Construction of polarization kernels of size 16 for low complexity processing

Grigorii Trofimiuk, Peter Trifonov
ITMO University, Russia
Email: {gtrofimiuk,pvtrifonov}@corp.ifmo.ru

***Abstract*—An algorithm for construction of binary 16×16 polarization kernels with polarization rate 0.51828 which admit low complexity processing is proposed. The considered processing algorithm exploits linear relationship of the considered kernels and Arikan transform. The proposed approach relies on restricted application of elementary row operations to Arikan transform matrix, which are chosen to have minimal impact on complexity of the window processing algorithm.**

The proposed construction resulted in relatively low number of kernels, which can be easily checked by computer-based search. Moreover, simulation results show that polar (sub)codes with obtained kernels can outperform polar codes with Arikan kernel, while having lower decoding complexity.

I. Introduction

Polar codes are a novel class of error-correcting codes, which achieve the symmetric capacity of a binary-input discrete memoryless channel W, have low complexity construction, encoding and decoding algorithms [1]. However, the performance of polar codes of practical length is quite poor. The reasons for this are the presence of imperfectly polarized subchannels and the suboptimality of the successive cancellation (SC) decoding algorithm. To improve performance, successive cancellation list decoding (SCL) algorithm [2], as well as various code constructions were proposed [3], [4], [5].

Polarization is a general phenomenon, and is not restricted to the case of Arikan matrix [6]. One can replace it by a larger matrix, called *polarization kernel*, which can provide higher polarization rate. Polar codes with large kernels were shown to provide asymptotically optimal scaling exponent [7]. Many kernels with various properties were proposed [6], [8], [9], [10]. Until recently, polar codes with large kernels were believed to be impractical due to very high decoding complexity.

The window processing algorithm for some 16×16 polarization kernels was introduced in [11]. This approach exploits the relationship between the considered kernels and the Arikan matrix. Essentially, the log-likelihood ratios (LLRs) for the input symbols of the considered kernels are obtained from the LLRs computed via the Arikan recursive expressions.

In this paper we present a construction method for polarization kernels, which admit efficient decoding by window based approach. The proposed method construct a class of polarization kernels, which are expected to be suitable for given processing method. The kernels are constructed by performing elementary row operations over Arikan transform matrix.

We show that with obtained kernels increasing list size in the SCL decoder provides much more significant performance gain compared to the case of Arikan kernel, and ultimately the proposed approach results in lower decoding complexity compared to the case of polar codes with Arikan kernel with the same performance.

II. Background

A. Channel polarization

Consider a binary-input memoryless channel with transition probabilities $W\{y|c\}, c \in \mathbb{F}_2, y \in \mathcal{Y}$, where $\mathcal{Y}$ is output alphabet. For a positive integer n, denote by $[n]$ the set of n integers $\{0, 1, \dots n-1\}$. A *polarization kernel* K is a binary invertible $l \times l$ matrix, which is not upper-triangular under any column permutation. The Arikan kernel is given by

$$F_m = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}^{\otimes m},$$

where $\otimes m$ is m-fold Kronecker product of matrix with itself.

An $(n = l^m, k)$ polar code is a linear block code generated by k rows of matrix $G_m = M^{(m)} K^{\otimes m}$, where $M^{(m)}$ is a digit-reversal permutation matrix, corresponding to mapping $\sum_{i=0}^{m-1} t_i l^i \rightarrow \sum_{i=0}^{m-1} t_{m-1-i} l^i, t_i \in [l]$. The encoding scheme is given by $c_0^{n-1} = u_0^{n-1} G_m$, where $u_i, i \in \mathcal{F}$ are set to some pre-defined values, e.g. zero (frozen symbols), $|\mathcal{F}| = n - k$, and the remaining values u_i are set to the payload data.

It is possible to show that a binary input memoryless channel W together with matrix G_m gives rise to bit subchannels $W_{m,K}^{(i)}(y_0^{n-1}, u_0^{i-1}|u_i)$ with capacities approaching 0 or 1, and fraction of noiseless subchannels approaching $I(W)$ [6]. Selecting $\mathcal{F}$ as the set of indices of low-capacity subchannels enables almost error-free communication. It is convenient to define probabilities

$$W_{m,K}^{(i)}(u_0^i|y_0^{n-1}) = \frac{W_{m,K}^{(i)}(y_0^{n-1}, u_0^{i-1}|u_i)}{2W(y_0^{n-1})} = \sum_{u_{i+1}^{n-1}} \prod_{i=0}^{n-1} W((u_0^{n-1} G_m)_i | y_i). \quad (1)$$

Let us further define $\mathbf{W}_m^{(j)}(u_0^j|y_0^{n-1}) = W_{m,K}^{(j)}(u_0^j|y_0^{n-1})$, where kernel K will be clear from the context. We also need

probabilities $W_t^{(j)}(u_0^j|y_0^{l-1}) = W_{1,F_t}^{(j)}(u_0^j|y_0^{l-1})$ for Arikan matrix F_t. Due to the recursive structure of G_m, one has

$$\mathbf{W}_m^{(sl+t)}(u_0^{sl+t}|y_0^{n-1}) = \sum_{u_{sl+t+1}^{l(s+1)-1}} \prod_{j=0}^{l-1} \mathbf{W}_{m-1}^{(s)}(\theta_K[u_0^{l(s+1)-1}, j]|y_{j\frac{n}{l}}^{(j+1)\frac{n}{l}-1}) \quad (2)$$

where $\theta_K[u_0^{(s+1)l-1}, j]_r = (u_{lr}^{l(r+1)-1}G_m)_j, r \in [s+1]$. A trellis-based algorithm for computing these values was presented in [12].

At the receiver side, one can successively estimate

$$\widehat{u}_i = \begin{cases} \arg\max_{u_i \in \mathbb{F}_2} \mathbf{W}_m^{(i)}(\widehat{u}_0^{i-1}.u_i|y_0^{n-1}), & i \notin \mathcal{F}, \\ \text{the frozen value of } u_i & i \in \mathcal{F}. \end{cases} \quad (3)$$

This is known as the successive cancellation (SC) decoding algorithm.

B. Rate of polarization

Let $W : \{0,1\} \to \mathcal{Y}$ be a symmetric binary-input discrete memoryless channel (B-DMC) with capacity $I(W)$. By definition,

$$I(W) = \sum_{y\in\mathcal{Y}} \sum_{x\in\{0,1\}} \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)}.$$

Also, let $Z(W) \in [0,1]$ denote the Bhattacharyya parameter of W, i.e., $Z(W) = \sum_{y\in\mathcal{Y}} \sqrt{W(y|0)W(y|1)}$.

Consider polarizing transform $K^{\otimes m}$, where K is an $l \times l$ polarization kernel, and bit subchannels $W_{m,K}^{(i)}(y_0^{n-1}, u_0^{i-1}|u_i)$, induced by it. Let $Z_m^{(i)} = Z(W_{m,K}^{(i)}(y_0^{n-1}, u_0^{i-1}|u_i))$ be a Bhattacharyya parameter of i-th subchannel, where i is uniformly distributed on the set $[l^m]$. Then, for any B-DMC W with $0 < I(W) < 1$, we will say that an $\ell \times \ell$ matrix K has polarization rate $E(K)$ if [6]

(i) For any fixed $\beta < E(K)$,

$$\liminf_{n\to\infty} \Pr[Z_n \le 2^{-\ell^{n\beta}}] = I(W).$$

(ii) For any fixed $\beta > E(K)$,

$$\liminf_{n\to\infty} \Pr[Z_n \ge 2^{-\ell^{n\beta}}] = 1.$$

That is, the rate of polarization shows how fast bit subchannels of $K^{\otimes m}$ approach neither almost noiseless or noisy channel with $n = l^m$.

Suppose we constructed (n,k) polar code $\mathcal{C}$ with kernel K. Let $P_e(n)$ be a block error probability of $\mathcal{C}$ under transmission over W and decoding by SC algorithm. It was proven [6], that if $n/k < I(W)$ and $\beta < E(K)$, then

$$P_e(n) \le 2^{-n^\beta}$$

It turns out that the rate of polarization is independent of channel W. Namely, let $\langle g_1, g_2, \ldots, g_k\rangle$ be a linear code, generated by vectors $g_1, g_2, \ldots, g_k$. Let $d_H(a,b)$ be the Hamming distance between a and b. Let $d_H(b,\mathcal{C}) = \min_{c\in\mathcal{C}} d_H(b,c)$ be a minimal distance between vector b and linear block code $\mathcal{C}$. We denote the i-th row of an $l \times l$ matrix M as $M[i], i \in [l]$.

The *partial distances* $\mathbb{D}_i, i = 0, \ldots, l-1, l \times l$ of the matrix K are defined as follows:

$$\mathbb{D}_i = d_H(K[i], \langle K[i+1], \ldots, K[l-1]\rangle), i = 0, \ldots, l-2,$$
$$\mathbb{D}_{l-1} = d_H(K[l-1], \mathbf{0}).$$

The vector $\mathbb{D}$ will be referred to as a *partial distances profile*. In work [6] it was shown that for any B-DMC W and any $l \times l$ polarization kernels K with partial distances $\{\mathbb{D}_i\}_{i=0}^{l-1}$, the rate of polarization $E(K)$ is given by

$$E(K) = \frac{1}{l} \sum_{i=0}^{l-1} \log_l \mathbb{D}_i. \quad (4)$$

The Arikan kernel F_1 has rate of polarization $E(F_1) = 0.5$, whereas random codes achieve $E = 1$. For polarization kernels of size 16 and 32 the kernels with rate of polarization 0.51828 and 0.53656 respectively can be obtained.

C. Scaling exponent

Let us fix a B-DMC W of capacity $I(W)$ and a desired block error probability P_e. Given W and P_e, suppose we wish to communicate at rate $I(W) - \Delta$ using a family of (n,k) polar codes with kernel K. It has been shown that this value of n scales as $O(\Delta^{-\mu(K)})$, where the constant $\mu(K)$ is known as the scaling exponent [8].

The scaling exponent depends on channel. Unfortunately, the algorithm of its computing is only known for the case on binary erasure channel (BEC) [13],[8].

The Arikan kernel F_1 has $\mu(K) = 3.627$, whereas random codes achieve optimal $\mu = 2$. The best known scaling exponent for 16×16 polarization kernel is 3.346 [11].

D. Computing kernel input symbols LLRs

1) General case: Our goal is to compute probabilities $\mathbf{W}_m^{(i)}(u_0^i|y_0^{n-1})$ for a given polarization transform $K^{\otimes m}$. Let us assume for the sake of simplicity that $m = 1$. The corresponding task will be referred to as *kernel processing*.

We propose to introduce approximate probabilities

$$\widetilde{\mathbf{W}}_1^{(j)}(u_0^j|y_0^{l-1}) = \max_{u_{j+1}^{l-1}} \mathbf{W}_1^{(l-1)}(u_0^{l-1}|y_0^{l-1}) = \max_{u_{j+1}^{l-1}} \prod_{i=0}^{l-1} W((u_0^{l-1}K)_i|y_i). \quad (5)$$

This is the probability of the most likely continuation of path u_0^j in the code tree, without taking into account possible freezing constraints on symbols $u_i, i > j$. Note that the same probabilities were introduced in [14], [15], and shown to provide substantial reduction of the complexity of sequential decoding of polar codes.

Decoding can be implemented using the log-likelihood ratios $\bar{\mathbf{S}}_{m,i} = \bar{\mathbf{S}}_m^{(i)}(u_0^{i-1}|y_0^{n-1}) = \ln \frac{\mathbf{W}_m^{(i)}(u_0^{i-1}.0|y_0^{n-1})}{\mathbf{W}_m^{(i)}(u_0^{i-1}.1|y_0^{n-1})}$. Hence, kernel output LLRs $\bar{\mathbf{S}}_{1,i}, i \in [l]$ can be approximated by

$$\bar{\mathbf{S}}_{1,i} \approx \mathbf{S}_{1,i} = \ln \frac{\widetilde{\mathbf{W}}_1^{(i)}(u_0^{i-1}.0|y_0^{l-1})}{\widetilde{\mathbf{W}}_1^{(i)}(u_0^{i-1}.1|y_0^{l-1})}$$
$$= \max_{u_{i+1}^{l-1}} \ln \mathbf{W}_1^{(l-1)}(u(0)^i|y_0^{l-1}) - \max_{u_{i+1}^{l-1}} \ln \mathbf{W}_1^{(l-1)}(u(1)^i|y_0^{l-1}), \tag{6}$$

where $u(a)^i = (u_0^{i-1}.a.u_{i+1}^{l-1})$. The above expression means that $\mathbf{S}_{1,i}$ can be computed by performing ML decoding of the code, generated by last $l-i+1$ rows of the kernel K, assuming that all $u_j, i < j < l$, are equiprobable.

2) Window processing: Straightforward evaluation of (6) for arbitrary kernel has complexity $O(2^l l)$. However, we have a simple explicit recursive procedure for computing these values for the case of the Arikan transform F_t.

Let $l = 2^t$. Consider encoding scheme

$$c_0^{l-1} = v_0^{l-1} F_t. \tag{7}$$

Similarly to (5), define approximate probabilities

$$\widetilde{W}_t^{(i)}(v_0^i|y_0^{l-1}) = \max_{v_{i+1}^{l-1}} W_t^{(l-1)}(v_0^{l-1}|y_0^{l-1})$$

and modified log-likelihood ratios

$$S_t^{(i)}(v_0^{i-1}, y_0^{l-1}) = \log \frac{\widetilde{W}_t^{(i)}(v_0^{i-1}.0|y_0^{l-1})}{\widetilde{W}_t^{(i)}(v_0^{i-1}.1|y_0^{l-1})}.$$

It can be seen that

$$S_\lambda^{(2i)}(v_0^{2i-1}, y_0^{N-1}) = \operatorname{sgn}(a)\operatorname{sgn}(b)\min(|a|,|b|) \tag{8}$$
$$S_\lambda^{(2i+1)}(v_0^{2i}, y_0^{N-1}) = (-1)^{v_{2i}} a + b, \tag{9}$$

where $N = 2^\lambda$, $a = S_{\lambda-1}^{(i)}(v_{0,e}^{2i-1} \oplus v_{0,o}^{2i-1}, y_{0,e}^{N-1})$, $b = S_{\lambda-1}^{(i)}(v_{0,o}^{2i-1}, y_{0,o}^{N-1})$. Then the log-likelihood of a path v_0^i can be obtained as [16]

$$R(v_0^i|y_0^{l-1}) = \log \widetilde{W}_t^{(i)}(v_0^i|y_0^{l-1})$$
$$= R(v_0^{i-1}|y_0^{l-1}) + \tau\left(S_t^{(i)}(v_0^{i-1}, y_0^{l-1}), v_i\right), \tag{10}$$

where $R(\epsilon|y_0^{l-1})$ can be set to 0, ϵ is an empty sequence, and

$$\tau(S, v) = \begin{cases} 0, & \operatorname{sgn}(S) = (-1)^v \\ -|S|, & \text{otherwise.} \end{cases}$$

It was suggested in [17] and [11] to express values $\mathbf{W}_1^{(i)}(u_0^i|y_0^{l-1})$ via $W_t^{(j)}(v_0^j|y_0^{l-1})$ for some j. Indeed, $TK = F_t$, where T is an $l \times l$ matrix. Let

$$c_0^{l-1} = v_0^{l-1} F_t = u_0^{l-1} K \Rightarrow u_0^{l-1} = v_0^{l-1} T.$$

Observe, that it is possible to reconstruct u_0^i from $v_0^{\tau_i}$, where τ_i is the position of the last non-zero symbol in the i-th column of T. For the sake of simplicity we assume that all $\tau_i, i \in [l]$ are distinct. The general case is considered in [11].

Indeed, vectors u_0^{l-1} and v_0^{l-1} satisfy the equation

$$u_i = \sum_{j=0}^{l-1} v_j T[j,i], \tag{11}$$

where $T[i,j]$ is a j-th element of row $T[i]$.

Let $h_i = \max_{i' \in [i+1]} \tau_{i'}$. It can be seen that[1]

$$\mathbf{W}_1^{(j)}(u_0^j|y_0^{l-1}) = \sum_{v_0^{h_j} \in \mathcal{Z}_j} W_t^{(h_j)}(v_0^{h_j}|y_0^{l-1})$$
$$= \sum_{v_0^{h_j} \in \mathcal{Z}_j} \sum_{v_{h_j+1}^{l-1}} W_t^{(l-1)}(v_0^{l-1}|y_0^{l-1}), \tag{12}$$

where $\mathcal{Z}_j$ is the set of vectors $v_0^{h_j}$, such that (11) holds for $i \in [j]$. Similarly we can rewrite the above expression for the case of the approximate probabilities

$$\widetilde{\mathbf{W}}_1^{(j)}(u_0^j|y_0^{l-1}) = \max_{v_0^{h_j} \in \mathcal{Z}_j} \widetilde{W}_t^{(h_j)}(v_0^{h_j}|y_0^{l-1})$$
$$= \max_{v_0^{h_j} \in \mathcal{Z}_j} \max_{v_{h_j+1}^{l-1}} W_t^{(l-1)}(v_0^{l-1}|y_0^{l-1}). \tag{13}$$

Let $\mathcal{Z}_{i,b} = \left\{v_0^{h_i} | v_0^{h_i} \in \mathcal{Z}_i, \text{where } u_i = b\right\}$. Hence, one obtains

$$\mathbf{S}_{1,i} = \max_{v_0^{h_i} \in \mathcal{Z}_{i,0}} R(v_0^{h_i}|y_0^{l-1}) - \max_{v_0^{h_i} \in \mathcal{Z}_{i,1}} R(v_0^{h_i}|y_0^{l-1}). \tag{14}$$

Observe that computing these values requires considering multiple vectors $v_0^{h_i}$ of input symbols of the Arikan transform F_t. Let

$$\mathcal{D}_i = [h_i + 1] \backslash \{\tau_0, \tau_1, \ldots, \tau_i\} \tag{15}$$

be a *decoding window*, i.e. the set of indices of independent (from u_0^{i-1}) components of $v_0^{h_i}$. Note that

$$|\mathcal{D}_i| = |[h_i+1]| - |\{\tau_0, \tau_1, \ldots, \tau_i\}| = h_i + 1 - (i+1) = h_i - i$$

since all τ_i are distinct and $\{\tau_0, \tau_1, \ldots, \tau_i\} \subseteq [h_i + 1]$. The calculation of LLRs $\mathbf{S}_{1,i}$ via (14) will be referred to as the *window processing* algorithm.

The number of path scores to be computed in (14), which determines the processing complexity, is equal to $2^{|\mathcal{D}_i|+1}$. Let $\mathcal{M}(K)$ denotes the $\max_{i \in [l]} |\mathcal{D}_i|$. In general, one has $\mathcal{M}(K) = O(l)$ for an arbitrary kernel K.

3) Complexity: The complexity of LLR $\mathbf{S}_{1,i}$ computation via the straightforward implementation of the window processing algorithm (14) consist of the several components. In this work we count the arithmetical complexity as a number summation and comparison operations, which is considered to be equal.

At first, to compute $\mathbf{S}_{1,i}$, one should obtain path scores $R(v_0^{h_i}|y_0^{l-1}), v_0^{h_i} \in \mathcal{Z}_i$. According to the expression (10), the path score $R(v_0^{h_i}|y_0^{l-1})$ is equal to

$$R(v_0^{h_i-1}|y_0^{l-1}) + \tau\left(S_t^{(h_i)}(v_0^{h_i-1}, y_0^{l-1}), v_i\right).$$

If one stores the intermediate results of (8) and (9), then the complexity of computing $S_t^{(h_i)} = S_t^{(h_i)}(v_0^{h_i-1}, y_0^{l-1})$ is

[1]The method given in [10] is a special case of this approach.

given by $2^{B(h_i)}-1$ operations, where $B(h)$ is a position of the last nonzero digit in the binary representation of h, i.e. $h=2^{b_0}+2^{b_1}+\cdots+2^{B(h)}$. If $h=0$ then $B(h)$ is assumed to be t.

Totally, $2^{|\mathcal{D}_i|}$ LLRs $S_t^{(h_i)}$ should be computed. Then, for LLR $S_t^{(h_i)}$ and $v_i \in [1]$ one should calculate the value $\tau\left(S_t^{(h_i)}(v_0^{h_i-1}, y_0^{l-1}), v_i\right)$, which can be done in one summation. In sum, it gives $2^{|\mathcal{D}_i|}$ operations more. Moreover, if $h_i - h_{i-1} > 1$, then the above described computations should be done for LLRs $S_t^{(h)}$, $h_{i-1} < h \leq h_i$. It can be observed, that the number of such LLRs is given by $2^{|\mathcal{D}_i|-(h_i-h)} = 2^{h-i}$.

In total, the complexity of path scores $R(v_0^{h_i}|y_0^{l-1})$, $v_0^{h_i} \in \mathcal{Z}_i$, calculation is given by

$$\Lambda(i) = \sum_{h_{i-1}+1}^{h_i} (2^{h-i}(2^{B(h)}-1)+2^{h-i}) = \sum_{h_{i-1}+1}^{h_i} 2^{h+B(h)-i}$$

and h_{-1} is assumed to be -1.

To complete the LLR $\mathbf{S}_{1,i}$ computation, the corresponding maximum of path scores should be computed, which requires $2^{|\mathcal{D}_i|+1}$ comparisons.

Note that in the case of $h_i = h_{i-1}$ we assume that all path scores are stored together with corresponding partial maximums, thus, one substraction needed only.

In sum, the complexity of the straightforward implementation of the window processing algorithm for kernel K can be estimated as

$$\Psi(K) = \sum_{i=0}^{l-1} \Phi(i), \tag{16}$$

where

$$\Phi(i) = \begin{cases} 2^{h_i-i+1} + \Lambda(i), & h_i > h_{i-1}, \\ 1, & \text{otherwise.} \end{cases}$$

III. Construction of polarization kernels

Our goal is to construct polarization kernels with polarization rate greater that 0.5, which admit low complexity processing. Such rate of polarization rate can be achieved for kernels of size $l=16$ and $l \geq 23$ [6]. In this work we focus on 16×16 polarization kernels.

The maximum rate of polarization among 16×16 kernels is equal to 0.51828, which can be achieved by the kernel with the partial distances profile

$$\mathbb{D}^{(*)} = [1,2,2,2,2,4,4,4,4,6,6,8,8,8,8,16].$$

There are polarization kernels with partial distance profile which corresponds to some permutation of $\mathbb{D}^{(*)}$, what will be demonstrated later, but the complete list of such permutations is unknown. Therefore, it is convenient to begin our investigation with kernels with monotonic increasing partial distances.

The minimization of the complexity (16) by the exhaustive search among all polarization kernels K of size 16×16 and partial distances $\mathbb{D}^{(*)}$ is intractable. Therefore, we are going to significantly reduce the search space to some restricted class of polarization kernels, which are expected to have moderate $\Psi(K)$.

A. Row permutation

Recall that τ_i is the position of the last non-zero symbol in the i-th column of $T = F_t K^{-1}$, $h_i = \max_{i' \in [i+1]} \tau_{i'}$ and $|\mathcal{D}_i| = h_i - i$.

It can be seen, that the value of $h_i - i$ increases once $\tau_i > i$ appears in T, therefore the heuristic minimization of $\Psi(K)$ can be done with minimization of $|\tau_i - i|, i \in [l]$.

The minimal value of $|\tau_i - i| = 0$ is achieved by Arikan transform F_t. The kernel with partial distances $\mathbb{D}^{(*)}$ can be derived by performing elementary operations over rows space of F_4, since F_4 is invertible.

The partial distance profile of the Arikan transform F_4 is given by

$$\mathbb{D}^{(F_4)} = [1,2,2,4,2,4,4,8,2,4,4,8,4,8,8,16].$$

Hence, we can begin construction procedure with row permutation of the matrix F_4 .

Let P_ρ be a permutation matrix, which corresponds to the permutation

$$\rho = \begin{pmatrix} 0 & 1 & \dots & 14 & 15 \\ \rho(0) & \rho(1) & \dots & \rho(14) & \rho(15) \end{pmatrix}.$$

For convenience, we enumerate elements of ρ from zero unlike standard notation. For brevity we will write ρ as $[\rho(1), \rho(2), \dots, \rho(16)]$. Consider the kernel $K_\rho = P_\rho F_4$, consequently,

$$T = F_4 K_\rho^{-1} = F_4 (P_\rho F_4)^{-1} = P_\rho^T.$$

Thus, $\tau_i, i \in [l]$ are given by $\rho(i)$. Therefore, the processing complexity for K_ρ directly depends on the permutation ρ.

We start our construction with permuted Arikan kernels K_β given by permutations

$$\beta = [0,1,2,4,8,w_0,w_1,w_2,w_3,w_4,w_5,7,11,13,14,15],$$

where w is an arbitrary permutation of the vector $[3,5,6,9,10,12]$. The indices of w are the indices of F_4 rows with Hamming weight 4. The obtained kernels have monotonic partial distance profile

$$\mathbb{D}^{(4)} = [1,2,2,2,2,4,4,4,4,4,4,8,8,8,8,16].$$

For instance, the permutation

$$\sigma = [0,1,2,4,8,3,5,6,9,10,12,7,11,13,14,15]$$

results in the permuted Arikan kernel K_σ with $E(K_\sigma) = 0.5$ and scaling exponent $\mu(K_\sigma) = 3.479$ [10]. It can be observed, that kernel K_σ has the least processing complexity $\psi(K)$ among all permuted F_4 kernels which have the partial distance profile $\mathbb{D}^{(4)}$. The maximal h_i for this kernel is given by 4, which results in relatively low complexity.

B. Row addition

To transform the kernel K_β into the kernel with partial distance $\mathbb{D}^{(*)}$, one should sum rows of K_β. It is proven [8], that addition of row K_i to row K_j with $i > j$ does not change the properties of the kernel K. Thus, we consider row additions with $i < j$ only.

The addition of two rows can also increase the maximal size of the decoding windows. Indeed, let $X_{i,j}$ be an elementary matrix which corresponds to addition of row i to row j. In other words, $X_{i,j}$ is an identity matrix with $X_{i,j}[j,i] = 1$. Then

$$K = X_{i,j}P_\rho F_4 \Rightarrow T = P_\rho^T X_{i,j},$$

which means that the column j has been added to the column i of the matrix $P_\rho^T X_{i,j}$. After row addition in K, $\tau_i = \max(\tau_i, \tau_j)$, which can increase the $\tau_i - i$. It can lead to increasing of the size of the corresponding decoding window. It means that one should use addition matrices $X_{i,j}$ with as small as possible values $|j - i|$.

To keep the processing complexity as small as possible, we suggest to sum only rows $K_\beta[i], i \in \{5, 6, 7, 8, 9, 10\}$ to each other. These rows have a Hamming weight 4. It was shown that sum x of these rows can produce vectors of weight ≥ 6 and, furthermore, there exist several x such as

$$d_H(x, \langle K_\beta[11], K_\beta[12], \ldots, K_\beta[15]\rangle) = 6$$

(see [18] page 429).

C. The construction algorithm

Let $\mathbb{M} = \{3, 5, 6, 9, 10, 12\}$. We propose to minimize the decoding window processing complexity over set $\mathbb{K}$ of 16×16 kernels, which is given by following constraints on kernel K:

- $K[i] = K_\beta[i], i \in \{0, 1, 2, 3, 4, 11, 12, 13, 14, 15\}$,
- $K[9], K[10] \in \mathbb{V}_0$, where $\mathbb{V}_0 = \{c \in \mathcal{C} | d_H(c, \mathbf{0}) = 6\}$, $\mathbf{0}$ is a zero element vector and $\mathcal{C} = \langle\{F_4[i], i \in \mathbb{M}\}\rangle$,
- $K_j \in \mathbb{V}_1$, where $\mathbb{V}_1 = \{F_4[i], i \in \mathbb{M}\}, j \in 5, 6, 7, 8$.

The above construction results in the search space of size

$$|\mathbb{K}| = |\mathbb{V}_0|^2 \cdot |\mathbb{V}_1|^4 = 27^2 \cdot 6^4 = 944784.$$

It is easy to observe, that the proposed construction can produce kernels with partial distances distinct from $\mathbb{D}^{(*)}$ and even to singular matrices. However, one does not need to compute the complete partial distance profile $\mathbb{D}$ for $K \in \mathbb{K}$, because the kernel K can be dropped once its partial distance does not match the $\mathbb{D}^{(*)}$. Of course, there are a lot of possible methods for reduction of $\mathbb{K}$, however, there is no need for them since computer-based search over $\mathbb{K}$ runs in several minutes.

IV. Numeric results

A. Kernel construction

1) Monotonic partial distances: Computer-based search results in set $\mathbb{K}_*$ of 60480 16×16 polarization kernels K with $E(K) = 0.51828$. For each K in $\mathbb{K}_*$ we compute its complexity $\Psi(K)$. Moreover, we also computed the BEC scaling exponent $\mu(K)$ for each kernel. The scaling exponent also affects on the error correction performance, so we write the minimal processing complexity for kernel with different scaling exponent.

$K_1, E = 0.51828, \mu = 3.346$

$$\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&0&1&1&0&1&1&0&0&0&0&0&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1
\end{pmatrix}$$

$K_2, E = 0.51828, \mu = 3.45$

$$\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&1&1&0&1&1&0&0&0&0&0&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1
\end{pmatrix}$$

Fig. 1. Examples of constructed polarization kernels

TABLE I
Scaling exponents of kernels from $\mathbb{K}_*$

$\mu(K)$	3.346	3.353	3.356	3.363	3.374	3.378	3.383	3.396
min complexity	740	692	740	660	692	692	660	660

Table I demonstrates all occurred scaling exponents of kernels from the set $\mathbb{K}_*$ together with minimal processing complexity. Furthermore, for each presented scaling exponent the kernel K with $\mathcal{M}(K) = 4$ is provided. It can be seen that the minimal complexity of 660 operations is provided by kernels with $\mu(K) = 3.363$ and kernels with the lowest $\mu(K) = 3.346$ requires the maximal complexity among other scaling exponents.

It turns out, that the complexity of window processing can be significantly reduced. For instance, the kernel K_1, illustrated in Figure 1, $K_1 \in \mathbb{K}_*$, $\mu(K_1) = 3.346$, was reported in [11] to have processing complexity of 472 arithmetic operations instead of 740.

For comparison, the general trellis-based algorithm [12] applied to processing of K_1 kernel has the complexity of 7530 operations, which is 10 times higher compared to the complexity of straightforward window processing algorithm. However, minimization of maximal size of the decoding windows is crucial, as far as complexity grows exponentially with it. For instance, 16×16 BCH kernel

$$K_{BCH} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&0&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&0&0&1&0&0&0&0&0&0&0&0&0\\
1&0&0&1&1&0&0&1&0&0&0&0&0&0&0&0\\
1&0&0&0&1&1&0&0&1&0&0&0&0&0&0&0\\
1&1&0&0&0&1&0&1&1&1&0&0&0&0&0&0\\
1&0&1&0&0&0&1&0&1&1&1&0&0&0&0&0\\
1&1&1&1&0&1&1&0&0&1&0&1&0&0&0&0\\
1&0&1&1&1&0&1&1&0&0&1&0&1&0&0&0\\
1&0&0&1&1&1&0&1&1&0&0&1&0&1&0&0\\
1&0&0&0&1&1&1&0&1&1&0&0&1&0&1&0\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1
\end{pmatrix},$$

with $E(K_{BCH}) = 0.51828$ and $\mu(K_{BCH} = 3.396)$, which consist of the sequence of nested generator matrices of extended BCH codes, has $\mathcal{M}(K_{BCH}) = 12$ and the

TABLE II
PROPERTIES OF PERMUTED K_1 KERNELS

$E(K)$	$\mu(K)$	$\mathcal{M}(K)$	$\Psi(K)$	$E(K)$	$\mu(K)$	$\mathcal{M}(K)$	$\Psi(K)$
0.58128	3.346	4	740	0.58128	3.405	3	377
0.58128	3.353	4	648	0.58128	3.414	3	331
0.58128	3.361	4	600	0.58128	3.415	3	320
0.58128	3.363	4	602	0.58128	3.432	3	306
0.58128	3.37	4	554	0.58128	3.45	3	292
0.58128	3.379	4	522	0.50914	3.484	3	268
0.58128	3.38	4	483	0.50914	3.513	3	252
0.58128	3.397	3	345	0.50914	3.53	2	173

processing complexity $\Psi(K_{BCH}) = 72563$. Whereas the algorithm [12] for K_{BCH} requires 12456 operations. This example shows us the importance of minimization of decoding windows sizes.

2) Permuted partial distances: In the previous section we showed how to find kernels of size 16 with monotonic partial distance profile $\mathbb{D}^{(*)}$. It resulted in kernels with the $\mathcal{M}(K) = 4$. For further complexity reduction we are going to perform row permutations over row space of the obtained kernels, which preserves the polarization rate.

Given kernel K, the value $\mathcal{M}(K)$ can be reduced by row permutation of K. By step-by-step exchange of the kernel rows, we performed an heuristic search of row permutation, which preserve the polarization rate of K_1.

Table II demonstrates the properties of kernels which we obtained by permutations of the kernel K_1. It can be observed, that higher scaling exponent requires lower processing complexity, furthermore, the maximal size of the decoding windows can be also reduced for kernels with polarization rate 0.51828. For instance, the kernel K_2, illustrated in Figure 1, has $E(K_2) = 0.51828$, $\mu(K) = 3.45$ and $\mathcal{M}(K) = 3$. The kernel K_2 is given by $\bar{\rho}K_1$, where

$$\bar{\rho} = [0, 1, 2, 7, 3, 4, 5, 6, 9, 10, 11, 12, 8, 13, 14, 15],$$

and has a partial distance profile

$$\bar{\mathbb{D}} = [1, 2, 2, 4, 2, 2, 4, 4, 6, 6, 8, 8, 4, 8, 8, 16],$$

which is not monotonic unlike $\mathbb{D}^{(*)}$. It was shown in [11] that the kernel K_2 can be processed with 183 operations instead of 293 operations in straightforward implementation.

Unfortunately, we do not have a proof that the kernel K_2 admits minimum possible complexity of window processing algorithm among all 16×16 polarization kernels with polarization rate 0.51828.

B. Performance of polar codes with the constructed kernels

We constructed $(4096, 2048)$ polar codes with kernels K_1 and K_2, obtained by the proposed construction, and investigated their performance for the case of AWGN channel with BPSK modulation. The sets of frozen symbols were obtained by Monte-Karlo simulations.

Figure 2 illustrates the performance of plain polar codes and polar subcodes [4],[19]. It can be seen that the codes based on kernels K_1 and K_2 with improved polarization rate

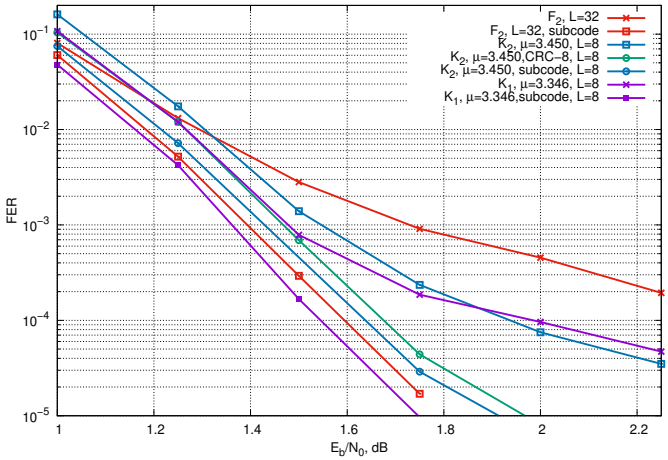

Fig. 2. Performance of $(4096, 2048)$ polar codes

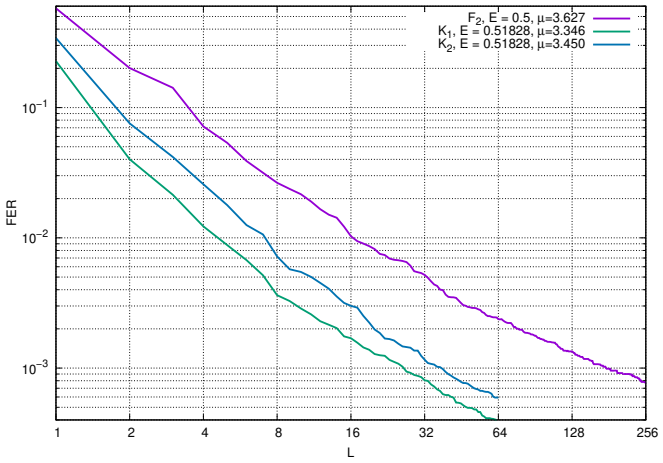

Fig. 3. Performance of SCL decoding

$E(K_1) = E(K_2) = 0.51828$ provide significant performance gain compared to polar codes with Arikan kernel. Moreover, polar subcodes with kernels K_1, K_2 under SCL with $L = 8$ have almost the same performance as polar subcodes with Arikan kernel under SCL with $L = 32$. Observe also that the codes based on kernels with lower scaling exponent exhibit better performance despite of the fact that scaling exponent is computed for the BEC.

Figure 3 presents simulation results for $(4096, 2048)$ polar subcodes with different kernels under SCL with different L at $E_b/N_0 = 1.25$ dB. It can be seen that the kernels with polarization rate 0.51828 require significantly lower list size L to achieve the same performance as the code with the Arikan kernel. Moreover, this gap grows with L. This is due to improved rate of polarization, which results in smaller number of unfrozen imperfectly polarized bit subchannels. The size of the list needed to correct possible errors in these subchannels grows exponentially with their number (at least for the genie-aided decoder considered in [20]). On the other hand, lower

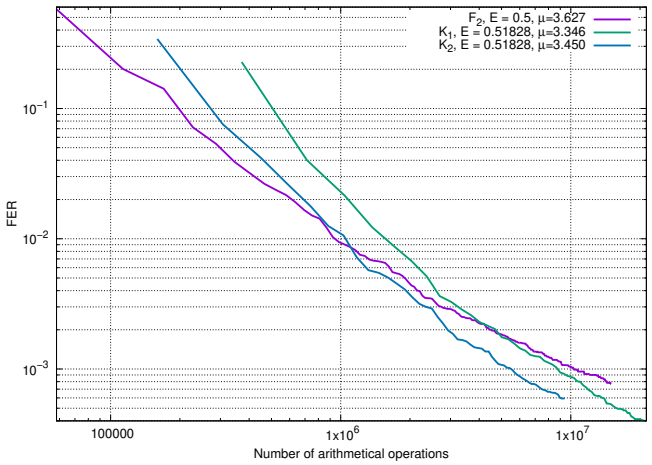

Fig. 4. Complexity of SCL decoding

scaling exponent gives better performance with the same list L, but the slope of the curve remains the same for both kernels K_1, K_2.

Figure 4 presents the same results in terms of the actual decoding complexity. Recall that proposed kernel processing algorithm uses only summations and comparisons. The SCL algorithm was implemented using the randomized order statistic algorithm for selection of the paths to be killed at each phase, which has complexity $O(L)$. Observe that the polar subcode based on kernel K_2 can provide better performance with the same decoding complexity for FER $\leq 8 \cdot 10^{-3}$. This is due to higher slope of the corresponding curve in Figure 3, which eventually enables one to compensate relatively high complexity of the LLR computation algorithm presented in [11].

Unfortunately, K_1 kernel, which provides lower scaling exponent, has greater processing complexity than K_2, so that its curve intersects the one for the Arikan kernel only at FER$= 2 \cdot 10^{-3}$.

V. Conclusions

In this paper the construction method for 16×16 polarization kernels with polarization rate 0.51828 were proposed. These kernels admits low complexity decoding by window processing algorithm. The construction method performs elementary operations over row space of the Arikan transform matrix. These elementary operations are chosen to have minimal impact on the complexity of the window processing algorithm.

It was shown that in the case of SCL decoding with sufficiently large list size, the constructed kernels results in lower decoding complexity compared to the case of polar (sub)codes with Arikan kernel with the same performance.

Extension of the proposed construction to the case of kernels with larger size remains an open problem.

References

[1] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, July 2009.

[2] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Transactions On Information Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[3] P. Trifonov and V. Miloslavskaya, "Polar subcodes," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 2, pp. 254–266, February 2016.

[4] P. Trifonov and G. Trofimiuk, "A randomized construction of polar subcodes," in *Proceedings of IEEE International Symposium on Information Theory*. Aachen, Germany: IEEE, 2017, pp. 1863–1867.

[5] T. Wang, D. Qu, and T. Jiang, "Parity-check-concatenated polar codes," *IEEE Communications Letters*, vol. 20, no. 12, December 2016.

[6] S. B. Korada, E. Sasoglu, and R. Urbanke, "Polar codes: Characterization of exponent, bounds, and constructions," *IEEE Transactions on Information Theory*, vol. 56, no. 12, pp. 6253–6264, December 2010.

[7] A. Fazeli, S. H. Hassani, M. Mondelli, and A. Vardy, "Binary linear codes with optimal scaling: Polar codes with large kernels," in *Proceedings of IEEE Information Theory Workshop*, 2018.

[8] A. Fazeli and A. Vardy, "On the scaling exponent of binary polarization kernels," in *Proceedings of 52nd Annual Allerton Conference on Communication, Control and Computing*, 2014, pp. 797 – 804.

[9] N. Presman, O. Shapira, S. Litsyn, T. Etzion, and A. Vardy, "Binary polarization kernels from code decompositions," *IEEE Transactions On Information Theory*, vol. 61, no. 5, May 2015.

[10] S. Buzaglo, A. Fazeli, P. H. Siegel, V. Taranalli, and A. Vardy, "On efficient decoding of polar codes with large kernels," in *Proceedings of IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, March 2017, pp. 1–6.

[11] G. Trofimiuk and P. Trifonov, "Efficient decoding of polar codes with some 16×16 kernels," in *Proceedings of IEEE Information Theory Workshop*, 2018.

[12] H. Griesser and V. R. Sidorenko, "A posteriory probability decoding of nonsystematically encoded block codes," *Problems of Information Transmission*, vol. 38, no. 3, 2002.

[13] S. H. Hassani, K. Alishahi, and R. Urbanke, "Finite-length scaling for polar codes," *IEEE Transactions On Information Theory*, vol. 60, no. 10, October 2014.

[14] V. Miloslavskaya and P. Trifonov, "Sequential decoding of polar codes with arbitrary binary kernel," in *Proceedings of IEEE Information Theory Workshop*. Hobart, Australia: IEEE, 2014, pp. 377–381.

[15] ——, "Sequential decoding of polar codes," *IEEE Communications Letters*, vol. 18, no. 7, pp. 1127–1130, 2014.

[16] P. Trifonov, "A score function for sequential decoding of polar codes," in *Proceedings of IEEE International Symposium on Information Theory*, Vail, USA, 2018.

[17] ——, "Binary successive cancellation decoding of polar codes with Reed-Solomon kernel," in *Proceedings of IEEE International Symposium on Information Theory*. Honolulu, USA: IEEE, 2014, pp. 2972 – 2976.

[18] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*. Amsterdam, The Netherlands: North-Holland, 1977.

[19] P. Trifonov, "Design of randomized polar subcodes with non-Arikan kernels," in *Proceedings of 16-th International Workshop on Algebraic and Combinatorial Coding Theory*, 2018.

[20] M. Mondelli, S. H. Hassani, and R. Urbanke, "Scaling exponent of list decoders with applications to polar codes," *IEEE Transactions On Information Theory*, vol. 61, no. 9, September 2015.