Design of BCH Polarization Kernels with Reduced Processing Complexity

Elizaveta Moskovskaya, Peter Trifonov *Member, IEEE*

***Abstract*—The paper introduces a construction of extended BCH polarization kernels, which admit simple computation of the probabilities or log-likelihood ratios arising in the successive cancellation decoding algorithm. The kernels are obtained by finding a column permutation of extended BCH kernels with natural column order, such that the total complexity of the associated trellises is minimized.**

***Index Terms*—Polar codes, large kernels, code trellis.**

I. Introduction

Polar codes are the first class of capacity-achieving codes, which have simple construction, encoding and decoding algorithms [1]. Although polar codes achieve the Shannon limit asymptotically, their finite-length performance is quite poor. One way to improve their performance is to replace the Arikan 2×2 kernel $F_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ with $l\times l$ kernels F [2]. It is possible to show that with $l \to \infty$ one can construct kernels with polarization rate and scaling exponent approaching optimal values of 1 and 2, respectively [2], [3], obtaining thus better finite length performance. In practice, polarization kernels can be constructed from a family of nested codes. Since BCH codes are among the best in terms of their minimum distance for small block length, they can be used to construct polarization kernels with near-optimal rate of polarization.

However, the decoding complexity of polar codes with large kernels still remains too high for a practical implementation. Namely, the complexity of kernel processing, i.e. evaluation of the probabilities, which arise in the successive cancellation decoding algorithm, in general, grows exponentially with kernel dimension l. In this paper we suggest a method for constructing BCH-based polarization kernels with reduced processing complexity.

II. Polar codes

Consider a matrix given by $G = BF^{\otimes n}$, where B is a digit-reversal permutation matrix, $F^{\otimes n}$ is the n-times Kronecker product of F with itself, and F is an $l\times l$ matrix, called kernel, which is invertible and not permutation-equivalent to an upper triangular matrix. It is possible to show that the transformation $c_0^{N-1} = u_0^{N-1}G$ together with a binary input memoryless channel $W(y|c)$ gives rise to $N = l^n$ bit subchannels [2]

$$W_n^{(i)}(y_0^{N-1}, u_0^{i-1}|u_i) = \sum_{u_{i+1}^{N-1}\in\mathbb{F}_2^{N-i-1}} \frac{1}{2^{N-1}} W_N(y_0^{N-1}|u_0^{N-1}), 0 \le i < N,$$

where $W_N(y_0^{N-1}|u_0^{N-1}) = \prod_{i=0}^{N-1} W(y_i|(u_0^{N-1}G)_i)$, and the capacities $I(W_n^{(i)})$ converge with $n \to \infty$ to 0 or 1, and the fraction of the subchannels with capacities close to 1 converges to the capacity of the underlying channel $W(y|c)$. This enables one to implement encoding as $c_0^{N-1} = u_0^{N-1}G, u_i = 0, i \in \mathcal{F}$. The frozen set $\mathcal{F}$ is typically selected as the set of indices i of bit subchannels $W_n^{(i)}(y_0^{N-1}, u_0^{i-1}|u_i)$ with capacity $I(W_n^{(i)}) \approx 0$, while the symbols $u_i, i \notin \mathcal{F}$, are set to the payload data. To determine the frozen set one can estimate error probability for subchannels using the Monte Carlo method.

Decoding of polar codes can be implemented using the successive cancelation (SC) algorithm

$$\hat{u}_i = \begin{cases} \arg\max\limits_{u_i\in\mathbb{F}_2} W_n^{(i)}\left(\hat{u}_0^{i-1}, u_i|y_0^{N-1}\right), & i \notin \mathcal{F}, \\ \text{frozen value } u_i & i \in \mathcal{F}, \end{cases} \quad (1)$$

where for $s = 0, \ldots, l^{n-1}$, $t = 0, \ldots, l-1$ one has

$$W_n^{(sl+t)}(u_0^{sl+t}|y_0^{l^n-1}) = \sum_{u_{sl+t+1}^{sl+l-1}\in\mathbb{F}_2^{l-t-1}} \prod_{j=0}^{l-1} W_{n-1}^{(s)}\left(v_j|y_{jl^{n-1}}^{(j+1)l^{n-1}-1}\right), \quad (2)$$

$v_j = \left(\left(u_{lp}^{lp+l-1}F\right)_j, p = 0, \ldots, s\right)$. The task of computing probabilities (2) will be further referred to as kernel processing. It can be seen that the complexity of this operation grows exponentially with l.

Let us consider without loss of generality the case of $n = 1$. It was suggested in [4] to simplify the kernel processing operation by considering only the maximal term in (2), i.e.

$$W_1^{(j)}(u_0^j|y_0^{l-1}) \approx \widetilde{W}_1^{(j)}(u_0^j|y_0^{l-1}) = \max_{u_{j+1}^{l-1}} \prod_{i=0}^{l-1} W((u_0^{l-1}F)_i|y_i). \quad (3)$$

Since vectors u_{j+1}^{l-1} are equiprobable, computing the latter expression is equivalent to maximum likelihood decoding of y_0^{l-1} in the coset of the code $\mathcal{C}^{(j+1)}$ generated by rows $j+1, \ldots, l-1$ of matrix F, where the coset representative is given by $a_0^{l-1} = u_0^j(F)_0^j$, and $(F)_p^q$ denotes a submatrix of matrix A consisting of its rows $p, \ldots, q$. For a BI-AWGN channel, this is equivalent to

$$\widetilde{W}_1^{(j)}(u_0^j|y_0^{l-1}) = \max_{u_{j+1}^{l-1}} \prod_{i=0}^{l-1} W((u_{j+1}^{l-1}(F)_{j+1}^{l-1})_i|y_i(-1)^{a_i}). \quad (4)$$

The authors are with ITMO University, Russia. Email: lizmoskovskaya@gmail.com

This work is supported by Government of Russian Federation (grant 08-08).

Equivalently, one can compute the log-likelihood ratios (LLR)

$$S_1^{(j)}(u_0^{j-1}, y_0^{l-1}) = \log \frac{\widetilde{W}_1^{(j)}(u_0^{j-1}, 0|y_0^{l-1})}{\widetilde{W}_1^{(j)}(u_0^{j-1}, 1|y_0^{l-1})}.$$

Let the correlation discrepancy of c_0^{l-1} with respect to the y_0^{l-1} be defined as $\mathcal{E}(c_0^{l-1}, y_0^{l-1}) = \sum_{i=0}^{l-1} \tau(c_i, y_i)$ where $\tau(c, y) = \begin{cases} -|S(y)|, & (-1)^c \neq \operatorname{sgn} S(y) \\ 0 & \text{otherwise,} \end{cases}$ and $S(y) = \log \frac{W(0|y)}{W(1|y)}$.

It is possible to show [5] that

$$S_1^{(j)}(u_0^{j-1}, y_0^{l-1}) = \mathcal{E}(c^{(0)}, y_0^{l-1}) - \mathcal{E}(c^{(1)}, y_0^{l-1}), \quad (5)$$

where $c^{(u_j)} = (u_0^j, U(u_0^{j-1}, u_j))F$, and $U(u_0^{j-1}, u_j)$ is the vector u_{j+1}^{l-1}, which delivers the maximum in (4). Employing the correlation discrepancy for computing the LLRs enables one to reduce the decoding complexity compared to a straightforward implementation based on (4).

III. Trellis based kernel processing

The Viterbi algorithm can be used to implement maximum likelihood decoding of codes $\mathcal{C}^{(j)}$, as needed for computing (4). Observe that at each step of the SC recursion (1)–(2) one has to compute $W_1^{(j)}\left(\hat{u}_0^{j-1}, u_j|y_0^{l-1}\right)$ for $u_j \in \mathbb{F}_2$. In order to solve this problem simultaneously for both values of u_j, it is convenient to define extended code $\overline{\mathcal{C}}^{(j)}$ with generator matrix

$$T_j = \begin{pmatrix} F_{j,*} & 1 \\ F_{j+1,*} & 0 \\ \vdots & \vdots \\ F_{l-1,*} & 0 \end{pmatrix},$$

where $F_{i,*}$ is the i-th row of F. Then computing $\widetilde{W}_1^{(j)}\left(\hat{u}_0^{j-1}, u_j|y_0^{l-1}\right)$ reduces to decoding of vector $(y_0, \ldots, y_{l-1}, \epsilon)$ in code $\overline{\mathcal{C}}^{(j)}$, where ϵ is the erasure symbol. This can be implemented by applying the Viterbi algorithm to the minimal trellis of $\overline{\mathcal{C}}^{(j)}$. The metric of trellis node s at layer i for the Viterbi algorithm can be defined as $M_{i,s} = \max_{s' \in N_{is}} \left(M_{i-1,s'} + \tau(c_{i-1,s',s}, y_{i-1})\right)$, where $M_{0,0} = 0$, N_{is} is the set of nodes s' at layer $i-1$, adjacent to node s at layer i, and $c_{i-1,s',s}$ is the label of the corresponding edge.

At the l-th layer this trellis has two states. It can be seen that the metrics computed by the Viterbi algorithm for these states are exactly the values $\mathcal{E}(c^{(x)}, y_0^{l-1}), x \in \mathbb{F}_2$, used in (5). Hence, the required LLR $S_1^{(j)}(u_0^{j-1}, y_0^{l-1})$ is obtained as the difference between these metrics. This approach is similar to the soft-output decoding algorithm for non-systematically encoded linear block codes presented in [6].

Example 1. For the Arikan kernel, one obtains the following generator matrices for codes $\overline{\mathcal{C}}^{(j)}$: $T_0 = \left(\begin{array}{cc|c} 1 & 0 & 1 \\ 1 & 1 & 0 \end{array}\right)$, $T_1 = \left(\begin{array}{cc|c} 1 & 1 & 1 \end{array}\right)$. The trellises for these codes are shown in Fig. 1. Dotted edges correspond to the auxiliary (erased) symbol.

The Viterbi algorithm performs at most $\mathbf{S}_j = \sum_{i=0}^{l-1} |E_i^{(j)}|$ additions and $\mathbf{C}_j = \sum_{i=0}^{l-1} |E_i^{(j)}| - \sum_{i=0}^{l-1} |V_i^{(j)}|$ comparisons, where $|V_i^{(j)}|$ is the number of nodes on the i-th level of the trellis and $|E_i^{(j)}|$ is the number of edges between i-th and $(i+1)$-th levels of the trellis for code $\overline{\mathcal{C}}^{(j)}$.

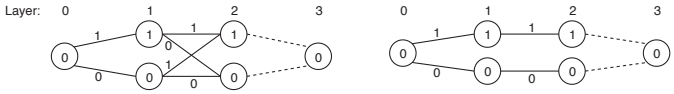

Figure 1. Trellises for the Arikan kernel

It is possible to show that the number of nodes and edges in a minimal trellis of code $\overline{\mathcal{C}}^{(j)}$ can be computed as [7]

$$|V_i^{(j)}| = 2^{\dim C^{(j)} - \dim C^{(j)}_{[0,i]} - \dim C^{(j)}_{[i,l]}} \quad (6)$$

$$|E_i^{(j)}| = 2^{\dim C^{(j)} - \dim C^{(j)}_{[0,i-1]} - \dim C^{(j)}_{[i,l]}}, \quad (7)$$

where $C^{(j)}_{[k,k']}$ is a subcode of $\overline{\mathcal{C}}^{(j)}$ for which $c_i = 0 \; \forall i \notin [k, k']$. Permuting the columns of the generator matrix of code $\overline{\mathcal{C}}^{(j)}$ may result in dramatic changes in the number of nodes and edges in the minimal trellises of the obtained codes. To minimize the complexity of the Viterbi algorithm, one needs to find a column permutation, which maximizes the dimensions of codes $C^{(j)}_{[0,i]}$ and $C^{(j)}_{[i,l]}$ for various i.

IV. BCH kernels optimization

In this section we present a method for finding a permutation of columns of an extended BCH kernel, which provides reduced complexity processing based on the Viterbi algorithm.

A. Extended BCH kernel

A polarization kernel can be obtained from a sequence of nested codes, e.g. BCH codes. The rate of polarization depends on the partial distances of the kernel [2], which are related to the minimum distances of the corresponding nested codes. Since short BCH codes provide near-optimal minimum distances, they are good candidates for use in a construction of polarization kernels.

(Extended) BCH kernel is an $l \times l$ matrix, such that its last k rows generate (l, k, d_k) subcode of the (extended) BCH code with the maximum possible minimal distance d_k, $k = 1, \ldots, l$. In particular, for $l = 2^m$ the extended BCH kernel can be obtained as a matrix F, where $(F_{i+1,1}, \ldots, F_{i+1,l-1})$ are coefficients of $x^j f_{i'}(x)$. Here $f_{i'}(x)$ is a generator polynomial of $(l-1, l-1-i')$ primitive narrow-sense BCH code, and j is a minimal natural number, such as $i = j + i', 0 \leq i < l-1$. Furthermore, the parity check symbol for each row is given by $F_{0,0} = 1$ and $F_{i+1,0} = \sum_{k=1}^{l-1} F_{i+1,k}$ [2] [8].

B. Trellis complexity minimization

Let us consider the problem of finding a column permutation of the kernel matrix, which minimizes the complexity of Viterbi-based processing. Observe that permuting the columns of a kernel does not affect its polarization properties. Exploring all $l!$ permutations is not computationally feasible in practice. Therefore, we propose to exploit the algebraic structure of extended BCH codes to obtain good (but not necessarily optimal) column permutations in terms of the complexity of the associated trellises.

Extended BCH codes have a direct-sum structure, which means that their generator matrix can be transformed by column permutations and elementary row operations into a form, containing a block-diagonal submatrix. This ensures the existence of subcodes $C_{[0,i]}$ and $C_{[i,l]}$ with sufficiently large dimension, i.e. small number of nodes and edges in the trellis.

A linear code C of length l is said to have a direct-sum structure if it contains a nontrivial subcode C' which is spanned by some h non-overlapping codes $C'_1, C'_2, \ldots, C'_h$. More precisely, let $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_h$ be a partition of the set $\{0, 1, \ldots, l-1\}$, and let the codes $C'_1, C'_2, \ldots, C'_h$ be such that for any $j \in \{1, 2, \ldots, h\}$ and for any $(c_0, c_1, \ldots, c_{l-1}) \in C'_j$ we have $c_i = 0$ for all $i \notin \mathcal{L}_j$. Then the direct-sum subcode C' consists of all the vectors of the form $\mathbf{c}_1 + \mathbf{c}_2 + \cdots + \mathbf{c}_h$ where $\mathbf{c}_j \in C'_j$ for $j = 1, 2, \ldots, h$ [9], [10].

Let C be an extended binary BCH code of length l and dimension k. Then, let $C[\mathfrak{I}]$ denote subcode C, which consists of codewords which are not equal to zero only in positions from $\mathfrak{I}$. Subcode $C(\mathfrak{I})$ is obtained from $C[\mathfrak{I}]$ by puncturing all zeroes that are not in $\mathfrak{I}$. Let $\alpha \in \mathbb{F}_l$ be a primitive $(l-1)$-th root of unity and $\mathfrak{I}$ be a subset of set $\{0, 1, \ldots, l-2, \infty\}$. Let us further assume that $\alpha^\infty = 0$. The following theorem shows that the structure of the additive group of $\mathbb{F}_l$ induces the direct-sum structure of the extended BCH code.

Theorem 1 ([9]). Let $\mathfrak{I}_1$ and $\mathfrak{I}_2$ be subsets of $\{0, 1, \ldots, l-2, \infty\}$, such that some $a \in \{0, 1, \ldots, l-2\}$ satisfies $\{\alpha^i : i \in \mathfrak{I}_2\} = \{\alpha^a + \alpha^i : i \in \mathfrak{I}_1\}$. Then $C(\mathfrak{I}_1) = C(\mathfrak{I}_2)$.

The dimensions of codes $C(\mathfrak{I}_1)$ depend strongly on the choice of $\mathfrak{I}_s, s \in \mathbb{F}_2$. This theorem means that if one finds a subset $\mathfrak{I}_1$ with sufficiently large $C(\mathfrak{I}_1)$, i.e. a large dimension component of the direct sum structure of the extended BCH code, then this implies the existence of at least one more large dimension direct sum component. This can be used to simplify search for good column permutations.

Let the columns of a generator matrix of the extended BCH code (or kernel) be labeled as $0, \alpha^0, \alpha^1, \ldots, \alpha^{l-2}$. This numbering will be referred to as *natural*. Recall, that the classical construction of the check matrix of the extended BCH code with design distance $d+1$ is obtained by raising these values to powers $0, \ldots, d-1$.

Consider the column permutation of the generator matrix of this code with labels satisfying

$$\pi(i + l/2^s) = \alpha^a + \pi(i), 0 \le i < l/2^s, \tag{8}$$

for some a and $s \le \log_2 l$, where $\pi(i)$ denotes the label of the i-th column after the permutation. The above theorem implies that if one permutes the columns of the generator matrix so that (8) holds, then the obtained generator matrix can be transformed by row operations into the block-diagonal form with a number of identical blocks. Furthermore, there exist subcodes $C_{[0,i]}$ and $C_{[i,l]}$ of the considered extended BCH code of sufficiently large dimension. This is valid for any extended BCH code of length l, so the above statement holds simultaneously for all codes of $\overline{\mathcal{C}}^{(j)}$ generated by submatrices of the considered extended BCH kernel.

There exist several ways to partition $\mathbb{F}_l$ into the subsets satisfying the conditions of Theorem 1, i.e. permute the columns of the generator matrix of the extended BCH codes, and obtain their direct sum subcodes of sufficiently large dimensions. Based on the approach suggested in [9], we propose to consider the permutations given by $\pi(i) = (\alpha^0, \ldots, \alpha^{\lambda-1})Q(i_0, \ldots, i_{\lambda-1})^T, 0 \le i < l$, where $i = \sum_{i=0}^{\lambda-1} i_j 2^j, i_j \in \mathbb{F}_2$, $l = 2^\lambda$, and Q is a non-singular binary matrix. The permutation given by $Q = I$ will be referred to as the *standard bit ordering*.

$$K_1 = \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \end{array} \\ \begin{pmatrix} 1&0&0&0&0&0&0&0 \\ 1&1&0&0&0&0&0&0 \\ 1&0&1&0&0&0&0&0 \\ 1&0&0&1&0&0&0&0 \\ 1&1&1&0&1&0&0&0 \\ 1&0&1&1&0&1&0&0 \\ 1&0&0&1&1&0&1&0 \\ 1&1&1&1&1&1&1&1 \end{pmatrix} \end{array}, \quad K_2 = \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & \alpha & \alpha+1 & \alpha^2 & \alpha^2+1 & \alpha^2+\alpha & \alpha^2+\alpha+1 \end{array} \\ \begin{pmatrix} 1&0&0&0&0&0&0&0 \\ 1&1&0&0&0&0&0&0 \\ 1&0&1&0&0&0&0&0 \\ 1&0&0&0&1&0&0&0 \\ 1&1&1&1&0&0&0&0 \\ 1&0&1&0&1&0&1&0 \\ 1&0&0&1&1&0&0&1 \\ 1&1&1&1&1&1&1&1 \end{pmatrix} \end{array}$$

Figure 2. Extended BCH 8×8 kernels with natural and standard ordering

Example 2. The extended BCH kernel of dimension 8 is given by the kernel K_1 in Fig. 2, where α is a primitive root of x^3+x+1. By applying the column permutation corresponding to the standard bit ordering, one obtains K_2 in Fig. 2.

By enumerating matrices Q, one obtains different column permutations of the original extended BCH kernel, which induce trellises with different number of nodes and edges. We propose to select matrix Q, which results in the smallest total number of summation and comparison operations in the Viterbi algorithm for all phases $j, 0 \le j < l$. i.e. the one which minimizes $\sum_{j=0}^{l-1}(\mathbf{S}_j + \mathbf{C}_j)$.

The total number of matrices Q is $\prod_{i=0}^{\lambda-1}(2^\lambda - 2^i)$. One can either enumerate them exhaustively for small λ, or perform randomized search.

V. Numeric results

Fig. 3 and 4 present extended BCH kernels of dimension 16 and 32 obtained with the proposed algorithm, as well as the corresponding column permutations given by $\mathbb{F}_l$ bases $(\alpha^0, \ldots, \alpha^{\lambda-1})Q$. Table I presents the total number of arithmetic operations performed by the Viterbi algorithm for computing the LLRs $S_1^{(j)}, 0 \le j < l$, for the obtained kernels. We consider the cases of the permutations given by natural column numbering, standard bit ordering and optimized matrices Q. It can be seen that the complexity of the Viterbi algorithm for the case of standard bit ordering is much less compared to the case of natural numbering. Employing the optimized column permutations results in further complexity reduction. In total, the proposed approach provides for the case of 32×32 BCH kernel complexity reduction by a factor of 10.5 compared to the natural numbering.

Observe that the Viterbi algorithm together with the obtained kernels provides much more efficient implementation of kernel processing compared to the algorithm presented in [11]. Indeed, the latter algorithm requires $16 \cdot 2487 = 39792$ operations, while the Viterbi algorithm for the same kernel requires 10386 operations in average.

Fig. 5 presents simulation results for polar codes under SC decoding, based on the approximate trellis-based kernel processing method. For comparison, we report also the results

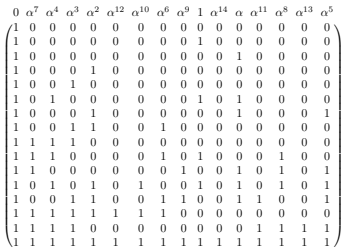

Figure 3. Optimized kernel for basis $(\alpha^3+\alpha+1, \alpha+1, \alpha^2, \alpha^3+\alpha)$ with polarization rate and scaling exponent equal to 0.518 and 3.396 respectively

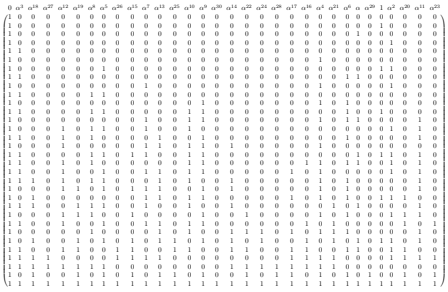

Figure 4. Optimized kernel for basis $(\alpha^3, \alpha+1, \alpha^3+\alpha^2+\alpha, \alpha^4+\alpha^2+\alpha+1, \alpha^4+\alpha^3+\alpha^2+1)$ with polarization rate and scaling exponent equal to 0.537 and 3.122 respectively

for $(256, 128)$ code obtained with the processing method based on [6]. It can be seen that employing approximation (3) has almost no impact on the performance. It can be also seen that polar codes with BCH kernels provide much better performance compared to those with the Arikan kernel. However, slightly better performance can be obtained with a kernel suggested by Lin et al. in [12] with scaling exponent 3.363, which is lower compared to the one of the BCH kernel.

In Fig. 6 it can be seen that the decoding complexity of polar codes with BCH kernel still remains higher compared to the case of codes with Arikan kernel with the same performance. However, the gap between the curves decreases while increasing the decoding complexity (i.e. list size). By employing more efficient decoding techniques (e.g. trellis sectionalization) it may be possible to further reduce the complexity of BCH kernel processing.

VI. Conclusions

In this paper the problem of computing the LLRs for the bit subchannels induced by a large polarization kernel was studied. It was shown that the LLRs can be computed by the Viterbi algorithm. For the case of extended BCH kernels, permuting their columns may result in dramatic changes in the complexity of the Viterbi algorithm. A method for finding a column permutation, which (approximately) minimizes the complexity of the Viterbi algorithm was presented.

Table I
BCH kernel processing complexity

Kernel	Total number of sums			Total number of comparisons		
	Natural	Standard	Optimised	Natural	Standard	Optimised
16×16	4137	2687	1673	8274	5374	3346
32×32	$1.05 \cdot 10^6$	168323	99745	$2.1 \cdot 10^6$	336646	199490

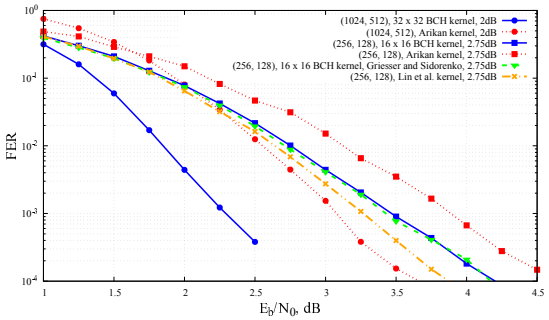

Figure 5. Performance of polar codes

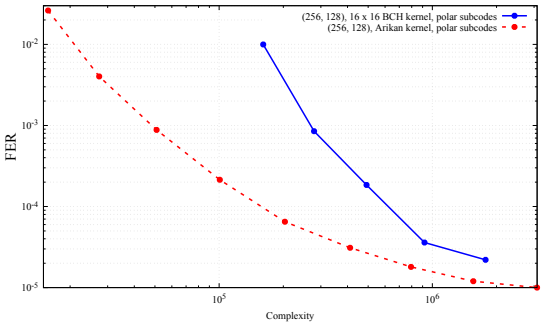

Figure 6. Performance of polar subcodes with 16×16 BCH kernel and the Arikan kernel under SC list decoding with various list sizes for SNR $= 3$ dB

References

[1] E. Arikan, "Channel polarization: A method for constructing capacity-achieving ccodes for symmetric binary-input memoryless channels," *IEEE Trans. on Inf. Th.*, vol. 55, no. 7, July 2009.

[2] S. Korada, E. Sasoglu, and R. Urbanke, "Polar codes: Characterization of exponent, bounds, and constructions," *IEEE Trans. on Inf. Th.*, vol. 56, no. 12, December 2009.

[3] 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.

[4] V. Miloslavskaya and P. Trifonov, "Sequential decoding of polar codes with arbitrary binary kernel," In *Proc. of IEEE ITW*, 2014.

[5] G. Trofimiuk and P. Trifonov, "Efficient decoding of polar codes with some 16x16 kernels," In *Proc. of IEEE ITW*, November 2018.

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

[7] A. Vardy, "Trellis structure of codes," in *Handbook of Coding Theory*, V. Pless and W. Huffman, Eds. Amsterdam, The Netherlands: Elsevier Science, 1998, vol. 2.

[8] V. Miloslavskaya and P. Trifonov, "Design of binary polar codes with arbitrary kernel," In *Proc. of IEEE ITW*, 2012.

[9] A. Vardy and Y. Be'ery, "Maximum-likelihood soft decision decoding of bch codes," *IEEE Trans. on Inf. Th.*, vol. 40, no. 2, December 1994.

[10] T. Kasami, T. Takata, E. Fujiwara, and S. Lin, "On the optimum bit orders with respect to the state complexity of trellis diagrams for binary linear codes," *IEEE Trans. on Inf. Th.*, vol. 39, no. 1, January 1993.

[11] Z. Huang, S. Zhang, F. Zhang, C. Duanmu, F. Zhong, and M. Chen, "Simplified successive cancellation decoding of polar codes with medium-dimensional binary kernels," *IEEE Access*, vol. 6, pp. 26 707–26 717, 2018.

[12] H.-P. Lin, S. Lin, and K. A. Abdel-Ghaffar, "Linear and nonlinear binary kernels of polar codes of small dimensions with maximum exponents," *IEEE Trans. on Inf. Th.*, vol. 61, no. 10, October 2015.