Design and Decoding of Polar Codes with a Mixture of Reed-Solomon and Arikan Kernels

Nikolai Iakuba and Peter Trifonov
ITMO University
{nyakuba, pvtrifonov}@itmo.ru

***Abstract*—In this paper a construction method for polar codes with mixed Reed-Solomon and Arikan kernels is proposed. A decoding algorithm for the proposed codes is also presented, which provides lower decoding latency compared to the case of Arikan polar codes with the same performance.**

I. Introduction

Polar codes are the class of codes, that are proven to achieve the symmetric capacity of memoryless channels with polynomial encoding and decoding complexity [1]. Despite good performance of long polar codes, sequential nature of decoding algorithms [2], [3] results in high decoding latency, which grows linearly with the length of code.

To reduce decoding latency, several approaches were proposed, including permutation decoding [4]–[6] and algorithms based on the belief propagation decoding algorithm [7], [8]. However, these algorithms cannot simultaneously provide reduced decoding latency and performance comparable to classical successive cancellation list decoding of polar codes.

In this paper, the problem of latency reduction is addressed by employing a non-binary polarization kernels. A construction of polar codes over $\mathrm{GF}(2^\mu)$ with mixed Reed-Solomon and Arikan kernels is investigated. We propose also a decoding algorithm for these codes. Instead of employing a highly complicated algorithm needed for computing the probabilities of input symbols for Reed-Solomon kernel, we represent it via Arikan kernel. This enables one to implement the decoder for the proposed codes as a number of instances of Arikan SCL decoder, which perform synchronization via dynamic freezing constraints induced by the Reed-Solomon kernel. In section III construction methods are discussed, and an equation for computing the minimum distance of the proposed codes are provided. Generalization of the list decoding algorithm [2] is described in section IV. Finally, simulation results are presented in section V.

II. Background

A. Channel polarization

Consider the symmetric memoryless channel $\mathcal{W}:\mathbb{F}_2\to\mathcal{Y}$ defined by transition probabilities $\mathcal{W}(y|c)$. Polar $(n=2^m,k)$ code is defined as the set of vectors

$$\left\{c_0^{n-1}=u_0^{n-1}A^{\otimes m}|u_0^{n-1}\in\mathbb{F}_2^n \text{ and } u_i=0,\forall i\in\mathcal{F}\right\}, \quad (1)$$

where $A=\left(\begin{smallmatrix}1&0\\1&1\end{smallmatrix}\right)$ is known as the Arikan kernel, $\otimes m$ denotes m-time Kronecker product of the matrix with itself, and $\mathcal{F}\subset[n]$ is the set of frozen symbols [1], where $[n]$ denotes the set $\{0,\dots,n-1\}$.

The matrix $A_m=A^{\otimes m}$ together with the channel $\mathcal{W}$ gives rise to n synthetic channels

$$\mathcal{W}_i(y_0^{n-1},u_0^{i-1}|u_i)\triangleq\sum_{u_{i+1}^{n-1}\in\mathbb{F}_2^{n-i-1}}\frac{1}{2^{n-1}}\mathcal{W}(y_0^{n-1},u_0^{n-1}) \quad (2)$$

whose capacities asymptotically tend to 0 or 1 with the increase of the code length n.

Therefore, it is possible to transmit payload data through the set $[n]\setminus\mathcal{F}$ of most reliable channels $\mathcal{W}_i$. At the receiver end symbols $\hat{u}_\phi,\phi=0,\dots,n-1$ can be estimated by the successive cancellation (SC) decoding algorithm one by one:

$$\hat{u}_\phi=\begin{cases}\arg\max_{u_\phi\in\mathbb{F}_2}\mathcal{W}_\phi(y_0^{n-1},\hat{u}_0^{\phi-1}|u_\phi),\phi\notin\mathcal{F}\\0,\phi\in\mathcal{F}.\end{cases} \quad (3)$$

Due to the recursive structure of the matrix A_m encoding and decoding can be done in $O(n\log n)$ operations, and decoding latency grows linearly with the code length.

B. List decoding algorithm

The SC decoding algorithm is highly suboptimal, since the estimates $\hat{u}_\phi$ are computed without taking into account frozen symbols $u_i,i>\phi,i\in\mathcal{F}$. This problem was addressed in [2], where the successive cancellation list (SCL) decoding algorithm was introduced. Here we revise the min-sum version of this algorithm.

The SCL decoder successively extends L vectors $\hat{u}_0^\phi$ of equal length trying to maximize their score

$$M(\hat{u}_0^\phi,y_0^{n-1})=\sum_{i=0}^{\phi}\tau(\hat{u}_i,S_m^{(i)}(\hat{u}_0^{i-1},y_0^{n-1})), \quad (4)$$

where

$$\tau(u,S)=\begin{cases}0,\ \text{if } (-1)^u=\mathrm{sgn}(S)\\-|S|,\ \text{otherwise}\end{cases} \quad (5)$$

is the penalty function. $S_m^{(i)}(\hat{u}_0^{i-1},y_0^{n-1})$ denotes the log-likelihood ratio given by

$$\begin{aligned}S_\lambda^{(2\phi)}(\hat{u}_0^{2\phi-1},y_0^{n-1})&=\mathrm{sgn}(a)\,\mathrm{sgn}(b)\min(|a|,|b|),\\S_\lambda^{(2\phi+1)}(\hat{u}_0^{2\phi},y_0^{n-1})&=(-1)^{\hat{u}_{2\phi}}a+b,\end{aligned} \quad (6)$$

978-1-7281-1944-1/19/$31.00 ©2019 IEEE

where $n = 2^{m-\lambda}$, $a = S^{\phi}_{\lambda-1}(\hat{u}^{2\phi-1}_{0,e} + \hat{u}^{2\phi-1}_{0,o}, y_0^{n/2-1})$, $b = S^{(\phi)}_{\lambda-1}(\hat{u}^{2\phi-1}_{0,o}, y^{n-1}_{n/2-1})$, and $S^{(i)}_0 = \log \mathcal{W}(y_i|0) - \log \mathcal{W}(y_i|1)$. The list size L defines the complexity $O(Ln \log n)$ of the decoding, and the latency is $O(n)$.

C. Polar subcodes

The performance of classic polar codes also can be improved by employing dynamic frozen symbols [9]. It was proposed to set some $u_i, i \in \mathcal{F}$ to pre-defined linear functions of previous symbols, i.e.

$$u_{j_i} = \sum_{s<j_i} V_{i,s} u_s, 0 \le i < n-k, \tag{7}$$

where $V \in \mathbb{F}_2^{n-k \times n}$ is a constraint matrix and j_i is the maximal index of the non-zero element of the row $V_{i,-}$.

Hence, encoding can be done by evaluation of $c_0^{n-1} = u_0^{n-1} W A_m$, where $W \in \mathbb{F}_2^{n \times n}, WV^T = 0$ is a precoding matrix. Since $A_m^{-1} = A_m$, any linear code of length n with a check matrix H can be represented as a polar code with dynamic frozen symbols by choosing V and W, such that $H = VA_m^T$ and $c_0^{n-1} H^T = u_0^{n-1} W V^T = 0$.

D. Polarization kernels

It is possible to show, that any binary $l \times l$ matrix K that is not upper-triangular under any column permutation can be used to construct polar codes [10]. The error probability of polar codes under SC decoding depends on the polarization kernel K and asymptotically scales as $O(2^{-n^{E(K)}})$, where $E(K)$ is called the rate of polarization [11].

For the case of polar codes with Arikan kernel $E(A) = 0.5$, however, higher rate of polarization can be reached by using kernels over $\mathrm{GF}(q = 2^\mu)$ [12]. In this paper we consider the Reed-Solomon kernel R_μ, which is defined by the matrix

$$R_\mu = \begin{pmatrix} 1 & 1 & \dots & 1 & 1 & 0 \\ \alpha_{q-1}^{q-2} & \alpha_{q-2}^{q-2} & \dots & \alpha_2^{q-2} & \alpha_1^{q-2} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \alpha_{q-1}^{1} & \alpha_{q-2}^{1} & \dots & \alpha_2^{1} & \alpha_1^{1} & 0 \\ 1 & 1 & \dots & 1 & 1 & 1 \end{pmatrix}, \tag{8}$$

where α_i are distinct elements of $\mathrm{GF}(2^\mu) \setminus \{0\}$. Note, that R_μ defines 2^μ nested extended Reed-Solomon codes, generated by last rows of the kernel matrix, and evaluation of probabilities $\mathcal{W}_i(y_0^{n-1}, u_0^{i-1}|u_i)$ for the SC decoder is equivalent to soft-decision decoding of those codes.

Any $(2^\mu, k, d)$ extended Reed-Solomon code can also be represented as a polar code with dynamic frozen symbols [13]. This representation corresponds to the encoding scheme $c_0^{2^\mu-1} = u_0^{2^\mu-1} W A_\mu$, where first $2^\mu - k$ symbols of an information vector $u_0^{2^\mu-1} \in \mathrm{GF}(2^\mu)^{2^\mu}$ are zero, and $W = R_\mu A_\mu^{-1}$.

The above constructions of polar codes were generalized in [14], [15], where polarizing transforms defined by $K = K_1 \otimes \dots \otimes K_N$ are considered, where K_i are different $l_i \times l_i$ kernel matrices. Those mixed-kernel constructions can be used to obtain polar codes of arbitrary length [16], and provide a tradeoff between desired decoding performance and complexity [15], [17].

III. Construction

In this work we are using the construction of polar codes with mixed kernels to provide a decoding algorihtm with reduced latency compared to the list decoding of classic polar codes. We consider codes over $\mathrm{GF}(2^\mu)$ generated by rows with indices $i \notin \mathcal{F}$ of the matrix $G_{m+\mu} = A_m \otimes R_\mu$.

The matrix $G_{m+\mu}$ and set of frozen symbols $\mathcal{F}$ define the $(n = 2^{m+\mu}, k = 2^{m+\mu} - |\mathcal{F}|)$ generalized concatenated code over $\mathrm{GF}(2^\mu)$ with outer extended Reed-Solomon codes $\mathcal{R}_i, i = 0, \dots, 2^m - 1$ and inner polar codes with Arikan kernel [18].

Hence, encoding can be done in the following way. First, an information vector $u_0^{n-1} \in \mathrm{GF}(2^\mu)^n$ with data symbols placed in $u_j, j \notin \mathcal{F}$ and $u_i = 0, i \in \mathcal{F}$ is arranged into $2^m \times 2^\mu$ matrix, where i-th row is equal to $u_{i2^\mu}^{i2^\mu+2^\mu-1}$. Next, each row is multiplied by R_μ, which is equivalent to encoding data by some outer Reed-Solomon code defined by frozen set $\mathcal{F}$. Finally, columns are multiplied by A_m, i.e. are encoded by an inner polar code.

Alternatively, outer Reed-Solomon codes can be represented as polar codes with dynamic frozen symbols [13], which results in the encoding scheme given by

$$c_0^{n-1} = u_0^{n-1} W A_{m+\mu}, \tag{9}$$

where W is a block-diagonal precoding matrix, which is defined as shown in Example 1.

Example 1. *Consider $G_4 = R_2 \otimes A_2$. Hence*

$$G_4 = \begin{pmatrix} R_2 & 0 & 0 & 0 \\ R_2 & R_2 & 0 & 0 \\ R_2 & 0 & R_2 & 0 \\ R_2 & R_2 & R_2 & R_2 \end{pmatrix} = \begin{pmatrix} W^{(0)} & 0 & 0 & 0 \\ 0 & W^{(1)} & 0 & 0 \\ 0 & 0 & W^{(2)} & 0 \\ 0 & 0 & 0 & W^{(3)} \end{pmatrix} \cdot A_4.$$

Let define the matrix $W^{(1)}$ and the corresponding constraint matrix $V^{(1)}$. Consider the block of symbols u_4^7 and assume $\{4, 5\} \subset \mathcal{F}$, i.e. data symbols u_6, u_7 are encoded by the $(4, 2, 3)$ extended Reed-Solomon code. Hence,

$$u_4^7 R_2 = u_4^7 \bar{W} A_2 = (0, 0, u_6, u_7) \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & \alpha^2 & 0 \\ 0 & 1 & \alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot A_2$$
$$= (0, u_6, u_6\alpha, u_7) A_2 = v_4^7 A_2.$$

Here by v_0^{n-1} we denote the input vector for the Arikan polarizing transform, where $v_4 = 0$ is frozen, $v_6 = \alpha v_5$ is dynamic frozen, and v_5, v_7 are unfrozen. The matrix $W^{(1)}$ is obtained as a submatrix of $\bar{W}$ with zero rows corresponding to u_4, u_5, i.e.

$$W^{(1)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & \alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, V^{(1)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \alpha & 1 & 0 \end{pmatrix}, W^{(1)}(V^{(1)})^T = 0.$$

To define a polar code one needs to establish the set of frozen symbols $\mathcal{F}$, or, equivalently, to choose rates of outer Reed-Solomon codes $\mathcal{R}_i$ generated by last rows of R_μ. Frozen set $\mathcal{F}$ depends on the desired rate of the code, channel $\mathcal{W}(y|x)$, and the decoding algorithm used. In the following sections

methods of choosing $\mathcal{F}$ based on distance properties of the code and BEC erasure probability are described.

A. Maximization of the minimum distance

The following theorem provides a simple equation for the minimum distance of the proposed codes, which can be also used to construct codes or enhance its distance properties.

Theorem 1. *The minimum distance of the* $(n = 2^{m+\mu}, k = \sum_{i=0}^{2^m-1} k_i, d)$ *generalized concatenated code* $\mathcal{C}$ *with outer* $(n_i = 2^\mu, k_i, d_i = 2^\mu - k_i + 1)$ *extended Reed-Solomon codes* $\mathcal{R}_i, i = 0, \ldots, 2^m$ *and inner polar codes with Arikan kernel is equal to*

$$d(\mathcal{C}) = \min\left\{w_i d_i \mid i = 0, \ldots, 2^m - 1\right\}, \tag{10}$$

where w_i *is the Hamming weight of the* i*-th row of* A_m*.*

Proof. Recall, that codewords are given by

$$c_0^{n-1} = \sum_{i=0}^{n-1} A_{i,-} \otimes (u_{i2^\mu}^{(i+1)2^\mu-1} R_\mu) = \sum_{i=0}^{n-1} A_{i,-} \otimes \mathbf{r}^{(i)},$$

where $A_{i,-}$ denotes the i-th row of A_m, and $\mathbf{r}^{(i)} \in \mathcal{R}_i$ is a codeword of the corresponding outer code.

Clearly $d \le w_i d_i, i = 0, \ldots, 2^m - 1$, since there exists the codeword $c_0^{n-1} = A_{i,-} \otimes \mathbf{r}^{(i)}$, which contains the vector $\mathbf{r}^{(i)} \in \mathcal{R}_i$ repeated w_i times. Therefore, we only need to prove, that $d \ge \min\left\{w_i d_i \mid i = 0, \ldots, 2^m - 1\right\}$.

Note [19], that the minimum distance of polar codes with Arikan kernel is equal to

$$d_A = \min_{i \notin \mathcal{F}} 2^{\mathrm{wt}(i)} = \min_{i \notin \mathcal{F}} w_i. \tag{11}$$

i.e. minimum-weight codewords of a classic polar code are produced using rows of A_m with the minimum weight.

Let $\hat{A}_m$ denote the sorted Arikan kernel of length 2^m, i.e. $\hat{A}_m$ consists of rows of A_m sorted by the increase of its Hamming weight. Consider a generalized concatenated code $\mathcal{C}$ with inner codes generated by last rows of A_m and outer codes $\mathcal{R}_i, i = 0, \ldots, 2^m - 1$.

Note, that $\mathcal{C}$ is equivalent to a generalized concatenated code with inner $(2^m, 2^m - j, \delta_j), j = 0, \ldots, 2^m - 1$ codes generated by $\hat{A}_m$, and outer $(2^\mu, k_j, d_j)$ codes $\hat{\mathcal{R}}_j, j = 0, \ldots, 2^m$, such that if $A_{i,-} = \hat{A}_{j,-}$, then $\mathcal{R}_i = \hat{\mathcal{R}}_j$.

Hence, using (11) and properties of GCC construction [18] one obtains

$$d(\mathcal{C}) \ge \min_{j=0,\ldots,2^m-1} \delta_j d_j = \min_{i=0,\ldots,2^m-1} w_i d_i.$$

□

The theorem 1 provides a simple way to construct the code with the required minimum distance $d_{\min}$ by choosing dimensions k_i of $\mathcal{R}_i$, such that $w_i(2^\mu - k_i + 1) \ge d_{\min}$.

B. Constructing codes for erasure channel

The construction method of polar codes for erasure channel, which is described in the original Arikan paper [1], can be immediately generalized to the case of the proposed code construction.

Consider a transmission of $u_0^{n-1}(A_m \otimes R_\mu)$ through the erasure channel $\mathcal{W} : \mathrm{GF}(2^\mu) \to \mathrm{GF}(2^\mu) \cup \{\epsilon\}$ with the erasure probability $p_0^{(0)}$. Assuming that u_0^{n-1} is chosen uniformly at random and the SC decoding algorithm is used, one can compute erasure probabilities $p_m^{(i)}$ of symbols of a codeword $\mathbf{r}^{(i)} = u_{i2^\mu}^{(i+1)2^\mu-1} R_\mu$, $\mathbf{r}^{(i)} \in \mathcal{R}_i$, $i = 0, \ldots, 2^m - 1$ using the following recursion:

$$p_\lambda^{(2i)} = 2 \cdot p_{\lambda-1}^{(i)} - \left(p_{\lambda-1}^{(i)}\right)^2, \quad p_\lambda^{(2i+1)} = \left(p_{\lambda-1}^{(i)}\right)^2. \tag{12}$$

Since the code $\mathcal{R}_i$ is an $(2^\mu, k_i, 2^\mu - k_i + 1)$ extended Reed-Solomon code, at most $2^\mu - k_i$ erasures can be corrected. Therefore, probability of restoring each symbol $u_j, j = i2^\mu, \ldots, (i+1)2^\mu - 1$ is given by

$$P_{i,k_i} = \sum_{t=0}^{2^\mu - k_i} \binom{2^\mu}{k} \left(p_m^{(i)}\right)^t \left(1 - p_m^{(i)}\right)^{2^\mu - t}. \tag{13}$$

Procedure ChooseDimensions is used to construct a $(2^{m+\mu}, k, d \ge d_{\min})$ code with mixed A_m and R_μ kernels for the channel with erasure probability $p_0^{(0)}$. In lines 1–5 erasure probabilities of codewords of outer codes are computed using (12). Then, probabilities $P_{i,j}$ given by (13) are evaluated in lines 6–11. Note, that values $P_{i,0} = 1$ are not considered in the procedure, since any zero-rate outer code corresponds to a block of frozen symbols and does not affect decoding performance.

In lines 15–18 the algorithm looks through the sorted list of $P_{i,j}$ to incrementally compute dimensions of outer codes, such that the minimum distance of the code $d \ge d_{\min}$. If the desired minimum distance $d_{\min}$ is not achievable, than the empty set is returned.

Given the set $\{k_0, \ldots, k_{2^m-1}\}$ of dimensions of outer Reed-Solomon codes $\mathcal{R}_i$, one should define matrices W and V for (9), as it is shown in Example 1.

IV. Decoding

This section describes a generalization of the SCL decoding algorithm [2], which can be applied to a proposed code construction. Consider $(n = 2^{m+\mu}, k)$ polar code over $\mathrm{GF}(2^\mu)$ generated by the matrix $G_{m+\mu} = A_m \otimes R_\mu$. A vector u_0^{n-1} with data symbols on positions $i \notin \mathcal{F}$ is encoded as

$$c_0^{n-1} = u_0^{n-1} W A_{m+\mu} = v_0^{n-1} A_{m+\mu}.$$

Let $a_0, \ldots, a_{\mu-1}$ be the standard basis of $\mathrm{GF}(2^\mu)$, and consider a binary image of v_0^{n-1} given by $v_i = \sum_{j=0}^{\mu-1} v_{i,j} a_j$, $v_{i,j} \in \mathbb{F}_2$. Since $v_0^{n-1} A_{m+\mu} = \sum_{j=0}^{\mu-1} (v_{0,j}, \ldots, v_{n-1,j}) A_{m+\mu} = \sum_{j=0}^{\mu-1} \mathbf{v}_j A_{m+\mu}$, it is possible to evaluate terms $\mathbf{v}_j A_{m+\mu}$ separately.

This is equivalent to the encoding of μ binary vectors $\mathbf{v}_j$ by a binary polar code with Arikan kernel. Hence, the receiver may use μ decoders to estimate $\mathbf{v}_j, j = 0, \ldots, \mu - 1$ (and, consequently, u_0^{n-1}) in parallel, while synchronizing on constraints induced by the matrix W.

Procedure ChooseDimensions($m, \mu, p_0^{(0)}, k, d_{\min}$)

input : log length of the Arikan kernel m,
log length of the Reed-Solomon kernel μ,
channel erasure probability $p_0^{(0)}$,
target code dimension k,
target minimum distance $d_{\min}$
output: set of dimensions k_i of outer Reed-Solomon codes $\mathcal{R}_i, i = 0, \dots, 2^m$

$p_0 \leftarrow p_0^{(0)}$, $p_1 \leftarrow 0, \dots, p_{2^m-1} \leftarrow 0$
for $i \leftarrow 0$ **to** $m-1$ **do**
  for $j \leftarrow 2^i - 1$ **to** 0 **do**
    $p_{2j+1} \leftarrow (p_j)^2$
    $p_{2j} \leftarrow 2p_j - (p_j)^2$
$P_{0,1} \leftarrow 0, \dots, P_{0,2^\mu} \leftarrow 0, P_{1,1} \leftarrow 0, \dots, P_{2^m-1,2^\mu} \leftarrow 0$
for $i \leftarrow 0$ **to** $2^m - 1$ **do** // an index of outer code
  for $n_e \leftarrow 0$ **to** $2^\mu - 1$ **do** // a number of erasures
    $\pi_e \leftarrow \binom{2^\mu}{n_e} (p_i)^{n_e} (1-p_i)^{2^\mu - n_e}$
    for $j \leftarrow 1$ **to** $2^\mu - n_e$ **do**
      $P_{i,j} \leftarrow P_{i,j} + \pi_e$
$\mathbf{P} \leftarrow (P_{0,1}, \dots, P_{2^m-1,2^\mu})$
Sort($\mathbf{P}$) // sort $P_{i,j}$ in descending order
$s \leftarrow 0$, $t \leftarrow 0$, $k_0 \leftarrow 0, \dots, k_{2^m-1} \leftarrow 0$
while $t < k$ **and** $s < 2^{m+\mu}$ **do**
  $(i, j) \leftarrow (i', j') : \mathbf{P}_s = P_{i',j'}$
  if $w_i(2^\mu - k_i) \geq d_{\min}$ **then** $k_i \leftarrow k_i + 1$, $t \leftarrow t + 1$
  $s \leftarrow s + 1$
if $\sum_{i=0}^{2^m-1} k_i < k$ **then return** $\{\}$
else return $\{k_0, \dots, k_{2^m-1}\}$

A. List decoding algorithm

Suppose, that binary vectors $\mathbf{v}_j, j = 0, \dots, \mu - 1$ are transmitted through the channel $\mathcal{W}$. Then, a channel output vector $y_0^{\mu n-1}$ is obtained, and log-likelihood values $S_0^{(jn+i)} = \log \mathcal{W}(y_{jn+i}|v_{i,j} = 0) - \log \mathcal{W}(y_{jn+i}|v_{i,j} = 1)$, $i = 0, \dots, n-1$ are passed to a decoder.

Procedure Decode provides a high-level description of the algorithm. The decoder consists of μ binary SCL decoders running concurrently. Initialization is done in lines 1–3 separately for each component decoder. These decoders operate with the equal list size L_b independently of each other until a synchronization phase $\phi \in \mathcal{S}$, $\mathcal{S} \subset [n]$ is reached. Functions *CalcS* and *UpdateC* in lines 6 and 14 evaluate log-likelihood values $S_{m+\mu}^{\phi}$ and partial sums for each binary path in the set $\mathcal{U}$, as described in section II-B.

In line 7 *SortContinuations* uses values $S_{m+\mu}^{(\phi)}$, which are computed for each path considered by the j-th SCL decoder, to produce sorted lists $\bar{\mathcal{U}}^{(j)}$ of at most $2L_b$ path continuations. Since dynamic freezing constraints given by (7) may involve bits corresponding to different decoders, SCL decoders should consider bits of dynamic frozen symbols unfrozen. Here $\mathcal{F}_s \subset \mathcal{F}$ denotes the set of static frozen symbols, whose values do not depend on data bits.

Lines 5–9 and 14 can be processed by each of the component decoders independently. Synchronization is done in line 12, where the decoder kills all binary paths, that are not involved in a generation of L paths over $\mathrm{GF}(2^\mu)$ with the L highest scores. Finally, in line 15 the path over $\mathrm{GF}(2^\mu)$ with the highest score is returned.

Procedure Decode($\mathcal{F}_s, V, L, L_b, \mathcal{S}, \mathbf{S}_0$)

input : set of static frozen symbols $\mathcal{F}_s$,
constraint matrix V (see the equation (7)),
list size L of the decoder,
list size L_b of component SCL decoders,
maximum number of paths to consider $T_{\max}$,
set of synchronization phases $\mathcal{S}$,
log-likelihood ratios $\mathbf{S}_0 = (S_0^{(0)}, \dots, S_0^{(\mu n-1)})$
output: estimated symbols $\hat{u}_0^{n-1} \in \mathrm{GF}(2^\mu)^n$

$\mathcal{U} \leftarrow \{\mathcal{U}^{(0)} \leftarrow \varnothing, \dots, \mathcal{U}^{(\mu-1)} \leftarrow \varnothing\}$ // paths
for $j \leftarrow 0$ **to** $\mu - 1$ **do**
  $\mathcal{U}^{(j)} \leftarrow$ AssignInitialPath($j, \mathbf{S}_0$)
for $\phi \leftarrow 0$ **to** $n-1$ **do**
  for $j \leftarrow 0$ **to** μ **do**
    CalcS($\mathcal{U}^{(j)}, \phi$)
    $\bar{\mathcal{U}}^{(j)} \leftarrow$ SortContinuations($\mathcal{F}_s, \phi, \mathcal{U}^{(j)}$)
    if $\phi \notin \mathcal{S}$ **then**
      $\mathcal{U}^{(j)} \leftarrow$ KillAndExtendBinary($L_b, \bar{\mathcal{U}}^{(j)}$)
  if $\phi \in \mathcal{S}$ **then**
    $\bar{\mathcal{U}} \leftarrow \{\bar{\mathcal{U}}^{(0)}, \dots, \bar{\mathcal{U}}^{(\mu-1)}\}$
    $\mathcal{U} \leftarrow$ KillAndExtend($V, L, L_b, T_{\max}, \bar{\mathcal{U}}$)
    if $\mathcal{U} = \varnothing$ **then return** $(\epsilon, \dots, \epsilon)$
  for $j \leftarrow 0$ **to** μ **do** UpdateC($\mathcal{U}^{(j)}, \phi$)
return GetBestPath($\mathcal{U}$)

Procedure KillAndExtend from line 12 successively constructs paths over $\mathrm{GF}(2^\mu)$ from sorted lists $\bar{\mathcal{U}}^{(0)}, \dots, \bar{\mathcal{U}}^{(\mu-1)}$ of possible path continuations. The function *Score* returns score $\mathcal{M}$ of a path over $\mathrm{GF}(2^\mu)$ from scores of binary paths provided by the equation (4):

$$\mathcal{M}(\bar{\mathcal{U}}^{(0)}, \dots, \bar{\mathcal{U}}^{(\mu-1)}) = \sum_{j=0}^{\mu-1} M\left(y_{j2^m}^{(j+1)2^m-1}, \bar{\mathcal{U}}^{(j)}\right). \quad (14)$$

Function *IsValid* in line 5 checks if given binary paths satisfy to all constraints, induced by the matrix V. It is assumed, that it takes $O(1)$ time, so the complexity of Procedure KillAndExtend is bounded by $O((L_b)^\mu \cdot q)$, where q is a complexity of finding the maximum in Q. With the growth of L_b and μ the complexity becomes high, so the procedure can be aborted after a fixed number of iterations $T_{\max}$ of the loop.

Procedure KillAndExtend($V, L, L_b, T_{\max}, \bar{\mathcal{U}}$)

input : constraint matrix V,
number of paths L to keep and list size L_b,
maximum number of loop iterations $T_{\max}$,
sorted paths $\bar{\mathcal{U}} = \{\bar{\mathcal{U}}^{(0)}, \ldots, \bar{\mathcal{U}}^{(\mu-1)}\}$

output: set $\mathcal{U}$ of binary paths to keep

$Q \leftarrow \{(0, \ldots, 0)\}$, $l \leftarrow 0$, $t \leftarrow 0$
for $j \leftarrow 0$ **to** $\mu - 1$ **do** $\mathcal{U}^{(j)} \leftarrow \varnothing$
while $Q \neq \varnothing$ **and** $l < L$ **and** $t < T_{\max}$ **do**
  $(i_0, \ldots, i_{\mu-1}) \leftarrow \underset{(j_0,\ldots,j_{\mu-1}) \in Q}{\operatorname{argmax}} \texttt{Score}(\bar{\mathcal{U}}^{(0)}_{j_0}, \ldots, \bar{\mathcal{U}}^{(\mu-1)}_{j_{\mu-1}})$
  if $\texttt{IsValid}(V, \bar{\mathcal{U}}^{(0)}_{i_0}, \ldots, \bar{\mathcal{U}}^{(\mu-1)}_{i_{\mu-1}})$ **then**
    for $j \leftarrow 0$ **to** $\mu - 1$ **do** $\mathcal{U}^{(j)} \leftarrow \mathcal{U}^{(j)} \cup \left\{\bar{\mathcal{U}}^{(j)}_{i_j}\right\}$
    $l \leftarrow l + 1$
  for $j \leftarrow 0$ **to** $\mu - 1$ **do**
    if $i_j + 1 < L_b$ **then**
      $Q \leftarrow Q \cup \{(i_0, \ldots, i_j + 1, \ldots, i_{\mu-1})\}$
  $Q \leftarrow Q \setminus \{(i_0, \ldots, i_{\mu-1})\}$, $t \leftarrow t + 1$
return $\{\mathcal{U}^{(0)}, \ldots, \mathcal{U}^{(\mu-1)}\}$

B. Decoding of polar codes with mixed kernels

To fully utilize the error correcting capabilities of the proposed codes, maximum-likelihood decoding of the outer Reed-Solomon codes should be applied. Recall, that a path corresponding to a transmitted vector (a correct path) is composed by μ binary paths, that are extended independently. Note, that the list decoder loses the correct path only if any of the constituent SCL decoders removes the correct binary path from the list.

To ensure maximum-likelihood decoding of the outer code $\mathcal{R}_i$ of length 2^μ it is necessary to guarantee, that the correct path remains in the list until the last frozen symbol in a block $u_{i2^\mu}^{(i+1)2^\mu-1} \in \mathrm{GF}(2^\mu)^{2^\mu}$ is processed.

Therefore, one may simply set $L_b = L \cdot 2^{2^\mu}$, $\mathcal{S} = \{2^\mu - 1, 2 \cdot 2^\mu - 1, \ldots, 2^m \cdot 2^\mu - 1\}$, and do not limit $T_{\max}$ in Procedure KillAndExtend. However, simulations show, that on practice one can still use small SCL list size L_b and limit $T_{\max}$, which results in suboptimal decoding of $\mathcal{R}_i$ but substantial complexity reduction.

It can be seen, that decoding latency depends on the number of loop iterations in KillAndExtend. When a synchronization phase $\phi = i2^\mu - 1$ is reached, the decoder aims to generate L valid non-binary paths, i.e. paths, which satisfy dynamic freezing constraints induced by outer codes $\mathcal{R}_0, \ldots, \mathcal{R}_{i-1}$. Observe, that increasing number of constraints lowers total number of valid paths, which can be possibly generated during KillAndExtend. As a consequence, decoding latency scales with the number of constraints considered by *IsValid*.

In order to limit number of constraints considered by *IsValid* we propose to keep track of L valid non-binary paths. Hence, on the next synchronization phase $\phi + 2^\mu$ it is possible to construct L sets of continuations of corresponding L paths generated on the phase ϕ. Therefore, one may run L copies of Procedure KillAndExtend independently for each set and then select L paths with highest scores. Hence, the total number of constraints considered by *IsValid* can be limited by $2^{\mu-1} - 1$.

V. Simulation results

Figures 1 and 2 illustrate performance of $(1024, 512)$ polar codes with mixed kernels over $\mathrm{GF}(2^\mu)$, where $\mu = 1, 2, 3$, constructed for the erasure probability 0.15 and 0.2, and $d_{\min} = 24$. It can be seen, that the proposed decoding algorithm provides better decoding performance with the increase of μ. Dashed lines on Figure 1 correspond to the scenario, where 2 or 3 SC decoders operate in parallel without any synchronization. The proposed list decoding algorithm outperforms these scenarios at the cost of increased complexity ($L_b = 8$ for the case of μ = 3).

Note, that the list size, required for the ML decoding of Reed-Solomon codes of length 8 is $L_b = 256$. Low value L_b results in low complexity overhead, however, it can be seen from Figure 2, that for the case $\mu = 3$ and $L \in \{2, 4\}$ list size $L_b = 32$ leads to approximately equal performance. Hence, with increase of L list size of binary SCL decoders L_b should be also increased. Even smaller complexity overhead by optimizing the set of synchronization phases $\mathcal{S}$ and developing more efficient synchronization methods. However, these directions require further research.

It is possible to estimate decoding latency of (n, k) code over $\mathrm{GF}(2^\mu)$ in the following way. Latency of SCL decoding is equal to $\mathcal{L}_1 = \mathcal{S}(2L) \cdot (k - \lfloor \log_2(L) \rfloor) + \mathcal{I} \cdot n$, where $\mathcal{S}(2L)$ is the latency of selecting L paths with highest score, and $\mathcal{I}$ is the average latency of the decoding iteration without path selection.

Latency of the proposed decoding algorithm can be estimated as $\mathcal{L}_2 = \mathcal{S}(2L_b) \cdot N_{sort} + L\mathcal{S}(\mu) \cdot N_t + (\mathcal{E} + L\mathcal{S}(\mu)) \cdot N_d + \mathcal{I} * n$, where N_{sort} is the number of phases where component SCL decoders perform selection of L_b paths with highest scores, N_d is the number of outer codes inducing freezing constraints, $N_t = n/2^\mu - N_d$, and $\mathcal{E}$ denotes average latency of KillAndExtend.

Simulations show, that above considered $(1024, 512)$ code over $\mathrm{GF}(2^3)$ requires decoding of $N_d = 33$ blocks with dynamic frozen symbols, leading to $N_{sort} = 184$, where $L = 4$ and $\mathcal{E} = 20$. Under the assumption that $\mathcal{I} = 1$ and selection of paths is performed by bitonic sorting network with time complexity $O(\log_2^2(n))$, one obtains $\mathcal{S}(2L_b) = 36$ and $\mathcal{L}_2 = 9332$. With $L_b = 64$, $\mathcal{S}(128) = 49$ and $N_{sort} = 111$ decoding latency becomes $\mathcal{L}_2 = 8147$, which is substantially lower, than the latency $L_1 = 6 \cdot 1534 + 4096 = 13300$ of SCL decoding of $(3072, 1536)$ punctured binary code.

VI. Conclusion

In this paper a method for construction and decoding of polar codes with mixed Arikan and RS kernels over $\mathrm{GF}(2^\mu)$ was proposed. The construction was shown to provide reduced decoding latency compared to the case of Arikan polar codes

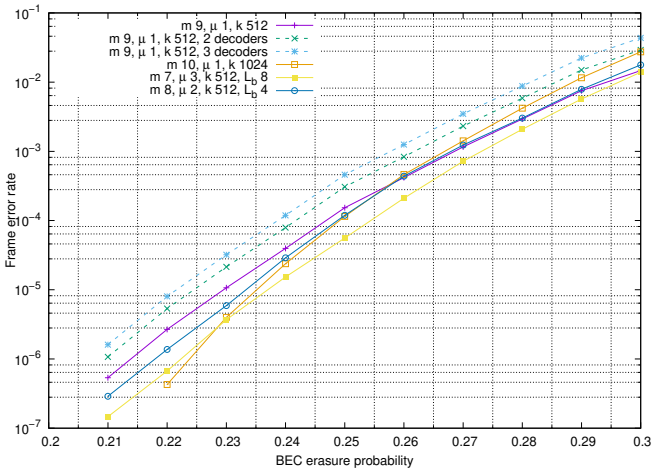

Figure 1: Performance of polar codes with mixed kernels in BEC. $L = 1, p_0 = 0.15, T_{\max} = \infty$

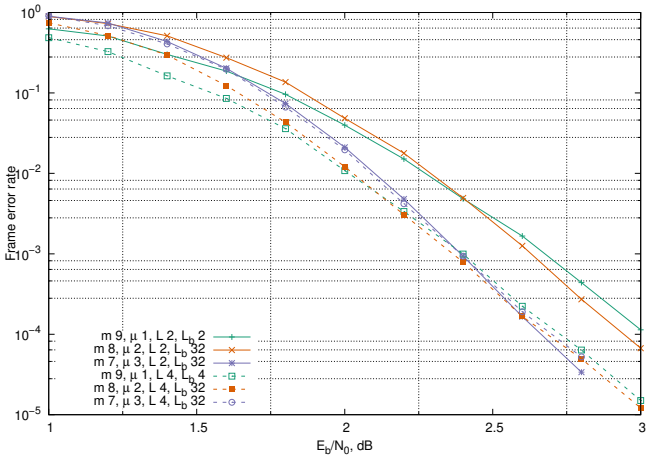

Figure 2: Performance of polar codes with mixed kernels in AWGN channel. $p_0 = 0.2, T_{\max} = 1024$

with the same performance. The proposed list decoding algorithm can be used to run several SCL decoders in parallel and it is most applicable for the case of small μ and list size L.

References

[1] 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.

[2] "List decoding of polar codes", *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[3] V. Miloslavkaya and P. Trifonov, "Sequential decoding of polar codes", *IEEE Communications Letters*, vol. 18, no. 7, pp. 1127–1130, Jul. 2014, ISSN: 1089-7798.

[4] N. Doan, S. A. Hashemi, M. Mondelli, and W. J. Gross, "On the decoding of polar codes on permuted factor graphs", pp. 1–6, Dec. 2018.

[5] S. A. Hashemi, N. Doan, M. Mondelli, and W. J. Gross, "Decoding reed-muller and polar codes by successive factor graph permutations", pp. 1–5, Dec. 2018.

[6] M. Kamenev, Y. Kameneva, O. Kurmaev, and A. Maevskiy, "Permutation decoding of polar codes", *arXiv:1901.05459 [cs, math]*, arXiv: 1901.05459.

[7] A. Elkelesh, M. Ebada, S. Cammerer, and S. ten Brink, "Improving belief propagation decoding of polar codes using scattered EXIT charts", in *2016 IEEE Information Theory Workshop (ITW)*, IEEE, Sep. 2016, pp. 91–95.

[8] S. Cammerer, M. Ebada, A. Elkelesh, and S. ten Brink, "Sparse graphs for belief propagation decoding of polar codes", pp. 1465–1469, Jun. 2018.

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

[10] M. Mondelli, S. H. Hassani, and R. Urbanke, "Unified scaling of polar codes: Error exponent, scaling exponent, moderate deviations, and error floors", *IEEE Trans. Inf. Theory*, vol. 62, no. 12, pp. 6698–6712, Dec. 2016.

[11] S. B. Korada, E. Sasoglu, and R. Urbanke, "Polar codes: Characterization of exponent, bounds, and constructions", *IEEE Trans. Inf. Theory*, vol. 56, no. 12, pp. 6253–6264, Dec. 2010.

[12] R. Mori and T. Tanaka, "Channel polarization on q-ary discrete memoryless channels by arbitrary kernels", pp. 894–898, Jan. 15, 2010.

[13] P. Trifonov, "Binary successive cancellation decoding of polar codes with reed-solomon kernel", in *2014 IEEE International Symposium on Information Theory*, IEEE, Jun. 2014, pp. 2972–2976.

[14] N. Presman, O. Shapira, and S. Litsyn, "Polar codes with mixed kernels", in *2011 IEEE International Symposium on Information Theory Proceedings*, Jul. 2011, pp. 6–10.

[15] N. Presman, O. Shapira, and S. Litsyn, "Mixed-kernels constructions of polar codes", *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 239–253, Feb. 2016.

[16] F. Gabry, V. Bioglio, I. Land, and J. Belfiore, "Multi-kernel construction of polar codes", in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2017, pp. 761–765.

[17] V. Bioglio, I. Land, F. Gabry, and J. Belfiore, "Flexible design of multi-kernel polar codes by reliability and distance properties", in *2018 IEEE 10th International Symposium on Turbo Codes Iterative Information Processing (ISTC)*, Dec. 2018, pp. 1–5.

[18] V. A. Zinoviev, «Generalized concatenated codes», Problems of Inf. Transmission, т. 12, с. 2—9, 1 1976.

[19] N. Hussami, S. B. Korada, and R. Urbanke, "Performance of polar codes for channel and source coding", in *2009 IEEE International Symposium on Information Theory*, Jun. 2009, pp. 1488–1492.