_i)} / \sigma_{\text{worst}}),$$  

and a lower bound on the error probability $\Pr \{ E^{(t)} \}$ can be obtained by

$$\Pr \{ E^{(t)} \} \geq \sum_{1 \leq i \leq |\mathcal{J}^{(t)}|-1} Q(\sqrt{w_H(v^{(i)}_i)} / \sigma_{\text{best}}) - \sum_{1 < i < j \leq |\mathcal{J}^{(t)}|-1} Q(\psi_{ij}(\sqrt{w_H(v^{(i)}_i)} / \sigma_{\text{best}}) - \sqrt{w_H(v^{(j)}_j)} / \sigma_{\text{best}}),$$

where

$$Q(x) \triangleq \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} e^{-t^2/2} dt,$$

and

$$\psi_{ij} \triangleq \frac{1}{\sqrt{2\pi}} \int_{0}^{\pi/4} \frac{1}{\cos^2 \theta + \sin^2 \theta} \exp \left[ \frac{x^2 - 1 - \rho \sin \theta}{2 (1 - \rho^2) \sin^2 \theta} \right] d\theta + \frac{1}{\sqrt{2\pi}} \int_{0}^{\pi/4} \frac{1}{\cos^2 \theta + \sin^2 \theta} \exp \left[ \frac{-y^2 - 1 + \rho \sin \theta}{2 (1 - \rho^2) \sin^2 \theta} \right] d\theta,$$

and

$$\rho_{ij} \triangleq \frac{w_H(v^{(i)}_i) + w_H(v^{(j)}_j) - w_H(v^{(i)}_i) - w_H(v^{(j)}_j)}{2 \sqrt{w_H(v^{(i)}_i) w_H(v^{(j)}_j)}}.$$  

In addition, by recursively computing the Bhattacharyya bounds [1], we can obtain the Bhattacharyya upper bound of the worst component channel, denoted by $Z_{\text{worst}}^{(t)}$. The error probability $\Pr \{ E^{(t)} \}$ can also be upper bounded by

$$\Pr \{ E^{(t)} \} \leq \sum_{1 \leq i \leq |\mathcal{J}^{(t)}|-1} Z_{\text{worst}}^{(t)} w_H(v^{(i)}_i).$$  

Finally, by substituting (33), (38) into (26) and substituting (34) into (27), we can derive the following bounds on the FER of the BBT polar code with the PSC decoding.

1) The upper bound based on GA (G-UB):

$$\text{FER} \leq \sum_{t=0}^{q-1} \sum_{1 \leq i \leq |\mathcal{J}^{(t)}|-1} Q(\sqrt{w_H(v^{(i)}_i)} / \sigma_{\text{worst}}).$$  

2) The upper bound based on Bhattacharyya bounds (B-UB):

$$\text{FER} \leq \sum_{t=0}^{q-1} \sum_{1 \leq i \leq |\mathcal{J}^{(t)}|-1} (Z_{\text{worst}}^{(t)} w_H(v^{(i)}_i)).$$
between the proposed PSCL(8) decoding with $\tau = 1$ and the conventional SC decoding are shown in Fig. 7. For $N = 384$ and $R \in \{1/4, 1/2, 3/4\}$.

3) The lower bound (LB):

$$\text{FER} \geq \max_{0 \leq t \leq q-1} \left\{ \sum_{1 \leq i \leq |\psi^{(t)}| - 1} Q(\sqrt{w_{\text{H}}(v_{i}^{(t)})/\sigma_{\text{best}}^{(t)}}) - \sum_{1 \leq i < j \leq |\psi^{(t)}| - 1} \psi_{ij} \sqrt{w_{\text{H}}(v_{i}^{(t)})/\sigma_{\text{best}}^{(t)} \cdot \sqrt{w_{\text{H}}(v_{j}^{(t)})/\sigma_{\text{best}}^{(t)}}} \right\},$$

(41)

C. Performance Evaluation and Latency Analysis

1) Decoding Performance: We conduct simulations for the BBT polar codes constructed by the PW method introduced in Subsection III-C over a BPSK-AWGN channel. For $N = 384$ and $R \in \{1/4, 1/2, 3/4\}$, the performance comparisons between the proposed PSC decoding with $\tau = 1$ and the conventional SC decoding are shown in Fig. 7. For $N = 384$ and $R = 1/2$, the performance comparisons between the proposed PSCL(8) decoding with $\tau \in \{1, 2, 3\}$ and the conventional SCL(8) decoding are shown in Fig. 8. We see that the proposed PSC/PSCL(8) decoding can have the same performance as the conventional SC/SCL(8) decoding. However, the PSC-based decoding avoids a complete traversal of a coding tree and can reduce the decoding latency. The G-UB (39), B-UB (40) and LB (41) of the PSC decoding with $\tau = 1$ are shown in Fig. 9. We see that, for $\tau = 1$, the G-UB is close to the LB in the high SNR region. We also see that the B-UB is looser than the G-UB. However, calculating the B-UBs is simpler.

2) Decoding Latency: We now compare the decoding latency of the proposed PSC decoder with state-of-art decoders [1], [24] under a similar assumption to that in [25], [31], and [32] that there is no resource limitation so that all the parallelizable operations can be performed simultaneously. First, the LLR calculations defined in (4) and (5) require one time step, while the hard-decisions and the bit-operations are performed instantaneously. Second, the repetition nodes, the single-parity-check nodes and the ML decodable nodes require one time step, while the rate-0 nodes and rate-1 nodes are performed instantaneously. Table I shows the numbers of decoding time steps for $N = 384$ and $R \in \{1/4, 1/2, 3/4\}$. We see that the PSC decoder requires less steps than the conventional SC decoder [1]. Moreover, for the PSC decoding, the larger the dimension threshold $\tau$ is, the less the number of decoding time steps is. Note that we can also take four special nodes (rate-0/rate-1/repetition/single-parity-check) into account in the proposed PSC decoding, resulting in fast PSC decoding. From Table I, we see that the fast PSC decoder requires less steps than the fast SC decoder [24].

V. CONCLUSION

In this paper, we have defined a coding tree and then proposed a length-flexible polar coding scheme. For code construction over BPSK-AWGN channels, we have presented an SNR-dependent and two SNR-independent methods. It is worth pointing out that the constructions can be adapted to general B-DMCs. For example, we may use the density evolution (similar to the Gaussian approximation proposed in this paper) where the LLRs of a root node are set to $\ln \frac{1-p}{p}$ for a binary symmetric channel with crossover probability $p$. Numerical results show that the error-rate performance of the proposed BBT polar codes is comparable to that of the 5G polar codes. Further, we have also proposed the partitioned SC-based decoding, which can be implemented over the
decoding sub-tree. Especially, we have derived three bounds to evaluate the performance of the PSC decoding. Numerical results show that the proposed PSC-based decoding performs as well as the conventional SC-based decoding but has less complexity and low latency.

REFERENCES

[1] E. Arikan, “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] 3GPP, “Multiplexing and channel coding v.15.3.0,” 3GPP, Sophia Antipolis, France, Tech. Rep. 38.212, 2018.

[3] A. Eslami and H. Pishro-Nik, “A practical approach to polar codes,” in Proc. IEEE Int. Symp. Inf. Theory, Jul. 2011, pp. 16–20.

[4] K. Niu, K. Chen, and J.-R. Lin, “Beyond turbo codes: Rate-compatible punctured polar codes,” in Proc. IEEE Int. Conf. Commun. (ICC), Budapest, Hungary, Jun. 2013, pp. 3423–3427.

[5] D.-M. Shin, S.-C. Lim, and K. Yang, “Design of length-compatible polar codes based on the reduction of polarizing matrices,” IEEE Trans. Commun., vol. 61, no. 7, pp. 2593–2599, Jul. 2013.

[6] R. Wang and R. Liu, “A novel puncturing scheme for polar codes,” IEEE Trans. Commun. Lett., vol. 18, no. 12, pp. 2081–2084, Dec. 2014.

[7] V. Mitrolievskaya, “Shortened polar codes,” IEEE Trans. Inf. Theory, vol. 61, no. 9, pp. 4852–4865, Sep. 2015.

[8] M. Jang et al., “Rate matching for polar codes based on binary domination,” IEEE Trans. Commun., vol. 67, no. 10, pp. 6668–6681, Oct. 2019.

[9] J. Zhao, W. Zhang, and Y. Liu, “A novel puncturing scheme of low rate polar codes based on fixed information set,” IEEE Commun. Lett., vol. 25, no. 7, pp. 2104–2108, Jul. 2021.

[10] T.-H. Han, B. Kim, and J.-H. Yim, “Rate-compatible punctured polar codes,” IEEE Commun. Lett., vol. 26, no. 4, pp. 753–757, Apr. 2022.

[11] K. Chen, K. Niu, and J. Lin, “A hybrid ARQ scheme based on polar codes,” IEEE Commun. Lett., vol. 17, no. 10, pp. 1996–1999, Oct. 2013.

[12] H. Saber and I. Marsland, “An incremental redundancy hybrid ARQ scheme via puncturing and extending of polar codes,” IEEE Trans. Commun., vol. 63, no. 11, pp. 3964–3973, Nov. 2015.

[13] L. Ma, J. Xiong, and Y. Wei, “An incremental redundancy HARQ scheme for polar codes,” IEEE Commun. Lett., vol. 27, no. 1, pp. 1708-0967, Sep. 2021.

[14] M.-M. Zhao, G. Zhang, C. Xu, H. Zhang, R. Li, and J. Wang, “An adaptive IR-HARQ scheme for polar codes by polarizing matrix extension,” IEEE Commun. Lett., vol. 22, no. 7, pp. 1306–1309, Jul. 2018.

[15] M. Jang, J.-H. Kim, S. Myung, H. Yang, and S.-H. Kim, “Structural extension of polar code via simplex kernels,” IEEE Trans. Commun., vol. 68, no. 12, pp. 7337–7351, Dec. 2020.

[16] V. Bioglio, F. Gabry, I. Land, and J.-C. Belfiore, “Multi-kernel polar codes,” IEEE Trans. Commun., vol. 68, no. 9, pp. 5350–5362, Sep. 2020.

[17] P. Trifonov, “Randomized chained polar subcodes,” in Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW), Barcelona, Spain, Apr. 2018, pp. 25–30.

[18] A. Cavatassi, T. Tonnellier, and W. J. Gross, “Asymmetric construction and design principles,” IEEE Trans. Commun., vol. 68, no. 9, pp. 5350–5362, Sep. 2020.

[19] P. Trifonov, “Randomized chained polar subcodes,” in Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW), Barcelona, Spain, Apr. 2018, pp. 25–30.

[20] A. Cavatassi, T. Tonnellier, and W. J. Gross, “Asymmetric construction and design principles,” IEEE Trans. Commun., vol. 68, no. 9, pp. 5350–5362, Sep. 2020.

[21] K. Niu and Y. Li, “Polar codes for fast fading channel: Design based on polar spectrum,” IEEE Trans. Veh. Technol., vol. 69, no. 9, pp. 10103–10114, Sep. 2020.

[22] K. Niu and Y. Li, “Polar coded diversity on block fading channels via polar spectrum,” IEEE Trans. Signal Process., vol. 67, no. 9, pp. 4007–4022, 2019.

[23] G. He et al., “Beta-expansion: A theoretical framework for fast and recursive construction of polar codes,” in Proc. IEEE Global Commun. Conf., Singapore, Dec. 2017, pp. 1–6.

[24] A. Alamdar-Yazdi and F. R. Kschischang, “A simplified successive-cancellation decoder for polar codes,” IEEE Commun. Lett., vol. 15, no. 12, pp. 1378–1380, Dec. 2011.

[25] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, “Fast polar decoders: Algorithm and implementation,” IEEE J. Sel. Areas Commun., vol. 32, no. 5, pp. 946–957, May 2014.

Xinyuanmeeng Yao received the B.S. degree in information and computing science from the Minzu University of China, China, in 2019. She is currently pursuing the Ph.D. degree with Sun Yat-sen University, China. Her research interests include polar codes, joint source, channel coding, and unequal message protection.

Xiao Ma (Member, IEEE) received the Ph.D. degree in communication and information systems from Xidian University, China, in 2000. He is currently a Professor with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. From 2000 to 2002, he was a Post-Doctoral Fellow with Harvard University, Cambridge, MA, USA. From 2002 to 2004, he was a Research Fellow with the City University of Hong Kong. His research interests include information theory, channel coding theory and their applications to communication systems, digital coding systems. He was a co-recipient with A. Kavčič and N. Varnica of the 2005 IEEE Best Paper Award in Signal Processing and Coding for Data Storage.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.