Achieving PAC Code Performance with SCL Decoding without Extra Computational Complexity

Samet Gelincik, Philippe Mary, Jean-Yves Baudais, Anne Savard

Achieving PAC Code Performance with SCL Decoding without Extra Computational Complexity

Samet Gelincik*, Philippe Mary*, Jean-Yves Baudais* and Anne Savard†‡
* Univ Rennes, INSA Rennes, CNRS, IETR-UMR 6164, F-35000 Rennes, France
† IMT Nord Europe, Institut Mines Télécom, Centre for Digital Systems, F-59653 Villeneuve d'Ascq, France
‡ Univ. Lille, CNRS, Centrale Lille, UPHF, UMR 8520 - IEMN, F-59000 Lille, France

***Abstract*—For finite blocklength polar codes, the minimum distance and the number of low weight codewords are essential to obtain good performance under successive cancellation list decoding with moderate and high list sizes. In this paper, we propose a code design method to decrease the number of low weight codewords for some information lengths with a very low computational complexity. In the proposed method, some information bits are encoded by several rows of the polar encoding matrix, i.e., each of the dynamic frozen bits is chosen the same as one of the preceding information bits. The dynamic frozen bit index set is determined by using the connection between the binary representation of the row indices and the number of common 1-bit positions of any given rows. The resulting design is shown to perform as well as polarization-adjusted-convolutional codes [9] under successive cancellation list decoding but with significant computational complexity savings. These findings pave the way for the use of polar codes in applications with stringent complexity and with low energy consumption constraints.**

***Index Terms*—Polar Codes, dynamic frozen function, Reed-Muller codes, successive cancellation list decoding**

I. Introduction

Polar codes are the first proven (symmetric) capacity-achieving block codes in asymptotic regime with low decoding complexity for discrete binary input memoryless channels [1]. Polar codes have recently been adopted for control channels in 5G networks and may be used for other purposes in future wireless networks. Indeed in beyond 5G networks, use cases such as traffic safety, industrial control, medical and internet services will rely on highly reliable low latency communications [2], [3]. Therefore, it is important to design polar codes that are efficient in short blocklength regime.

However, for short and moderate code sizes, polar codes under successive cancellation decoding performs poorly due to the incomplete polarization of the channel, and hence, designing codes with good minimum distance properties becomes very important to improve performance. For instance in [4], cyclic redundancy check (CRC) concatenation is proposed to improve the minimum distance properties of the code to help successive cancellation list (SCL) decoding. In [5] and [6], the CRC polynomial has been optimized to increase minimum Hamming distance and minimizing the number of low weight codewords by an exhaustive search over a wide range of CRC polynomials.

This work has been partially supported by IRCICA, CNRS USR 3380, Lille, France and the French National Agency for Research (ANR) under grant ANR-16-CE25-0001 ARBurst.

Authors in [7] proposed to improve the minimum distance properties of a polar code by choosing the frozen bits as a function of preceding information bits. This technique has been called dynamic freezing and has been extended in [8] by introducing the concept of precoded polar codes, in which any linear codes can be represented with a precoding matrix followed by a standard polar transformation matrix. In precoded polar codes, information bit indices are not necessarily chosen as it is in the original polar codes. In this article, we use the term *polar-like* codes to represent a general class of linear precoded codes with information and frozen bits[1].

Recently, polarization-adjusted convolutional (PAC) code has been proposed by Arikan [9] and studied further in [10] and [11]. The information indices are chosen according to the Reed-Muller (RM) design rule, i.e., the rows of the encoding matrix with the highest Hamming weights are selected as information rows. PAC codes have been shown to perform close to the normal approximation (NA) of the second order rate [12]. This method allows to reduce drastically the number of minimum weight codewords by employing an outer convolutional encoder that can be represented with an upper triangular matrix that in turn has the property to reduce the number of minimum weight codewords [13]. However, the improved performance comes at the cost of extra computational complexity. For instance, the decoding complexity of PAC codes is of the order of magnitude of SCL decoder's itself for short blocklengths. In [6], state of the art has been moved a step forward by combining variable list size SCL decoding with a sphere decoder. The technique has been shown to approach the achievable error probability in finite block length but at the cost of very high complexity of the sphere decoder that may prevent from its usage in applications with low computational capability.

In this paper, we propose an alternative way for designing polar-like codes in the sense that some frozen bits are determined with a dynamic freezing function with a single argument, and not with a combination of preceding information bits, as it has been done in literature. The proposed process of determining the dynamic freezing functions is named *row merging*. The proposal is based on the relationship between the binary representation of row indices of the polar encoding

[1] In [8], the name *polar-like* is used to represent original polar codes of [1] with dynamic frozen bits.

matrix and the number of common 1-bit positions among the rows. In order to construct the code, the information bit indices are first chosen according to the Reed-Muller rule, which provides the highest minimum distance among the codewords without dynamic frozen bits. Then, we search for the rows that can be merged together in order to decrease the number of low weight codewords. Our method provides codes that perform as well as the PAC codes [9] in terms of frame error rate (FER) in binary-input additive white Gaussian noise (BI-AWGN) channel but at a much smaller computational complexity.

II. Preliminaries

A. Notations

Vectors of length N are represented in row and denoted with sans serif font x. The j-th entry of the vector x is x_j with $j \in \mathcal{N} = \{0, 1, \cdots, N-1\}$. The set of integers from j to $k-1$ are represented by $[j, k)$ and the set of integers from j to k are represented by $[j, k]$. A sub-vector drawn from x is denoted as $\mathsf{x}_j^k = [x_j, \cdots, x_k]$, $\forall (j,k) \in \mathcal{N}^2$, $j < k$, and simply as x^k if $j = 0$. Index sets are denoted by caligraphic letters, e.g. $\mathcal{A}$, and are sorted in the *ascending* order. For a vector x and set $\mathcal{A} \subset \mathcal{N}$, $\mathsf{x}_\mathcal{A} = \{x_j : j \in \mathcal{A}\}$. The matrices are denoted with uppercase sans serif font, e.g., G. For a matrix $\mathsf{G} \in \{0,1\}^{N \times N}$ and index set $\mathcal{A} \subseteq [0, N-1]$, $\mathsf{G}_\mathcal{A}$ denotes the matrix consisting of rows of G indexed by $\mathcal{A}$. Modulo-2 binary addition is denoted by $\oplus$.

For any index $j \in \mathcal{N}$, $N = 2^n$, the n-bit binary representation of j is denoted by b_j and is a vector of length n. The indexing of elements in b_j is started from the least significant bit, i.e., the rightmost bit, and $b_{j,k}$ is the k-th least significant bit, $k \in [0, n-1]$. The number of ones and zeros in any binary vector x is denoted by $i_1(\mathsf{x})$ and $i_0(\mathsf{x})$, respectively. For any $\mathsf{x}_1, \mathsf{x}_2 \in \{0,1\}^N$, $\mathsf{x}_1 \barwedge \mathsf{x}_2$ is the element-wise logical $'$AND$'$ operation and $\mathsf{x}_1 \barvee \mathsf{x}_2$ is the element-wise logical $'$OR$'$ operation. Moreover, $\mathcal{P}_1(\mathsf{x}), \mathcal{P}_0(\mathsf{x})$ denotes the index sets of 1's and 0's, respectively, of any binary vector x.

B. Polar-like and Polar Codes

Let $\mathcal{C}(N = 2^n, k, \mathcal{A})$, $n \in \mathbb{N}$, be a polar-like code defined as a binary linear block code with generator matrix G_N,

$$\mathsf{G}_N = \begin{bmatrix} \mathsf{g}_0^T & \mathsf{g}_1^T & \cdots & \mathsf{g}_{N-1}^T \end{bmatrix}^T = \mathsf{G}_{\text{Ker}}^{\otimes n}, \ \mathsf{G}_{\text{Ker}} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

where $\otimes$ is the Kronecker product and $[\cdot]^T$ the transpose operator. The input bits are denoted by $\mathsf{u} \in \{0,1\}^N$, where k of them are information bits and indexed by $\mathcal{A} \subseteq \mathcal{N}$, and the rest are indexed by $\mathcal{F} = \mathcal{N} \setminus \mathcal{A}$ and called *frozen* bits since they can be set to any binary value and are known *a priori* by the decoder. Alternatively, the jth row g_j of generator matrix G_N is represented as

$$\mathsf{g}_j = \hat{\mathsf{g}}_{b_{j,(n-1)}} \otimes \hat{\mathsf{g}}_{b_{j,(n-2)}} \otimes \cdots \otimes \hat{\mathsf{g}}_{b_{j,0}} \tag{1}$$

where $\hat{\mathsf{g}}_0 = [1\ 0]$ and $\hat{\mathsf{g}}_1 = [1\ 1]$.

Let $W : \mathcal{X} \to \mathcal{Y}$ be a binary-input symmetric discrete memoryless channel, where $\mathcal{X} = \{0,1\}$, with *a priori* probabilities $p_X(0) = p_X(1) = 1/2$, and $W(y|x)$ are transition probabilities $\forall (x,y) \in \mathcal{X} \times \mathcal{Y}$. It has been shown in [1] that the application of a linear mapping such as $\mathsf{x} = \mathsf{u}\mathsf{G}_N$ for any $\mathsf{u} \in \mathcal{X}^N$ creates synthetic subchannels $W_N^{(j)} : \mathcal{X} \to \mathcal{Y}^N \times \mathcal{X}^j$ such as

$$W_N^{(j)}(\mathsf{y}, \mathsf{u}^{j-1} | u_j) = \sum_{\mathsf{u}_{j+1}^{N-1}} \frac{1}{2^{N-1}} W^N(\mathsf{y} | \mathsf{u}\mathsf{G}_N) \tag{2}$$

in which $p(\mathsf{u}) = 1/2^N$ and $W^N : \mathcal{X}^N \to \mathcal{Y}^N$ denotes the vector channel which, in memoryless channel, becomes

$$W^N(\mathsf{y} | \mathsf{x}) = \prod_{j=0}^{N-1} W(y_j | x_j). \tag{3}$$

The polar code determines the information bit index set $\mathcal{A}$ as the k most reliable bits and the rest of the indices are the frozen bit set $\mathcal{F}$ and transmitted as $u_j = 0, \forall j \in \mathcal{F}$. To exploit the polarization phenomenon imposed by the linear transformation G_N, the SC decoding has been proposed [1]:

$$\hat{u}_j = \begin{cases} 0 \text{ if } W_N^{(j)}(\mathsf{y}, \hat{\mathsf{u}}^{j-1} | 0) \geq W_N^{(j)}(\mathsf{y}, \hat{\mathsf{u}}^{j-1} | 1) \\ 1 \text{ otherwise,} \end{cases}, \ j \in \mathcal{A} \tag{4}$$

where $\hat{u}_j = 0, \ \forall j \in \mathcal{F}$.

The *dynamic* frozen bits concept for polar codes has been first introduced in [7]. Here, we generalize the definition of [7] for polar-like codes and name them as *dynamic* polar-like codes, which is the same as precoded polar codes of [8] and [14]. In these codes, some frozen bits $j \in \mathcal{F}$ are determined dynamically using previous bits[2] with a Boolean function $h_j : \{0,1\}^j \to \{0,1\}$:

$$u_j = h_j(u_0, u_1, \cdots, u_{j-1}), \quad j \in \mathcal{F} \tag{5}$$

For dynamic polar-like codes, the encoding process is:

$$\mathsf{x} = \Big(\bigoplus_{m \in \mathcal{A}} u_m \cdot \mathsf{g}_m \Big) \oplus \Big(\bigoplus_{j \in \mathcal{F}} h_j \cdot \mathsf{g}_j \Big), \tag{6}$$

where h_j is a short-hand of $h_j(u_0, u_1, \cdots, u_{j-1})$ and $h_j(\cdot) = 0$ if the jth bit is not a dynamic frozen bit.

The SC decoding of dynamic polar-like codes is the same as polar codes for information bits. However, the dynamic frozen bits are obtained with their corresponding functions

$$\hat{u}_j = h_j(\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_{j-1}) \tag{7}$$

during the course of decoding. The reader may refer to [4] for SCL decoding details.

C. Distance Spectrum and Symbol Error Probability

For a $\mathcal{C}(N, k)$ binary linear block code, the minimum distance $d_{\min}$ is the minimum Hamming distance between any two codewords $\mathsf{x}, \mathsf{x}' \in \mathcal{C}$, that is to say

$$\min_{\substack{\mathsf{x} \neq \mathsf{x}' \\ \mathsf{x}, \mathsf{x}' \in \mathcal{C}}} i_1(\mathsf{x} \oplus \mathsf{x}') = \min_{\mathsf{x} \neq \mathsf{0}, \mathsf{x} \in \mathcal{C}} i_1(\mathsf{x}) \tag{8}$$

[2] In [7], (5) is defined as a function of the preceding information bits. Here, we give a general definition.

Let $A_w(\mathcal{C})$ be the number of codewords in $\mathcal{C}$ with the Hamming weight w, i.e.,

$$A_w(\mathcal{C}) := |\{\mathsf{x} : i_1(\mathsf{x}) = w, \mathsf{x} \in \mathcal{C}\}|, \tag{9}$$

Then, the distance properties of the code $\mathcal{C}$ is described by the set $A_{\mathcal{C}} = \{A_0, A_1, \cdots, A_N\}$, which is called *distance spectrum* (or the weight enumerator function).

The decoding error probability of any code $\mathcal{C}(N,k)$ under maximum likelihood (ML) decoding, P_{ML}, is upper bounded by the union bound. For the binary phase-shift keying modulation over AWGN channel, the union bound is given as [15]:

$$P_{\text{ML}} \leq P_{ub} = \sum_{w=1}^{N} A_w Q(\sqrt{w \cdot \mathsf{SNR}}). \tag{10}$$

For high SNR, the union bound is well approximated by [5]

$$P_{ub} \approx A_{d_{\min}} Q(\sqrt{d_{\min} \cdot \mathsf{SNR}}), \tag{11}$$

which indicates that, at high SNR's, the minimum weight codewords contribute more to the union bound than codewords with higher Hammming weights. To estimate the distance spectrum, we adopt the SCL decoding method proposed in [16]. For any polar-like code with $\mathsf{u}_{\mathcal{F}} = 0$, the minimum distance is given as $d_{\min} = 2^{\min_{j\in\mathcal{A}} i_1(\mathsf{b}_j)}$ in [17].

III. New Code Design with Dynamic Frozen Bits

This section summarizes our main contribution in this paper.

A. Number of Common 1-bit Positions in G_N Rows

The jth row of G_N, g_j, in (1) can be divided into n disjoint vectors $\mathsf{r}_j(\ell)$, such that for $\ell \in [1, n-1]$:

$$\mathsf{r}_j(\ell) = \begin{cases} \mathsf{0}_0^{2^\ell - 1} & \text{if } b_{j,\ell} = 0 \\ [\mathsf{r}_j(0)\mathsf{r}_j(1)\cdots\mathsf{r}_j(\ell-1)] & \text{if } b_{j,\ell} = 1 \end{cases} \tag{12}$$

and $\mathsf{r}_j(0) = \hat{\mathsf{g}}_{b_{j,0}}$. The length of $\mathsf{r}_j(\ell)$ is 2^ℓ if $\ell \geq 1$ and is equal to 2 if $\ell = 0$. A close look at the recursive nature of $\mathsf{r}_j(\ell)$ reveals that the entries of g_j, indexed by the set $\mathcal{K}_\ell$, defined as

$$\mathcal{K}_\ell = \{k : b_{k,\ell} = 1, k \in \mathcal{N}\}, \tag{13}$$

are zeros if $b_{j,\ell} = 0, \ell \in [0, n-1]$. More precisely, the set $\mathcal{K}_\ell$ gets the following particular form

$$\mathcal{K}_\ell = \{k : k = a\,2^{\ell+1} - b, a \in [1, 2^{n-1-\ell}], b \in [0, 2^\ell - 1]\}, \tag{14}$$

and $\mathsf{g}_{j,\mathcal{K}_\ell} = 0$ if $b_{j,\ell} = \mathsf{0}$ whatever $\mathsf{b}_{j,\ell'}, \ell' \neq \ell$, is.

We define the complementary set as $\mathcal{K}_\ell^c = \mathcal{N} \setminus \mathcal{K}_\ell$. Note that, by (13), $|\mathcal{K}_\ell| = |\mathcal{K}_\ell^c| = 2^{n-1}$ for any $\ell \in [0, n-1]$ since there are $n-1$ free positions in b_k if the ℓth bit is set to 1. Moreover

$$|\bigcap_{\ell\in\mathcal{B}} \mathcal{K}_\ell| = |\bigcap_{\ell\in\mathcal{B}} \mathcal{K}_\ell^c| = N/2^{|\mathcal{B}|} \tag{15}$$

for any subset $\mathcal{B} \subseteq [0, n)$.

Theorem 1. *Let G_N be the generator matrix for a polar-like code $(N = 2^n, k)$ with arbitrary information bit indices set $\mathcal{A}$ and frozen bit indices set $\mathcal{F}$. For any $\mathcal{T} \subseteq \mathcal{N}$, the number of common 1-bit positions for all $j \in \mathcal{T}$ is given as:*

$$i_1\big(\overline{\bigcap}_{j\in\mathcal{T}} \mathsf{g}_j\big) = 2^{i_1(\overline{\cap}_{j\in\mathcal{T}} \mathsf{b}_j)}. \tag{16}$$

Proof. By (12) note that if $b_{j,\ell} = 0$, all entries of g_j, i.e., $g_{j,k}$, indexed by $k \in \mathcal{K}_\ell$ are zero. Therefore, the number of zeros at row vector g_j is given by

$$i_0\left(\mathsf{g}_j\right) = |\bigcup_{\substack{\ell=0,\\ b_{j,\ell}=0}}^{n-1} \mathcal{K}_\ell| \tag{17}$$

and the number of ones is

$$\begin{aligned} i_1(\mathsf{g}_j) = N - i_0\big(\mathsf{g}_j\big) &\overset{\text{(a)}}{=} |\,\mathcal{N} \setminus \bigcup_{\substack{\ell=0,\\ b_{j,\ell}=0}}^{n-1} \mathcal{K}_\ell\,| \\ &\overset{\text{(b)}}{=} |\bigcap_{\substack{\ell=0,\\ b_{j,\ell}=0}}^{n-1} \mathcal{K}_\ell^c| \overset{\text{(c)}}{=} \frac{2^n}{2^{i_0(\mathsf{b}_j)}} = 2^{i_1(\mathsf{b}_j)} \end{aligned} \tag{18}$$

where (a) and (b) are due to standard operations on sets and (c) is due to (15) since $|\mathcal{B}| = i_0(\mathsf{b}_j)$ in (b). In general, for any $\mathcal{T} \subseteq \mathcal{N}$, the number of common ones is given as

$$\begin{aligned} i_1\big(\overline{\bigcap}_{j\in\mathcal{T}} \mathsf{g}_j\big) = N - i_0\big(\bar{\bigcup}_{j\in\mathcal{T}} \mathsf{g}_j\big) &= |\,\mathcal{N} \setminus \bigcup_{\substack{\ell:\ b_{j,\ell}=0,\\ j\in\mathcal{T}}} \mathcal{K}_\ell\,| \\ &= |\bigcap_{\substack{\ell:\ b_{j,\ell}=0,\\ j\in\mathcal{T}}} \mathcal{K}_\ell^c| \overset{\text{(a)}}{=} \frac{2^n}{2^{i_0(\bar{\cup}_{j\in\mathcal{T}}\mathsf{b}_j)}} \\ &= 2^{n - i_0(\bar{\cup}_{j\in\mathcal{T}}\mathsf{b}_j)} = 2^{i_1(\bar{\cap}_{j\in\mathcal{T}}\mathsf{b}_j)}, \end{aligned} \tag{19}$$

where (a) is due to (15). To see this, note that

$$\bigcap_{\substack{\ell:\ b_{j,\ell}=0,\\ j\in\mathcal{T}}} \mathcal{K}_\ell^c = \bigcap_{\ell:\ \mathsf{b}_{\mathcal{T}}(\ell)=0} \mathcal{K}_\ell^c \tag{20}$$

where $\mathsf{b}_{\mathcal{T}} = \bar{\cup}_{j\in\mathcal{T}}\mathsf{b}_j$. □

B. Hamming Weight of the Sum of G_N Rows

Let $\mathcal{T} \subseteq \mathcal{N}$ be any subset of row indices of polar-like code generator matrix G_N and $\mathsf{g}_{\mathcal{T}}$ be

$$\mathsf{g}_{\mathcal{T}} = \bigoplus_{j\in\mathcal{T}} \mathsf{g}_j, \tag{21}$$

and $\mathcal{T}^w \subseteq \mathcal{T}$ be any subset of $\mathcal{T}$ with $|\mathcal{T}^w| = w$. Then, the Hamming weight of $\mathsf{g}_{\mathcal{T}}$ is given by the following theorem:

Theorem 2. *Let $\mathcal{T} \subseteq \mathcal{N}$ be any subset of row indices of polar-like code generator matrix G_N. Then, the Hamming weight of the sum of the rows $\mathsf{g}_j,\ j \in \mathcal{T}$ is given by*

$$i_1\big(\mathsf{g}_{\mathcal{T}}\big) = \sum_{w=1}^{|\mathcal{T}|} (-2)^{w-1} \sum_{\mathcal{T}^w\in\mathcal{T}} 2^{i_1\left(\bar{\cap}_{j\in\mathcal{T}^w}\mathsf{b}_j\right)}. \tag{22}$$

Proof. Let us proceed by induction. For any two rows $(\mathsf{g}_t, \mathsf{g}_m)$, $t, m \in \mathcal{N}$ of G_N, the Hamming weight of $\mathsf{g}_j \oplus \mathsf{g}_k$ is given by:

$$i_1(\mathsf{g}_t \oplus \mathsf{g}_m) = i_1(\mathsf{g}_t) + i_1(\mathsf{g}_m) - 2 \cdot i_1(\mathsf{g}_t \bar{\cap} \mathsf{g}_m). \tag{23}$$

Note that (23) complies with the following expression

$$i_1\big(\bigoplus_{j\in\mathcal{T}}\mathsf{g}_j\big)=\sum_{j\in\mathcal{T}}i_1(\mathsf{g}_j)+\sum_{w=2}^{|\mathcal{T}|}(-2)^{w-1}\sum_{\mathcal{T}^w\in\mathcal{T}}i_1\big(\bar{\cap}_{j\in\mathcal{T}^w}\mathsf{g}_j\big), \quad (24)$$

where $\mathcal{T}=\{t,m\}$.

Now, assume that it holds for arbitrary $\mathcal{T}\subset\mathcal{N}$. By (23), $i_1(\mathsf{g}_{\mathcal{T}'})$ for $\mathcal{T}'=\mathcal{T}\cup\{t'\}$ can be written as

$$\begin{aligned} i_1(\mathsf{g}_{\mathcal{T}'}) &= i_1(\mathsf{g}_{\mathcal{T}})+i_1(\mathsf{g}_{t'})-2\cdot i_1(\mathsf{g}_{\mathcal{T}}\bar{\cap}\mathsf{g}_{t'}) \\ &= i_1(\mathsf{g}_{\mathcal{T}})+i_1(\mathsf{g}_{t'})-2\cdot i_1(\bigoplus_{j\in\mathcal{T}}(\mathsf{g}_j\bar{\cap}\mathsf{g}_{t'})) \end{aligned} \quad (25)$$

Substituting (24) in (25) provides (26) on top of the next page where $\mathcal{T}^w$ is any subset of $\mathcal{T}$, i.e., $\mathcal{T}^w\subseteq\mathcal{T}$, with $|\mathcal{T}^w|=w$. Substituting (16) in (24) results in (22) once noted that $i_1(\mathsf{g}_j)=2^{i_1(\bar{\bigcap}_{j\in\mathcal{T}_b^1}\mathsf{b}_j)}=2^{i_1(\mathsf{b}_j)}$ for $\mathcal{T}_b^1=\{\mathsf{b}_j\}$. □

C. Code Design with Row Merging

The proposed code design is based on encoding some information bits with more than one row of the generator matrix G_N such that some frozen bits are set to their corresponding preceding information bits, in a sense that it will be made clear in the following. For a given blocklength $N=2^n$, we define the *critical information lengths* $k=k_\ell$, $\ell\in[2,n-1]$, such that

$$k_\ell=\sum_{p=\ell}^{n}\binom{n}{p}. \quad (27)$$

In order to guarantee the highest possible $d_{\min}$ without dynamically frozen bits, the set of information bits, $\mathcal{A}$, is chosen as the set of row indices which have the highest weights, which is called the *Reed-Muller* (RM) rule [1] selection, that is to say, for $k=k_\ell$:

$$\mathcal{A}=\bigcup_{p=\ell}^{n}\mathcal{N}_p \quad (28)$$

where $\mathcal{N}_p:=\{j\in\mathcal{N}:i_1(\mathsf{b}_j)=p\}$.

Even though RM row selection guarantees a polar-like code with the highest $d_{\min}$ with $\mathsf{u}_{\mathcal{F}}=0$, the number of codewords with Hamming weight $d_{\min}$ is still high as it is illustrated in Table I. This table gives the number of codewords A_d with the Hamming weight d, for the RM design, our proposed design (PD) and PAC codes. To decrease the number of low weight codewords, we propose a *heuristic* algorithm by which either the Hamming weight of the merged rows are increased or they remain unchanged but the positions of 1's in the merged rows are moved, which changes the regular structure of the positions of 1's of the information rows selected by the RM rule. For a given k_ℓ, the Hamming weight of a row $j\in\mathcal{N}_\ell$ can be increased by merging it with a row $m\in\mathcal{N}_{\ell-1}$ such that

$$i_1(b_j\bar{\cap}b_m)=\ell^*,\ m>j \quad (29)$$

where $\max\{0,2\ell-1-n\}\le\ell^*<\ell-2$. The merging with any row m such that

$$i_1(b_j\bar{\cap}b_m)=\ell-2,\ m\in\mathcal{N}_{\ell-1},\ m>j \quad (30)$$

results in a merged row with $i_1(\mathsf{g}_j\oplus\mathsf{g}_m)=2^\ell$. However, by Theorems 1 and 2, the 1's at the indices $\mathcal{P}_1(\mathsf{g}_j\bar{\cap}\mathsf{g}_m)$ are moved to the indices $\mathcal{P}_1(\mathsf{g}_m)\setminus\mathcal{P}_1(\mathsf{g}_j\bar{\cap}\mathsf{g}_m)$.

After having merged two rows (j,m) satisfying (29) one can still increase the Hamming weight by merging them with a third row $t>j,\ t\in\mathcal{N}_{\ell-2}$ such that

$$\begin{aligned} &\mathcal{P}_1(\mathsf{b}_j\bar{\cap}\mathsf{b}_m\bar{\cap}\mathsf{b}_t)=\mathcal{P}_1(b_j\bar{\cap}b_m) \text{ and} \\ &\mathcal{P}_1(\mathsf{b}_t)\subset(\mathcal{P}_1(\mathsf{b}_j\bar{\cap}\mathsf{b}_m)\cup\mathcal{P}_0(\mathsf{b}_j\bar{\cup}\mathsf{b}_m)) \text{ if } i_1(\mathsf{b}_j\bar{\cup}\mathsf{b}_m)<n \end{aligned} \quad (31)$$

For (j,m) pairs satisfying (30), the third row $t>j, t\in\mathcal{N}_{\ell-2}$ to increase the Hamming weight can be chosen as

$$\begin{aligned} &\mathcal{P}_1(\mathsf{b}_t\bar{\cap}\mathsf{b}_j)=\mathcal{P}_1(\mathsf{b}_t\bar{\cap}\mathsf{b}_m) \text{ and } i_1(\mathsf{b}_j\bar{\cap}\mathsf{b}_m\bar{\cap}\mathsf{b}_t)=i_1(\mathsf{b}_j\bar{\cap}\mathsf{b}_m)-1 \\ &\text{and}\,\mathcal{P}_1(\mathsf{b}_t)\subset(\mathcal{P}_1(\mathsf{b}_j\bar{\cap}\mathsf{b}_m)\cup\mathcal{P}_0(\mathsf{b}_j\bar{\cup}\mathsf{b}_m))\,\text{if}\,i_1(\mathsf{b}_j\bar{\cup}\mathsf{b}_m)<n. \end{aligned} \quad (32)$$

Based on these observations, for each $\ell^*\in[\max\{0,2\ell-1-n\},\ell-2]$ and for a given $k=k_\ell$, we implement Algorithm 1 to obtain pairs and triples to merge. The algorithm chooses the smallest $m\in\mathcal{N}_{\ell-1}$ satisfying (29) or (30) starting from the smallest $j\in\mathcal{N}_\ell$ for any given ℓ^*. Then, it chooses the smallest $t\in\mathcal{N}_{\ell-2}$ satisfying (31) or (32) regarding the value of ℓ^*. To obtain the codewords for any $\ell^*\in[\max\{0,2\ell-1-n\},\ell-2]$, the row merging operation is implemented at the encoding stage by choosing

$$u_m\overset{(a)}{=}h_m(u_0,\cdots,u_{m-1})=u_j, \quad (33)$$

for $m\in\mathcal{F}$, $j\in\mathcal{A}$ and $j<m$, where (a) is due to (5), or

$$\begin{aligned} h_m(u_0,\cdots,u_{m-1}) &= h_t(u_0,\cdots,u_{t-1}) \\ &= u_j, \end{aligned} \quad (34)$$

for $(m,t)\in\mathcal{F}$, $j\in\mathcal{A}$, $j<m$ and $j<t$, if there exist corresponding merging pairs or triples, respectively. These operations only require reading from the memory in practice.

Let $\mathcal{C}_{\ell^*}$ denote the codebooks obtained by applying the row merging operations proposed by Algorithm 1. Then, the codebook with the best distance spectrum is chosen, i.e., the one with the lowest number of minimum weight codewords. If the distance spectrum of two codebooks are equal for the first $w-1$ smallest terms, $w\ge2$, then the one with the smallest number of codewords at the w-th term is chosen.

IV. Numerical Results

A. Distance Spectrum and FER Performance

Table I shows the number of codewords with different Hamming weights for blocklength $N=128$ and $k=29,64,$ and 99 by employing the same experiment as the one conducted in [16]. For $k=99$, $k=64$ and $k=29$ the minimum list size is 7×10^5, 3×10^5 and 1.5×10^5 respectively. For PAC codes with $k=29$ and $k=99$, the generator polynomials have been optimized for the memory length $m=9$, following the experimentin [16] in order to produce the most favorable case of PAC codes. The chosen generator polynomial for $k=29$ and $k=99$ are $[1\,0\,1\,1\,1\,0\,0\,0\,0\,1]$ and $[1\,0\,0\,1\,0\,1\,1\,1\,1\,1]$, respectively. We observe that the proposed

$$i_1(\mathsf{g}_{\mathcal{T}'}) = \sum_{j\in\mathcal{T}} i_1(\mathsf{g}_j)+i_1(\mathsf{g}_{t'}) - 2\Big(\sum_{w=2}^{|\mathcal{T}|}(-2)^{w-2}\sum_{\mathcal{T}^w\in\mathcal{T}} i_1\big(\bar{\cap}_{j\in\mathcal{T}^w}\mathsf{g}_j\big) + \sum_{j\in\mathcal{T}} i_1\big(\mathsf{g}_j\bar{\cap}\mathsf{g}_{t'}\big) + \sum_{w=2}^{|\mathcal{T}|}(-2)^{w-1}\sum_{\mathcal{T}^w\in\mathcal{T}} i_1\big((\bar{\cap}_{j\in\mathcal{T}^w}\mathsf{g}_j)\bar{\cap}\mathsf{g}_{t'}\big)\Big)$$
$$= \sum_{j\in\mathcal{T}'} i_1(\mathsf{g}_j)+\sum_{w=2}^{|\mathcal{T}|}(-2)^{w-1}\cdot\Big(\sum_{\mathcal{T}^w\in\mathcal{T}} i_1\big(\bar{\cap}_{j\in\mathcal{T}^w}\mathsf{g}_j\big) + \sum_{\mathcal{T}^{w-1}\in\mathcal{T}} i_1\big((\bar{\cap}_{j\in\mathcal{T}^{w-1}}\mathsf{g}_j)\bar{\cap}\mathsf{g}_{t'}\big)\Big) + (-2)^{|\mathcal{T}'|-1} i_1(\bar{\cap}_{j\in\mathcal{T}'}\mathsf{g}_j)$$
$$= \sum_{j\in\mathcal{T}'} i_1(\mathsf{g}_j)+\sum_{w=2}^{|\mathcal{T}'|}(-2)^{w-1}\cdot\sum_{\mathcal{T}',w\in\mathcal{T}} i_1\big(\bar{\cap}_{j\in\mathcal{T}',w}\mathsf{g}_j\big) \tag{26}$$

Algorithm 1: Row Merging Pairs and Triples for ℓ^*.

Set $\mathcal{M}\leftarrow\mathcal{N}_{\ell-1}$ and $\mathcal{W}\leftarrow\mathcal{N}_{\ell-2}$;
for *i from 1 to* $|\mathcal{N}_\ell|$ **do**
  Set $j\leftarrow\mathcal{N}_\ell(i)$;
  Set $\mathcal{M}_j\leftarrow\{m : i_1(b_j\bar{\cap}b_m)=\ell^*, m>j, m\in\mathcal{M}\}$;
  if $\mathcal{M}_j\neq\emptyset$ **then**
    Set $(j,m)\leftarrow(j,\mathcal{M}_j(1))$;
    Set $\mathcal{M}\leftarrow\mathcal{M}\setminus m$;
    if $\ell^*<\ell-2$ **then**
      Set $\mathcal{W}_{j,m}\leftarrow\{t : t>j, t\in\mathcal{W},$ satisf. (31)$\}$;
    else
      Set $\mathcal{W}_{j,m}\leftarrow\{t : t>j, t\in\mathcal{W},$ satisf. (32)$\}$;
    end
    Set $(j,m,t)\leftarrow(j,m,\mathcal{W}_{j,m}(1))$;
    Set $\mathcal{W}\leftarrow\mathcal{W}\setminus t$ if $\mathcal{W}_{j,m}\neq\emptyset$;
  end
end

Table I: Number of low weight codewords in some codes.

	A_8	A_{10}	A_{16}	A_{18}	A_{32}	A_{34}
$(128,29)$ **RM**	–	–	–	–	10668	–
$(128,29)$ **PD**	–	–	–	–	952	–
$(128,29)$ **PAC**	–	–	–	–	348	360
$(128,64)$ **RM**	–	–	94488	NC	NC	NC
$(128,64)$ **PD**	–	–	2778	410	NC	NC
$(128,64)$ **PAC**	–	–	3120	2696	NC	NC
$(128,99)$ **RM**	188976	–	NC	NC	NC	NC
$(128,99)$ **PD**	19226	$>10^5$	NC	NC	NC	NC
$(128,99)$ **PAC**	14432	$>10^5$	NC	NC	NC	NC

'–': no codeword at this Hamming weight; 'NC': not computed

design allows to reduce the number of minimum weight codewords to the same order of magnitude of those obtained with optimized PAC codes. For $(128,64)$-code, the numbers of codewords at Hamming weights 16 and 18 obtained with our proposal are even lower than for the ones obtained with PAC codes. For $(128,29)$-code the chosen code does not have any codewords with Hamming weights 34. The number of codewords at certain Hamming weights are not computed due to the prohibitive amount of computations that they require.

Figures 1 and 2 compare the FER achieved with our design with those obtained with RM and PAC codes for three different codes, i.e., $(128,29)$, $(128,64)$ and $(128,99)$, w.r.t. the uncoded energy per bit, i.e. E_b/N_0. Indeed, since codes with different rates are compared, E_b/N_0 is a fairer base of comparison than SNR. This is the reason why one may observe a difference about 3 dB for $R=1/2$ between our results compared to PAC codes in [9]. The performance of the codes are also compared w.r.t. the lower bound of the saddle-point approximation [18] of the meta-converse bound [12] in BI-AWGN channel. This bound, referred as converse saddle-point approximation (CSA) on Figures 1 and 2, is a strict lower bound that cannot be outperformed. At the receiver side, SCL decoding with list size $L=256$, that almost reaches the ML performance, is performed. It is seen that our method performs as well as the optimized Arikan's PAC codes for various information length k, with a lower complexity, as it will be explained in next Section. Moreover, FER achieved with our design is at most at 0.5dB from CSA at 10^{-4} for $k=29$, as it has been reported in Figure 2. As expected, PD outperforms RM codes due to the huge difference in the number of lowest weight codewords between the two schemes.

B. Complexity Comparison with PAC Codes

As stated in [19], the difference of PAC encoding scheme from standard polar encoding is the rate-1 convolutional encoder that takes place before polar transformation, which is implemented through a length-m memory as follows [19]:

$$u_i=\sum_{j=0}^{m} c_j v_{i-j}, \qquad i\in\mathcal{N} \tag{35}$$

where $\mathsf{c}\in\mathbb{F}_2^{1\times(m+1)}$ with $c_0=c_m=1$ is the generator polynomial of the convolutional encoder, $\mathsf{v}_\mathcal{A}\in\mathbb{F}_2^{1\times k}$ is the information vector and $\mathsf{v}_\mathcal{F}=\mathsf{0}_0^{N-k-1}$ is the frozen vector, respectively. At the encoder side the PAC code requires $O(m\cdot N)$ additional floating point operations besides extra memory requirement [10]. Note that the extra computational complexity is equivalent to standard polar encoding for short block lengths if $m\approx\log_2 N$, which holds for the undertaken scenario, for instance.

The SCL decoding of PAC codes is implemented by adding a convolutional decoder after each SC decoder of list decoder as follows [19]:

$$\hat{v}_i=\hat{u}_i-\sum_{j=1}^{m} c_j\hat{v}_{i-j}, \qquad i\in\mathcal{A} \tag{36}$$

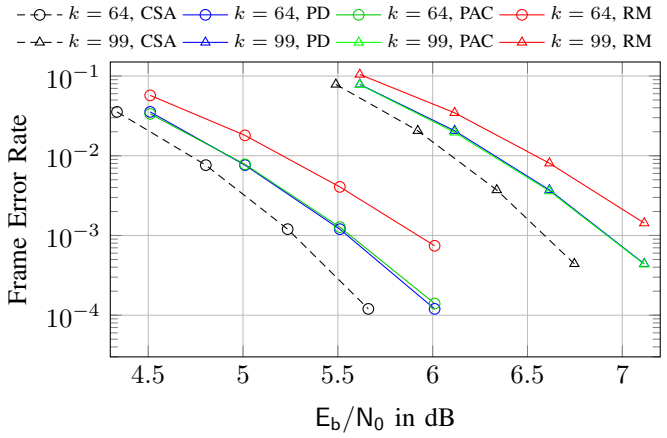

Figure 1: FER obtained with our scheme (PD) w.r.t. PAC codes and CSA for $N = 128$, $k = 64$ and $k = 99$.

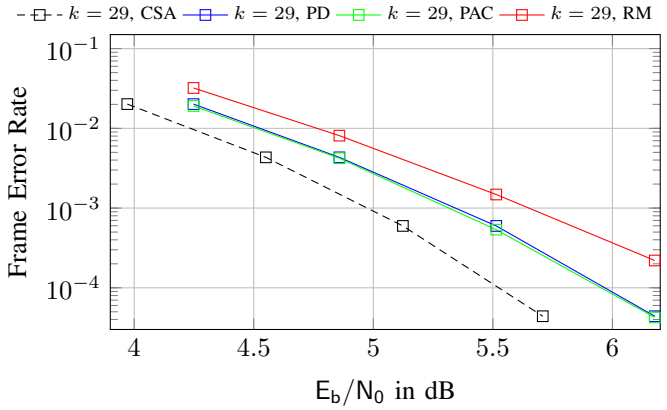

Figure 2: FER obtained with our scheme (PD) w.r.t. PAC codes and CSA for $N = 128$, $k = 29$.

where $\hat{v}_i = 0$ if $i \in \mathcal{F}$. Therefore, in terms of hardware complexity the PAC SCL decoder requires $O(L \cdot m)$ additional memory on top of $O(L \cdot N)$ of standard SCL decoding [4].

As for computational complexity, PAC SCL decoding requires extra $O(m \cdot L \cdot k)$ [10] addtitional floating point operations compared to standard SCL decoding, which is even equivalent in complexity of SCL decoder $O(L \cdot N \cdot \log N)$ [4], for short blocklengths if $k \cdot m \approx N \cdot \log N$ as it is the case of $k = 99$.

On the contrary, the complexity of the proposed method is almost equivalent to the one of SCL decoder as it requires few memory readings due to the single argument of dynamic frozen functions as stated in (33) and (34).

V. Conclusion

In this paper, we proposed a practical polar-like code construction method that performs similar, in terms of FER, to PAC codes, at short blocklengths but with a much smaller computational complexity. The method is based on the row merging that encodes some information bits with more than one rows of the polar encoding matrix. The proposed scheme has been constructed using the connection between the binary representation of row indices and the number of common 1-bit positions of the corresponding rows in the polar matrix. Our design decreases the number of low weight codewords compared to the RM selection rule. The proposed scheme does not require more computational resources than classical SCL decoding and hence our scheme is a good candidate to implement polar codes in applications with stringent complexity constraints and with short packet transmission. As further work, the extension to higher blocklengths is under investigation since it may require increasing the list size decoding for proper performance.

References

[1] E. Arikan, "Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels," *IEEE Trans. on Inf. Th.*, vol. 55, no. 7, pp. 3051–3073, July 2009.

[2] "Service requirements for the 5G system; Stage 1 (Release 17)," 3GPP TS 22.261, December 2019.

[3] ITU-T, "Representative use cases and key network requirements for network 2030", January 2020.

[4] I. Tal and A. Vardy, "List Decoding of Polar Codes," *IEEE Trans. on Inf. Th.*, vol. 61, no. 5, pp. 2213-2226, May 2015.

[5] Q. Zhang, A. Liu, X. Pan and K. Pan, "CRC Code Design for List Decoding of Polar Codes," *IEEE Commun. Lett.*, vol. 21, no. 6, pp. 1229–1232, June 2017.

[6] J. Piao, K. Niu, J. Dai and C. Dong, "Approaching the Normal Approximation of the Finite Blocklength Capacity Within 0.025 dB by Short Polar Codes," *IEEE Wireless Commun. Lett.*, vol. 9, no. 7, pp. 1089–1092, July 2020.

[7] P. Trifonov and V. Miloslavskaya, "Polar codes with dynamic frozen symbols and their decoding by directed search," *In Proc. of 2013 IEEE Information Theory Workshop (ITW)*, Sevilla, 2013, pp. 1–5.

[8] V. Miloslavskaya and B. Vucetic, "Design of Short Polar Codes for SCL Decoding," *IEEE Trans. on Commun.*, vol. 68, no. 11, pp. 6657–6668, Nov. 2020.

[9] E. Arikan, "From sequential decoding to channel polarization and back again", preprint available as 'arxiv:org/abs/1908.09594', September 2019.

[10] E. Arıkan, "Systematic Encoding and Shortening of PAC Codes," Entropy, vol. 22, no. 11, p. 1301, Nov. 2020.

[11] T. Thibaud and W. J. Gross, "On Systematic Polarization-Adjusted Convolutional (PAC) Codes," *IEEE Commun. Lett.*,vol. 25, no. 7, pp. 2128-2132, July 2021.

[12] Y. Polyanskiy, H. V. Poor and S. Verdu, "Channel Coding Rate in the Finite Blocklength Regime," *IEEE Trans. on Inf. Th.*, vol. 56, no. 5, pp. 2307–2359, May 2010.

[13] B. Li, H. Zhang and J. Gu, "On Pre-transformed Polar Codes," preprint available as 'arXiv:1912.06359', December 2019.

[14] V. Miloslavskaya, B. Vucetic, Y. Li, G. Park and O. -S. Park, "Recursive Design of Precoded Polar Codes for SCL Decoding," *IEEE Trans. on Commun.*, early access, 2021.

[15] X. Ma, J. Liu and B. Bai, "New Techniques for Upper-Bounding the ML Decoding Performance of Binary Linear Codes," *IEEE Trans. on Commun.*, vol. 61, no. 3, pp. 842–851, March 2013.

[16] B. Li, H. Shen and D. Tse, "An Adaptive Successive Cancellation List Decoder for Polar Codes with Cyclic Redundancy Check," *IEEE Commun. Lett.*, vol. 16, no. 12, pp. 2044–2047, December 2012.

[17] S.B. Korada, "Polar codes for channel and source coding", Doctoral Thesis, 2009, EPFL.

[18] D. Anade, J. M. Gorce, P. Mary and S. M. Perlaza, "An Upper Bound on the Error Induced by Saddlepoint Approximations-Applications to Information Theory," *Entropy*, vol. 22, no. 6: 690, December 2020.

[19] H. Yao, A. Fazeli and A. Vardy, "List Decoding of Arıkan's PAC Codes," *IEEE International Symposium on Information Theory (ISIT)*, 2020, pp. 443-448