On Sequential Decoding Metric Function of Polarization-Adjusted Convolutional (PAC) Codes

Mohsen Moradi, *Student Member, IEEE*

***Abstract*— In this paper, we present a sequential decoding metric function, which leads to significantly improved computational complexity while maintaining the superiority of polarization-adjusted convolutional (PAC) codes' error-correction performance. With the proposed metric function, the PAC codes' decoding computational complexity is comparable to the computational complexity of sequential decoding of conventional convolutional codes (CCs). Moreover, simulation results show an improvement in the error-correction performance of low rate PAC codes when using the proposed metric function. Simulation results also show that using the proposed metric, the upper bound on the PAC codes' computational complexity has a Pareto distribution. To reduce the worst-case latency of PAC sequential decoder, we limit the number of searches performed by sequential decoder. The results show that for PAC codes of length 128, search-limited sequential decoding can achieve an error-correction performance close to the error-correction performance of polar codes with successive cancellation list decoding with list size 64 and CRC length 11 with considerably less computational complexity.**

***Index Terms*— PAC codes, sequential decoding, metric function, threshold spacing, bias value.**

I. Introduction

SHORT block-length codes with low complexity and probability of error are of interest from a practical perspective. Over binary-input additive white Gaussian noise (BI-AWGN) channels, polarization-adjusted convolutional (PAC) codes with sequential decoding [1] are shown to have an error-correction performance close to the dispersion approximation [2] (best achievable performance of any given finite block-length code) for certain code rates [1], [3].

It is beneficial to consider a PAC code as a convolutional code (CC) with sequential decoding, which sees a polarized channel. In this way, the encoding process of PAC codes can be presented as an irregular binary tree code (tree codes only bifurcate for data bits), and sequential decoding of PAC codes can be considered as a walk through the tree. The decoder's task is to find the correct path in the tree guided by a metric function.

Sequential decoding was introduced by Wozencraft [4], and two well-known sequential decoding algorithms are the Fano [5] and stack [6], [7] algorithms. Due to its low memory requirements, Fano decoding is more favorable for practical hardware implementations [8]. For this reason, in this paper we use the Fano algorithm in our simulations.

The Fano metric is the most common metric function used in the sequential decoding of CCs, and the code rate is mostly used as a fixed bias term in the metric function [5]. In [9], Massey gives analytical justification of the optimality of the Fano metric. Also, Gallager [10] proves that by using a fixed bias value less than or equal to the cutoff rate R_0 of a binary-input discrete memoryless channel (B-DMC), the computational complexity of the Fano algorithm has a finite average.

Since each output bit of the CC in a PAC code sees a synthesized channel created by the polar transform, using a different metric function for sequential decoding of PAC codes is required. In [3], fixed bias values for different code rates were used in the metric function. In [11], because the CC in a PAC code is a one-to-one transform, a fixed bias value equal to the CC rate was used, and an heuristic metric function was introduced. Similar to the heuristic path metric function of [11], an heuristic path metric function for decoding polar codes was introduced in [12] with a different initial value. In [13], a metric function for stack decoding of polar codes was introduced that only updates the path metric function if the extended branch is an information bit; this update is obtained by adding the logarithm of the probability of the bit-channel output for a given bit-channel input. In [14], a path metric function for sequential decoding of polar codes was introduced, which maximizes the most probable continuation of a partially explored path. For this metric, the expected value of the logarithm of the probability of the partially decoded correct path (obtained based on the cumulative function of the evolving log-likelihood ratios (LLRs)) is subtracted from the metric function, which makes the expectation of the metric (expectation over the partially explored correct path) equal to zero.

In PAC codes, because of the channel polarization effect, the bit-channel cutoff rates $E_0(1, W_N^{(i)})$ are boosted close to the bit-channel capacities $I(W_N^{(i)})$. We review the Gaussian approximation for calculating $E_0(1, W_N^{(i)})$ and $I(W_N^{(i)})$ values in the next section. We then propose a suboptimal metric function, which, by using $E_0(1, W_N^{(i)})$ or $I(W_N^{(i)})$ as bias values, maintains the superior error-correction performance of PAC codes while requiring lower computational complexity compared to the fixed bias values used in [1], [3], [11].

Manuscript received December 30, 2020; revised May 5, 2021 and July 15, 2021; accepted August 28, 2021. Date of publication September 7, 2021; date of current version December 16, 2021. The associate editor coordinating the review of this article and approving it for publication was L. Chen.

The author is with the Department of Electrical Electronics Engineering, Bilkent University, 06800 Ankara, Turkey (e-mail: moradi@ee.bilkent.edu.tr).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2021.3111018.

Digital Object Identifier 10.1109/TCOMM.2021.3111018

Moreover, using our proposed metric function, the computational complexity of PAC sequential decoding is comparable to the computational complexity of sequential decoding of CCs. Compared to the previously used fixed bias values for low code rate PAC codes of [1], [3], using either of the proposed bias values improves the error-correction performance while maintaining low computational complexity.

Using metric bias values less than or equal to the bit-channel cutoff rates, we then empirically obtain an upper bound with Pareto distribution for the computational complexity of the PAC sequential decoder. Since the computational complexity has a Pareto distribution, the PAC sequential decoder requires a high computational complexity to decode a small fraction of the codewords. Search-limited PAC sequential decoding can be employed [3] to address this drawback. Moreover, results show that a search-limited PAC$(128, 64)$ code with an average number of decoded bit visits less than 10 has almost the same error correction performance as a polar code with successive cancellation list (SCL) decoding [15] with list size 64 and CRC length 11. Through simulations, we also obtain the best choice of Δ for PAC sequential decoder.

Simulation results show that using a fixed bias value in the PAC codes' sequential decoding metric function results in satisfactory error-correction performance. Nevertheless, the computational complexity is much higher than the computational complexity of sequential decoding of CCs [3], and we offer an explanation for why using a fixed bias value can result in exponential growth in computational complexity.

The rest of this paper is organized as follows. In Section II, the calculation of bit-channel mutual information and cutoff rate is explained, and the idea of PAC codes is briefly reviewed. In Section III, the optimality of the metric used in the PAC sequential decoder is proved. In Section IV, the behavior of the partial path (accumulated) metric function is analyzed using different bias values. In Section V, it is demonstrated that both frozen bits and information bits can be used in the decoding process. Moreover, it is shown that by using the bit-channel capacity or cutoff rate as the metric bias, PAC codes can have excellent error-correction performance while requiring low computational complexity. Section VI shows that the PAC codes' computation has a Pareto distribution upper bound. In Sections VII, techniques for finding the best threshold spacing values for sequential decoding are proposed. Finally, Section VIII concludes this paper with a brief summary.

II. Preliminaries

This section reviews the Gaussian approximation method of calculating the bit-channel mutual information and the bit-channel cutoff rate, which we frequently use in this paper.

A. Bit-Channel Mutual Information

Consider a BI-AWGN channel with binary phase-shift keying (BPSK) modulation. If the channel input is a uniform random variable $X \in \{-1, +1\}$ and the channel output is a random variable $Y \in \mathbb{R}$, the mutual information between the input and the output is defined as

$$\begin{aligned} I(W) &= I(X;Y) \\ &:= \sum_{x=\pm 1} \int_{-\infty}^{+\infty} \frac{1}{2} P_{Y|X}(y|x) \log_2 \frac{P_{Y|X}(y|x)}{P_Y(y)} dy \\ &= \frac{1}{\sqrt{8\pi\sigma^2}} \int_{-\infty}^{+\infty} a\left(1 - \log_2\left(\frac{b}{a}+1\right)\right) \\ &\quad + b\left(1 - \log_2\left(\frac{a}{b}+1\right)\right) dy, \end{aligned} \tag{1}$$

where

$$a := e^{\frac{-(y-1)^2}{2\sigma^2}} \quad \text{and} \quad b := e^{\frac{-(y+1)^2}{2\sigma^2}}, \tag{2}$$

and σ^2 is the noise variance of the BI-AWGN channel. Since $y = x + n$ has a Gaussian distribution, where n is Gaussian with zero mean and variance σ_n^2, according to [16] the mutual information is given by

$$I(X;Y) = J\left(\frac{2}{\sigma_n^2}\right), \tag{3}$$

where

$$J(t) = 1 - \frac{1}{\sqrt{2\pi t^2}} \int_{-\infty}^{+\infty} e^{\frac{-(u-\frac{t^2}{2})^2}{2t^2}} \log_2(1 + e^{-u}) du. \tag{4}$$

The monotonically increasing function $I(X;Y) = J(t)$ has a unique inverse function $t = J^{-1}(I(X;Y))$. An approximation for the function $J(t)$ and its inverse is given in [17] as

$$\begin{aligned} J(t) &= \left[1 - 2^{-0.3073 t^{2\times 0.8935}}\right]^{1.1064}, \quad J^{-1}(I(X;Y)) \\ &= \left[-\frac{1}{0.3073} \log_2\left(1 - I(X;Y)^{\frac{1}{1.1064}}\right)\right]^{\frac{1}{2\times 0.8935}}. \end{aligned} \tag{5}$$

In every step of the polarization process, independent copies of the channel W are transformed into polarized binary input channels W^+ and W^-. If we represent this operation by a tree with its root being initialized by (1), at each node the mutual information is polarized using the left-branch operation (parity operation) f_c and the right-branch operation (node operation) f_v, which are defined as

$$\begin{aligned} f_c(t) &= 1 - J\left[\sqrt{2} J^{-1}(1-t)\right], \\ f_v(t) &= J\left[\sqrt{2} J^{-1}(t)\right], \end{aligned} \tag{6}$$

and the leaf nodes provide approximations of the bit-channel mutual information $I(W_N^{(i)})$ for $i \in \{1, \ldots, N\}$. Fig. 1 plots the bit-channel mutual information for length $N = 128$ and rate $R = 1/2$ at a signal-to-noise ratio (SNR) of 2.5 dB. Information bit locations are chosen according to Reed-Muller rate profile as explained in [1].

Another important parameter of the channel W is the Bhattacharyya parameter which is defined as

$$Z(W) := \sum_{y \in \mathcal{Y}} \sqrt{W(y|0) W(y|1)}. \tag{7}$$

For a BI-AWGN channel W, $Z(W)$ is

$$Z(W) = e^{\frac{-1}{2\sigma^2}}, \tag{8}$$

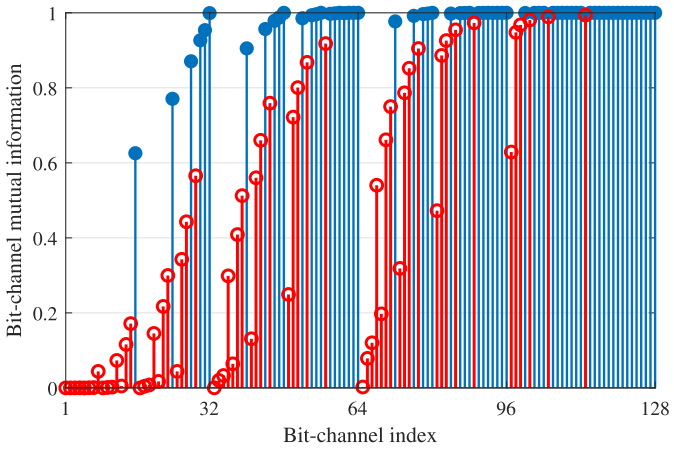

Fig. 1. Bit-channel capacity for information bits (solid circles) and frozen bits (hollow circles) at an SNR of 2.5 dB.

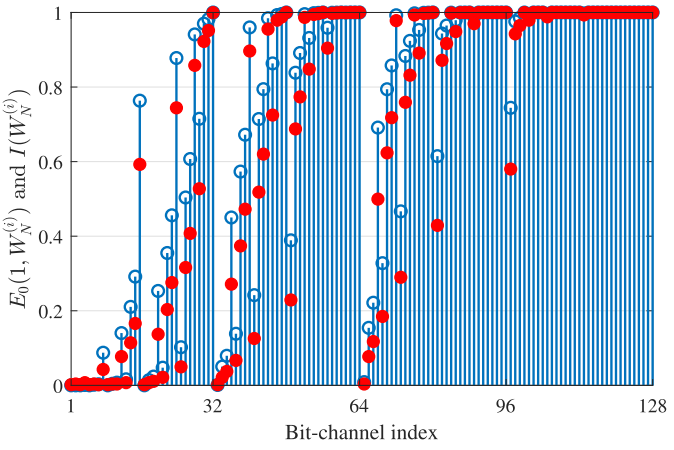

Fig. 2. (Solid circles) bit-channel cutoff rate and (hollow circles) bit-channel mutual information at an SNR of 2.5 dB.

where σ^2 is the noise variance. The log-likelihood ratio of the output of the channel W has a Gaussian distribution with mean $m_0^{(1)} = 2/\sigma^2$ and variance $2m_0^{(1)}$. With $m_0^{(1)}$ at the root of the tree, the bit-channel means $m_N^{(i)}$ at the leaf level of the tree can be approximated using the check-operation f_c and bit-operation f_v as [18]

$$\begin{aligned} f_c(t) &= \phi^{-1}\left(1-(1-\phi(t))^2\right), \\ f_v(t) &= 2\phi(t), \end{aligned} \tag{9}$$

where

$$\phi(t) = \begin{cases} 1 - \dfrac{1}{\sqrt{4\pi t}} \displaystyle\int_{\mathbb{R}} \tanh(\frac{z}{2}) e^{-\frac{(z-t)^2}{4t}} dz, & t > 0, \\ 1, & t < 0, \end{cases} \tag{10}$$

and $\phi^{-1}(t)$ can be calculated numerically. Finally, the bit-channel Bhattacharya parameters can be approximated as

$$Z(W_N^{(i)}) = e^{\frac{-1}{2(\sigma_N^{(i)})^2}}, \tag{11}$$

where $m_N^{(i)} = 2/(\sigma_N^{(i)})^2$.

For a given B-DMC W and any $\rho \geq 0$, the error exponent of the channel W with probability distribution $q(x)$ on the inputs is defined as [10]

$$E_0(\rho, W) = -\log_2 \sum_{y\in\mathcal{Y}} \left[\sum_{x\in\mathcal{X}} q(x) W(y|x)^{\frac{1}{1+\rho}} \right]^{1+\rho}. \tag{12}$$

By substituting $\rho = 1$ in (12), we can obtain the channel cutoff rate as

$$R_0(W, q) := E_0(1, W). \tag{13}$$

If the input distribution is uniform, the error exponent becomes

$$E_0(1, W) = \log_2 \frac{2}{1 + Z(W)}, \tag{14}$$

which is a lower bound on $I(W)$. According to (14), the polarization of $Z(W)$ results in the polarization of $E_0(1, W)$. We refer to $E_0(1, W_N^{(i)})$ as the bit-channel cutoff rate. Fig. 2 compares the bit-channel cutoff rate to the bit-channel mutual information.

B. Sequential Decoding

The stack and Fano algorithms are two commonly used sequential decoding algorithms for CCs. The stack decoding algorithm requires a larger memory than the Fano algorithm, and it maintains the sequence with the best metric on top of the stack. Because both the Fano and stack algorithms eventually traverse the same paths, the set of nodes visited by both algorithms is identical [19]. While the stack decoding algorithm can visit each node at most once, the Fano algorithm can visit a node multiple times. For low memory requirements, the Fano algorithm is more desirable for hardware implementation.

In this paper, we use the Fano decoding algorithm as explained in [10] to decode a PAC code. The Fano algorithm begins from the root of the code tree and moves forward to the child node with a higher branch metric. To continue, the Fano algorithm can make a forward move to a node if its path metric is greater than a running threshold T, which is an integer multiple of a constant threshold spacing parameter Δ. In a forward move, the threshold T is increased in steps of size Δ, with an upper limit of the path metric. If the path metric values of both children are less than T, the Fano decoder checks the preceding node path metric. If the preceding node has a path metric value greater than T, the decoder makes a backward move; otherwise, it lowers T by Δ and tries to make a forward move again.

The computational complexity of a sequential decoder is a random variable. This paper represents this random variable by counting the total number of nodes visited after a forward move during a single decoding session. For a sufficiently large number of simulation trials, we use the notion of the average number of visits (ANV), which corresponds to the empirical average of the number of nodes visited after a forward move. We also use the maximum number of visits (MNV) to denote the maximum number of nodes that the decoder is allowed to visit during a single decoding session.

Throughout the paper, we only consider PAC codes with a codeword of length 128. All codes are over the binary Galois field $\mathbb{F}_2 = \{0, 1\}$. We use boldface notation for vectors and for a vector $\mathbf{u} = (u_1, u_2, \ldots, u_N) \in \mathbb{F}_2^N$, $\mathbf{u}^i$ denotes the subvector $(u_1, u_2, \ldots, u_i)$ and $\mathbf{u}_i^j$ denotes the subvector $(u_i, \ldots, u_j)$ for

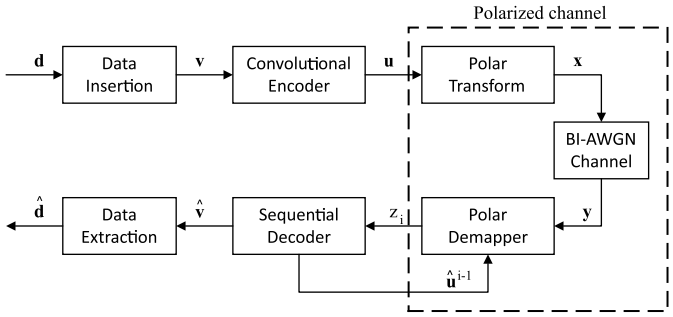

Fig. 3. Flowchart of the PAC coding scheme.

$i \leq j$. For any subset of indices $\mathcal{A} \subset \{1, 2, \ldots, N\}$, $\mathcal{A}^c$ denotes the complement of $\mathcal{A}$ and $\mathbf{u}_\mathcal{A}$ represents the subvector $(u_i : i \in \mathcal{A})$.

C. PAC Coding Scheme

Fig. 3 shows a block diagram of the PAC coding scheme. $\mathbf{d} = (d_1, \ldots, d_K)$ is the data generated uniformly at random over all possible source words of length K in a binary field. The rate profile module maps these K bits into a data carrier vector $\mathbf{v}$ in accordance with the data index set $\mathcal{A}$, thus inducing a code rate of $R = K/N$. The data index set has size $|\mathcal{A}| = K$ and is a subset of $\{1, 2, \cdots, N\}$. In this paper, the polar and RM score functions are employed to determine the set $\mathcal{A}$. The polar score function chooses the indices of the bit-channels with the highest reliability (the K indices with higher bit-channel cutoff rates), and the RM score function chooses an index i if the binary representation of $i - 1$ has a majority of 1s. After $\mathbf{v}$ is obtained by setting $\mathbf{v}_\mathcal{A} = \mathbf{d}$ and $\mathbf{v}_{\mathcal{A}^c} = \mathbf{0}$, it is sent to a rate 1 convolutional encoder and encoded as $\mathbf{u} = \mathbf{vT}$, where $\mathbf{T}$ is an upper-triangular Toeplitz matrix constructed with a connection polynomial $\mathbf{c}(x)$. In all of our simulations, we use the connection polynomial $\mathbf{c}(x) = x^{10} + x^9 + x^7 + x^3 + 1$ ($\mathbf{c} = 3211$ in octal form) [3]. Then $\mathbf{u}$ is transformed to $\mathbf{x}$ using the standard polar transformation $\mathbf{F}^{\otimes n}$, where $\mathbf{F}^{\otimes n}$ is the Kronecker power of the kernel matrix $\mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ with $n = \log_2 N$. After the polar transformation, $\mathbf{x}$ is sent through a BI-AWGN channel. The polar demapper receives the channel output $\mathbf{y}$ and the previously decoded bits and calculates the likelihood ratio value of the current bit z_i. Finally, the sequential decoder outputs an estimate $\hat{\mathbf{v}}$ of the data carrier vector, from which the K data bits can be extracted according to $\mathcal{A}$.

A PAC decoder consists of polar demapper and sequential decoder blocks. Assume that the sequential decoder is in a forward move to the ith node. Similar to successive cancellation (SC) decoding, the polar demapper receives channel output $\mathbf{y}$ and generates

$$z_i = \frac{P(\mathbf{y}, \hat{\mathbf{u}}^{i-1} | u_i = 0)}{P(\mathbf{y}, \hat{\mathbf{u}}^{i-1} | u_i = 1)}, \tag{15}$$

where, unlike the SC decoder, the $\hat{\mathbf{u}}^{i-1}$ vector is provided by the sequential decoder, and the polar demapper does not estimate u_i, but instead passes the soft value z_i to the sequential decoder. By using z_i in the metric function explained in Section III, sequential decoding generates $\hat{v}_i$. Also, by using an encoder replica, the sequential decoder obtains $\hat{u}_i$ from $\hat{v}_i$ and passes $\hat{u}^i$ back to the polar demapper. Then using $\hat{u}^i$, the polar demapper similarly obtains z_{i+1} and decoding continues in this way until $\hat{v}_N$ is determined or a predefined stopping rule finishes the decoding process.

As explained in Section II-B, if the path metrics of both children fall below the running threshold T and the path metric of the preceding node is above T, the Fano decoder makes a backward move. Now assume that the sequential decoder is at the ith node, makes a backward move to the $(i-1)$th node, and decoder passes $\hat{u}^{i-1}$ to the polar demapper. When calculating z_{i-1}, to prevent the polar demapper from starting the demapping process from scratch, storing all the intermediate LLR values is required. By doing so, to obtain z_{i-1}, the polar demapper can begin its operation from the common ancestor of $(i-1)$th and ith nodes. For this reason, the polar demapper that we use in this paper stores all the intermediate LLRs and uses a memory of $N \log_2 N$.

The channel that the sequential decoder of PAC codes sees is a polarized channel with memory. For this reason, the metric used by the sequential decoder must be compatible with the channel. In the following sections, we investigate the metric function. To implement the sequential decoder, we use the Fano algorithm, which is modified to be compatible with the PAC codes. Two important parameters of the Fano algorithm are the bias and the threshold spacing. In the rest of this paper, we investigate the effects of the bias and threshold spacing on the error-correction performance and computational complexity of PAC codes under search-limited and search-unlimited sequential decoding.

III. Metric

The most commonly used metric for sequential decoding in a B-DMC is the Fano metric [5], defined as

$$\gamma(x_i; y_i) = \log_2 \frac{P(y_i|x_i)}{P(y_i)} - b, \tag{16}$$

where $P(y_i|x_i)$ is the channel transition probability, $P(y_i)$ is the channel output probability, and b is a constant bias. Massey [9] showed that, for a B-DMC, the Fano metric is a locally optimum metric for comparing paths of different lengths. However, the polarized channel in PAC codes is a channel with memory since, for a polarized channel, the demapping of any bit requires the values of all the previously decoded bits. Hence the Fano metric must be modified to be suitable for decoding an irregular tree code sent over a channel with memory. By adopting the notation used in [1], [3] and recalling that the rate one convolution operation and polar transform are one-to-one transforms, the partial path metric for the first i branches is given by

$$\Gamma(\mathbf{u}^i; \mathbf{y}) = \log_2 \left(\frac{P(\mathbf{y}|\mathbf{u}^i)}{P(\mathbf{y})} \right) - B_i, \tag{17}$$

where $\mathbf{y}$ is the channel output, $\mathbf{u}^i$ is the path vector from the root of the tree to node i, $B_i = \sum_{j=1}^{i} b_j$ is the partial path bias value up to the ith bit, and b_j is a design parameter.

Every time a new branch in the tree is examined, it is more convenient to calculate the branch metric rather than

calculating the partial path metric of (17). For decoding u_j, the decoder knows the channel output $\mathbf{y}$ and the previous bits u_1 to u_{j-1}. By defining $\gamma(u_j;\mathbf{y},\mathbf{u}^{j-1})$ as the jth branch metric, (17) can be written as

$$\Gamma(\mathbf{u}^i;\mathbf{y})=\sum_{j=1}^{i}\gamma(u_j;\mathbf{y},\mathbf{u}^{j-1}), \tag{18}$$

and we have

$$\begin{aligned}\gamma(u_j;\mathbf{y},\mathbf{u}^{j-1}) &= \Gamma(\mathbf{u}^j;\mathbf{y})-\Gamma(\mathbf{u}^{j-1};\mathbf{y})\\ &= \left[\log_2\left(\frac{P(\mathbf{y}|\mathbf{u}^j)}{P(\mathbf{y})}\right)-B_j\right]\\ &\quad-\left[\log_2\left(\frac{P(\mathbf{y}|\mathbf{u}^{j-1})}{P(\mathbf{y})}\right)-B_{j-1}\right]\\ &= \log_2\left(\frac{P(\mathbf{y}|\mathbf{u}^j)}{P(\mathbf{y}|\mathbf{u}^{j-1})}\right)-(B_j-B_{j-1})\\ &= \log_2\left(\frac{P(\mathbf{y}|\mathbf{u}^j)}{P(\mathbf{y}|\mathbf{u}^{j-1})}\right)-b_j\\ &= \log_2\left(\frac{P(\mathbf{y},\mathbf{u}^{j-1}|u_j)}{P(\mathbf{y},\mathbf{u}^{j-1})}\right)-b_j. \end{aligned}\tag{19}$$

For a binary input channel with a uniform input distribution, u_j can be either 0 or 1. For $u_j=0$, (19) becomes

$$\begin{aligned}\gamma(u_j=0;\mathbf{y},\mathbf{u}^{j-1}) &= \log_2\left(\frac{P(\mathbf{y},\mathbf{u}^{j-1}|u_j=0)}{P(\mathbf{y},\mathbf{u}^{j-1})}\right)-b_j\\ &= \log_2\left(\frac{P(\mathbf{y},\mathbf{u}^{j-1}|u_j=0)}{\frac{1}{2}\left[P(\mathbf{y},\mathbf{u}^{j-1}|u_j=0)+P(\mathbf{y},\mathbf{u}^{j-1}|u_j=1)\right]}\right)\\ &\quad-b_j. \end{aligned}\tag{20}$$

Dividing the numerator and denominator of the argument by $P(\mathbf{y},\mathbf{u}^{j-1}|u_j=0)$ and defining

$$z_j:=\frac{P(\mathbf{y},\mathbf{u}^{j-1}|u_j=0)}{P(\mathbf{y},\mathbf{u}^{j-1}|u_j=1)} \tag{21}$$

as the likelihood ratio of bit u_j, we can rewrite (20) as

$$\gamma(u_j=0;\mathbf{y},\mathbf{u}^{j-1})=1-\log_2\left(1+\frac{1}{z_j}\right)-b_j. \tag{22}$$

Similarly, we can obtain the jth branch metric for $u_j=1$ as

$$\gamma(u_j=1;\mathbf{y},\mathbf{u}^{j-1})=1-\log_2\left(1+z_j\right)-b_j. \tag{23}$$

In conclusion, the jth branch metric for an irregular tree code transmitted over a polarized channel is calculated using

$$\gamma(u_j;\mathbf{y},\mathbf{u}^{j-1})=\begin{cases}1-\log_2\left(1+\dfrac{1}{z_j}\right)-b_j, & \text{if } u_j=0;\\ 1-\log_2(1+z_j)-b_j, & \text{if } u_j=1.\end{cases} \tag{24}$$

In sequential decoding, the metric function determines which path should be traversed in the decoding tree. Intuitively, to find the correct path in a tree (the path corresponding to $\mathbf{u}$), the branch metric function value should increment for every branch u_j on the correct path. For this reason, the branch metric value on a correct branch should be higher than the branch metric value on an incorrect branch $\tilde{u}_j$ at level j of the tree. Quantifying this intuition for a specific code is complicated. Alternatively, we consider an ensemble of PAC codes for the given channel and code length, which casts this problem in a more tractable setting. The expectation of (19) over the ensemble of input bit u_j and the bit-channel joint ensemble $P(\mathbf{y},\mathbf{u}^{j-1}|u_j)$ is

$$\begin{aligned}&\mathbb{E}_{u_j,(\mathbf{y},\mathbf{u}^{j-1})}\left[\gamma(u_j;\mathbf{y},\mathbf{u}^{j-1})\right]\\ &\quad=\sum_{u_j}q(u_j)\sum_{(\mathbf{y},\mathbf{u}^{j-1})}P(\mathbf{y},\mathbf{u}^{j-1}|u_j)\gamma(u_j;\mathbf{y},\mathbf{u}^{j-1})\\ &\quad=\sum_{u_j}\sum_{(\mathbf{y},\mathbf{u}^{j-1})}q(u_j)P(\mathbf{y},\mathbf{u}^{j-1}|u_j)\\ &\qquad\times\left[\log_2\left(\frac{P(\mathbf{y},\mathbf{u}^{j-1}|u_j)}{P(\mathbf{y},\mathbf{u}^{j-1})}\right)-b_j\right]=I(W_N^{(j)})-b_j, \end{aligned}\tag{25}$$

where $I(W_N^{(j)})$ is the symmetric capacity of the bit-channel. If the expectation in (25) is positive, the average branch metric increment on a correct path is always positive. Choosing the bit-channel bias less than the symmetric capacity of the bit-channels guarantees that the expectation in (25) is positive.

Now, assume that $\tilde{u}_j$ is an incorrect branch at level j with a metric of $\gamma(\tilde{u}_j;\mathbf{y},\mathbf{u}^{j-1})$. By averaging this quantity over the correct path and incorrect branch, we obtain

$$\begin{aligned}&\mathbb{E}_{u_j,\tilde{u}_j,(\mathbf{y},\mathbf{u}^{j-1})}\left[\gamma(\tilde{u}_j;\mathbf{y},\mathbf{u}^{j-1})\right]\\ &\quad=\sum_{\tilde{u}_j}q(\tilde{u}_j)\sum_{u_j}q(u_j)\sum_{(\mathbf{y},\mathbf{u}^{j-1})}P(\mathbf{y},\mathbf{u}^{j-1}|u_j)\gamma(\tilde{u}_j;\mathbf{y},\mathbf{u}^{j-1})\\ &\quad=\sum_{\tilde{u}_j}q(\tilde{u}_j)\sum_{(\mathbf{y},\mathbf{u}^{j-1})}P(\mathbf{y},\mathbf{u}^{j-1})\gamma(\tilde{u}_j;\mathbf{y},\mathbf{u}^{j-1})\\ &\quad=\sum_{\tilde{u}_j}\sum_{(\mathbf{y},\mathbf{u}^{j-1})}q(\tilde{u}_j)P(\mathbf{y},\mathbf{u}^{j-1})\\ &\qquad\times\left[\log_2\left(\frac{P(\mathbf{y},\mathbf{u}^{j-1}|\tilde{u}_j)}{P(\mathbf{y},\mathbf{u}^{j-1})}\right)-b_j\right]\\ &\quad\le\sum_{\tilde{u}_j}\sum_{(\mathbf{y},\mathbf{u}^{j-1})}q(\tilde{u}_j)P(\mathbf{y},\mathbf{u}^{j-1})\left[\frac{P(\mathbf{y},\mathbf{u}^{j-1}|\tilde{u}_j)}{P(\mathbf{y},\mathbf{u}^{j-1})}-1\right]-b_j\\ &\quad=-b_j. \end{aligned}\tag{26}$$

Note that in deriving (26), we used the property $\log_2(x)\le x-1$. Since the information bit bias value is positive, the above expectation shows that the branch metric decreases by a constant value for an incorrect branch on average.

A. Optimality of the Proposed Metric

In the ith step of decoding, given the channel output $\mathbf{y}$, the optimal metric (in the sense of error probability) is the one that chooses $\hat{\mathbf{u}}^i$ as the sequence $\mathbf{u}^i$ for which $p(\mathbf{u}^i|\mathbf{y})$ is maximized. Using Bayes' rule, this probability is given by

$$p(\mathbf{u}^i|\mathbf{y})=\frac{p(\mathbf{y}|\mathbf{u}^i)}{p(\mathbf{y})}p(\mathbf{u}^i). \tag{27}$$

Since any monotonically increasing function preserves maximality, an optimal metric also maximizes

$$\log_2\ p(\mathbf{u}^i|\mathbf{y})=\log_2\frac{p(\mathbf{y}|\mathbf{u}^i)}{p(\mathbf{y})}-\log_2\frac{1}{p(\mathbf{u}^i)}$$

$$= \log_2 \frac{p(\mathbf{y}|\mathbf{u}^i)}{p(\mathbf{y})} - \sum_{j=1}^{i} \log_2 \frac{1}{p(u_j)}, \quad (28)$$

where the last equality is obtained using the fact that convolution is a one-to-one and deterministic operation. Since $\log_2 \frac{1}{p(u_j)}$ is the self-information provided about the bit u_j, which equals $I(W_N^{(j)})$ on average, the path metric defined by

$$\Gamma(\mathbf{u}^i; \mathbf{y}) = \log_2 \frac{p(\mathbf{y}|\mathbf{u}^i)}{p(\mathbf{y})} - \sum_{j=1}^{i} I(W_N^{(j)}) \quad (29)$$

maximizes the error-correction probability, and the jth branch metric becomes

$$\begin{aligned} \gamma(u_j; \mathbf{y}, \mathbf{u}^{j-1}) &= \Gamma(\mathbf{u}^j; \mathbf{y}) - \Gamma(\mathbf{u}^{j-1}; \mathbf{y}) \\ &= \log_2 \frac{P(\mathbf{y}, \mathbf{u}^{j-1}|u_j)}{P(\mathbf{y}, \mathbf{u}^{j-1})} - I(W_N^{(j)}). \quad (30) \end{aligned}$$

For any bias value b_j and a given output $(\mathbf{y}, \mathbf{u}^{j-1})$ of a bit-channel $W_N^{(j)}$, we have $\gamma(u_j; \mathbf{y}, \mathbf{u}^{j-1}) \geq \gamma(\tilde{u}_j; \mathbf{y}, \mathbf{u}^{j-1})$ if and only if $P(\mathbf{y}, \mathbf{u}^{j-1}|u_j) \geq P(\mathbf{y}, \mathbf{u}^{j-1}|\tilde{u}_j)$. This happens if and only if $h_j(\mathbf{y}, \mathbf{u}^{j-1}) = u_j$, where

$$h_j(\mathbf{y}, \mathbf{u}^{j-1}) = \begin{cases} 0, & P(\mathbf{y}, \mathbf{u}^{j-1}|0) \geq P(\mathbf{y}, \mathbf{u}^{j-1}|1), \\ 1, & \text{otherwise}, \end{cases} \quad (31)$$

which implies that, for a bit channel $W_N^{(j)}$, the branch metric with any bias value follows the bit-channel maximum-likelihood (ML) rule. This proves that extending the path with a larger branch metric maximizes the probability that the extended path is part of the optimal path in the PAC code. Moreover, for a uniform input distribution, the branch metric is upper bounded as

$$\begin{aligned} \gamma(u_j; \mathbf{y}, \mathbf{u}^{j-1}) &= \log_2 \frac{P(\mathbf{y}, \mathbf{u}^{j-1}|u_j)}{P(\mathbf{y}, \mathbf{u}^{j-1})} - b_j \\ &= \log_2 \frac{P(\mathbf{y}, \mathbf{u}^{j-1}, u_j)}{P(u_j)P(\mathbf{y}, \mathbf{u}^{j-1})} - b_j \\ &= \log_2 \ P(u_j|\mathbf{y}, \mathbf{u}^{j-1}) \\ &\quad + \log_2 \frac{1}{P(u_j)} - b_j \leq 1 - b_j. \quad (32) \end{aligned}$$

From this, we can conclude that choosing the bias value less than 1 results in a locally optimum (ML) decision rule with a positive branch metric. Choosing the bias value less than or equal to $I(W_N^{(j)})$ also results in a positive branch metric on average. This paper aims to benefit from the channel polarization effect by designing a proper design rule for b_j.

IV. Design Rules for Bias

This section studies the effect of different possible bias values for use in sequential decoding of PAC codes. Various choices of bias values can result in different error-correction performance and computational complexity.

We use the notation $b_i^{(F)}$ and $b_i^{(I)}$ to represent the bias values for the frozen and information bits, respectively. We use b_i to indicate the bit-channel bias values whenever we do not specify whether a bit is frozen or information.

For a sequential decoder to have better error-correction performance and lower computational complexity, the codewords must have a good distance profile [20]. As discussed in the last section, by averaging over the correct and incorrect paths, we have:

$$\begin{aligned} &\textit{Correct path:} \quad b_i \leq I(W_N^{(i)}), \\ &\textit{Wrong path:} \quad -b_i \leq 0. \end{aligned} \quad (33)$$

In the following subsections, we discuss two different design rules for the bit-channel bias values. Both rules maintain the PAC codes' superior error-correction performance, but their computational complexity for rates above R_0 is different in the two cases.

A. Design Rule 1

Design rule 1 uses a metric with the bias values $b_i^{(F)} = 0$ and fixed $b_i^{(I)}$ s.t. $b_i^{(I)} > I(W_N^{(i)})$. More specifically, [3] uses $b_i^{(I)} = 1.4, 1.35$, and 1.14 for K $= 29, 64, 99$, respectively. Smaller $b_i^{(I)}$ values sacrifice the PAC code's error-correction performance. On the other hand, larger $b_i^{(I)}$ values increase the computational complexity. Note that for $i \in \mathcal{A}^c$, $b_i^{(F)} = 0 < I(W_N^{(i)})$ and hence both inequalities of (33) are satisfied. As a result, the decoder can decode the frozen bits faster, and the partial path metric is positive for the frozen bits. On the other hand, by assigning a large value to the information-bit bias values (say $b_i^{(I)} = 1.35$ for $K = 64$), since $b_i^{(I)} \nleq I(W_N^{(i)})$, the decoder will have some difficulty to find the correct path, and the partial path metric for the information bits becomes negative on average.

One drawback of choosing $b_i^{(I)} > I(W_N^{(i)})$ is that if the code has long stretches of consecutive information bits, the information bits' partial path metric will decrease. In [3], the reason for choosing smaller $b_i^{(I)}$ values for longer messages is to reduce the effect of negative partial path metrics; otherwise, decoding would require an exponentially larger computational complexity.

Fig. 4 shows the partial path metric averaged over 10^4 decoding trials for these three values of K along with their corresponding bias values. The figure demonstrates that the partial path metric increases for the frozen bits and decreases for the information bits on average by choosing a fixed $b_i^{(I)} > I(W_N^{(i)})$ and $b_i^{(F)} = 0 < I(W_N^{(i)})$, where the larger values reached for $K = 29$ reflect the fact that lower rates code contain more frozen bits.

The advantages of choosing $b_i^{(F)} = 0$ in this case is that both inequalities of (33) are satisfied for the frozen bits, and consequently the decoder can decode the frozen bits with less computational complexity. As long as $I(W_N^{(i)}) > b_i^{(F)}$, the partial path metric is positive on average.

B. Design Rule 2

Design rule 2, on the other hand, is motivated by the fact that sequential decoding is a greedy tree-search algorithm similar to the shortest path graph-search Dijkstra algorithm. The (locally optimum) Dijkstra algorithm always finds the shortest path on a graph when the edges have positive weights

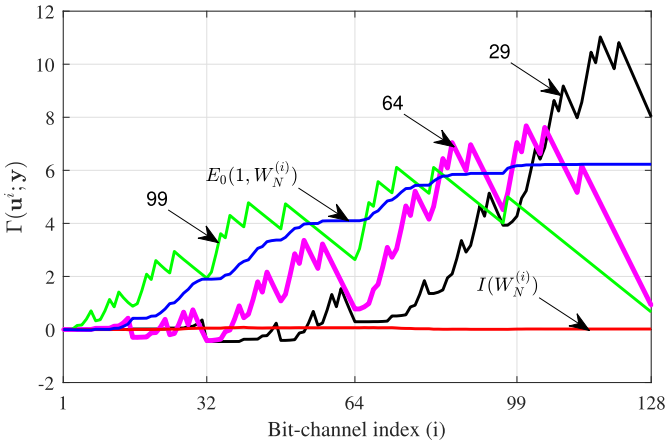

Fig. 4. Partial path metrics v. bit indices.

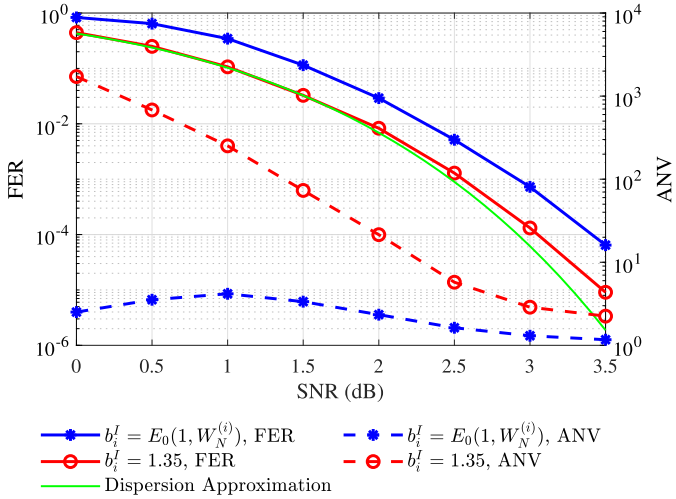

Fig. 5. FER and ANV performance comparison of PAC codes with different bias values.

and may or may not succeed in finding the shortest path when some edges have negative weights. Design rule 1 yields excellent error correction performance for the PAC codes tested but typically requires large computational complexity.

Design rule 2 uses bit-channel capacities or cutoff rates as the bit-channel bias values. By choosing $b_i^{(I)}$ and $b_i^{(F)}$ both equal to the bit-channel mutual information $I(W_N^{(i)})$, both inequalities of (33) are satisfied and the partial path metric of each branch becomes zero on average. For all the bits, since $-b_i = -I(W_N^{(i)}) < 0$, rule 2 prevents the decoder from following an incorrect path most of the time. Fig. 4 demonstrates that, using design rule 2 (for $K = 64$), the average partial path metric at each level is 0. Hence, by employing design rule 2, the decoder can decode every bit with a lower computational complexity than for design rule 1.

Alternatively, using $b_i = E_0(1, W_N^{(i)})$ for the bit-channel bias values satisfies both inequalities of (33) strictly. Hence, for these bias values, the partial path metrics should trend positive on average. Fig. 4 supports the latter statement (for $K = 64$). As a result, the computational complexity with bias $b_i = E_0(1, W_N^{(i)})$ should be lower than the one with bias $b_i = I(W_N^{(i)})$, with negligible loss in error performance. This computational complexity improvement is due to the polarization effect on both $E_0(1, W_N^{(i)})$ and $I(W_N^{(i)})$, where the difference in the two cases depends on the difference in polarization between the channel cutoff rate and the channel capacity shown in Fig. 2. The bit-channel capacities or cutoff rates used in sequential decoding of PAC codes must be pre-computed. A hardware implementation of sequential decoding of PAC codes can use a one-bit quantization of these bias values [8]. Note that, for the conventional CCs, choosing the bias value greater than the cutoff rate and close to the capacity results in the exponential growth of computational complexity.

V. Computational Complexity and Error-Correction Performance Results

Considering (33), only design rule 2 of the previous section satisfies both inequalities. To understand the effect of the information and frozen bits on error-correction performance and computational complexity of PAC codes, we first investigate the effect of the information bits on the error-correction performance and computational complexity of design rule 2, followed by an investigation of the effect of the frozen bits.

A. Effect of Information Bits

As the discussion of design rule 2 suggests, choosing the bit-channel bias values according to $E_0(1, W_N^{(i)})$ or $I(W_N^{(i)})$ can result in lower decoding computational complexity. In this subsection, we choose the bit-channel bias values of the information bits to be $E_0(1, W_N^{(i)})$ and we fix the bit-channel bias values of the frozen bits to zero. Fig. 5 compares this choice with using $b_i^I = 1.35$ and $b_i^F = 0$ bias values. As we can observe from the figure, there is a significant improvement in the computational complexity when using $b_i^I = E_0(1, W_N^{(i)})$ and $b_i^F = 0$ bias values. Note that the decrease in computational complexity comes at some cost in frame error rate (FER). The FER of the dispersion approximation is also provided in this figure.

Fig. 6 compares a PAC$(128, 59)$ code using $b_i^I = E_0(1, W_N^{(i)})$ and $b_i^F = 0$ bias values with a CC$(140, 64)$. As the figure shows, there is a significant improvement in the FER performance of the PAC code, and the empirical average of the number of visits (ANV) of PAC sequential decoding is almost the same as for CC sequential decoding, where the improvement of the PAC code's FER performance is due to the polarization effect on the channel. We also note that the computational complexity suffers from the cutoff rate phenomenon (the ANV value for both increases for SNR values less than 2 dB, which corresponds to rates above the channel cutoff rate). Note that to calculate the ANV for both codes, we count the number of forward visits per codeword and divide it by 128.

B. Effect of Frozen Bits

In sequential decoding of conventional CCs, because of the gap between the cutoff rate and the channel capacity, choosing the metric function bias value close to the channel

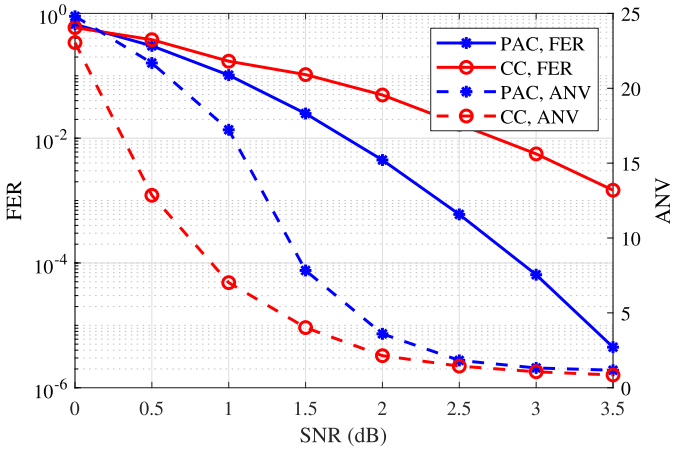

Fig. 6. FER and ANV performance comparison of PAC and CC codes with sequential decoding.

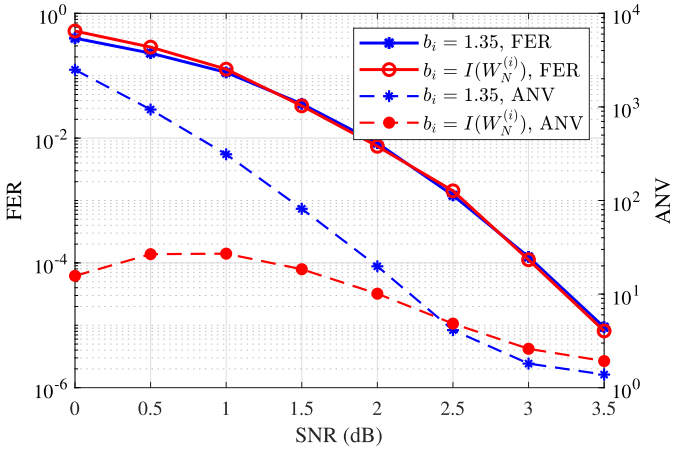

Fig. 7. Comparing the FER and ANV performance of PAC codes with $K = 64$.

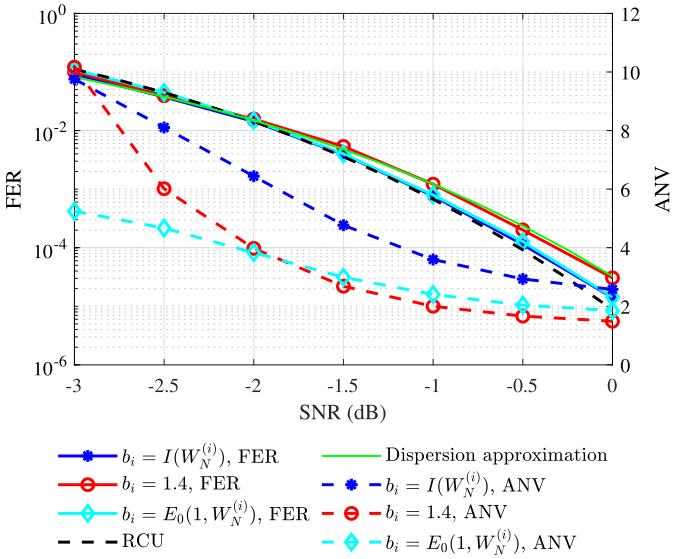

Fig. 8. FER and ANV performance comparison of PAC codes with K = 29.

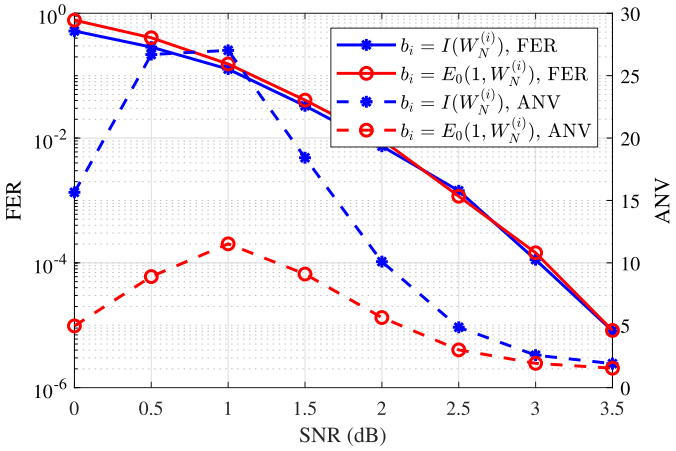

Fig. 9. FER and ANV performance of PAC codes with bias $I(W_N^{(i)})$ and $E_0(1, W_N^{(i)})$.

capacity results in exponential growth of the computational complexity. However, for PAC codes, because of the small gap between the bit-channel cutoff rates $E_0(1, W_N^{(i)})$ and the bit-channel capacities $I(W_N^{(i)})$, there is a negligible difference in the decoding computational complexity and error-correction performance in using either as bias values.

Fig. 7 plots the FER and ANV of a PAC$(128, 64)$ code using bias values $b_i = I(W_N^{(i)})$ versus using bias values $b_i^I = 1.35$ and $b_i^F = 0$. From this figure, we see that both choices result in the same FER performance, with bias $b_i = I(W_N^{(i)})$ having much smaller computational complexity.

Fig. 8 plots the FER and ANV of a PAC$(128, 29)$ code using bit-channel bias values $b_i = I(W_N^{(i)})$ and $b_i = E_0(1, W_N^{(i)})$ versus using fixed bias values $b_i^I = 1.4$ and $b_i^F = 0$. From this figure, we see that choosing $b_i = I(W_N^{(i)})$ or $b_i = E_0(1, W_N^{(i)})$ results in better FER performance and a comparable computational complexity to choosing $b_i^I = 1.4$ and $b_i^F = 0$. The random-coding union bound and dispersion approximation are also provided in this figure [2].

Fig. 9 compares the FER and ANV of a PAC$(128, 64)$ code using the bias values $b_i = I(W_N^{(i)})$ and $b_i = E_0(1, W_N^{(i)})$. This figure supports our previous claim that, since $E_0(1, W_N^{(i)})$ is slightly lower than $I(W_N^{(i)})$, the computational complexity when using $b_i = I(W_N^{(i)})$ is higher, and the (marginal) FER performance improvement occurs only at very low SNR values (very noisy channels). Also note that the computational complexity advantage of using $E_0(1, W_N^{(i)})$ rather than $I(W_N^{(i)})$ or fixed bias values is mainly for rates below the channel cutoff rate. For rates above the cutoff rate, as for conventional CCs, fixed bias values also tend to result in constant ANV values.

Fig. 10 compares the FER performance and computational complexity of Fano decoding of the PAC$(128, 64)$ code using the metric function with bias values $b_i = E_0(1, W_N^{(i)})$ to the search constrained and unconstrained Fano decoders of [11], which uses an heuristic metric function and stores $N - 1$ intermediate LLR values. For a fair comparison, the computational complexity is expressed in terms of the average number of time steps. Each time step corresponds to the time required to execute the check/bit node operation in one stage

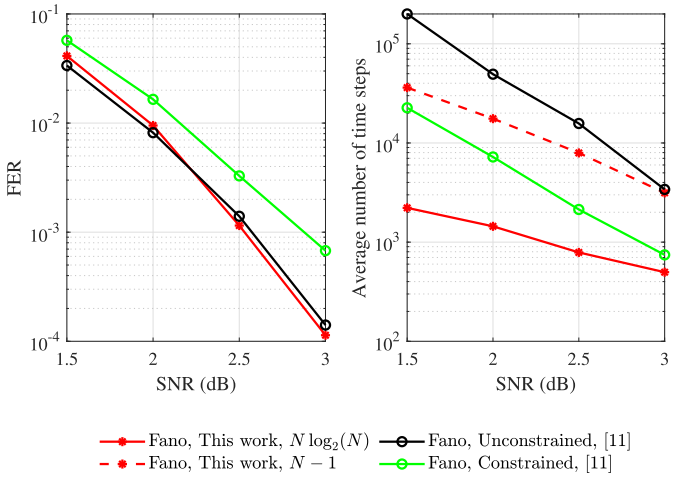

Fig. 10. Performance comparison of Fano decoder using the metric function with bias values $b_i = E_0(1, W_N^{(i)})$ and heuristic metric function of [11].

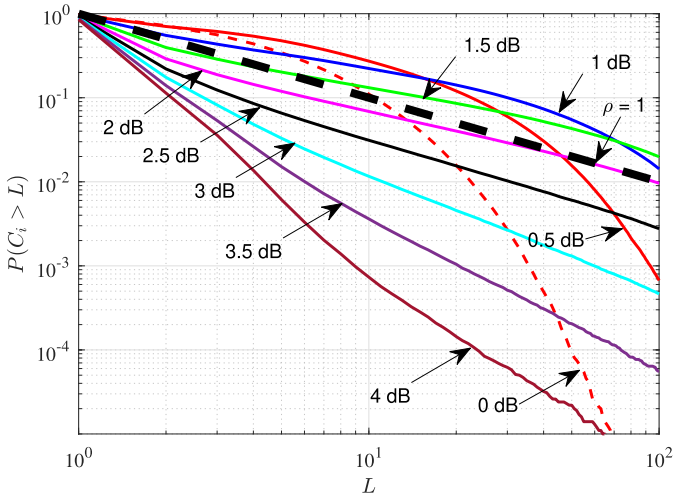

Fig. 11. CCDF of the number of node visits for PAC decoding with bias values $b_i = E_0(1, W_N^{(i)})$.

of the successive cancellation factor graph. To perform a fair comparison with [11], we also provide the FER performance and computational complexity of a Fano decoder that uses our metric function but the polar demapper of [11] which requires $N-1$ memory. In [11], the authors use tree search constraining methods to reduce the computational complexity of the Fano decoder at some cost in FER performance. As Fig. 10 shows, using the proposed metric function, the Fano decoder has an FER performance very close to the FER performance of the unconstrained decoder of [11]. By storing $N-1$ intermediate LLRs, the Fano decoder using the proposed metric function exhibits a computational complexity less than the computational complexity of the unconstrained decoder of [11]. There is a trade-off between the latency and memory usage of polar demapper, and by storing $N\log_2 N$ intermediate LLRs, Fig. 10 shows a significant amount of latency reduction.

VI. Distribution of Computational Complexity

In this section, we study the distribution of the number of forward visits during the decoding of PAC codes. For sequential decoding of CCs, it was shown in [21] that, for rates below the cutoff rate, the distribution of computation required to advance any level in the tree is upper bounded as

$$P(C_i > L) < AL^{-\beta}, \tag{34}$$

where $A > 0$ and $\beta > 0$ are constants and C_i is the average number of forward visits per decoded bit. This implies that the complementary cumulative distribution function (CCDF) of the computational complexity of sequential decoding of CCs has a Pareto distributed upper bound. Smaller values of β corresponds to larger tails of the Pareto distribution, and it has a finite mean for $\beta > 1$ and a finite variance for $\beta > 2$.

For PAC$(128, 64)$ decoding with bias values $b_i = E_0(1, W_N^{(i)})$, the CCDF of the number of nodes visited for all decoded codewords is plotted in Fig. 11 for 10^6 decoding trials at different SNR values. The results demonstrate that: 1) $P(C_i > L) < L^{-1}$ for SNR values greater than 2 dB, and consequently the mean of the upper bound distribution is finite; 2) For SNR values 3.5 dB and 4 dB, we have $\beta > 2$, which gives a finite variance. Fig. 11 also illustrates the trade-off between outage probability (the probability that C_i exceeds some limit L) and computational complexity of sequential decoding. For example, there is a 1% probability that a decoded codeword requires more than ten visits per branch at an SNR of 3.0 dB, and the probability that the number of visits for a decoded codeword exceeds L goes to zero as L increases.

Moreover, for SNR values less than 2 dB, Fig. 11 shows that the outage probability drops rapidly as L increases. Whereas, for SNR values higher than 2 dB, the drop in outage probability is smoother. In contrast, for fixed bias values, PAC codes at low SNR values experience extremely high computational complexity.

VII. Threshold Spacing

For sequential decoding of CCs, the threshold spacing value is normally chosen between 2 and 8 [20, p. 625]. In this section, we study the impact of the threshold spacing parameter Δ on the error-correction performance and computational complexity of PAC codes. Different values of Δ result in trade-off between FER and ANV values for sequential decoding, so deciding on the most suitable value of Δ is an important design consideration.

Fig. 12 shows the effect of Δ on the FER performance and ANV value of a PAC$(128, 64)$ code at an SNR of 2.5 dB, where the bit-channel bias values are set to $E_0(1, W_N^{(i)})$. As we see, the FER performance degrades as Δ increases; with a Δ value between 0.5 and 2 resulting in the best FER performance. We also see that, the ANV value decreases with increasing Δ, and we can conclude that choosing $\Delta = 2$ provides a good trade-off between the FER performance and computational complexity. Notice that in sequential decoding of conventional CCs, choosing $\Delta = 2$ also minimizes the computational complexity upper bound [21, p. 475].

Finally, based on the sequential decoding parameters selected above, in Fig. 13 we compare the FER performance of the search-limited PAC$(128, 64)$ code with MNV $= 2^{14}$,

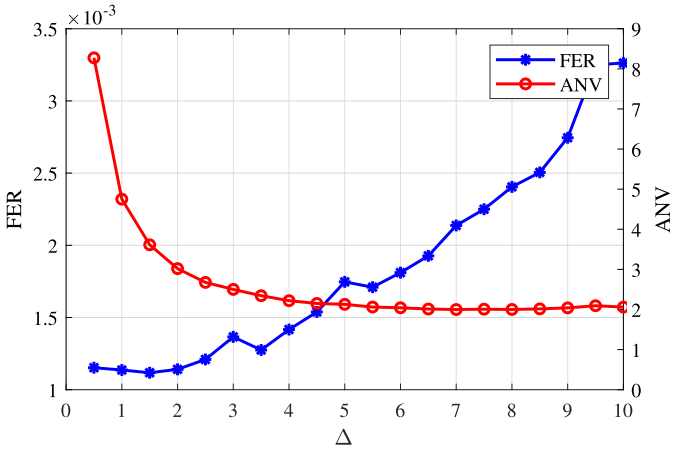

Fig. 12. FER and ANV v. threshold spacing Δ parameter for a search-unlimited PAC$(128, 64)$ code.

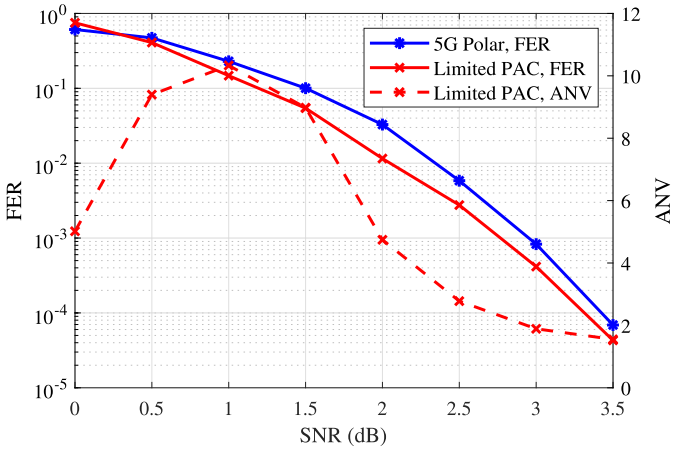

Fig. 13. Search-limited PAC code with MNV $= 2^{14}$ v. polar code with SCL decoding with list size 64 and CRC length of 11.

$b_i = E_0(1, W_N^{(i)})$, and $\Delta = 2$ to successive cancellation list (SCL) decoding of the 5G polar code [15] with list size 64 and CRC length 11. From this figure, we see that the FER performance of the PAC codes is comparable to that of polar codes, and the PAC code ANV values are much less than the fixed polar code list size of 64 for all SNR values.

VIII. Conclusion

In this paper, we proposed a metric function that uses the bit-channel mutual information and cutoff rate values as the bias and results in a favorable trade-off between error-correction performance and computational complexity for sequential decoding of PAC codes. Moreover, we investigated the selection of proper value of the threshold spacing parameter that improves the error-correction and computational complexity trade-off for sequential decoding of PAC codes. We also showed that, by using bias values less than the bit-channel cutoff rates, sequential decoding of PAC codes exhibits a Pareto distribution upper bound on its computational complexity. From this, the probability of having a computational complexity greater than L goes to zero with increasing L. Using the bit-channel capacities and cutoff rates as bias values, simulation results demonstrated that the PAC codes' superior error-correction performance is maintained while their computational complexity is reduced.

Acknowledgment

The author is grateful to all anonymous reviewers for their constructive comments.

References

[1] E. Arikan, "From sequential decoding to channel polarization and back again," 2019, *arXiv:1908.09594*. [Online]. Available: http://arxiv.org/abs/1908.09594
[2] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.
[3] M. Moradi, A. Mozammel, K. Qin, and E. Arikan, "Performance and complexity of sequential decoding of PAC codes," 2020, *arXiv:2012.04990*. [Online]. Available: http://arxiv.org/abs/2012.04990
[4] J. M. Wozencraft, "Sequential decoding for reliable communication," Res. Lab. Elect., MIT, Cambridge, MA, USA, Tech. Rep. 325, 1957.
[5] R. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inf. Theory*, vol. IT-9, no. 4, pp. 64–74, Apr. 1963.
[6] K. Zigangirov, "Some sequential decoding procedures," *Problem Peredachi Inf.*, vol. 2, no. 4, pp. 13–25, 1966.
[7] F. Jelinek, "Fast sequential decoding algorithm using a stack," *IBM J. Res. Develop.*, vol. 13, no. 6, pp. 675–685, Nov. 1969.
[8] A. Mozammel, "Hardware implementation of Fano decoder for polarization-adjusted convolutional (PAC) codes," 2020, *arXiv:2011.09819*. [Online]. Available: http://arxiv.org/abs/2011.09819
[9] J. Massey, "Variable-length codes and the Fano metric," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 1, pp. 196–198, Jan. 1972.
[10] R. G. Gallager, *Information Theory and Reliable Communication*. New York, NY, USA: Wiley, 1968.
[11] M. Rowshan, A. Burg, and E. Viterbo, "Polarization-adjusted convolutional (PAC) codes: Sequential decoding vs list decoding," *IEEE Trans. Veh. Technol.*, vol. 70, no. 2, pp. 1434–1447, Feb. 2021.
[12] M.-O. Jeong and S.-N. Hong, "SC-Fano decoding of polar codes," *IEEE Access*, vol. 7, pp. 81682–81690, 2019.
[13] K. Niu and K. Chen, "Stack decoding of polar codes," *Electron. Lett.*, vol. 48, no. 12, pp. 695–697, Jun. 2012.
[14] P. Trifonov, "A score function for sequential decoding of polar codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 1470–1474.
[15] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.
[16] F. Brannstrom, L. K. Rasmussen, and A. J. Grant, "Convergence analysis and optimal scheduling for multiple concatenated codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 9, pp. 3354–3364, Sep. 2005.
[17] F. Brannstrom, "Convergence analysis and design of multiple concatenated codes," Ph.D. dissertation, Dept. Comput. Eng., Chalmers Univ., Gothenburg, Sweden, 2004.
[18] H. Li and J. Yuan, "A practical construction method for polar codes in AWGN channels," in *Proc. Tencon-Spring*, Apr. 2013, pp. 223–226.
[19] J. M. Geist, "Algorithmic aspects of sequential decoding," Dept. Elect. Eng., Univ. Notre Dame, Notre Dame, IN, USA, Tech. Rep. EE-702, Aug. 1970.
[20] S. Lin and D. J. Costello, *Error Control Coding*, vol. 2, no. 4. Upper Saddle River, NJ, USA: Prentice-Hall, 2001.
[21] J. Wozencraft and I. Jacobs, *Principles of Communication Engineering*. New York, NY, USA: Wiley, 1965.

Mohsen Moradi (Student Member, IEEE) was born in Ghorveh, Iran, in 1989. He received the B.S. degree in pure mathematics from the University of Isfahan, Isfahan, Iran, in 2011, and the M.S. degree in pure mathematics from Tehran Polytechnic, Tehran, Iran, in 2013. He is currently pursuing the Ph.D. degree with the Department of Electrical and Electronics Engineering, Bilkent University, Ankara, Turkey. His main research interests include information theory, error correction coding, and analysis of algorithms.