A Simplified Lower Bound on Minimum Distance of Convolutional Polar Codes

Ruslan Morozov
ITMO University, Russia
rmorozov@itmo.ru

***Abstract*—A simplified method for estimation of the minimum distance of convolutional polar codes is proposed. It is based on a novel algorithm for computing the minimum weight of cosets of the codes induced by the convolutional polarizing transformation. The proposed method also enables one to reduce complexity of procedure of construction of convolutional polar subcodes.**

***Index Terms*—Convolutional polar codes, minimum distance.**

I. Introduction

Convolutional polar codes (CvPCs) are introduced in [2]. Their finite-length performance under successive cancellation (SC) decoding is better than that of classical polar codes [1], due to higher rate of polarization, which is conjectured in [3] to be 0.61 for the case of binary erasure channel (BEC).

In [6], [7], [4] implementations of SC and list SC decoding were proposed. It was shown in [4] that convolutional polar codes under list SC decoding experience error floor due to low minimum distance. Addressing this problem, a construction of convolutional polar subcodes was proposed in [5].

Convolutional polar subcodes (CvPSs) were shown to outperform CRC-concatenated convolutional polar codes and polar subcodes [9] under list SC decoding [8], [4] with the same list size and even with the same complexity if the list size is sufficiently large. CvPSs exploit the idea of type-A dynamic frozen symbols for polar subcodes [9] to reduce error coefficient and improve minimum distance. This construction requires one to know the minimum weight of coset of code, generated by last rows $\varphi, \varphi+1, \dots, n-1$ of the polarizing transformation of size n for each $\varphi = 0, ..., n-1$. Let us call such minimum weight as *the φ-th coset weight*.

In the case of classical Arikan polarizing transformation of size n, the φ-th coset weight is given by $2^{\mathbf{wt}\varphi}$, where $\mathbf{wt}\varphi$ is the number of ones in the binary representation of integer φ. In the case of convolutional polarizing transformation, no closed formula is known. Recursive formulae and the algorithm for computing the φ-th coset weights are provided in [5], which follow from the proof of the lower bound on minimum distance of convolutional polar codes.

In this paper, a simple algorithm to compute the φ-th coset weights for the case of convolutional polar codes is proposed. It allows one to reduce the complexity of constructing CvPSs. Moreover, simpler relations between the described cosets can be used in future research on CvPCs.

II. Background

A. Notations

The following notations are used throughout the paper. $\mathbb{F}$ denotes the Galois field of two elements. For integer n we denote the set $[n] = \{0, 1, \dots n-1\}$. Symbol a_b^c denotes vector $(a_b, a_{b+1}, \dots, a_c)$. For $m \times n$ matrix A and sets $\mathcal{X} \subseteq [m]$, $\mathcal{Y} \subseteq [n]$, by $A_{\mathcal{X},\mathcal{Y}}$ we denote the submatrix of A with rows from set $\mathcal{X}$ and columns from set $\mathcal{Y}$, where indexing of rows and columns starts with zero. Similar notations are applied to vectors as well. If $\mathcal{X} = *$ or $\mathcal{Y} = *$, this means that all rows or all columns of the original matrix are in the submatrix. Furthermore, $A_{\overline{\mathcal{X}},\overline{\mathcal{Y}}}$ denotes a submatrix of A consisting of rows and columns with indices that are not in $\mathcal{X}$ and $\mathcal{Y}$, respectively. The vector of i zeroes is denoted by $\mathbf{0}^i$, or just by $\mathbf{0}$, if i is clear from the context.

We also use for two vectors/matrices/elements a and b symbol $a.b$ for their concatenation, which has lower precedence than $+$, i.e., $a.b + c = (a, b + c)$. For two sets of vectors A and B denote by $A.B$ the set of all concatenations $a.b$, where $a \in A$, $b \in B$. For vector a_b^c we denote the sum of its elements by $\Sigma a_b^c = \sum_{i=b}^{c} a_i$.

B. Convolutional Polar Codes

Convolutional polar codes [3] (CvPCs) are a family of linear block codes of length $n = 2^m$. The generator matrix of a CvPC consists of rows of $n \times n$ non-singular matrix $Q^{(n)}$, called convolutional polarizing transformation (CvPT), defined as

$$Q^{(n)} = \left(X^{(n)}Q^{(n/2)}.Z^{(n)}Q^{(n/2)}\right), \tag{1}$$

where $Q^{(1)} = (1)$, $X^{(l)}$ and $Z^{(l)}$ are $l \times l/2$ matrices, defined for even l as

$$X_{i,j}^{(l)} = \begin{cases} 1, & \text{if } 2j \le i \le 2j+2 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$Z_{i,j}^{(l)} = \begin{cases} 1, & \text{if } 2j < i \le 2j+2 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

For example, $X^{(4)} = \begin{pmatrix} 1110 \\ 0011 \end{pmatrix}^T$, $Z^{(4)} = \begin{pmatrix} 0110 \\ 0001 \end{pmatrix}^T$. Expansion (1) corresponds to one layer of the CvPT, which is depicted in Fig. 1. The m-th layer of the CvPT is a

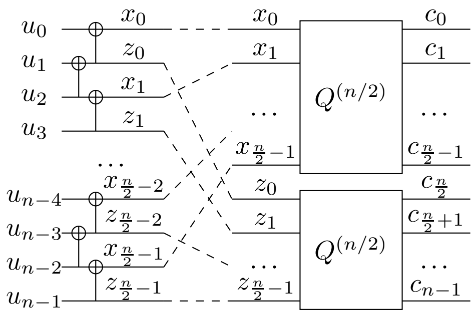

Fig. 1: Convolutional polarizing transformation $Q^{(n)}$

mapping of vector u_0^{n-1} to vectors $x_0^{n/2-1} = u_0^{n-1} X^{(n)}$ and $z_0^{n/2-1} = u_0^{n-1} Z^{(n)}$, where

$$x_i = u_{2i} + u_{2i+1} + u_{2i+2}, z_i = u_{2i+1} + u_{2i+2}, i \leq \frac{n}{2} - 2$$
$$x_{n/2-1} = u_{n-2} + u_{n-1}, z_{n/2-1} = u_{n-1}. \quad (4)$$

A CvPC is defined as a set of codewords

$$\left\{ u_0^{n-1} Q^{(n)} | u_{\mathcal{I}} \in \mathbb{F}^k, u_{\mathcal{F}} = \mathbf{0} \right\}, \mathcal{I} \subseteq [n], |\mathcal{I}| = k, \quad (5)$$

$\mathcal{I}$ is called the information set and $\mathcal{F} = [n] \setminus \mathcal{I}$ is called the frozen set.

C. Successive Cancellation Decoding and a Lower Bound on Minimum Distance

Consider transmission of codeword $c_0^{n-1} = \overline{u}_0^{n-1} Q^{(n)}$ through a binary-input memoryless channel $\mathcal{W} : \mathbb{F} \to \mathcal{Y}$. Let y_0^{n-1} be the output of this channel. After demodulation, the probabilities $W(c_i|y_i) = \mathcal{W}(y_i|c_i) / (\mathcal{W}(y_i|0) + \mathcal{W}(y_i|1))$ for $c_i \in \mathbb{F}$ are provided to the decoding algorithm. Given the prior hard decisions $\hat{u}_0 \ldots \hat{u}_{\varphi-1}$, at phase φ the SC decoding algorithm calculates probabilities $W_n^{(\varphi)}(\hat{u}_0^{\varphi-1}.u_\varphi | y_0^{n-1})$, defined as

$$W_n^{(\varphi)}(u_0^{\varphi} | y_0^{n-1}) = \sum_{u_{\varphi+1}^{n-1} \in \mathbb{F}^{n-\varphi-1}} W^n(u_0^{n-1} Q^{(n)} | y_0^{n-1}), \quad (6)$$

where $W^n(c_0^{n-1} | y_0^{n-1}) = \prod_{i=0}^{n-1} W(c_i|y_i)$. Then, the hard decision on v_φ is made by

$$\hat{u}_\varphi = \begin{cases} 0, & \varphi \in \mathcal{F} \\ \arg\max_{u_\varphi \in \mathbb{F}} W_n^{(\varphi)}(\hat{u}_0^{\varphi-1}.u_\varphi | y_0^{n-1}), & \varphi \notin \mathcal{F}. \end{cases}$$

Assume that the SC decoder knows exactly the values of $\overline{u}_0^{\varphi-1}$, $\varphi \in \mathcal{I}$. For the sake of simplicity, assume that $\overline{u}_0^{\varphi-1} = \mathbf{0}^\varphi$. For $a \in \mathbb{F}$, denote the coset of the code, generated by the last $n - \varphi - 1$ rows of $Q^{(n)}$, by

$$\mathcal{C}_n^{(\varphi)}(a) = \left\{ (\mathbf{0}^\varphi.a.u_{\varphi+1}^{n-1}) Q^{(n)} | u_{\varphi+1}^{n-1} \in \mathbb{F}^{n-1-\varphi} \right\}. \quad (7)$$

Observe that in (6) SC decoder computes probabilities of $\mathcal{C}_n^{(\varphi)}(a)$ for $a \in \mathbb{F}$.

Define *the φ-th coset weight* as

$$d_n^{(\varphi)} = \min_{c_0^{n-1} \in \mathcal{C}_n^{(\varphi)}(1)} \mathbf{wt}(c_0^{n-1}). \quad (8)$$

It is known (see [5]), that if d is the minimum distance of a CvPC (5), then

$$d \geq \min_{\varphi \in \mathcal{I}} d_n^{(\varphi)}. \quad (9)$$

The algorithm which computes the φ-th coset weight $d_n^{(\varphi)}$ was proposed in [5]. In this paper we simplify the algorithm and prove its correctness.

III. Computing the φ-th Coset Weight

A. Recursive Expressions for the φ-th Coset Weight

In the case of CvPCs, one cannot obtain recursive expressions for $W_n^{(\varphi)}(\hat{u}_0^{\varphi-1}.u_\varphi | y_0^{n-1})$ in closed form. However, recursive expressions for values of

$$W_n^{(\varphi+2)}(\hat{u}_0^{\varphi-1}.u_\varphi^{\varphi+2} | y_0^{n-1}) = \sum_{u_{\varphi+3}^{n-1} \in \mathbb{F}^{n-\varphi-3}} W^n\left((\hat{u}_0^{\varphi-1}.u_\varphi^{n-1}) Q^{(n)} | y_0^{n-1} \right), \quad (10)$$

are available [4], so at each layer of a CvPT, values of $W_n^{(\varphi+2)}(\hat{u}_0^{\varphi-1}.u_\varphi^{\varphi+2} | y_0^{n-1})$ are computed for each value of $u_\varphi^{\varphi+2}$. After that, one can compute probabilities of values of u_φ by

$$W_n^{(\varphi)}(\hat{u}_0^{\varphi-1}.u_\varphi | y_0^{n-1}) = \sum_{u_{\varphi+1}^{\varphi+2} \in \mathbb{F}^2} W_n^{(\varphi+2)}(\hat{u}_0^{\varphi-1}.u_\varphi^{\varphi+2} | y_0^{n-1}).$$

Let us generalize definition (7) according to (10). For $t < n - \varphi$ and $a_0^{t-1} \in \mathbb{F}^t$ denote

$$\mathcal{C}_n^{(\varphi)}(a_0^{t-1}) = \left\{ (\mathbf{0}^\varphi.a_0^{t-1}.u_{\varphi+t}^{n-1}) Q^{(n)} | u_{\varphi+t}^{n-1} \in \mathbb{F}^{n-\varphi-t} \right\}. \quad (11)$$

Denote also

$$d_n^{(\varphi)}(a_0^{t-1}) = \min_{c_0^{n-1} \in \mathcal{C}_n^{(\varphi)}(a_0^{t-1})} \mathbf{wt}(c_0^{n-1}) = \min_{u_{\varphi+t}^{n-1} \in \mathbb{F}^{n-\varphi-t}} \mathbf{wt}\left[(\mathbf{0}^\varphi.a_0^{t-1}.u_{\varphi+t}^{n-1}) Q^{(n)} \right]. \quad (12)$$

Observe that for any $t > 0$:

$$d_n^{(\varphi)} = \min_{u_{\varphi+1}^{n-1} \in \mathbb{F}^{n-\varphi-1}} \mathbf{wt}\left[(\mathbf{0}^\varphi.1.u_{\varphi+1}^{n-1}) Q^{(n)} \right]$$
$$= \min_{a_0^{t-2} \in \mathbb{F}^{t-1}} \min_{u_{\varphi+t}^{n-1} \in \mathbb{F}^{t-1}} \mathbf{wt}\left[(\mathbf{0}^\varphi.1.a_0^{t-2}.u_{\varphi+t}^{n-1}) Q^{(n)} \right] \quad (13)$$
$$= \min_{a_0^{t-2} \in \mathbb{F}^{t-1}} d_n^{(\varphi)}(1.a_0^{t-2}). \quad (14)$$

The structure of CvPT $Q^{(n)}$ allows one to express $d_n^{(\varphi)}(a_0^2)$ through some $d_{n/2}^{(\psi)}(b_0^2)$. The recursive expressions are given by the following theorem.

Theorem 1.

$$d_n^{(0)}(a_0^2) = \min_{v\in\mathbb{F}} \left(d_{n/2}^{(0)}(\Sigma a_0^2.a_2+v) + d_{n/2}^{(0)}(a_1+a_2.v) \right) \quad (15)$$

$$d_n^{(2\psi+1)}(a_0^2) = \min_{v,w\in\mathbb{F}} \Big(d_{n/2}^{(\psi)}(a_0 + a_1.a_1 + a_2 + v.v + w) + d_{n/2}^{(\psi)}(a_0+a_1.a_2 + v.w) \Big) \quad (16)$$

$$d_n^{(2\psi+2)}(a_0^2) = \min_{v\in\mathbb{F}} \Big(d_{n/2}^{(\psi)}(a_0.a_0 + a_1 + a_2.a_2 + v) + d_{n/2}^{(\psi)}(a_0.a_1 + a_2.v) \Big) \quad (17)$$

$$d_n^{(n-3)}(a_0^2) = d_{n/2}^{(n/2-3)}(0.a_0 + a_1.a_1 + a_2) + d_{n/2}^{(n/2-3)}(0.a_0 + a_1.a_2), \quad (18)$$

where we use shortcut $\Sigma a_b^c = \sum_{i=b}^{c} a_i$ for the sum of vector elements.

Proof. Let us prove (15). Recall that $u_0^{n-1}Q^{(n)} = x_0^{n/2-1}Q^{(n/2)}.z_0^{n/2-1}Q^{(n/2)}$, where x_i and z_i are defined by (4). One can expand the definition (11) as

$$\mathcal{C}_n^{(0)}(a_0^2) = \left\{ (a_0^2.u_3^{n-1})Q^{(n)} | u_3^{n-1} \in \mathbb{F}^{n-3} \right\} = \Big\{ (\Sigma a_0^2, a_2+u_3+u_4, \Sigma u_4^6, \ldots, \Sigma u_{n-2}^{n-1})Q^{(n/2)}. (\Sigma a_1^2, \Sigma u_3^4, \Sigma u_5^6, ..., u_{n-1})Q^{(n/2)} | u_3^{n-1} \in \mathbb{F}^{n-3} \Big\} =$$
$$\Big\{ (\Sigma a_0^2, a_2+v, \Sigma u_4^6, \ldots, \Sigma u_{n-2}^{n-1})Q^{(n/2)}. (\Sigma a_1^2, v, \Sigma u_5^6, ..., u_{n-1})Q^{(n/2)} | v \in \mathbb{F}, u_4^{n-1} \in \mathbb{F}^{n-4} \Big\} \overset{(1)}{=}$$
$$\Big\{ (\Sigma a_0^2, a_2+v, x_2^{n/2-1})Q^{(n/2)}.(a_1+a_2, v, z_2^{n/2-1})Q^{(n/2)} | v \in \mathbb{F}, x_2^{n/2-1}, z_2^{n/2-2} \in \mathbb{F}^{n/2-2} \Big\}$$
$$= \bigcup_{v\in\mathbb{F}} \mathcal{C}_{n/2}^{(0)}(\Sigma a_0^2, a_2+v).\mathcal{C}_{n/2}^{(0)}(a_1+a_2, v), \quad (19)$$

Equality $\overset{(1)}{=}$ is given by the fact that the matrix $A = \left(X^{(n)}_{\overline{[4]},\overline{[2]}} . Z^{(n)}_{\overline{[4]},\overline{[2]}} \right)$ is non-singular and $x_2^{n/2-1}.z_2^{n/2-1} = u_4^{n-1}A$, so running over all possible values of u_4^{n-1} is equivalent to running over all pairs $x_2^{n/2-1}, z_2^{n/2-1}$. Also we used shortcut $v = u_3 + u_4$. Taking minimum from left and right side of (19), one obtains (15).

Let us prove (16). Note that if $u_0^{2\psi} = \mathbf{0}$, then $x_0^{\psi-1} = z_0^{\psi-1} = \mathbf{0}$. Similarly to (19), one can obtain

$$\mathcal{C}_n^{(2\psi+1)}(a_0^2) = \left\{ (0_0^{2\psi}.a_0^2.u_{2\psi+4}^{n-1})Q^{(n)} | u_{2\psi+4}^{n-1} \in \mathbb{F}^{n-2\psi-4} \right\}$$
$$= \Big\{ (0_0^{\psi-1}.\Sigma a_0^1.\Sigma a_1^2 + u_{2\psi+4}.\Sigma u_{2\psi+4}^{2\psi+6} \ldots \Sigma u_{n-2}^{n-1})Q^{(n/2)}. (0_0^{\psi-1}.\Sigma a_0^1, a_2 + u_{2\psi+4}.\Sigma u_{2\psi+5}^{2\psi+6} \ldots u_{n-1})Q^{(n/2)} \Big\}_{u_{2\psi+4}^{n-1} \in \mathbb{F}^{n-2\psi-4}}$$
$$\overset{(2)}{=} \Big\{ (0_0^{\psi-1}.\Sigma a_0^1.\Sigma a_1^2 + v.v + w.x_{\psi+3}^{n/2-1})Q^{(n/2)}. (0_0^{\psi-1}.\Sigma a_0^1, a_2 + v.w.x_{\psi+3})Q^{(n/2)} \Big\}_{v,w\in\mathbb{F}, x_{\psi+3}^{\frac{n}{2}-1}, z_{\psi+3}^{\frac{n}{2}-1} \in \mathbb{F}^{\frac{n}{2}-\psi-3}}$$
$$= \bigcup_{v,w\in\mathbb{F}} \mathcal{C}_{n/2}^{(\psi)}(\Sigma a_0^1.\Sigma a_1^2 + v.v + w).\mathcal{C}_{n/2}^{(\psi)}(\Sigma a_0^1, a_2 + v.w), \quad (20)$$

where $\overset{(2)}{=}$ follows from the fact that matrix $B = \left(X^{(n)}_{\overline{[2\psi+6]},\overline{[\psi+3]}} . Z^{(n)}_{\overline{[2\psi+6]},\overline{[\psi+3]}} \right)$ is non-singular and $x_{\psi+3}^{n/2-1}.z_{\psi+3}^{n/2-1} = u_{2\psi+6}^{n-1}B$, so one can replace running over $u_{2\psi+4}^{n-1}$ by running over $v = u_{2\psi+4}$, $w = \Sigma u_{2\psi+5}^{2\psi+6}$, and all values of pairs $x_{\psi+3}^{n/2-1}$, $z_{\psi+3}^{n/2-1}$. Equality (20) implies (16) by taking minima of sets at the left and the right side of (20).

In a similar manner one can prove (17), and (18) simply follows from (1)–(3) by the definition of $d_n^{(i)}(a_0^{t-1})$. □

Corollary 1. *For any $n = 2^m \geq 4$, the values of $d_n^{(0)}(a_0^2)$ are given by*

$$d_n^{(0)}(\mathbf{0}^3) = 0 \quad (21)$$
$$\forall a_1^2 \neq \mathbf{0} : d_n^{(0)}(0.a_1^2) = 2 \quad (22)$$
$$\forall a_1^2 \in \mathbb{F}^2 : d_n^{(0)}(1.a_1^2) = 1 \quad (23)$$

Proof. For $n = 4$ the corollary is obvious from

$$Q^{(4)} = \begin{pmatrix} 1000 \\ 1100 \\ 0110 \\ 1111 \end{pmatrix}. \quad (24)$$

Assume that the corollary is true for size of CvPT $n/2$. From (12) one can see that $d_{n/2}^{(0)}(\mathbf{0}^2) = \min_a d_{n/2}^{(0)}(\mathbf{0}^2, a)$, so the assumption implies

$$d_{n/2}^{(0)}(0.0) = 0, d_{n/2}^{(0)}(0.1) = d_{n/2}^{(0)}(1.1) = 2, d_{n/2}^{(0)}(1.0) = 1. \quad (25)$$

Applying (15) to (25) one can obtain (21)–(23). □

Corollary 2. *For $n = 2^m \geq 4$, the values of $d_n^{(n-3)}(a_0^2)$ are*

$$d_n^{(n-3)}(\mathbf{0}^3) = 0 \quad (26)$$
$$d_n^{(n-3)}(\mathbf{0}^2.1) = n \quad (27)$$
$$\forall a_0^2 \notin \left\{ \mathbf{0}^3, \mathbf{0}^2.1 \right\} : d_n^{(0)}(a_0^2) = \frac{n}{2} \quad (28)$$

Proof. The corollary is true for $n = 4$ (see (24)). For $n > 4$ the corollary can be obtained by induction on n via (18). □

B. The Algorithm of Computing $d_n^{(\varphi)}$

The top-level function which computes $d_n^{(\varphi)}$ is shown in Alg. 1. The auxiliary functions `C0`, `Cn_3`, `Codd`, `Ceven` are given in Alg. 2, which return $d_n^{(0)}(a_0^2)$, $d_n^{(n-3)}(a_0^2)$, $d_n^{(2\psi+1)}(a_0^2)$, $d_n^{(2\psi+2)}(a_0^2)$, respectively. Inside these functions indexing of arrays of 8 integers is performed as follows: for array A of 8 elements, symbol $A[a_0, a_1, a_2]$ denotes $A_{I(a_0^2)}$, where

$$I(a_0^2) = a_0 + 2a_1 + 4a_2.$$

Function `C0` returns $d_n^{(0)}(a_0^2)$ by Corollary 1. Function `Cn_3(n)` returns $d_n^{(n-3)}(a_0^2)$ by Corollary 2.

Functions `Codd(`$\widetilde{D}_0^7$`)` and `Ceven(`$\widetilde{D}_0^7$`)` take as input an array of values of $d_{n/2}^{(\psi)}(a_0^2)$ for all $a_0^2 \in \mathbb{F}^3$, and return arrays

Algorithm 1: Algorithm for computing $d_n^{(\varphi)}$

Input: m

1.1 $\widetilde{D}, D \leftarrow$ new arrays of $8 \cdot 2^m$ integers

1.2 $\widetilde{D}_0^7 \leftarrow$`C0()`, $D_0^7 \leftarrow$`C0()`, $D_8^{15} \leftarrow$ `Cn_3(4)`

1.3 **for** $t = 3 \ldots m$ **do**

1.4 $\quad n \leftarrow 2^t$

1.5 $\quad$ swap D and $\widetilde{D}$

1.6 $\quad$ **for** $\psi = 0 \ldots n/2 - 3$ **do**

1.7 $\quad\quad \Psi \leftarrow 8\psi$

1.8 $\quad\quad D_{16\psi+8}^{16\psi+15} \leftarrow$ `Codd(`$\widetilde{D}_{8\psi}^{8\psi+7}$`)`

1.9 $\quad\quad D_{16\psi+16}^{16\psi+23} \leftarrow$ `Ceven (`$\widetilde{D}_{8\psi}^{8\psi+7}$`)`

1.10 $\quad D_{(n-3)\cdot 8}^{(n-3)\cdot 8+7} \leftarrow$ `Cn_3(`n`)`

1.11 $\widetilde{D}_0 \leftarrow 1,\ \widetilde{D}_1 \leftarrow 2$

1.12 **for** $i = 0 \ldots 2^m - 3$ **do** $\widetilde{D}_{i+2} \leftarrow D_{8i+4}$

1.13 **return** $\widetilde{D}_0^{n-1}$

Algorithm 2: Auxiliary functions for computing $d_n^{(\varphi)}$

2.1 **Function** `C0()`

2.2 $\quad$ **return** $[0, 1, 2, 1, 2, 1, 2, 1]$

2.3 **Function** `Codd(`$\widetilde{D}_0^7$`)`

2.4 $\quad$ **for** $a_0^2 \in \mathbb{F}^3$ **do**

2.5 $\quad\quad D[a_0^2] \leftarrow \min_{v,w\in\mathbb{F}} \widetilde{D}[a_0 + a_1, a_1 + a_2 + v, v + w] + \widetilde{D}[a_0 + a_1, a_2 + v, w]$

2.6 $\quad$ **return** D_0^7

2.7 **Function** `Ceven(`$\widetilde{D}_0^7$`)`

2.8 $\quad$ **for** $a_0^2 \in \mathbb{F}^3$ **do**

2.9 $\quad\quad D[a_0^2] \leftarrow \min_{v\in\mathbb{F}} \widetilde{D}[a_0, a_0+a_1+a_2, a_2+v] + \widetilde{D}[a_0, a_1 + a_2, v]$

2.10 $\quad$ **return** D_0^7

2.11 **Function** `Cn_3(`n`)`

2.12 $\quad$ **return** $[0, \frac{n}{2}, \frac{n}{2}, \frac{n}{2}, n, \frac{n}{2}, \frac{n}{2}, \frac{n}{2}]$

of $d_n^{(2\psi+1)}(a_0^2)$ and $d_{n/2}^{(2\psi+2)}(a_0^2)$ for all $a_0^2 \in \mathbb{F}^3$, respectively, by Theorem 1.

In the top-level function in Alg. 1, the value of $d_n^{(\psi)}(a_0^2)$ is stored in $D_{8\psi+I(a_0^2)}$. Array $\widetilde{D}$ stores the content of D on the previous iteration of the loop in lines 1.3–1.10, which we call the main loop. Values of D_0^7 and $\widetilde{D}_0^7$ are initialized with values of $d_4^{(0)}(a_0^2)$ for $a_0^2 \in \mathbb{F}^3$ before the main loop. Both D_0^7 and $\widetilde{D}_0^7$ are never rewritten in the main loop, because value of $d_n^{(0)}(a_0^2)$ does not depend on n, see Corollary 1. Values of D_0^7 are initialized to the values of $d_4^{(1)}(a_0^2)$ by Corollary 2.

In the main loop values of $d_n^{(\varphi)}(a_0^2)$ are computed by Theorem 1 and stored in array D. The values of $d_{n/2}^{(\psi)}(a_0^2)$ are in array $\widetilde{D}$. At the beginning of the iteration D and $\widetilde{D}$ are swapped (line 1.3), so that D on the previous iteration becomes $\widetilde{D}$ on the current iteration.

After the last iteration of the main loop, array D contains values of $d_{2^m}^{(i)}(a_0^2)$. Array of $\widetilde{D}$ is reused for computing $d_{2^m}^{(i)}$ by

$$d_{2^m}^{(i+2)} = d_{2^m}^{(i)}(\mathbf{0}^2.1)$$

in line 1.12. By Corollary 1 one can see that $d_{2^m}^{(0)} = 1$, $d_{2^m}^{(1)} = 2$, which is reflected in line 1.11.

Example 1. For the case of $n = 2^4$, the sequence $d_{16}^{(\varphi)}$, $\varphi \in [16]$ is $(1, 2, 2, 2, 4, 2, 4, 4, 6, 4, 8, 4, 8, 8, 8, 16)$.

The complexity of the proposed algorithm is dominated by the complexity of the main loop, which has $\sum_{t=3}^{m} 2^{t-1} \approx 2^m$ calls of procedures `Codd` and `Ceven`. The complexity of procedure `Codd` is $8 \cdot 7 = 56$ operations, and the complexity of procedure `Ceven` is $8 \cdot 3 = 24$ operations. Thus, the total complexity of the proposed algorithm is $\approx 80 \cdot 2^m$ operations. The complexity of the original algorithm [5] is approximately $520 \cdot 2^m$ operations, which is 6.5 times higher.

IV. Convolutional Polar Subcodes

After computing $d_n^{(\varphi)}$ for $\varphi \in [n]$ one can obtain a lower bound on minimum distance of a convolutional polar code of length n with frozen set $\mathcal{F}$ by (9).

As well as polar codes, CvPCs have low minimum distance, and their finite-length performance under list SC decoding is not good, especially in the high-SNR region, where ML decoding error probability is almost equal to the SC decoding error probability, so employing list SC decoding does not provide any performance gain. Addressing this problem, convolutional polar subcodes (CvPSs) were proposed in [5].

List SC decoding with sufficiently large list size can provide near-ML decoding performance. In high-SNR region, ML decoding performance is influenced mostly by the minimum distance and the number of minimum-weight codewords.

CvPSs are a generalization of convolutional polar codes. A CvPS is defined as a set of vectors

$$\left\{ u_0^{n-1} Q^{(n)} | u_{\mathcal{S}} = 0, u_{\mathcal{I}} \in \mathbb{F}^k, u_i = \sum_{j\in[i]\cap\mathcal{I}} u_j V_{i,j}, i \in \mathcal{D} \right\}, \tag{29}$$

where coefficients $V_{i,j} \in \mathbb{F}$, $\mathcal{S}$ is the static frozen set, $\mathcal{D} = [n]\backslash(\mathcal{S}\cup\mathcal{I})$ is the set of dynamic frozen symbols. Set $\mathcal{S}$ should be chosen as the set of least reliable symbols. Denote the CvPC with the frozen set $\mathcal{S}$ by $\mathcal{C}_0$. Let the minimum distance of $\mathcal{C}_0$ be d. The question is, how to choose $\mathcal{D} \subset [n] \setminus \mathcal{S}$, i.e., how to choose subcode $\mathcal{C} \subset \mathcal{C}_0$ of given dimension, such that the number of weight-d codewords in $\mathcal{C}$ is small?

If one chooses the set of dynamic frozen symbols $\mathcal{D}$ as the largest elements of $[n] \setminus \mathcal{S}$ and coefficients $V_{i,j}$ are uniformly distributed over $\mathbb{F}$, one obtains a random uniformly distributed subcode of $\mathcal{C}_0$. Such subcode has in average $E/2^{|\mathcal{D}|}$ codewords of weight d, where E is the number of minimum-weight codewords of $\mathcal{C}_0$. This can be a good strategy for ML-decoding performance, since random codes are good, but is not suitable for the list SC decoding, since the list SC decoder can lose the

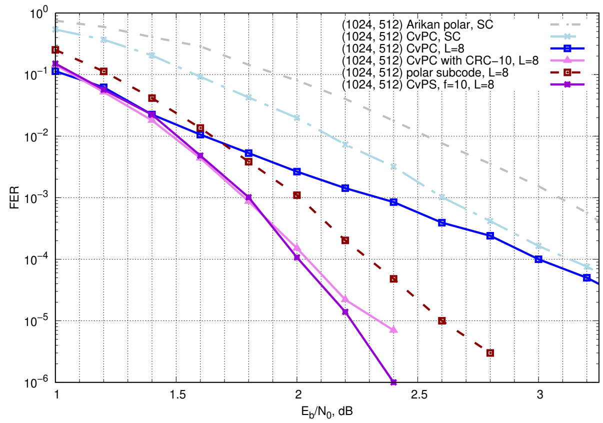

Fig. 2: Performance of $(1024, 512)$ CvPS with $|\mathcal{D}| = 10$

correct path before processing the dynamic frozen symbols, which would be estimated at the end of decoding process.

Observe that each weight-d codeword from $\mathcal{C}_0$ corresponds to input vector u_0^{n-1} with at least one non-zero element u_i, s.t. $d_n^{(i)} \leq d$. In other words, only u_i with the minimum $d_n^{(i)}$ are responsible for generating minimum-weight codewords. Thus, one can choose set $\mathcal{D}$ so that the dynamic freezing constraints cover all such symbols. It can be achieved by choosing set $\mathcal{D}$ as a set of highest $i \notin \mathcal{S}$ with the lowest $d_n^{(i)}$.

So, the construction of a convolutional polar subcode can be done as follows:

1) Choose the static frozen set $\mathcal{S}$ as a set of $n-k-f$ least reliable symbols, which can be obtained by Monte-Carlo simulations or via recursive expressions (see [3]) for the case of BEC channel.
2) Choose the set of dynamic frozen symbols $\mathcal{D}$ as f maximum indices i, corresponding to the minimum value of $d_n^{(i)}$. Assign $u_i = \sum_{j \in [i] \cap \mathcal{I}} V_{i,j} u_j$, where $V_{i,j}$ are chosen randomly uniformly distributed over $\mathbb{F}$.
3) The set of information symbols is $\mathcal{I} = [n] \setminus (\mathcal{S} \cup \mathcal{D})$.

The optimal value of f can be obtained by simulations.

The algorithm, proposed in this paper, enables one to compute $d_n^{(i)}$ for step 2 with reduced complexity compared to the originally published algorithm [5].

Fig. 2 presents the performance of a $(1024, 512)$ CvPS with $|\mathcal{D}| = 10$, a polar code and a randomized polar subcode [9] in AWGN channel. The polar code and the polar subcode are constructed for AWGN channel with $E_b/N_0 = 2$ dB using Gaussian approximation of density evolution [10], and the CvPC and the CvPS are constructed for the same channel using Monte-Carlo simulations for subchannels qualities. One can see that the CvPS outperforms the polar subcode, the CvPC and the CvPC concatenated with CRC-10. More detailed complexity and performance comparison can be found in [5].

V. Conclusions

A novel algorithm for computing minimum weight of cosets of codes, which are generated by last rows of convolutional polarizing transformation, is given. It has lower complexity, both conceptual and computational, compared to the originally proposed algorithm. The obtained algorithm may be used to simplify computation of a lower bound on minimum distance and reduce the complexity of construction of convolutional polar subcodes.

References

[1] E. Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, 55(7):3051–3073, July 2009.

[2] Andrew J. Ferris and David Poulin. Branching MERA codes: a natural extension of polar codes. *CoRR*, abs/1312.4575, 2013.

[3] Andrew James Ferris, Christoph Hirche, and David Poulin. Convolutional polar codes. *CoRR*, abs/1704.00715, 2017.

[4] R. Morozov and P. Trifonov. Efficient SC decoding of convolutional polar codes. In *Proceedings of International Symposium on Information Theory and Applications*, pages 442–446, Singapore, Singapore, 2018. IEEE.

[5] R. Morozov and P. Trifonov. On distance properties of convolutional polar codes. *IEEE Transactions on Communications*, 67(7):4585–4592, July 2019.

[6] Tobias Prinz and Peihong Yuan. Successive cancellation list decoding of BMERA codes with application to higher-order modulation. In *2018 International Symposium on Turbo Codes and Iterative Information Processing (ITW)*, December 2018.

[7] H. Saber, Y. Ge, R. Zhang, W. Shi, and W. Tong. Convolutional polar codes: LLR-based successive cancellation decoder and list decoding performance. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 1480–1484, June 2018.

[8] Ido Tal and A. Vardy. List decoding of polar codes. *IEEE Transactions On Information Theory*, 61(5):2213–2226, May 2015.

[9] P. Trifonov and G. Trofimiuk. A randomized construction of polar subcodes. In *Proceedings of IEEE International Symposium on Information Theory*, pages 1863–1867, Aachen, Germany, 2017. IEEE.

[10] Peter Trifonov. Efficient design and decoding of polar codes. *IEEE Transactions on Communications*, 60(11):3221 – 3227, November 2012.