Successive Cancellation Ordered Search Decoding of Modified G_N-Coset Codes

Peihong Yuan, *Member, IEEE*, and Mustafa Cemil Coşkun, *Member, IEEE*

***Abstract*— A tree search algorithm called successive cancellation ordered search (SCOS) is proposed for G_N-coset codes that implements maximum-likelihood (ML) decoding with adaptive complexity for transmission over binary-input AWGN channels. Unlike bit-flip decoders, no outer code is needed to terminate decoding; therefore, SCOS also applies to G_N-coset codes modified with dynamic frozen bits. The average complexity is close to that of successive cancellation (SC) decoding at practical frame error rates (FERs) for codes with wide ranges of rate and lengths up to 512 bits, which perform within 0.25 dB or less from the random coding union bound and outperform Reed–Muller codes under ML decoding by up to 0.5 dB. Simulations illustrate simultaneous gains for SCOS over SC-Fano, SC stack (SCS) and SC list (SCL) decoding in FER and the average complexity at various SNR regimes. SCOS is further extended by forcing it to look for candidates satisfying a threshold, thereby outperforming basic SCOS under complexity constraints. The modified SCOS enables strong error-detection capability without the need for an outer code. In particular, the $(128, 64)$ polarization-adjusted convolutional code under modified SCOS provides gains in overall and undetected FER compared to CRC-aided polar codes under SCL/dynamic SC flip decoding at high SNR.**

***Index Terms*— Complexity-adaptive maximum-likelihood decoding, error detection, polar codes, Reed–Muller codes, dynamic frozen bits.**

I. Introduction

THE G_N-coset codes are a class of block codes [3] that include polar codes [3], [4] and Reed-Muller (RM) codes [5], [6]. Polar codes achieve capacity over binary-input discrete memoryless channels (B-DMCs) under low-complexity successive cancellation (SC) decoding [3] and RM codes achieve capacity over binary erasure channels (BECs) under maximum-likelihood (ML) decoding [7].[1] However, their performance under SC decoding [4] is not competitive for short- to moderate-lengths, e.g., from 64 to 512 bits [9].[2] Significant research effort hast been put into approaching ML performance by improved decoding algorithms with an SC decoding schedule [12], [13], [14], [15], [16], [17], [18], improving the distance properties [19], [20], [21], [22], [23], [24], [25], [26] or both [27], [28].

The idea of *dynamic* frozen bits lets one represent any linear block code as a *modified* G_N-coset code [21]. This concept unifies the concatenated polar code approach, e.g., with a high-rate outer cyclic redundancy check (CRC) code, to improve the distance spectrum of polar codes so that they can be decoded with low to moderate complexity [27].

This paper proposes *successive cancellation ordered search (SCOS)* as an ML decoder for modified G_N-coset codes. The decoding complexity adapts to the channel noise and an extension of SCOS limits the worse-case complexity while still permitting near-ML decoding for various code lengths $N \in \{64, 128, 256, 512\}$ and wide ranges of rate from low to high. The decoder can be used for G_N-coset codes, CRC-concatenated G_N-coset codes as well as those with dynamic frozen bits. Numerical results show that RM and RM-polar codes with dynamic frozen bits of block length $N \in \{64, 128, 256, 512\}$, i.e., dynamic RM (dRM) [11] and the proposed dRM-polar codes, perform within 0.25 dB of the random-coding union (RCU) bound [29] with an average complexity close to that of SC decoding at low enough frame error rates (FERs). Remarkably, dRM codes under SCOS outperform ML performance of RM codes up to 0.5 dB. To illustrate the benefits of the proposed algorithm, simulations with successive cancellation Fano (SC-Fano), successive cancellation stack (SCS) and successive cancellation list (SCL) decoding algorithms are also provided for codes of length 128 as examples. Compared to all, SCOS provides lower average decoding complexity with better FER performance at various operating regimes. SCOS is further extended by limiting attention to codeword candidates satisfying an optimized threshold test, which improves the performance when a maximum complexity constraint is imposed. In addition, the threshold test lets the

Manuscript received 11 June 2023; revised 2 November 2023 and 28 December 2023; accepted 4 February 2024. Date of publication 9 February 2024; date of current version 18 June 2024. This work was partially supported by the German Research Foundation (DFG) under Grant KR 3517/9-1. An earlier version of this paper was presented in part at the IEEE Information Theory Workshop (ITW), October 2021, Kanazawa, Japan [DOI: 10.1109/ITW48936.2021.9611445]. The associate editor coordinating the review of this article and approving it for publication was I. Tal. *(Corresponding author: Mustafa Cemil Coşkun.)*

Peihong Yuan was with the Institute for Communications Engineering (LNT), Technical University of Munich (TUM), 80333 Munich, Germany. He is now with the Research Laboratory of Electronics (RLE), Massachusetts Institute of Technology (MIT), Cambridge, MA 02139 USA (e-mail: phyuan@mit.edu).

Mustafa Cemil Coşkun was with the Institute for Communications Engineering (LNT), Technical University of Munich (TUM), 80333 Munich, Germany. He is now with the Radio Systems Research Laboratory, Nokia Bell Labs, Murray Hill, NJ 07974 USA (e-mail: mustafa.coskun@nokia-bell-labs.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2024.3364989.

Digital Object Identifier 10.1109/TCOMM.2024.3364989

[1] RM codes achieve capacity over B-DMCs under bit-wise ML decoding [8], i.e., the average bit error probability vanishes asymptotically in the block length.

[2] Long RM codes are not well-suited for SC decoding [10], [11]: the error probability of long RM codes under SC decoding is lower-bounded by $1/2$ [3, Section X].

decoder avoid making a decision [30] which provides simultaneous gains in overall FER and undetected frame error rate (uFER) for the $(128, 64)$ polarization-adjusted convolutional (PAC) code as compared to a CRC-concatenated polar code under SCL and dynamic successive cancellation flip (DSCF) decoding algorithms, where the CRC is optimized for the lower tail of the distance spectrum [31].

SCOS borrows ideas from SC-based flip [17], [18], sequential [14], [15], [16], [32], [33] and list decoders [27], [34], [35], [36]. It is a tree search algorithm that flips the bits of valid paths to find a leaf with higher likelihood than other leaves, if such a leaf exists, and repeats until the ML decision is found. The search stores the branches in the list that is updated progressively while running partial SC decoding by flipping the bits of the most-likely leaf at each iteration. The order of the candidates is chosen according to the probability that they provide the ML decision. SCOS does not require an outer code (as for flip-decoders) or parameter optimization for the performance vs. complexity trade-off (as for sequential decoders).

This paper is organized as follows. Section II gives background on the problem. Section III presents the SCOS algorithm with the pseudo codes. The complexity of SCOS is discussed together with the numerical results in Section IV. Then, Section V proposes modifications and provides numerical results in comparison to the original algorithm. Section VI compares SCOS to the other existing complexity-adaptive decoders and Section VII concludes the paper.

II. Preliminaries

Let x^a be the vector $(x_1, x_2, \ldots, x_a)$; if $a = 0$, then the vector is empty. Given x^N and a set $\mathcal{A} \subset [N] \triangleq \{1, \ldots, N\}$, let $x_{\mathcal{A}}$ be the subvector $(x_i : i \in \mathcal{A})$. For set $\mathcal{A}$, we define an intersection set as $\mathcal{A}^{(i)} \triangleq \mathcal{A} \cap [i]$, $i \in [N]$. Uppercase letters refer to random variables (RVs) and lowercase letters to realizations. A B-DMC is denoted as $W : \mathcal{X} \to \mathcal{Y}$, with input alphabet $\mathcal{X} = \{0, 1\}$, output alphabet $\mathcal{Y}$, and transition probabilities $W(y|x)$ for $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ [37, Sec. 4]. The transition probabilities of N independent uses of the same channel are denoted as $W^N(y^N|x^N)$ and can be factored as $W^N(y^N|x^N) = \prod_{i=1}^{N} W(y_i|x_i)$. Capital bold letters refer to matrices, e.g., $\boldsymbol{B}_N$ denotes the $N \times N$ *bit reversal* matrix [3] and $\boldsymbol{G}_2$ denotes the 2×2 Hadamard matrix.

A. $\boldsymbol{G}_N$-Coset Codes

Consider the matrix $\boldsymbol{G}_N = \boldsymbol{B}_N \boldsymbol{G}_2^{\otimes n}$, where $N = 2^n$ with a non-negative integer n and $\boldsymbol{G}_2^{\otimes n}$ is the n-fold Kronecker product of $\boldsymbol{G}_2$. For the set $\mathcal{A} \subseteq [N]$ with $|\mathcal{A}| = K$, let $U_{\mathcal{A}}$ have entries that are independent and identically distributed (i.i.d.) uniform *information* bits, and let $U_{\mathcal{A}^c} = u_{\mathcal{A}^c}$ be fixed or *frozen*, where $\mathcal{A}^c \triangleq [N] \setminus \mathcal{A}$. The mapping $c^N = u^N \boldsymbol{G}_N$ defines a $\boldsymbol{G}_N$-coset code [3]. Polar and RM codes are $\boldsymbol{G}_N$-coset codes with different selections of $\mathcal{A}$ [3], [4].

Using $\boldsymbol{G}_N$, the transition probability from u^N to y^N is $W_N(y^N|u^N) \triangleq W^N(y^N|u^N \boldsymbol{G}_N)$. The transition probabilities of the i-th *bit-channel*, a synthesized channel with the input u_i and the output (y^N, u^{i-1}), are defined by

$$W_N^{(i)}(y^N, u^{i-1}|u_i) \triangleq \sum_{u_{i+1}^N \in \mathcal{X}^{N-i}} \frac{1}{2^{N-1}} W_N(y^N|u^N). \quad (1)$$

An (N, K) polar code is designed by placing the K most reliable bit-channels with indices $i \in [N]$ into the set $\mathcal{A}$ that can, e.g., be found using density evolution [3], [38]. An r-th order RM code of length-N and dimension $K = \sum_{i=0}^{r} \binom{n}{i}$, where $0 \le r \le n$, is denoted as RM(r, n). Its set $\mathcal{A}$ consists of the indices, $i \in [N]$, with Hamming weight at least $n - r$ for the binary expansion of $i - 1$. For both codes, one sets $u_i = 0$ for $i \in \mathcal{A}^c$.

We make use of *dynamic* frozen bits [21]. A frozen bit is dynamic if its value depends on a subset of information bits preceding it; the resulting codes are called modified $\boldsymbol{G}_N$-coset codes. Dynamic frozen bits tend to improve the performance of near-ML decoders [11], [23], [28], [31], [39] because the weight spectrum of the resulting code tends to improve as compared to the underlying code [23], [24], [25], [31], [40], [41]. For the numerical results, we will consider short- to moderate-length RM codes with dynamic frozen bits, called dRM codes [11, Def. 1]. An important instance from the ensemble is the PAC codes with RM rate-profiling [28]. However, the average complexity gets large for (near-)ML decoding of dRM codes as they get longer. Therefore, the formal definition of the modified RM-polar codes is given below.

Definition 1: The (N, K) dRM-polar ensemble is the set of codes specified by the set $\mathcal{A}$ of an (N, K) RM-polar code and choosing

$$u_i = \bigoplus_{j \in \mathcal{A}^{(i-1)}} v_{j,i} u_j, \quad \forall i \in \mathcal{A}^c \quad (2)$$

with all possible $v_{j,i} \in \{0, 1\}$ and $\mathcal{A}^{(0)} \triangleq \varnothing$, where $\bigoplus$ denotes XOR summation and $u_i \triangleq 0$ if $\mathcal{A}^{(i-1)} = \varnothing$ for any $i \in \mathcal{A}^c$.

B. Related Decoding Algorithms

1) Successive Cancellation Decoding: Let c^N and y^N be the transmitted and received words, respectively. SC decoding makes the decision for the i-th bit-channel sequentially from $i = 1$ to $i = N$ as follows. For $i \in \mathcal{A}^c$, set $\hat{u}_i$ to its (dynamic) frozen value. For $i \in \mathcal{A}$, compute the soft message $\ell_i\left(\hat{u}_1^{i-1}\right)$ defined as

$$\ell_i\left(\hat{u}_1^{i-1}\right) \triangleq \log \frac{P_{U_i|Y^N U^{i-1}}(0|y^N, \hat{u}^{i-1})}{P_{U_i|Y^N U^{i-1}}(1|y^N, \hat{u}^{i-1})} \quad (3)$$

assuming that the previous decisions $\hat{u}_1^{i-1}$ are correct and the frozen bits after u_i are uniformly distributed. Then, it makes a hard decision as

$$\hat{u}_i = \begin{cases} 0 & \text{if } \ell_i\left(\hat{u}_1^{i-1}\right) \ge 0 \\ 1 & \text{otherwise.} \end{cases} \quad (4)$$

Any erroneous decision $\hat{u}_i \neq u_i$, $i \in \mathcal{A}$, cannot be corrected by SC decoding and results in a frame error. In the following, we review techniques to overcome this problem.

2) Successive Cancellation List Decoding: SCL decoding tracks several SC decoding paths [27] in parallel. At each decoding phase $i \in \mathcal{A}$, instead of making a hard decision on u_i, two possible decoding paths are continued in parallel threads. The maximum number 2^K of paths implements ML decoding but with exponential complexity in K. To limit complexity, one may keep up to L paths at each phase. The reliability of decoding path v^i is quantified by a *path metric (PM)* defined as [42]

$$M\left(v^i\right) \triangleq -\log P_{U^i|Y^N}\left(v^i|y^N\right) \tag{5}$$

$$= M\left(v^{i-1}\right) + \log\left(1+e^{-(1-2v_i)\ell_i\left(v_1^{i-1}\right)}\right) \tag{6}$$

$$\approx \begin{cases} M\left(v^{i-1}\right), & \text{if sign}\left(\ell_i\left(v^{i-1}\right)\right) = 1-2v_i \\ M\left(v^{i-1}\right) + \left|\ell_i\left(v^{i-1}\right)\right|, & \text{otherwise} \end{cases} \tag{7}$$

where (7) can be computed recursively using SC decoding with $M\left(v^0\right) \triangleq 0$. At the end of N-th decoding phase, a list $\mathcal{L}$ of paths is collected. Finally, the output is the bit vector minimizing the PM:

$$\hat{u}^N = \underset{v^N \in \mathcal{L}}{\operatorname{argmin}} M\left(v^N\right). \tag{8}$$

3) Flip Decoding: successive cancellation flip (SCF) decoding [17] aims to correct the first erroneous bit decision by sequentially flipping the unreliable decisions. This procedure requires an error-detecting outer code, e.g., a CRC code.

The SCF decoder starts by performing SC decoding for the inner code to generate the first estimate v^N. If v^N passes the CRC test, it is declared as the output $\hat{u}^N = v^N$. If not, then the SCF algorithm attempts to correct the bit errors at most $T_{\max}$ times. At the t-th attempt, $t \in [T_{\max}]$, the decoder finds the index i_t of the t-th least reliable decision in v^N according to the amplitudes of the soft messages (3). The SCF algorithm restarts the SC decoder by flipping the estimate v_{i_t} to $v_{i_t} \oplus 1$. The CRC is checked after each attempt. This decoding process continues until the CRC passes or $T_{\max}$ is reached.

Introducing a bias term to account for the reliability of the previous decisions enhances the performance [18]. The improved metric is calculated as

$$Q(i) = \left|\ell_i\left(v^{i-1}\right)\right| + \sum_{j \in \mathcal{A}^{(i)}} \frac{1}{\alpha} \log\left(1+e^{-\alpha\left|\ell_j\left(v^{j-1}\right)\right|}\right) \tag{9}$$

where $\alpha > 0$ is a scaling factor.

SCF decoding can be generalized to flip multiple bit estimates at once, leading to DSCF decoding [18]. The reliability of the initial estimates $\tilde{u}_{\mathcal{E}}$, $\mathcal{E} \subseteq \mathcal{A}$, is described by

$$Q(\mathcal{E}) = \sum_{i \in \mathcal{E}} \left|\ell_i\left(v^{i-1}\right)\right| + \sum_{j \in \mathcal{A}^{(i_{\max})}} \frac{1}{\alpha} \log\left(1+e^{-\alpha\left|\ell_j\left(v^{j-1}\right)\right|}\right) \tag{10}$$

where $i_{\max}$ is the largest element in $\mathcal{E}$. The set of flipping positions is chosen as the one minimizing the metric (10) and is constructed progressively.

4) Sequential Decoding: We review two sequential decoding algorithms, namely SCS decoding [13], [14], [15] and SC-Fano decoding [16], [28].

SCS decoding stores the D most reliable paths (possibly) with different lengths and discards the rest whenever the stack is full. At each iteration, the decoder selects the most reliable path and creates two possible decoding paths based on this path. The winning word is declared once a path length becomes N. To limit its worst-case complexity similar to that of SCL decoding with list size L, the decoding is limited to have at most L visits each node in the decoding tree, which is finished in at most LN node-visits. SC-Fano decoding deploys a Fano search [32] that allows backward movement in the decoding tree and that uses a dynamic threshold. The dynamic threshold is initialized as $T = 0$. During the Fano search, if one cannot find a path with score less than T then the dynamic threshold is updated to $T + \Delta$, where Δ is called the threshold spacing and controls the performance vs. complexity tradeoff.

Sequential decoding compares paths of different lengths. The probabilities $P_{U^i|Y^N}\left(v^i|y^N\right)$, $v^i \in \{0,1\}^i$, however, cannot capture the effect of the path's length. A new score is introduced in [43] and used in [16] to account for the expected error rate of the future bits as

$$S\left(v^i\right) \triangleq -\log \frac{P_{U^i|Y^N}\left(v^i|y^N\right)}{\prod_{j=1}^{i}\left(1-p_j\right)} \tag{11}$$

$$= M\left(v^i\right) + \sum_{j=1}^{i} \log\left(1-p_j\right) \tag{12}$$

where p_j is the probability of the event that the first bit error occurred for u_j in SC decoding. The probabilities p_i can be computed via Monte Carlo simulations [3], [4] or they can be approximated via density evolution [38] offline. In the following, one may generalize the score as

$$S\left(v^i\right) = M\left(v^i\right) + b_i \tag{13}$$

where b_i is called a bias term. We discuss in Section V-A how the bias term affects the proposed decoding algorithm.

III. SC Ordered Search Decoding

SCOS uses the metrics (7) and (11). We first define

$$\overline{M}\left(v^i\right) \triangleq M\left(v^{i-1}\overline{v_i}\right) = -\log P_{U^i|Y^N}\left(v^{i-1}\overline{v_i}\,\middle|\,y^N\right) \tag{14}$$

$$\overline{S}\left(v^i\right) \triangleq \overline{M}\left(v^i\right) + b_i \tag{15}$$

where $\bar{v}_i \triangleq v_i \oplus 1$, $v^{i-1}v_i \triangleq v^i$ and one may choose b_i as the second term in the right hand side (RHS) of (12). Figure 1 shows an example of the decoding for $N = 4$ and $K = 4$. SCOS starts by SC decoding to provide an output v^N as the current most-likely leaf, e.g., the black path (0111) in Figure 1. This initial SC decoding computes and also stores the PM $\overline{M}\left(v^i\right)$ and the score $\overline{S}\left(v^i\right)$ associated with the flipped versions of the decisions v_i for all $i \in \mathcal{A}$, e.g., illustrated as the red paths in Figure 1. Every index $i \in \mathcal{A}$ with $\overline{M}\left(v^i\right) < M(v^N)$ constitutes a flipping set, i.e., each set is a singleton at this stage. All flipping sets are stored in a min heap $\mathcal{L}$ [44] and each member is visited in ascending order according to its score.

Let $\mathcal{E}$ be the flipping set with the smallest score in the heap $\mathcal{L}$ and let index $j \in [N]$ be the deepest common node of the current most-likely leaf and the branch node defined by $\mathcal{E}$ in the decoding tree (see the brown dot in Figure 1(d)). The

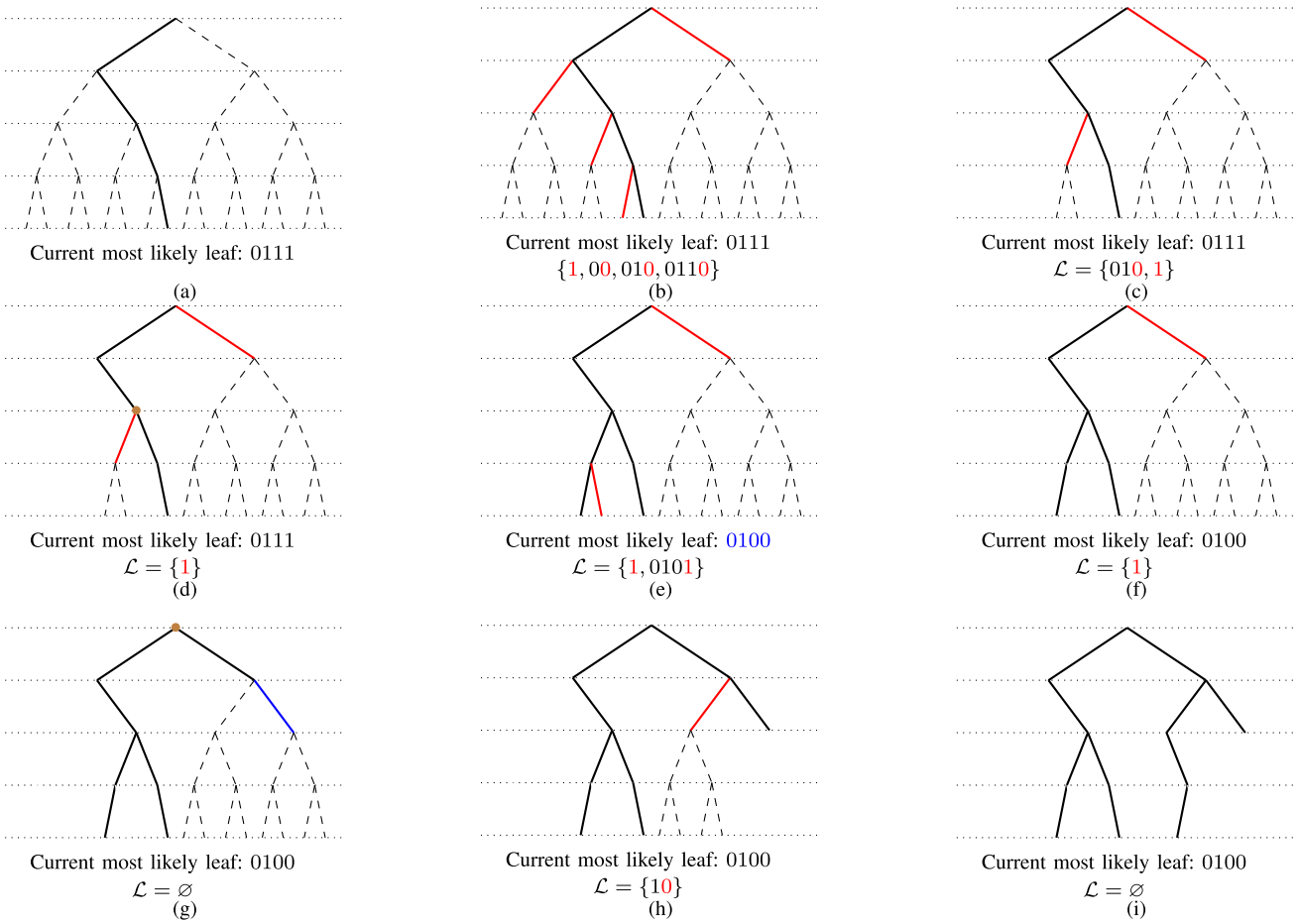


Fig. 1. (a) Initial SC decoding outputs $v^N = (0111)$ with the corresponding PM $M(v^N)$. (b) During the initial SC decoding, the PMs and scores are computed for branch nodes $\{1, 00, 010, 0110\}$. (c) The branch nodes with PMs larger than that of the current most likely leaf are pruned, e.g., we have $M(00), M(0110) > M(0111)$. Suppose also that $S(010) < S(1)$. Then, $\mathcal{L}$ stores all branch nodes with PMs smaller than that of current most likely leaf, where $\mathcal{L}$ is a min heap according to the scores of its members. (d) The candidate with smallest score is popped from the heap and the decoder returns to the deepest (or nearest) common node. (e) The decision is flipped and SC decoding continues. During decoding, the heap $\mathcal{L}$ and the current most likely leaf are updated. (f) The branch nodes with PMs larger than that of the current most likely leaf are pruned as in (c) (in this case, a leaf node is removed). (g) Repeat the procedure as in step (d), where we assume $M(11) > M(0100)$. (h) The heap $\mathcal{L}$ is updated when the branch (11) was visited. (i) The decoder examines the last member of the heap $\mathcal{L}$ and pops it from $\mathcal{L}$. After reaching the N-th decoding phase, suppose that there is no branch node left, which has a smaller PM than that of the current most likely leaf, i.e., $\mathcal{L} = \varnothing$. The current most likely leaf is declared as the decision $\hat{u}^N$.

decoder now flips the decision v_j and SC decoding continues. The set $\mathcal{E}$ is popped from the heap $\mathcal{L}$. The PMs (7) and scores (11) are calculated again for the flipped versions for decoding phases with $i > j$, $i \in \mathcal{A}$, and the heap $\mathcal{L}$ is enhanced by new flipping sets progressively (similar to [18]). The branch node, including all of its child nodes, is discarded if at any decoding phase its PM exceeds that of the current most-likely leaf, i.e., $M(v^N)$.[3] Such a branch cannot output the ML decision, since for any valid path v^i the PM (7) is non-decreasing for the next stage, i.e., we have

$$M\left(v^i\right) \leq M\left(v^{i+1}\right), \forall v_{i+1} \in \{0,1\}. \tag{16}$$

For instance, suppose that $M(11) > M(0111)$ in Figure 1(g). Then any path $\tilde{v}^N$ with $\tilde{v}^2 = (1,1)$ cannot be the ML decision; hence, it is pruned. If a leaf with lower PM is found, then it replaces the current most-likely leaf. The procedure is repeated until one cannot find a more reliable path by flipping decisions, i.e., until $\mathcal{L} = \varnothing$. Hence, SCOS decoding implements an ML decoder.

[3]This pruning method is similar to the adaptive skipping rule proposed in [36] for ordered-statistics decoding [34], [35].

A. Detailed Description

This section provides the details of the proposed SCOS decoding with the pseudo codes, where a simulation code is provided in [45]. In the following, we use type-writer font for the data structures (with an exception for sets) and 1-based indexing arrays. The required data structures together with their size are listed in Table I. As we explain the algorithms, we will revisit the relevant data structure from the table. We start with arrays `L` and `C`, which contain LLRs and hard decisions, respectively. Recall that there are $\log_2 N+1$ layers in a polar code graph and both `L` and `C` store N elements in each layer (in contrast to [27] where in total only $2N-1$ elements are stored) since we reuse some decoding paths to decrease the computational complexity (similar to SC-Fano decoding). The entry in position (i, j) of array `L` (`C`) is denoted as `L`$[i, j]$ (`C`$[i, j]$), which is calculated via Algorithm 5 (6). These routines, namely recursivelyCalcL and

TABLE I
DATA STRUCTURES FOR SCOS DECODING

name	size	data type	description
$\mathtt{L}$	$(\log_2 N+1)\times N$	float	LLR
$\mathtt{C}$	$(\log_2 N+1)\times N$	binary	hard decision
$\mathtt{F}$	1	$\langle$set, float, float$\rangle$	structure of a flipping set
$\mathcal{E}$, $\mathcal{E}_\mathrm{p}$	$\leq K$	integer (set of indices)	flipping set
$\mathcal{L}$	$\leq \eta$	type of $\mathtt{F}$	heap of flipping structures
$\hat{\mathtt{u}}$, $\mathtt{v}$	N	binary	decoding path
$\mathtt{b}$	N	float	precomputed bias term
$\mathtt{M}$, $\overline{\mathtt{M}}$, $\overline{\mathtt{S}}$	N	float	metric
$\mathtt{M}_{\mathtt{cml}}$	1	float	PM of the current most likely leaf

Algorithm 1 FindStartIndex $(\mathcal{E}, \mathcal{E}_\mathrm{p})$

Input : flipping sets $\mathcal{E}$ and $\mathcal{E}_\mathrm{p}$
Output: first different index

```
for i = 1, 2, ..., N do
    if (i ∈ E) ⊕ (i ∈ E_p) then
        return i
```

recursivelyCalcC, are the LLR-based versions [42] of [27, Alg. 3] and [27, Alg. 4], respectively, and provided as Algorithm 5 and Algorithm 6 in the appendix for completeness. We also name indices λ and ϕ as layer and phase, respectively, by adopting the convention of [27], which are integer-valued inputs to Algorithms 5 and 6. Unlike [27], the layer and phase satisfy $1 \leq \lambda \leq \log_2 N + 1$ and $1 \leq \phi < 2^\lambda$ due to 1-based indexing.

Flipping set structures, denoted by $\mathtt{F}$, are triplets containing a set of integer indices (flipping set $\mathcal{E}$), a PM and a score. The heap $\mathcal{L}$ contains multiple flipping structures $\mathtt{F} = \langle \mathcal{E}, \overline{\mathtt{M}}_\mathcal{E}, \overline{\mathtt{S}}_\mathcal{E} \rangle$,[4] where $\overline{\mathtt{M}}_\mathcal{E}$ and $\overline{\mathtt{S}}_\mathcal{E}$ are the respective PM and the score associated to the flipping set $\mathcal{E}$, as defined in (14) and (15). The size of $\mathcal{L}$ is constrained by a user-defined parameter η, which will define the space complexity of the decoder. Given two flipping sets, namely $\mathcal{E}$ and $\mathcal{E}_p$, Algorithm 1 is the procedure used to find the decoding stage to which the decoder should return, i.e., the deepest common node as illustrated in Figure 1(d).

Algorithm 2 is generalized SC decoding, which can start SC decoding at any decoding phase and continue decoding until a termination criterion is satisfied. Then, it returns the phase at which the decoding is terminated. The modifications compared to the original SC decoding are highlighted as blue in the pseudo code. Before their detailed descriptions, we recall data structures needed from Table I. The notation $\mathtt{v}[i]$ refers to the i-th entry of an array $\mathtt{v}$, where binary vectors $\mathtt{v}$ and $\hat{\mathtt{u}}$ are the currently processed path and the current most-likely one, respectively. Unless otherwise stated, the entries of vector $\mathtt{b}$ are computed offline via

$$\mathtt{b}[i] = \sum_{j=1}^{i} \log\left(1 - p_j\right), \ i \in [N]. \tag{17}$$

The entries of length-N vectors $\mathtt{M}$ and $\overline{\mathtt{M}}$ correspond to PMs along traversed paths and the flipped versions, respectively.

[4]Observe that the heap $\mathcal{L}$ in Figure 1 is slightly different for simplicity.

Vector $\overline{\mathtt{S}}$ contains the scores used for the search schedule of the proposed decoder. The modified SC decoding takes as input an integer $i_\text{start} \in [N]$ and a flipping set $\mathcal{E}$ and outputs another index i_end such that $i_\text{start} < i_\text{end} \leq N$. Along the way, the algorithm updates the vectors containing PMs and scores, namely $\mathtt{M}$ and $\overline{\mathtt{M}}$ and $\overline{\mathtt{S}}$, where the details are itemized as follows.

- The standard subroutine HardDec (`lines 8` and `10`) takes a real-valued LLR as the input and returns a decision according to (4). In addition, CalcPM (`lines 12` and `14`) takes a real-valued PM, a binary decision and a real-valued LLR as inputs and updates the PM using (7).
- One can start at any decoding phase i_start with no additional computational cost (`line 2`).
- The i-th entry of vector $\mathtt{M}$ is updated in each decoding phase i (`line 14`).
- The decisions are flipped at the decoding phases corresponding to the current flipping set, i.e., if $i \in \mathcal{E}$ (`lines 7-8`).
- The PMs and the scores of the potential flipping sets are computed for decoding phases after the largest one in the current flipping sets, i.e., $\overline{\mathtt{M}}[i]$ and $\overline{\mathtt{S}}[i]$ with $i \in \mathcal{A}$ and $i >$ maximum $(\mathcal{E})$ (`lines 11-13`).
- If a more likely leaf (i.e., a path of length-N with smaller PM) is found, $\hat{\mathtt{u}}$ and $\mathtt{M}_\mathtt{cml}$ are updated and the decoding phase N is returned as i_end (`lines 20-24`).
- For any i, if PM $\mathtt{M}[i]$ is larger than $\mathtt{M}_\mathtt{cml}$, stop SCDec function and return the current phase i as i_end (`line 15-16`).
- If i-th bit is dynamically frozen, then the computation of $\mathtt{v}[i]$ follows the constraints, i.e., using RHS of (2) where the coefficients $v_{j,i}$ are specified by the construction (`line 5`).

Algorithm 3 is the main loop of SCOS decoding. Naturally, the heap of flipping structures and the previous flipping set are initialized as null and the PM of the current most-likely leaf as $+\infty$. After the initial SC decoding (`line 4`), $\mathtt{M}_\mathtt{cml}$ is updated to the PM of the SC estimate. Then, a tree search is performed in order to find the most-likely estimate (`lines 8-16`), where the candidates are ordered by their scores. Many sub-trees are pruned thanks to the threshold $\mathtt{M}_\mathtt{cml}$ (`lines 6-7 and 14-15`), i.e., the PM of the current most likely leaf. The stopping condition of the "while loop" (`line 8` with $\mathcal{L} = \varnothing$) implies that the most likely codeword is found, i.e., there cannot be any other codeword with a smaller

Algorithm 2 SCDec $(i_{\text{start}}, \mathcal{E})$

Input : start index i_{start}, flipping set $\mathcal{E}$
Output: end index i_{end}

1 $m = \log_2 N$
2 **for** $i = i_{start}, \ldots, N$ **do**
3 recursivelyCalcL $(m+1, i-1)$
4 **if** $i \notin \mathcal{A}$ **then**
5 $\mathtt{v}[i] = 0$ // compute $\mathtt{v}[i]$ if dynamic frozen
6 **else**
7 **if** $i \in \mathcal{E}$ **then**
8 $\mathtt{v}[i] = \text{HardDec}(\mathtt{L}[m+1, i]) \oplus 1$
9 **else**
10 $\mathtt{v}[i] = \text{HardDec}(\mathtt{L}[m+1, i])$
11 **if** $i > \mathit{maximum}(\mathcal{E})$ **then**
12 $\overline{\mathtt{M}}[i] =$ CalcPM $(\mathtt{M}[i-1], \mathtt{v}[i] \oplus 1, \mathtt{L}[m+1, i])$
13 $\overline{\mathtt{S}}[i] = \overline{\mathtt{M}}[i] + \mathtt{b}[i]$
14 $\mathtt{M}[i] = \text{CalcPM}(\mathtt{M}[i-1], \mathtt{v}[i], \mathtt{L}[m+1, i])$
15 **if** $\mathit{M}[i] \geq \mathit{M}_{\mathit{cm1}}$ **then**
16 **return** i
17 $\mathtt{C}[m+1, i] = \mathtt{v}[i]$
18 **if** $i \mod 2 = 0$ **then**
19 recursivelyCalcC $(m+1, i-1)$
20 **if** $\mathit{M}[N] < \mathit{M}_{\mathit{cm1}}$ **then**
21 $\mathtt{M}_{\mathtt{cm1}} = \mathtt{M}[N]$
22 **for** $i = 1, 2, \ldots, N$ **do**
23 $\hat{\mathtt{u}}[i] = \mathtt{v}[i]$
24 **return** N

Algorithm 3 SCOS (ℓ^N)

Input : LLRs ℓ^N
Output: $\hat{\mathtt{u}}$

1 $\mathcal{L} = \varnothing, \mathcal{E}_{\mathrm{p}} = \varnothing, \mathtt{M}_{\mathtt{cm1}} = +\infty$
2 **for** $i = 1, 2, \ldots, N$ **do**
3 $\mathtt{L}[1, i] = \ell_i$
4 SCDec $(1, \varnothing)$
5 **for** $i = 1, 2, \ldots, N$ **do**
6 **if** $i \in \mathcal{A}$ *and* $\overline{\mathtt{M}}[i] < \mathit{M}_{\mathit{cm1}}$ **then**
7 InsertHeap $(\langle \{i\}, \overline{\mathtt{M}}[i], \overline{\mathtt{S}}[i] \rangle)$
8 **while** $\mathcal{L} \neq \varnothing$ **do**
9 $\langle \mathcal{E}, \overline{\mathtt{M}}_{\mathcal{E}}, \overline{\mathtt{S}}_{\mathcal{E}} \rangle = \text{popMin}(\mathcal{L})$
10 **if** $\overline{\mathtt{M}}_{\mathcal{E}} < \mathit{M}_{\mathit{cm1}}$ **then**
11 $i_{\text{start}} = \text{FindStartIndex}(\mathcal{E}, \mathcal{E}_{\mathrm{p}})$
12 $i_{\text{end}} = \text{SCDec}(i_{\text{start}}, \mathcal{E})$
13 **for** $i = \mathit{maximum}(\mathcal{E}) + 1, \ldots, i_{\mathit{end}}$ **do**
14 **if** $i \in \mathcal{A}$ *and* $\overline{\mathtt{M}}[i] < \mathit{M}_{\mathit{cm1}}$ **then**
15 InsertHeap $(\langle \mathcal{E} \cup \{i\}, \overline{\mathtt{M}}[i], \overline{\mathtt{S}}[i] \rangle)$
16 $\mathcal{E}_{\mathrm{p}} = \mathcal{E}$
17 **return** $\hat{u}$

PM. The estimate with PM $\mathtt{M}_{\mathtt{cm1}}$ is output as the decision (line 17).

Remark 1: Observe that each member in the heap $\mathcal{L}$, where $|\mathcal{L}| \leq \eta$, stores a set of integers with maximum size of K and two float metrics. Hence, $\mathcal{L}$ stores at most $K\eta$ integers and 2η floats. In addition, recall that the arrays $\mathtt{L}$ and $\mathtt{C}$ store $N \log_2 N + N$ elements each in contrast to $\eta \times (2N-1)$, which is the case, e.g., in SCS decoding with stack size $D = \eta$ [14], [15]. Other data structures listed in Table I are of size at most N each. In total, SCOS stores at most $N \log_2 N + 5N + 2\eta + 1$ floats, $NR\eta$ integers, and $N \log_2 N + 3N$ bits, where R is the code rate.

IV. Complexity and Performance Considerations

We adopt *number of node-visits* in the decoding tree as a proxy for the complexity. A node visit occurs each time line 3 is executed in Algorithm 2, i.e., each time a node is visited in decoding tree, which is, for instance, provided for the case of $N = 4$ in Figure 1. Note that this does not refer to the exact complexity; however, it still provides a very good proxy [24, Sec. 4.2], which is used very often in prior works, see, e.g., [16], [28], [46], and [47], among many other references.[5] To this end, let $\lambda(y^N)$ be the number of node-visits in the decoding tree by SCOS decoding normalized by block length for channel output y^N. Similar to other sequential decoders [48], it is a RV defined as $\Lambda \triangleq \lambda(Y^N)$. On the contrary, the number $\lambda_{\text{SC}}(y^N)$ of normalized node-visits for SC decoding is simply 1 independent of the channel output y^N.

A. Average Number of Node-Visits for ML Decoding

In the following, we are interested in the average behaviour of Λ when there is no limit in the number of node-visits. To this end, consider the set of partial input sequences v^i, $i \in [N]$, with a smaller PM than the ML decision $\hat{u}_{\text{ML}}^N$. Observe that there are i node-visits for SC decoding for any decoding path v^i.

Definition 2: For the channel output y^N and the binary sequence $v^N \in \mathcal{X}^N$, define the set

$$\mathcal{V}(v^N, y^N) \triangleq \bigcup_{i=1}^{N} \left\{ u^i \in \{0,1\}^i : M(u^i) \leq M(v^N) \right\}. \tag{18}$$

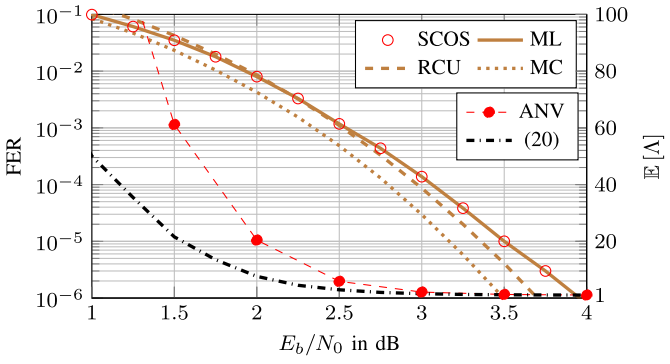
Lemma 1: We have

$$N\lambda(y^N) \geq \left| \mathcal{V}\left(\hat{u}_{\text{ML}}^N(y^N), y^N\right) \right| \tag{19}$$

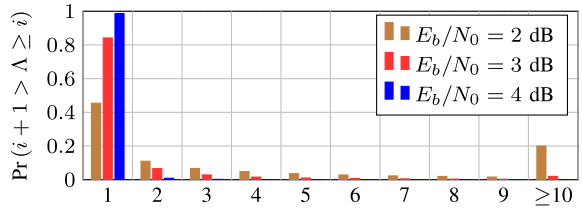
and the average number of node-visits (ANV) normalized to that of SC decoding is lower bounded as

$$\mathbb{E}[\Lambda] \geq \frac{1}{N} \mathbb{E}\left[\left| \mathcal{V}\left(\hat{u}_{\text{ML}}^N(Y^N), Y^N\right) \right| \right]. \tag{20}$$

[5] In our simulations, we count the number of arithmetic operations as well, which will be provided later in Sections IV-B and VI for various comparisons.



(a) FER/ANV vs. E_b/N_0



(b) Histogram vs. E_b/N_0

Fig. 2. FER/ANV/histogram for node visits vs. E_b/N_0 over the biAWGN channel for the $(128, 64)$ PAC code under SCOS decoding compared to relative RCU bound/lower bound (20).

Proof: Observe that each member of set $\mathcal{V}\left(v^N, y^N\right)$ corresponds to a node in the decoding tree. Then, inequality (19) follows from (18) by replacing v^N with the ML decision $\hat{u}^N_{\text{ML}}$ and observing that SCOS decoding needs to visit each node with a PM smaller than or equal to that of the ML decision. Since (19) is valid for any y^N, the bound (20) follows. □

From now on, we refer to $\mathbb{E}[\Lambda]$ as ANV by keeping in mind that it is normalized to the block length. Figure 2 provides performance, the ANV and the histogram for the node visits vs. signal-to-noise ratio (SNR) (in E_b/N_0, where E_b is here the energy per information bit and N_0 is the single-sided noise power spectral density) over the binary-input additive white Gaussian noise (biAWGN) channel [37, Sec. 4] for the $(128, 64)$ PAC code [28] under SCOS decoding where $\eta = \infty$. Observe that the lower bound on the ANV given by (20) is validated[6] and is tight for high SNR. However, the bound appears to be loose at low SNR values mainly for two reasons: (i) usually the initial SC decoding estimate v^N is not the ML decision and extra nodes in the difference set $\mathcal{V}\left(v^N, y^N\right) \setminus \mathcal{V}\left(v^N_{\text{ML}}, y^N\right)$ are visited and (ii) SCOS decoding may visit the same node multiple times and this cannot be tracked by a set definition. The histogram for the node visits, where the intervals are given as integer multiples of node visit of SC decoding, reveals the efficiency of the proposed decoder for high SNR regime. In particular, the probability that SCOS decoding needs a number of node visits larger than that of 8 times of SC decoding to guarantee returning ML decision is roughly 2×10^{-4} when $E_b/N_0 = 4$ dB.

[6]After finding the PM of ML decision for each transmission, the number of nodes in the decoding tree with lower PM than that of the ML decision is counted via a modified SCOS decoding, which is introduced in Section V-B. Then, its average provides the RHS of Eq. (20).

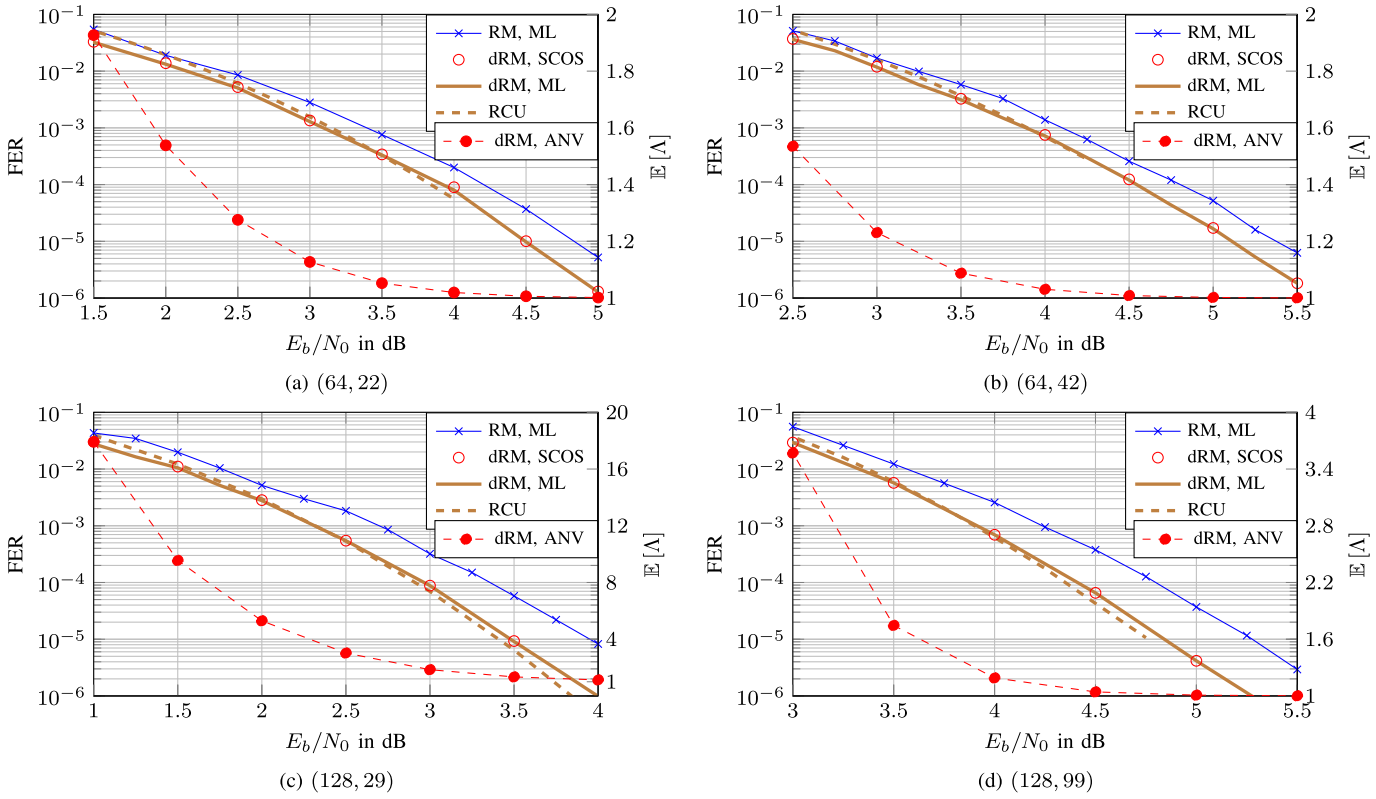
Remark 2: Recall that the PM (7) is calculated using the SC decoding schedule, i.e., it ignores the frozen bits coming after the current decoding phase i. This means the size of the set (18) tends to be smaller for codes more suited for SC decoding, e.g., polar codes, while it is larger for other codes such as RM codes. This principle is also observed when decoding via SCL decoding, i.e., the required list size to approach ML performance grows when one "interpolates" from polar to RM codes [11], [19], [20]. This observation motivates us to introduce dRM-polar code ensemble in Definition 1, whose random instances provide a good performance vs. complexity trade-off under SCOS decoding for moderate code lengths, e.g., $N = 256$ bits.

B. Performance Under Maximum Complexity Constraints

The proxy of node-visits is particularly useful when one wants to limit the worst-case complexity of *polar code decoders* leveraging the structure of the Hadamard matrix in a unified manner. Observe that, for a given code construction specified by set $\mathcal{A}$, the number $\lambda_{\text{SCL}}(L, \mathcal{A})$ of node-visits for SCL decoding with list size L is constant and upper bounded as $\lambda_{\text{SCL}}(L, \mathcal{A}) \leq L$. Given a polar code decoder, one may force it to satisfy a maximum number of node-visits such that $\lambda\left(y^N\right) \leq L$, for some positive integer L at each decoding attempt with the hope that the worst-case complexity can be comparable to de facto reference SCL decoding with list size L. For instance, sequential decoders use similar parameters, call L, to limit their worst-case complexity comparable to that of SCL decoding with list size L [15, Sec. III], [46, Sec. V]. Similarly, SCOS is modified by returning the existing most-likely candidate whenever a pre-defined maximum number of node-visits λ_{max} is reached, at the expense of suboptimality. Note that if $\lambda_{\text{max}} \geq 1$, SCOS decoding will always return a valid codeword.

Figures 3 and 4 provide performance and the ANV vs. SNR (in E_b/N_0) for short- and moderate-length dRM and dRM-polar codes, i.e., $N \in \{64, 128\}$ and $N \in \{256, 512\}$, respectively, of various rates $(0.14 < R < 0.92)$ under SCOS decoding, where λ_{max} and η are both set to 10, 100 and 5000 for cases where $N = 64$, $N = 128$ and $N \in \{256, 512\}$ for the simulations, respectively. For dRM codes, the information sets are the same as the RM code with the same block length and dimension, where the dynamic frozen bit constraints are randomly chosen. The $(256, 154)$ dRM-polar code with a rate of ≈ 0.6 is chosen uniformly at random from the ensemble of Definition 1, where the information set $\mathcal{A}$ is defined as in [19] with the mother $\text{RM}(4, 8)$ code and the polar rule given by setting $\beta = 2^{1/4}$ in [49]. Empirical *ML lower bounds* of [27] are also plotted. ML decoding performance is *approached* if the performance matches the simulated lower bounds.

For all simulated codes, the performance approaches to that of ML decoding (if not the same), which outperforms the ML performance of RM codes of the same parameters by up to 0.5 dB, which is provided as reference. Note that SCOS decoding parameters are set to the same complexity constraints for simulating RM codes, where the FERs match the ML



(a) $(64, 22)$ (b) $(64, 42)$ (c) $(128, 29)$ (d) $(128, 99)$

Fig. 3. FER/ANV vs. E_b/N_0 over the biAWGN channel for short dRM codes under SCOS decoding with $\lambda_{\max} = \eta = 10$ and $\lambda_{\max} = \eta = 100$ for $N = 64$ and $N = 128$, respectively, compared to relative RM codes under ML decoding and RCU bounds.

TABLE II
RM CODES UNDER SCOS DECODING

(N, K)	E_b/N_0	Addition	Compare	XOR	(21)	SC (22)
$(64, 22)$	5.00	300	192	193	3745	2880
$(64, 42)$	5.75	339	192	193	4057	2880
$(128, 29)$	4.00	724	504	507	9323	6720
$(128, 99)$	5.50	774	448	450	9330	6720
$(256, 37)$	3.25	17943	12367	12204	229950	15360
$(256, 219)$	5.50	1733	1032	1034	21090	15360
$(512, 466)$	5.50	4121	2515	2521	50579	34560

TABLE III
DRM CODES UNDER SCOS DECODING

(N, K)	E_b/N_0	Addition	Compare	XOR	(21)	SC (21)
$(64, 22)$	5.00	300	192	269	3821	2948
$(64, 42)$	5.50	339	192	266	4130	2954
$(128, 29)$	4.00	724	505	763	9585	6956
$(128, 99)$	5.50	774	449	708	9594	6945
$(256, 37)$	3.25	18309	12793	19418	242648	16036
$(256, 219)$	5.50	1733	1033	1697	21759	16027
$(512, 466)$	5.50	4155	2562	4553	53165	36338

lower bounds. Observe also that this remarkable performance is attained with $\mathbb{E}[\Lambda] \approx 1$ at high SNR regime except for the case of $(256, 37)$ codes, i.e., the proxy implies that the complexity will be close to that of SC decoding at low FERs (e.g., 10^{-5} and below).

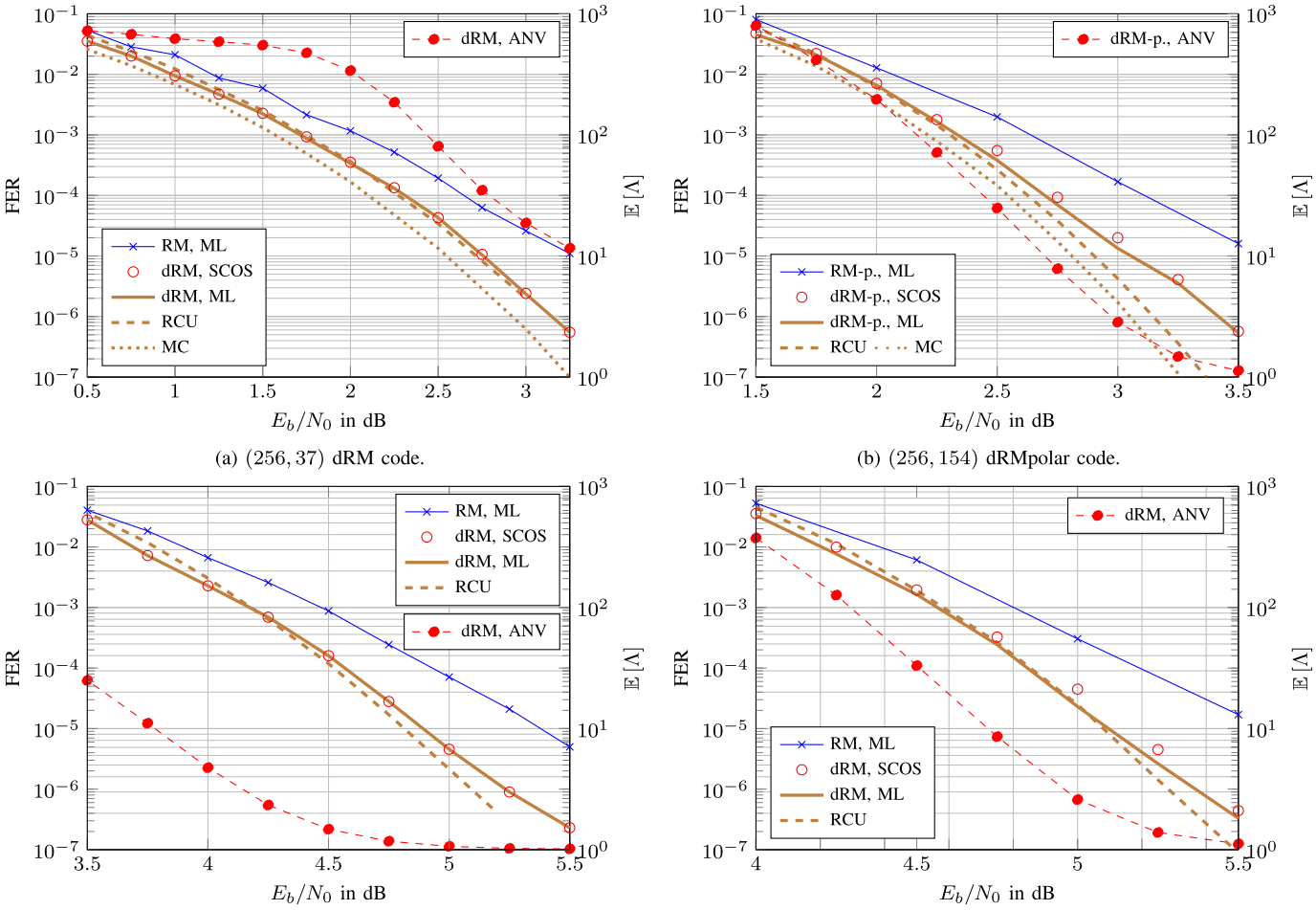
Since the number of node-visits may not refer to the exact complexity, Tables II and III provide the respective average complexity scores for SCOS, where $\lambda_{\max}$ and η are the same as in Figs. 3 and 4, as well as SC decoding of the codes as follows. During our simulations, we count the number of arithmetic operations, which include the floating-point additions and comparisons as well as binary XORs. In order to provide a unified complexity score, we assume that 1 floating-point addition corresponds roughly to 8 binary operations and 1 floating-point comparison correspond to 6 binary operations, i.e., an average complexity score is computed as

$$8 \cdot A + 6 \cdot C + B \tag{21}$$

where A, B and C are the average number of additions, XORs and comparisons during the simulations. These factors are motivated by the potential use of 8-bit representation of real numbers in order to limit quantization errors and the fact that the comparison of two real numbers may be terminated before comparing all 8 bits. Note that considered decoders do not require multiplications as the min-sum approximation is adopted. For instance, SC decoding requires the same number of $\frac{1}{2} N \log_2 N$ additions, comparisons as well as XORs for any code of length N, resulting in a fixed complexity score as

$$15 \times \frac{1}{2} N \log_2 N \tag{22}$$

if all frozen bits are set to 0. Except for the case of $(256, 37)$ codes, the average complexity score of SCOS decoding is within a factor of 1.5 from SC decoding for all RM and dRM codes at given SNR values provided in the tables.



(a) $(256, 37)$ dRM code.

(b) $(256, 154)$ dRMpolar code.

(c) $(256, 219)$ dRM code.

(d) $(512, 466)$ dRM code.

Fig. 4. FER/ANV vs. E_b/N_0 over the biAWGN channel for moderate-length dRM and dRM-polar codes under SCOS decoding $\lambda_{\max} = \eta = 5000$ compared to relative RM and RM-polar codes and RCU bounds.

V. Further Improvements

An interesting modification to SCOS is proposed by [50], which avoids using PM or any reliability score for the search as follows. After finishing an instance of the initial SC decoding, the flipping sets are prioritized according to a predefined depth-first or breadth-first search order. Numerical results show that the latter required less number of SC decoding attempts for various RM codes of length up to 512 bits for approaching their ML decoding performance compared to SCOS decoding although each attempt is expected to require more node-visits. Nevertheless, the results imply that SCOS provides robust performance with various search schedules. In the following, we investigate the effect of search schedule via simulations when the bit reliability under SC decoding is ignored, i.e., when only PMs are used for the search.

A. Bias Term Robustness

Consider the bias terms b_i given in (13), which impacts the search priority but not the performance if the maximum complexity constraints are unbounded. This means that a *suboptimal* bias term does not change the performance of SCOS decoding with unbounded complexity (which is still ML decoding), but it may increase the complexity.

In order to compute b_i, we assume that the all-zero codeword is transmitted thanks to the channel symmetry and the linearity of the codes under consideration. Let $f_N^{(i)}$ denote the probability density function (PDF) of the RV corresponding to $\ell_i(\mathbf{0})$, where $\mathbf{0}$ denotes an all-zero vector of length i and the source of randomness is the channel output Y^N. Over general B-DMCs, the densities can be computed recursively as

$$f_N^{(2i-1)} = f_{N/2}^{(i)} \circledast f_{N/2}^{(i)} \tag{23}$$

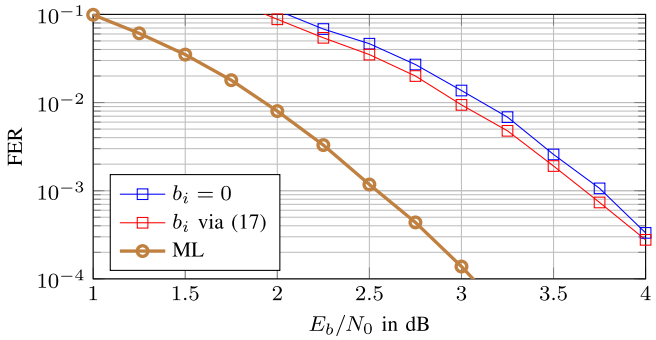
$$f_N^{(2i)} = f_{N/2}^{(i)} \circledcirc f_{N/2}^{(i)} \tag{24}$$

where $f_1^{(1)}$ is the PDF of the i.i.d. LLRs at the channel output, and $\circledast$ and $\circledcirc$ denote the check and variable node convolutions, respectively, as defined in [37, Ch. 4]. Then, terms p_j in Eq. (17) can be computed via $f_N^{(i)}$ as

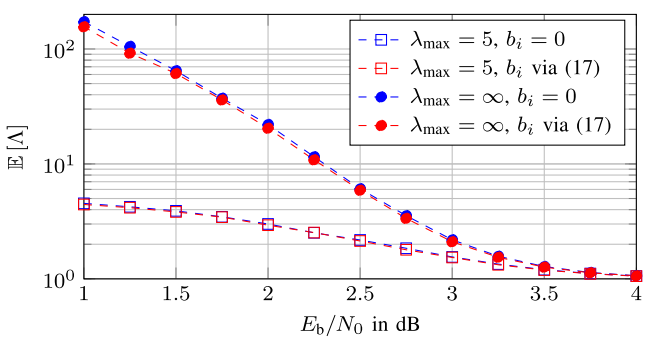
$$p_j = \lim_{z \to 0} \left(\int_{-\infty}^{-z} f_N^{(j)}(x)dx + \frac{1}{2} \int_{-z}^{+z} f_N^{(j)}(x)dx \right). \tag{25}$$

The computation of (23), (24) and (25) can be carried out, for instance, via quantized density evolution [51], yielding an accurate estimate of the RHS of (25).

Figure 5 illustrates the effect of various bias terms outlined below on the performance of SCOS decoding with bounded and unbounded $\lambda_{\max}$.



(a) Performance with $\lambda_{\max} = 5$.



(b) ANV.

Fig. 5. FER/ANV vs. E_b/N_0 over the biAWGN channel for the $(128, 64)$ PAC code under SCOS decoding with various bias terms and maximum complexity constraints such that $\eta = \lambda_{\max}$.

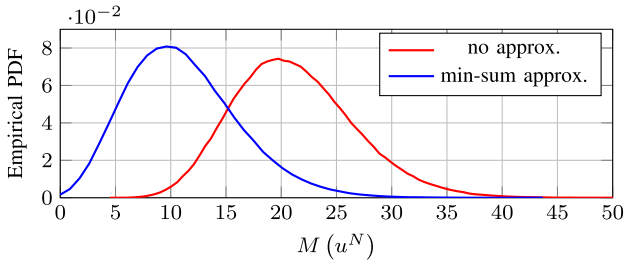


Fig. 6. Empirical PDF (via 10^7 samples) of the $M(u^N)$ over the biAWGN channel at 3.5 dB SNR for the $(128, 64)$ PAC code. The both curves are obtained via genie-aided SC decoding [3], where the blue line uses the min-sum approximation and the red line without any approximation.

- The bias terms are computed via the RHS of (17) using quantized density evolution for each SNR point.
- The bias terms are set to zero, i.e., $b_i = 0$, $i \in [N]$ which results in using the PM as score as well.

The complexity reduction is limited if (17) is used instead of setting the bias terms to zero. Nevertheless, setting them to zero slightly degrades the performance (by ≈ 0.12 dB) when the maximum number of node-visits is constraint to five times that of SC decoding with almost no savings in the average complexity. Hence, we conclude that SCOS decoding is not very sensitive to the choice of bias terms unless the maximum complexity is required to be very low.

Algorithm 4 SCOS with maximum PM $\left(\ell^N, M_{\max}\right)$

Input : input LLRs ℓ^N, $M_{\max}$
Output: output vector $\hat{u}$, decoding state ω

1 $\mathcal{L} = \varnothing, \mathcal{E}_{\mathrm{p}} = \varnothing, \mathtt{M}_{\mathtt{cm1}} = M_{\max}$, $\omega = 0$
2 **for** $i = 1, 2, \ldots, N$ **do**
3 $\mathtt{L}[1, i] = \ell_i$
4 $i_{\text{end}} =$ SCDec $(1, \varnothing)$
5 **if** $i_{end} = N$ **then** $\omega = 1$
6 **for** $i = 1, 2, \ldots, N$ **do**
7 **if** $i \in \mathcal{A}$ *and* $\overline{\mathtt{M}}[i] < M_{cm1}$ **then**
8 InsertHeap $(\langle \{i\}, \overline{\mathtt{M}}[i], \overline{\mathtt{S}}[i] \rangle)$
9 **while** $\mathcal{L} \neq \varnothing$ **do**
10 $\langle \mathcal{E}, \overline{\mathtt{M}}_{\mathcal{E}}, \overline{\mathtt{S}}_{\mathcal{E}} \rangle =$ popMin $(\mathcal{L})$
11 **if** $\overline{M}_{\mathcal{E}} < M_{cm1}$ **then**
12 $i_{\text{start}} =$ FindStartIndex $(\mathcal{E}, \mathcal{E}_{\mathrm{p}})$
13 $i_{\text{end}} =$ SCDec $(i_{\text{start}}, \mathcal{E})$
14 **if** $i_{end} = N$ **then** $\omega = 1$
15 **for** $i =$ *maximum* $(\mathcal{E}) + 1, \ldots, i_{end}$ **do**
16 **if** $i \in \mathcal{A}$ *and* $\overline{\mathtt{M}}[i] < M_{cm1}$ **then**
17 InsertHeap $(\langle \mathcal{E} \cup \{i\}, \overline{\mathtt{M}}[i], \overline{\mathtt{S}}[i] \rangle)$
18 $\mathcal{E}_{\mathrm{p}} = \mathcal{E}$
19 **return** $\hat{u}$, ω

B. SC Ordered Search Decoding with Maximum Path Metric

Monte Carlo simulation under genie-aided SC decoding [3] can be used to approximate the PDF of the PM for the transmitted message at a given SNR. Note that $M(u^N)$ is a RV where the source of randomness is the channel output. Since we consider symmetric B-DMCs and a linear code with uniform distribution, the PDF of $M(u^N)$ could be computed with an all-zero codeword assumption. For instance, Figure 6 provides the PDF for the $(128, 64)$ PAC code at $E_b/N_0 = 3.5$ dB. Observe that $\Pr\left(M(u^N) > 50\right) \approx 0$, i.e., if the decoder discards the paths having PMs larger than 50, then the performance degradation is negligible while reducing computational complexity. Such a modification is particularly relevant when a maximum complexity constraint is imposed on SCOS decoding. In this case, unnecessary node-visits drain the computation budget and increase the number of suboptimal decisions. Moreover, the threshold test lets the decoder reject unreliable decisions and reduces the number of undetected errors if the threshold is carefully optimized, see [30]. In the following, we modify SCOS decoding by setting a maximum PM $M_{\max}$ as shown in Algorithm 4.

The differences in SCOS decoding with a maximum PM are highlighted in red. The algorithm requires the input $M_{\max}$ to discard candidates with PMs larger than $M_{\max}$, where the PM of the current most-likely path is initialized to this threshold (`line 1`). In addition, the algorithm has a binary output ω, which is initialized to 0 (`line 1`), is set to 1 if any estimate with PM less than $M_{\max}$ is found (`lines 5 and 14`) and stays as 0 otherwise. Note that the initial SC decoding does not reach a leaf node in case SC decoding path is exceeding $M_{\max}$ (`line 4`).

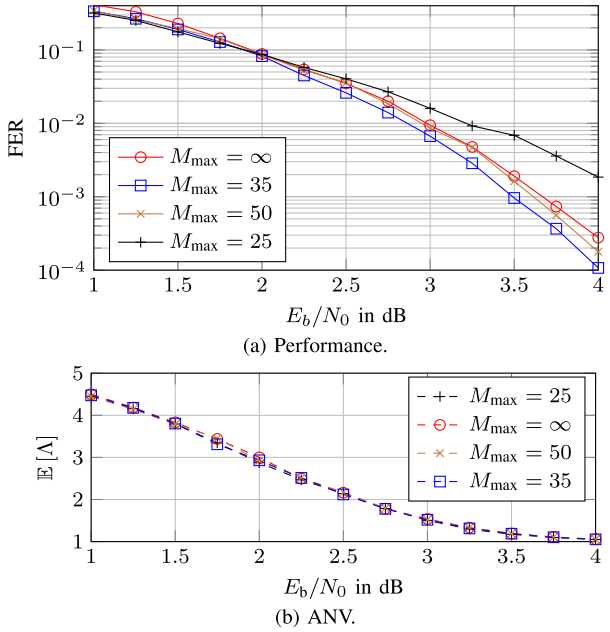


Fig. 7. FER/ANV vs. E_b/N_0 over the biAWGN channel for the $(128, 64)$ PAC code under SCOS decoding with various maximum PMs and fixed maximum complexity constraints $\lambda_{\max} = \eta = 5$.

Figure 7 compares the performance of SCOS decoding with maximum PM constraint (Algorithm 4) to that of original SCOS decoding. The former gains ≈ 0.2 dB with the same maximum complexity constraint $\lambda_{\max} = 5$ if $M_{\max} = 35$. Note that the average complexity is similar.

Observe now that the proposed modification enables SCOS decoding to reject an *unreliable* estimate inherently. For a given threshold $M_{\max}$, define the binary RV

$$\Omega = \mathbb{1}\left\{M(\hat{u}^N) \leq M_{\max}\right\} \tag{26}$$

where the indicator function $\mathbb{1}\{\mathrm{P}\}$ takes on the value 1 if the proposition P is true and 0 otherwise. The proposition of the indicator function (26) reads as "the modified SCOS decoding finds an estimate $\hat{u}^N$ with a PM smaller than $M_{\max}$". The undetected error probability of the algorithm is

$$\Pr\left(\hat{U}^N \neq U^N, \Omega = 1\right). \tag{27}$$

The overall error probability is the sum of the detected and undetected error probabilities, i.e., we have

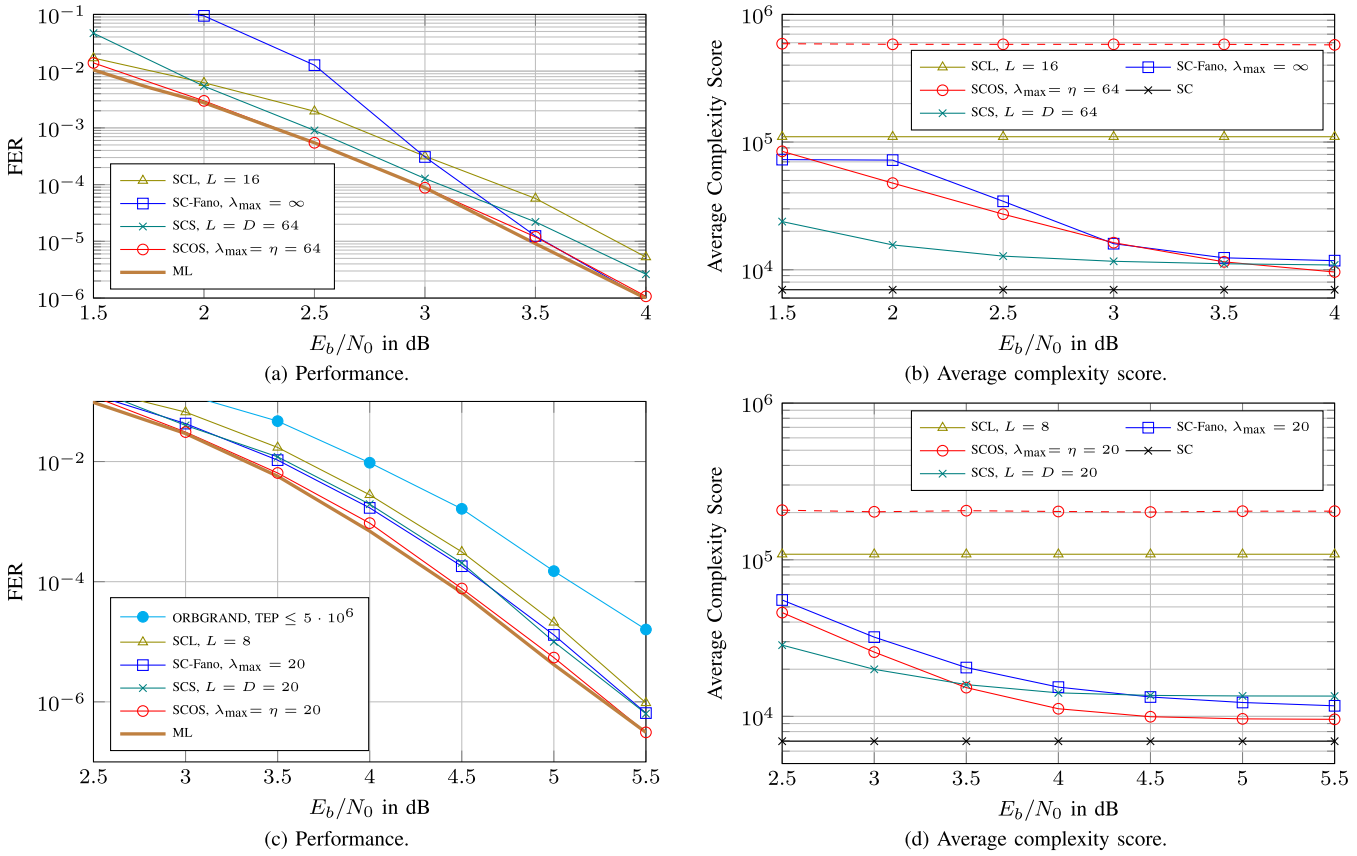
$$\Pr\left(\hat{U}^N \neq U^N\right) = \sum_{\omega \in \{0,1\}} \Pr\left(\hat{U}^N \neq U^N, \Omega = \omega\right) \tag{28}$$

which follows from the law of total probability. The parameter $M_{\max}$ controls the FER and uFER tradeoff [30], [52]. In particular, (27) is the left hand side (LHS) of (28) if $M_{\max} = \infty$. Numerical results illustrating benefits in uFER is provided towards the end of the next section.

VI. Comparison to Existing Polar Decoders

This section compares the proposed SCOS decoding to SC-Fano, SCL, SCS as well as ordered reliability bits guessing random additive noise decoding (ORBGRAND) [53] algorithms in the short block length regime, e.g., for $N = 128$, for two different code dimensions $K = 29$ and $K = 99$, over the biAWGN channels. Note that the score is given by (12) for SCOS, SC-Fano, and SCS decoding algorithms, where we use min-sum approximation for the first term and quantized density evolution for the bias as explained in Section V-A. We provide FERs and average complexity scores computed via (21) for these codes.

Figures 8(a) and 8(b) provides the results for the $(128, 29)$ code, where SCOS with maximum complexity constraints $\lambda_{\max} = \eta = 64$ approaches to its ML performance outperforming other decoding algorithms. This performance is achieved with the lowest complexity at high SNR. Its performance is followed by SCS decoding with stack size $D = 64$ and maximum node-visits $L = 64$ in a wide SNR regime robustly with 0.25 dB difference. It provides lower complexity compared to SCOS and SC-Fano decoding algorithms at SNR values smaller than 3.5 dB; however, it requires more complexity at higher SNR values with worse performance. Although its space complexity is already larger than other competitors for the chosen parameters, the performance of SCS decoding can be further improved if the stack size is chosen to be at least a few times of L, which is, e.g., chosen to be $D = LN$ in [15, Sec. IV]. However, this comes at the expense of also potentially higher computational especially at low and mid SNR ranges. SC-Fano decoding (even with unbounded complexity) does not perform well in particular at relatively low and medium FERs, which might require a careful optimization of the search parameter Δ. Nevertheless, it is very competitive to SCOS decoding at low FERs in performance and average complexity score. In addition, it does not require large space complexity as it is the case for SCS decoding. Also for the $(128, 99)$ code, SCOS decoding with $\lambda_{\max} = \eta = 20$ approaches ML decoding tightly (see Figure 8(c)). Figure 8(d) shows that its average complexity score is lower than all the competitors for SNR values larger than 3.5 dB, i.e., except for high FERs. Its complexity score is within 1.4 times of SC decoding at high SNR values in both cases. Although the complexity of SCOS decoding, at least in the proxy of ANV, is reaching to that of SC decoding at high SNR values,it requires extra operations similar to other improved decoding algorithms, e.g., in `lines 12-14` of Algorithm 2 for the computation of scores and PMs independent of operating SNR, which hinders its complexity score to hit that of SC decoding at high SNR values. In the case of SCS decoding, this is partially due to the push and pop operations as well as for the calculation of the scores for the pushed paths. To complement the numerical results of Figure 8, we provide Table IV for SCS decoding, where the parameters of the decoder is kept the same as in the figure. As expected, SCL decoding requires much larger complexity compared to all SC-based complexity-adaptive decoding algorithms. In addition, we also provided the worst-case complexity scores obtained during simulations at each SNR value for the proposed SCOS algorithm, which are a few times more than that of SCL decoding due to the higher $\lambda_{\max}$ of the latter. Note that the performance of the $(128, 99)$ code is also provided under ORBGRAND, which is an efficient approximation of soft GRAND [54], where



(a) Performance.

(b) Average complexity score.

(c) Performance.

(d) Average complexity score.

Fig. 8. FER and average complexity score vs. E_b/N_0 over the biAWGN channel for $(128, 29)$ (a-b) and $(128, 99)$ (c-d) dRM codes under SCOS compared to SC-Fano, SCL and SCS decoding algorithms under various complexity constraints. As reference, the worst-case complexity scores obtained via simulations for each SNR value are also provided for SCOS as dashed curves with solid marks.

TABLE IV
dRM Codes Under SCS Decoding

(N, K)	E_b/N_0	Addition	Compare	XOR	(21)	SC (21)
$(128, 29)$	4.00	771	613	1040	10889	6956
$(128, 99)$	5.50	902	852	1149	13480	6945

the maximum number of test error patterns (TEPs) is set to 5×10^6. Observe that it performs within 0.7 dB from the ML decoding performance for the considered case. The average complexity score[7] reaches roughly to 5×10^6 at an SNR of $E_b/N_0 = 5.5$ dB, which we skip in Figure 8(d) as it is an order of magnitude larger than that of SCL decoding with $L = 20$. As Remark 2 hints, SCOS decoding provides advantage over universal decoders like ORBGRAND if the underlying code is somewhat suited for SC-based decoders.

Finally, Figure 9 illustrates that the $(128, 64)$ PAC code under modified SCOS decoding gains in overall FER and in uFER as compared to a $(128, 71)$ polar code concatenated with a CRC-7 (resulting in a $(128, 64)$ overall code) under SCL decoding with $L = 16$ at high SNR. Furthermore, the code outperforms DSCF decoding with the maximum number $T_{\max} = 70$ of bit flips.

[7]The codeword membership test costs a binary vector-matrix multiplication using the parity-check matrix of the underlying dRM code similar to [55]. This way of membership test is not mandatory for GRAND-based decoders and more efficient methods might be used depending on the underlying code, e.g., applying a polar transform is sufficient for polar and RM codes.

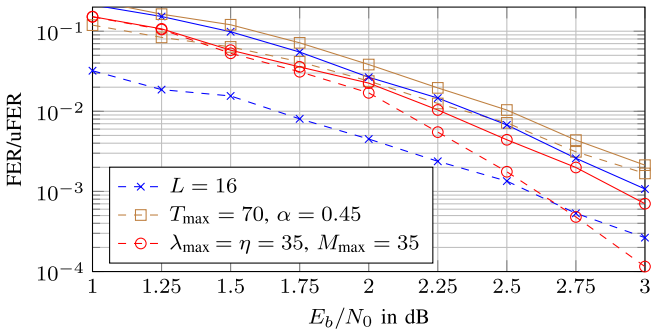


Fig. 9. FER (solid)/uFER (dashed) vs. E_b/N_0 over the biAWGN channel for the $(128, 64)$ PAC code under modified SCOS decoding with a maximum PMs and a fixed maximum complexity constraint compared to a $(128, 64)$ modified polar code with an outer CRC-7 having the generator polynomial $g(x) = x^7 + x^6 + x^5 + x^2 + 1$ [31].

VII. Conclusion

The SCOS algorithm was proposed that implements ML decoding. The complexity adapts to the channel quality and approaches the complexity of SC decoding for the illustrated short- to moderate-length $\mathbf{G}_N$-coset codes with RM rate profiles at high SNR values. Unlike existing alternatives, the algorithm does not need an outer code or a separate parameter optimization. In addition, it provides better performance compared to SC-Fano and SCS decoding algorithms with a lower complexity at high SNR values.

Algorithm 5 recursivelyCalcL (λ, ϕ)

Input : layer λ and phase ϕ

1 **if** $\lambda = 1$ **then return**
2 $\psi = \lfloor \phi/2 \rfloor, t = 2^{\lambda-2}$
3 **if** $\phi \mod 2 = 0$ **then**
4 recursivelyCalcL $(\lambda - 1, \psi)$
5 **for** $\beta = 0, 1, \ldots, 2^{\log_2 N - \lambda + 1} - 1$ **do**
6 **if** $\phi \mod 2 = 0$ **then**
7 $\mathtt{L}[\lambda, \phi + 2\beta t + 1] = f^-(\mathtt{L}[\lambda - 1, \psi + 2\beta t + 1], \mathtt{L}[\lambda - 1, \psi + (2\beta + 1)t + 1])$
8 **else**
9 $\mathtt{L}[\lambda, \phi + 2\beta t + 1] = f^+(\mathtt{L}[\lambda - 1, \psi + 2\beta t + 1], \mathtt{L}[\lambda - 1, \psi + (2\beta + 1)t + 1], \mathtt{C}[\lambda, \phi + 2\beta t])$

Algorithm 6 recursivelyCalcC (λ, ϕ)

Input : layer λ and phase ϕ

1 $\psi = \lfloor \phi/2 \rfloor, t = 2^{\lambda-2}$
2 **for** $\beta = 0, 1, \ldots, 2^{\log_2 N - \lambda + 1} - 1$ **do**
3 $\mathtt{C}[\lambda - 1, \psi + 2\beta t + 1] = \mathtt{C}[\lambda, \phi + 2\beta t] \oplus \mathtt{C}[\lambda, \phi + 2\beta t + 1]$
4 $\mathtt{C}[\lambda - 1, \psi + (2\beta + 1)t + 1] = \mathtt{C}[\lambda, \phi + 2\beta t + 1]$
5 **if** $\psi \mod 2 = 1$ **then**
6 recursivelyCalcC $(\lambda - 1, \psi)$

A modification to SCOS was proposed, which provides further gains compared to the original algorithm when there is stringent maximum complexity constraints. The modification has a potential to provide a trade-off between the overall and undetected error probabilities as a byproduct. Using the modified SCOS, the $(128, 64)$ PAC code provides simultaneous gains in the overall and undetected frame error rates compared to CRC-concatenated polar codes under SCL decoding.

APPENDIX

Here, we provide the standard routines recursivelyCalcL and recursivelyCalcC required for SCOS decoding as Algorithms 5 and 6 similar to [27, Alg. 3] and [27, Alg. 4], respectively. Note that Algorithm 6 denotes the check and variable node operations as f^- (`line 8`) and f^+ (`line 10`), where the former may be implemented using min-sum approximation.

ACKNOWLEDGMENT

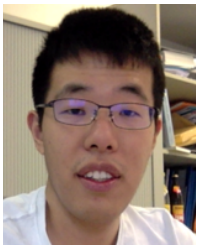
The authors would like to thank Gerhard Kramer (TUM) for the comments on an early version of this manuscript, also would like to thank the Associate Editor and the anonymous reviewers for their valuable comments, which improved the presentation and the content of the work significantly, and also would like to thank the reviewer who directed them to use heap structure in the implementations to reduce the complexity.

Some of the results are published in Chapter 4 of the Ph.D. thesis of the first author [2].

REFERENCES

[1] P. Yuan and M. C. Coskun, "Complexity-adaptive maximum-likelihood decoding of modified GN-coset codes," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Oct. 2021, pp. 1–6.
[2] P. Yuan, "Polar coding with complexity-adaptive decoding and time-varying channels," Ph.D. dissertation, Inst. Commun. Eng., Technische Universität München, Munich, Germany, 2021.
[3] 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.
[4] N. Stolte, "Rekursive codes mit der plotkin-konstruktion und ihre decodierung," Ph.D. dissertation, Dept. Elektrotechnik Informationstechnik, TU Darmstadt, Darmstadt, Germany, 2002.
[5] I. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *Trans. IRE Prof. Group Inf. Theory*, vol. 4, no. 4, pp. 38–49, Sep. 1954.
[6] D. E. Müller, "Application of Boolean algebra to switching circuit design and to error detection," *Trans. IRE Prof. Group Electron. Comput.*, vol. 3, no. 3, pp. 6–12, 1954.
[7] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Sasoglu, and R. L. Urbanke, "Reed–Müller codes achieve capacity on erasure channels," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4298–4316, Jul. 2017.
[8] G. Reeves and H. D. Pfister, "Reed–Müller codes on BMS channels achieve vanishing bit-error probability for all rates below capacity," *IEEE Trans. Inf. Theory*, vol. 70, no. 2, pp. 920–949, Apr. 2024.
[9] M. C. Coşkun et al., "Efficient error-correcting codes in the short blocklength regime," *Phys. Commun.*, vol. 34, pp. 66–79, Jun. 2019.
[10] K. Ivanov and R. L. Urbanke, "On the efficiency of polar-like decoding for symmetric codes," *IEEE Trans. Commun.*, vol. 70, no. 1, pp. 163–170, Jan. 2022.
[11] M. C. Coskun and H. D. Pfister, "An information-theoretic perspective on successive cancellation list decoding and polar code design," *IEEE Trans. Inf. Theory*, vol. 68, no. 9, pp. 5779–5791, Sep. 2022.
[12] I. Dumer and K. Shabunov, "Soft-decision decoding of Reed–Müller codes: Recursive lists," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 1260–1266, Mar. 2006.
[13] K. Niu and K. Chen, "Stack decoding of polar codes," *Electron. Lett.*, vol. 48, no. 12, pp. 695–697, Jun. 2012.
[14] V. Miloslavkaya and P. Trifonov, "Sequential decoding of polar codes," *IEEE Commun. Lett.*, vol. 18, no. 7, pp. 1127–1130, Jul. 2014.
[15] P. Trifonov, "A score function for sequential decoding of polar codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 1470–1474.
[16] M. Jeong and S. Hong, "SC-fano decoding of polar codes," *IEEE Access*, vol. 7, pp. 81682–81690, 2019.
[17] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, "A low-complexity improved successive cancellation decoder for polar codes," in *Proc. 48th Asilomar Conf. Signals, Syst. Comput.*, Nov. 2014, pp. 2116–2120.
[18] L. Chandesris, V. Savin, and D. Declercq, "Dynamic-SCFlip decoding of polar codes," *IEEE Trans. Commun.*, vol. 66, no. 6, pp. 2333–2345, Jun. 2018.
[19] B. Li, H. Shen, and D. Tse, "A RM-polar codes," 2014, *arXiv:1407.5483*.
[20] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "From polar to Reed–Müller codes: A technique to improve the finite-length performance," *IEEE Trans. Commun.*, vol. 62, no. 9, pp. 3084–3091, Sep. 2014.
[21] P. Trifonov and V. Miloslavskaya, "Polar subcodes," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 254–266, Feb. 2016.
[22] M. Kamenev, Y. Kameneva, O. Kurmaev, and A. Maevskiy, "Permutation decoding of polar codes," in *Proc. 14th Int. Symp. 'Problems Redundancy Inf. Control Syst.'*, Oct. 2019, pp. 1–6.
[23] B. Li, J. Gu, and H. Zhang, "Performance of CRC concatenated pre-transformed RM-polar codes," 2021, *arXiv:2104.07486*.
[24] H. Yao, A. Fazeli, and A. Vardy, "List decoding of Arikan's PAC codes," *Entropy*, vol. 23, no. 7, p. 841, 2021.
[25] 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.
[26] V. Miloslavskaya and B. Vucetic, "Design of short polar codes for SCL decoding," *IEEE Trans. Commun.*, vol. 68, no. 11, pp. 6657–6668, Nov. 2020.
[27] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.
[28] E. Arıkan, "From sequential decoding to channel polarization and back again," 2019, *arXiv:1908.09594*.
[29] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.
[30] G. Forney, "Exponential error bounds for erasure, list, and decision feedback schemes," *IEEE Trans. Inf. Theory*, vol. IT-14, no. 2, pp. 206–220, Mar. 1968.

[31] P. Yuan, T. Prinz, G. Boecherer, O. Iscan, R. Boehnke, and W. Xu, "Polar code construction for list decoding," in *Proc. SCC ; 12th Int. ITG Conf. Syst., Commun. Coding*, Feb. 2019, pp. 1–6.
[32] R. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inf. Theory*, vol. IT-9, no. 2, pp. 64–74, Apr. 1963.
[33] K. Niu and K. Chen, "CRC-aided decoding of polar codes," *IEEE Commun. Lett.*, vol. 16, no. 10, pp. 1668–1671, Oct. 2012.
[34] B. Dorsch, "A decoding algorithm for binary block codes and J-ary output channels (corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 3, pp. 391–394, May 1974.
[35] M. P. C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inf. Theory*, vol. 41, no. 5, pp. 1379–1396, Sep. 1995.
[36] Y. Wu and C. N. Hadjicostis, "Soft-decision decoding using ordered recodings on the most reliable basis," *IEEE Trans. Inf. Theory*, vol. 53, no. 2, pp. 829–836, Feb. 2007.
[37] T. Richardson and R. Urbanke, *Modern Coding Theory*. New York, NY, USA: Cambridge Univ. Press, 2008.
[38] R. Mori and T. Tanaka, "Performance and construction of polar codes on symmetric binary-input memoryless channels," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Feb. 2009, pp. 1496–1500.
[39] M. C. Coşkun, J. Neu, and H. D. Pfister, "Successive cancellation inactivation decoding for modified Reed–Müller and eBCH codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Sep. 2020, pp. 437–442.
[40] B. Li, H. Zhang, and J. Gu, "On pre-transformed polar codes," 2019, *arXiv:1912.06359*.
[41] Y. Li, H. Zhang, R. Li, J. Wang, G. Yan, and Z. Ma, "On the weight spectrum of pre-transformed polar codes," 2021, *arXiv:2102.12625*.
[42] A. Balatsoukas-Stimming, M. B. Parizi, and A. Burg, "LLR-based successive cancellation list decoding of polar codes," *IEEE Trans. Signal Process.*, vol. 63, no. 19, pp. 5165–5179, Oct. 2015.
[43] P. Trifonov and V. Miloslavskaya, "Polar codes with dynamic frozen symbols and their decoding by directed search," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Sep. 2013, pp. 1–5.
[44] M. D. Atkinson, J.-R. Sack, N. Santoro, and T. Strothotte, "Min-max heaps and generalized priority queues," *Commun. ACM*, vol. 29, no. 10, pp. 996–1000, Oct. 1986.
[45] M. C. Coşkun and P. Yuan. *Complexity-Adaptive Maximum-Likelihood Decoding Algorithms With a Successive Cancellation Schedule*. Accessed: Apr. 30, 2023. [Online]. Available: https://github.com/mcemilcoskun/scml
[46] M. Moradi, A. Mozammel, K. Qin, and E. Arikan, "Performance and complexity of sequential decoding of PAC codes," 2020, *arXiv:2012.04990*.
[47] M. Moradi, "On sequential decoding metric function of polarization-adjusted convolutional (PAC) codes," *IEEE Trans. Commun.*, vol. 69, no. 12, pp. 7913–7922, Dec. 2021.
[48] I. Jacobs and E. Berlekamp, "A lower bound to the distribution of computation for sequential decoding," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 2, pp. 167–174, Apr. 1967.
[49] G. He et al., "Beta-expansion: A theoretical framework for fast and recursive construction of polar codes," in *Proc. IEEE Global Commun. Conf.*, Dec. 2017, pp. 1–6.
[50] S. A. Hashemi, N. Doan, W. J. Gross, J. Cioffi, and A. Goldsmith, "A tree search approach for maximum-likelihood decoding of Reed–Müller codes," in *Proc. IEEE Globecom Workshops*, Dec. 2021, pp. 1–6.
[51] S.-Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, no. 2, pp. 58–60, Feb. 2001.
[52] E. Hof, I. Sason, and S. Shamai (Shitz), "Performance bounds for erasure, list, and decision feedback schemes with linear block codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 8, pp. 3754–3778, Aug. 2010.
[53] K. R. Duffy, W. An, and M. Médard, "Ordered reliability bits guessing random additive noise decoding," *IEEE Trans. Signal Process.*, vol. 70, pp. 4528–4542, 2022.
[54] A. Solomon, K. R. Duffy, and M. Médard, "Soft maximum likelihood decoding using GRAND," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.
[55] P. Yuan, K. R. Duffy, E. P. Gabhart, and M. Médard, "On the role of quantization of soft information in GRAND," in *Proc. IEEE Globecom Workshops*, Jun. 2023, pp. 1–6.



Peihong Yuan (Member, IEEE) was born in Shanghai, China, in 1988. He received the M.Sc. and Dr.-Ing. (Ph.D. equivalent) degrees in electrical and computer engineering from the Technical University of Munich (TUM) in 2015 and 2021, respectively. Since 2022, he has been a member of the Research Laboratory of Electronics (RLE), Massachusetts Institute of Technology (MIT), as a Post-Doctoral Associate. During his graduate studies, he was a Lecturer with TUM Asia, Singapore, in 2021, and an invited Lecturer with Tongji University, Shanghai, from 2019 to 2021. He was an Exemplary Reviewer of IEEE TRANSACTIONS ON COMMUNICATIONS in 2022. His research interests include applied probability, information theory, and coding theory.

Mustafa Cemil Coşkun (Member, IEEE) received the B.Sc. degree (summa cum laude) in electrical and electronics engineering from Boğaziçi University, Turkey, in 2014, and the M.Sc. degree (summa cum laude) in communications engineering and the Dr.-Ing. degree in electrical and computer engineering from the Technical University of Munich (TUM), Germany, in 2017 and 2022, respectively.

He is currently a member of technical staff with Nokia Bell Labs, Murray Hill, NJ, USA. During his graduate studies, he was also a member of the Information Transmission Group, German Aerospace Center, from 2016 to 2020; a Visiting Scholar with Duke University, USA, from 2019 to 2020; and a Lecturer with TUM Asia, Singapore, from 2020 to 2021. He was a Co-Organizer of "TUM-COM Workshop on Ultra-Reliable Low-Latency Communications (URLLCs) and Applications for 5G" in 2018 and an Exemplary Reviewer of IEEE TRANSACTIONS ON COMMUNICATIONS in 2021. His research interests include applied probability, information theory, and coding theory with applications to wireless communications.