Hybrid Decoding of Interlinked Generalized Concatenated Codes

Kirill Ivanov, Peter Trifonov
Saint Petersburg Polytechnic University
Email: {ivanov,petert}@dcn.icc.spbstu.ru

***Abstract*—The problem of efficient decoding of interlinked generalized concatenated codes and polar subcodes is considered. A novel decoding algorithm is proposed, which allows one to significantly reduce the complexity of block sequential decoding of these codes.**

I. Introduction

Polar codes, introduced in 2008 by E. Arikan, are able to achieve the capacity of an arbitrary binary memoryless channel [1]. Low-complexity procedures can be used for their coding and decoding. However, polar codes show quite poor performance due to their low minimum distance.

Polar subcodes [2], [3] are a generalization of polar codes with improved minimum distance. Instead of setting all frozen symbols to zero, as in the case of classical polar codes, polar subcodes allow some of the frozen symbols to be a linear combination of non-frozen ones. These codes can be efficiently decoded by the algorithms developed for classical polar codes with minor modifications.

Polar subcodes themselves can be considered as a special case of the interlinked GCC (IGCC), which are an extension of generalized concatenated codes. In [3], a construction of IGCC with outer extended BCH codes was proposed and was shown to outperform polar subcodes.

One can decode these codes with the block sequential decoding algorithm [4]. It represents a polar code as a Plotkin concatenation of some outer codes, and decodes these codes with maximum likelihood soft list decoders. A priority queue is used in order to store different paths in the code tree, and select the most probable one.

Straightforward implementation of this approach is based on the list Viterbi algorithm for decoding of outer codes. However, its complexity grows exponentially with the length of outer codes. In this paper we propose more efficient techniques for decoding of IGCC with outer extended BCH codes.

II. Background

A. Interlinked Generalized Concatenated Codes

1) Polar subcodes: $(n = 2^m, k, d)$ polar subcode is a set of vectors

$$c_0^{n-1} = u_0^{n-1} A_m,$$

where

$$u_{j_i} = \sum_{s=0}^{j_i-1} u_s V_{i,s}, 0 \leq i < n-k, \tag{1}$$

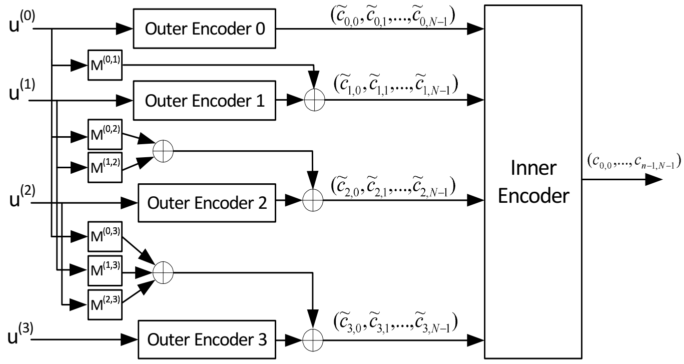

Figure 1. IGCC encoder

for some $j_i \in \{0, \ldots, n-1\}, V_{i,s} \in \mathbb{F}_2, A_m = F^{\otimes m}$, where $F = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$. The coefficients $V_{i,s}$ are selected in such way, so that any c_0^{n-1} is also a codeword of some parent $(n, k' > k, d' \leq d)$ code. Extended primitive narrow-sense BCH codes were shown to be good candidates for parent codes [3].

2) Generalized concatenated code: Consider a family of nested inner (n, k_i, d_i) codes $\mathcal{C}_i : \mathcal{C}_0 \supset \mathcal{C}_1 \supset ... \supset \mathcal{C}_{n-1}$, where $k_i = k_{i+1} + 1$, $k_n = 0$, and a family of outer (N, K_i, D_i) codes $\mathbb{C}_i$ over $\mathbb{F}_q$, $0 \leqslant i < n$. Let $\mathcal{G}$ be a $n \times n$ matrix, such that its rows $i, ..., n-1$ generate code $\mathcal{C}_i$. Encoding with a generalized concatenated code (GCC) is performed as follows. First, partition a data vector into n blocks of size $K_i, 0 \leqslant i < n$. Second, encode these blocks with codes $\mathbb{C}_i$ to obtain codewords $(c_{i,0}, ..., c_{i,N-1})$. Finally, multiply vectors $(c_{0,j}, ..., c_{n-1,j}), 0 \leqslant j < N$, by $\mathcal{G}$ to obtain a GCC codeword.

3) Interlinked GCC: The above described GCC construction can be extended as follows [3]. Subvectors of a data vector can be encoded not with codes $\mathbb{C}_i$, but with their cosets given by $\mathbb{C}_i + \sum_{s=0}^{i-1} u^{(i)} M^{(s,i)}$, where $M^{(s,i)} \in \mathbb{F}_q^{K_s}$ are some matrices, as shown in Figure 1. This construction will be referred to as interlinked GCC (IGCC).

Decoding of IGCC can be performed with the multistage decoding algorithm [5]. Decoding in cosets of outer codes can be implemented using standard decoders of $\mathbb{C}_i$, provided that their input LLRs are appropriately adjusted. IGCC representation enables one to use GCC decoding techniques for other types of error-correcting codes.

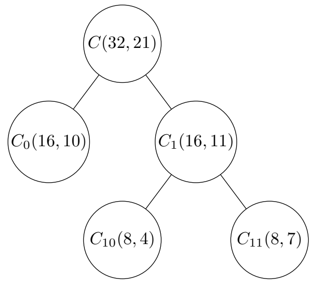

Figure 2. Recursive decomposition of code $(32, 21, 6)$

4) Generalized Plotkin Decomposition: It was shown in [3] that any $(2n, k, d)$ linear code $\mathcal{C}$ has a generator matrix given by

$$G = \begin{pmatrix} I_{k_0} & 0 & \tilde{I} \\ 0 & I_{k_1} & 0 \end{pmatrix} \begin{pmatrix} G_0 & 0 \\ G_1 & G_1 \\ G_2 & G_2 \end{pmatrix}, \tag{2}$$

where I_l is a $l \times l$ identity matrix, $G_i, i = 0, 1, 2$, are some $k_i \times n$ matrices, $\tilde{I}$ is obtained by stacking a $(k_0 - k_2) \times k_2$ zero matrix and I_{k_2}, where $k_2 \leqslant k_0$. Such representation, known as generalized Plotkin decomposition of code $\mathcal{C}$, essentially corresponds to an IGCC with outer codes $\mathbb{C}_0$ and $\mathbb{C}_1$ generated by G_0 and G_1, respectively, and inner codes are generated by rows of polarizing transformation matrix F.

GPD can be applied recursively. Figure 2 presents an example of the codes obtained by recursive GPD of a $(32, 21, 6)$ extended BCH code.

B. Block sequential decoding of polar codes

Block sequential decoding is an improvement of the sequential decoding method for polar and IGCC codes [6]. It recursively applies GPD to a polar code, until one obtains outer codes which admit efficient maximum likelihood soft decision list decoding. Let n_s be the lengths of the obtained outer codes $\mathcal{C}_{B_s}$. For example, the extended $(32, 21, 6)$ BCH code is decomposed into codes with $n_s = \{16, 8, 8\}$, as shown in Figure 2.

The decoder maintains a stack (priority queue), where paths $u_0^{\phi_s - 1}$ are stored together with their scores. At each iteration the decoder extracts a path with the highest score, constructs its possible continuations $u_0^{\phi_s + n_s - 1}$, where $\phi_{s+1} = \phi_s + n_s$, computes their scores, and pushes them into the stack. Construction of path continuations reduces to finding a number of most probable codewords of an outer code, corresponding to a given vector of log-likelihood ratios. In order to reduce the complexity, the codewords of outer codes are recovered one-by-one, as described in [4]. Furthermore, the decoder is allowed to construct at most L paths of any length ϕ_s. L can be considered as an equivalent of list size in the Tal-Vardy algorithm [7].

Path score is defined as

$$M(u_0^{\phi-1}, y_0^{n-1}) = R(u_0^{\phi-1}, y_0^{n-1}) + \psi(\phi - 1),$$

where

$$R(u_0^{\phi-1}, y_0^{n-1}) = \max_{u_\phi^{n-1} \in \mathbb{F}_2^{n-\phi}} \log W_m^{(n-1)}(u_0^{n-1} | y_0^{n-1}) + \rho,$$

where $W_m^{(n-1)}(u_0^{n-1} | y_0^{n-1}) = \prod_{j=0}^{n-1} W((u_0^{n-1})_j | y_j)$, ρ does not depend on $u_0^{\phi-1}$ and $\psi(\phi-1)$ is a heuristic function, which enables one to compare paths of different length stored in the stack.

The performance and complexity of the block sequential decoder (BSD) strongly depends on the efficiency of outer decoders. If an IGCC with outer codes of length N is decoded with the BSD, one needs to implement low complexity ML list decoders with list size l for all outer codes, where l is a parameter. Any outer decoder should meet following requirements. It provides two procedures, *preprocess* and *nextCW*. The former one performs some preprocessing of the input LLR vector $\hat{y}_0^{n_s - 1}$. The latter procedure on the i-th call should produce a pair $(c^{(i)}, E(c^{(i)}, \hat{y}_0^{n_s - 1})), 0 \leq i < l, c^{(i)} \in \mathcal{C}_{B_s}$ so that the obtained codewords are arranged in the ascending order of their ellipsoidal weights, and $E(c^{(i)}, \hat{y}_0^{n_s - 1})$ are the smallest possible.

III. Efficient decoding of outer codes

In this section we present an efficient algorithm, which can be used for implementation of the outer decoders of short extended BCH codes. The proposed algorithm is also based on the GPD.

A. Soft decision decoding of codes via their generalized Plotkin decomposition

Soft-decision decoding of code $\mathcal{C}$ can be performed as follows. Let y_0^{2n-1} be the vector of log-likelihood ratios corresponding to the result of transmission of codeword $c_0^{2n-1} \in \mathcal{C}$ over a memoryless channel. The ellipsoidal weight (correlation discrepancy) of vector z_0^{2n-1} is defined as

$$E(z_0^{2n-1}, y_0^{2n-1}) = \sum_{\substack{i=0 \\ (-1)^{z_i} \neq \operatorname{sgn} y_i}}^{2n-1} |y_i|.$$

Maximum likelihood decoding is equivalent to finding

$$\hat{\mathbf{c}} = \arg\min_{\mathbf{c} \in \mathcal{C}} E(\mathbf{c}, y_0^{2n-1}).$$

Let

$$\tilde{y}_i = Q(y_i, y_{i+n}) = \operatorname{sgn}(y_i) \operatorname{sgn}(y_{i+n}) \min(|y_i|, |y_{i+n}|), \tag{3}$$

$0 \leq i < n$, and let $\tilde{\mathbf{c}} = \tilde{x} G_0$ be a codeword of $\mathbb{C}_0$. Let us further define

$$\overline{y}_i = P(\tilde{c}_i, y_i, y_{i+n}) = (-1)^{\tilde{c}_i} y_i + y_{i+n}, 0 \leq i < n. \tag{4}$$

Let $\tilde{\mathbf{c}}$ be an element of $\mathbb{C}_1 + \tilde{x} G_2$.

Lemma 1: For any $\hat{\mathbf{c}} = (\tilde{\mathbf{c}} + \overline{\mathbf{c}} | \overline{\mathbf{c}})$

$$E(\hat{\mathbf{c}}, y_0^{2n-1}) = E(\tilde{\mathbf{c}}, \tilde{y}) + E(\overline{\mathbf{c}}, \overline{y})$$

Proof: It is sufficient to prove the statement for $n = 1$, so we can drop subscripts i. It is also sufficient to consider the case of $\tilde{c} = \overline{c} = 0$. Consider the following cases:

- $y_0, y_1 > 0$: It can be seen that $\tilde{y}, \overline{y} > 0$, so $E(0,\tilde{y}) = E(0,\overline{y}) = E((0|0),(y_0|y_1)) = 0$.
- $y_0, y_1 < 0$: One obtains $\tilde{y} > 0, \overline{y} = y_1 + y_0 < 0$, so $E(0,\tilde{y}) = 0, E(0,\overline{y}) = E((0|0),(y_0|y_1)) = |y_0| + |y_1|$.
- $y_0 > 0 > y_1$: One obtains $E(0,\tilde{y}) = \min(|y_0|,|y_1|)$, and $E(0,\overline{y}) = \begin{cases} |y_0+y_1|, & \text{if } |y_1| > |y_0| \\ 0 & \text{otherwise.} \end{cases}$ In both cases $E(0,\tilde{y}) + E(0,\overline{y}) = E(0,y_0) + E(0,y_1)$
- The case of $y_1 > 0 > y_0$ can be shown in a similar way.

■

Hence, the maximum likelihood decoding problem for code $\mathcal{C}$ can be equivalently stated as

$$\hat{\mathbf{c}} = (\tilde{\mathbf{c}} + \overline{\mathbf{c}}|\overline{\mathbf{c}}) = \arg \max_{\substack{\tilde{\mathbf{c}} \in \mathbb{C}_0 \\ \overline{\mathbf{c}} \in \mathbb{C}_1 + \tilde{u} G_2}} (E(\tilde{\mathbf{c}},\tilde{y}) + E(\overline{\mathbf{c}},\overline{y})) \quad (5)$$

It can be verified using the above lemma that the function $R(u_0^{\phi-1}, y_0^{n-1})$, which is used in the block sequential decoding algorithm, can be calculated as

$$R(u_0^{\phi_{s+1}-1}, y_0^{n-1}) = R(u_0^{\phi_s-1}, y_0^{n-1}) - E(u_{\phi_s}^{\phi_s+n_s-1}, \hat{y}_0^{n_s-1}),$$

where $\hat{y}_0^{n_s-1}$ is recursively computed from the received LLRs as follows. Let B_s be the label of the s-th outer code in the GPD tree (see Figure 2). Then

$$\hat{y}_i = T(B_s, y_0^{n-1}),$$

where

$$\begin{aligned} T(B.0, y_0^{n-1}) &= Q(T(B, y_0^{n/2-1}), T(B, y_{n/2}^{n-1})) \\ T(B.1, y_0^{n-1}) &= P(c^{(B.0)}, T(B, y_0^{n/2-1}), T(B, y_{n/2}^{n-1})) \\ T(\emptyset, y_0^{n-1}) &= y_0^{n-1}. \end{aligned}$$

Here application of functions Q and P to vectors denotes the vector obtained by their elementwise application to the arguments, and $c^{(B.0)}$ denotes the corresponding codeword of the code $\mathcal{C}_{B.0}$.

B. Hybrid decoding algorithm

We propose a novel algorithm for decoding of linear codes, which can be considered as a generalization of the classical method for decoding of codes obtained via the Plotkin construction.

Consider a linear $(N = 2^M, k, d)$ code $\mathcal{C}$ with generator matrix G. Let G_0 and G_1 be the generator matrices of *second order outer codes* $\mathbb{C}_0, \mathbb{C}_1$ obtained by applying the GPD to code $\mathcal{C}$. Let us assume that list decoders for $\mathbb{C}_0$ and $\mathbb{C}_1$ are available. Such decoders may be obtained either by recursive application of the below described algorithm, or using some other techniques, e.g. list Viterbi decoder [8]. The proposed approach makes use of one instance of the decoder for code $\mathbb{C}_0$ and l instances of the decoder for code $\mathbb{C}_1$.

Given a vector of log-likelihood ratios y_0^N, a list of (approximately) most probable codewords of $\mathcal{C}$ can be obtained by computing $\tilde{y}_i = Q(y_i, y_{i+N/2}), 0 \le i < N/2$, finding most probable codewords $\tilde{c}^{(j)} = \tilde{u}^{(j)} G_0 \in \mathbb{C}_0, 0 \le j < l$, computing

$$\overline{y}_i^{(j)} = (-1)^{(\tilde{u}^{(j)} \tilde{I} G_2)_i} P(\tilde{c}_i^{(j)}, y_i, y_{i+N/2}),$$

and decoding each $\overline{y}^{(j)}$ in code $\mathbb{C}_1$ to obtain $\overline{c}^{(j,t_j)} \in \mathbb{C}_1, 0 \le t_j < l$. Finally, one should construct codewords $(\tilde{c}^{(j)} + \overline{c}^{(j,t_j)} + \tilde{u}^{(j)} \tilde{I} G_2, \overline{c}^{(j,t_j)} + \tilde{u}^{(j)} \tilde{I} G_2) \in \mathcal{C}$ and select l of them with the smallest values of $E(\tilde{c}^{(j)}, \tilde{y}) + E(\overline{c}^{(j,t_j)}, \overline{y}^{(j)})$. Observe that it may happen that the obtained list does not contain all l most probable codewords of $\mathcal{C}$, since some of them may correspond to codewords $\tilde{c}^{(j)} \in \mathbb{C}_0, j \ge l$. The probability of such event decreases with l and increases with the rate of $\mathbb{C}_0$.

The algorithm can be further simplified. Since the block sequential decoding algorithm requires obtaining codewords of outer codes one-by-one, it is possible to construct $\overline{c}^{(j,t_j)}$ on demand.

Let us assume that t_j codewords of $\mathbb{C}_1$ have been obtained corresponding to some $\tilde{c}^{(j)}$. Let e_j be a lower bound on the ellipsoidal weight of a codeword of code $\mathcal{C}$, which corresponds to $\tilde{c}^{(j)} \in \mathbb{C}_0$ and $\overline{c}^{(j,t_j)} \in \mathbb{C}_1$. It can be seen from Lemma 1 that

$$e_j = \begin{cases} E(\tilde{c}^{(j)}, \tilde{y}), & t_j = 0 \\ E(\tilde{c}^{(j)}, \tilde{y}) + E(\overline{c}^{(j,t_j-1)}, \overline{y}^{(j)}), & t_j > 0. \end{cases}$$

The proposed algorithm is presented at Figure 3. $PreprocessHybrid$ subroutine performs data structures initialization and preprocessing of log-likelihood ratios corresponding to code $\mathbb{C}_0$. In the $NextCWHybrid$ procedure the index q minimizing $(E(\tilde{c}^{(q)}, \tilde{y}) + E(\overline{c}^{(q,t_q)}, \overline{y}^{(q)}))$ is found and corresponding codeword c with its ellipsoidal weight w is returned. $Preprocess(\mathbb{C}_i, y)$ denotes a call to the corresponding procedure of code $\mathbb{C}_i$ decoder, where y is a log-likelihood ratios vector. $NextCW(\mathbb{C}_i, y)$ returns a pair $(c^{(j)}, E(c^{(j)}, y)), 0 \le j < l, c^{(j)} \in \mathbb{C}_i$ and additionally a vector of information symbols $u^{(j)}$. If the element p_j of length-l array p is zero, the decoder of code $\mathbb{C}_1$ corresponding to codeword $\tilde{c}^{(j)}$ of code $\mathbb{C}_0$ has returned less than l codewords. Thus, one should consider only codewords $\tilde{c}^{(j)}$ corresponding to $p_j = 0$. If $\tilde{w}^{(k)}$ is the minimum weight found among the pairs of codewords $(\tilde{c}^{(j)}, \overline{c}^{(j,t_j)}), 0 \le j < k$ and $\tilde{w}^{(k)} < E(\tilde{c}^{(k+1)})$, one does not need to compute $\overline{c}^{(k+1,0)}$ and $\tilde{c}^{(k+2)}$. In order to implement this approach, the auxiliary array f of length l is used, where

$$f_j = \begin{cases} 0, \text{ if } \tilde{c}^{(j)} \text{ was not obtained yet} \\ 1, \text{ if } \tilde{c}^{(j)} \text{ was already obtained and } \overline{c}^{(j,0)} \text{ was not} \\ 2, \text{ otherwise.} \end{cases}$$

The complexity of the proposed decoding algorithm depends on the complexities of the decoders of codes $\mathbb{C}_i$. Let the complexities of the preprocessing and codeword recovery algorithms for these codes be $f_i(N,l)$, $b_i(N,l)$, respectively. The PreprocessHybrid procedure consists of $\tilde{y}$ calculation, their preprocessing in code $\mathbb{C}_0$ decoder and arrays initialization. Hence, its complexity is $O(N + f_0(N,l) + l)$. The NextCWHybrid procedure complexity is a random variable,

PREPROCESSHYBRID$(\mathcal{C}, y_0^{N-1})$
$\tilde{y}_i = Q(y_i, y_{i+N/2}), 0 \leq i < N/2$
PREPROCESS$(\mathbb{C}_0, \tilde{y})$
for $j \leftarrow 0$ **to** $l-1$
do $t_j \leftarrow 0; f_j \leftarrow 0; p_j \leftarrow 0$

NEXTCWHYBRID$(\mathcal{C}, y_0^{N-1})$
$w \leftarrow \infty$
for $j \leftarrow 0$ **to** $l-1$
do if $f_j = 0$
  then $(\tilde{c}^{(j)}, \tilde{u}^{(j)}) =$ NEXTCW$(\mathbb{C}_0, \tilde{y})$
    $E_j = E(\tilde{c}^{(j)}, \tilde{y});\ \ \hat{c}^{(j)} = \tilde{u}^{(j)}\tilde{I}G_2;\ \ f_j \leftarrow 1$
  if $f_j = 1$
   then if $E_j \leq w$
     then for $i \leftarrow 0$ **to** $\frac{N}{2}-1$
       do $\overline{y}_i^{(j)} \leftarrow P(\tilde{c}_i^{(j)}, y_i, y_{i+\frac{N}{2}})$
         $\overline{y}_i^{(j)} \leftarrow (-1)^{\hat{c}_i^{(j)}}\overline{y}_i^{(j)}$
       PREPROCESS$(\mathbb{C}_1, \overline{y}^{(j)})$
       $(\overline{c}^{(j,0)}, \tilde{u}^{(j)}) =$ NEXTCW$(\mathbb{C}_1, \overline{y}^{(j)})$
       $e_j \leftarrow E_j + E(\overline{c}^{(j,0)}, \overline{y}^{(j)})$
       $f_j \leftarrow 2$
     else **break**
  if $f_j = 2 \wedge p_j = 1 \wedge e_j \leq w$
   then $w \leftarrow e_j; q \leftarrow j$
$c \leftarrow (\overline{c}^{(q)} + \overline{c}^{(q,t_q)} + \hat{c}^{(q)}; \overline{c}^{(q,t_q)} + \hat{c}^{(q)})$
$t_q \leftarrow t_q + 1$
if $t_q < l$
  then $\overline{c}^{(q,t_q)} =$ NEXTCW$(\mathbb{C}_1, \overline{y}^{(q)})$
    $e_q \leftarrow E_q + E(\overline{c}^{(q,t_q)}, \overline{y}^{(q)})$
  else $p_q \leftarrow 1$
return c

Figure 3. Hybrid decoding algorithm

which depends on the noisy vector being decoded and decoding error probability of $\mathbb{C}_0$. In the worst case, all l codewords $\tilde{c}^{(j)}$ of code $\mathbb{C}_0$ are obtained and corresponding log-likelihood ratios $\overline{y}^{(j)}$ and codewords $\overline{c}^{(j,0)}$ of code $\mathbb{C}_1$ are calculated. This goes into the $O(l(b_0(N,l) + N + f_0(N,l) + b_1(N,l)))$ complexity. However, in most cases there are only few calls to $\mathbb{C}_i$ decoders procedures.

Although the proposed algorithm can be applied to any code, its efficiency depends on the parameters of the codes arising in the GPD of the considered code. Extended primitive narrow sense BCH codes are known to be subcodes of Reed-Muller codes [9]. This ensures that the dimension of $\mathbb{C}_0$ and its (list) decoding error probability are sufficiently small, so that with high probability all l most probable codewords of $\mathcal{C}$ can be obtained from l most probable codewords of $\mathbb{C}_0$.

C. Decoding of second order outer codes

The above described decoding algorithm can be applied recursively. This corresponds to recursive GPD (see Figure 2). The recursion should be terminated as soon as the GPD results in codes, which admit efficient decoding. Alternatively, generic decoding algorithms (e.g., tree-trellis Viterbi algorithm [8], list box-and-match algorithm [10]) can be used.

The following techniques can be used for decoding of short second order outer codes arising in the GPD.

1) Codes of dimension $k \leq 2$ can be decoded by exhaustive enumeration of all codewords.
2) First order Reed-Muller codes can be decoded using the fast Hadamard transform [11]. The same approach can be used for codes which can be represented as a union of a few cosets of a first order Reed-Muller code.
3) Single parity check and rate-1 codes can be decoded by flipping a few least reliable bits in a hard decision vector.

Table I
OUTER EBCH CODES

$\mathcal{C}$	$\mathbb{C}_0$		$\mathbb{C}_1$	
	Parameters	Decoding	Parameters	Decoding
(32, 8)	(16,2)	(1)	(16,6)	(2)
(32, 10)	(16,4)	(1)	(16,6)	(2)
(32, 11)	(16,5)	(1)	(16,6)	(2)
(32, 15)	(16,4)	(1)	(16,11)	GR
(32, 19)	(16,8)	GR	(16,11)	GR
(32, 21)	(16,10)	GR	(16,11)	GR
(32, 24)	(16,9)	GR	(16,15)	(3)
(32, 26)	(16,11)	GR	(16,15)	(3)
(32, 27)	(16,11)	GR	(16,16)	(3)
(32, 28)	(16,13)	GR	(16,15)	(3)
(32, 29)	(16,13)	GR	(16,16)	(3)

Table I presents various outer codes of length 32 arising in the GPD of an $(1024, 512, 28)$ IGCC presented in [3], as well as the corresponding decoding methods from the above list. Letters GR correspond to recursive application of the hybrid decoding algorithm or a generic decoder.

IV. NUMERICAL RESULTS

In this section we present numeric results illustrating the complexity and performance of the proposed algorithm. LTE turbo codes, polar codes with 16-bit CRC (Arikan-CRC) [12], polar subcodes and IGCC codes were considered. Polar subcodes and IGCC codes were constructed as described in [3]. Block sequential decoding algorithm [4] (BSD(s)) was used for decoding of polar codes with CRC, polar subcodes and IGCC, where 2^s is the minimal length of outer codes which are decoded using a maximum likelihood decoding algorithm. We consider the following algorithms for decoding of such outer codes:

- tree-trellis list Viterbi algorithm [8].
- the proposed decoding algorithm, where second order outer codes are decoded with the tree-trellis list Viterbi algorithm.
- the proposed decoding algorithm, which is applied recursively, until codes from the list in Section III-C are obtained.

Figures 4 and 5 illustrate the performance and the decoding complexity of various codes. The complexity is expressed in

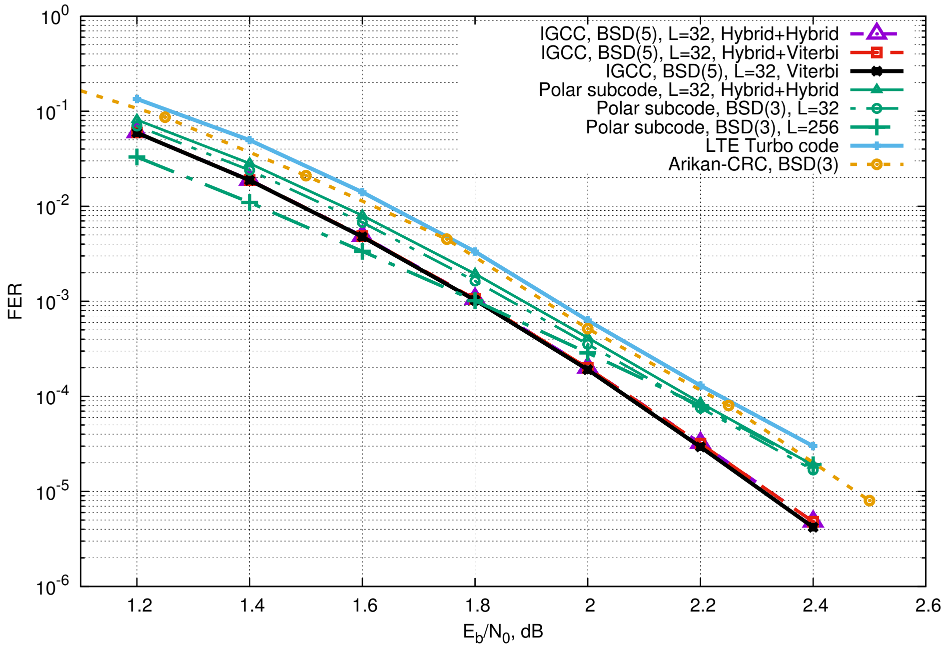

Figure 4. Performance of (1024, 512) codes

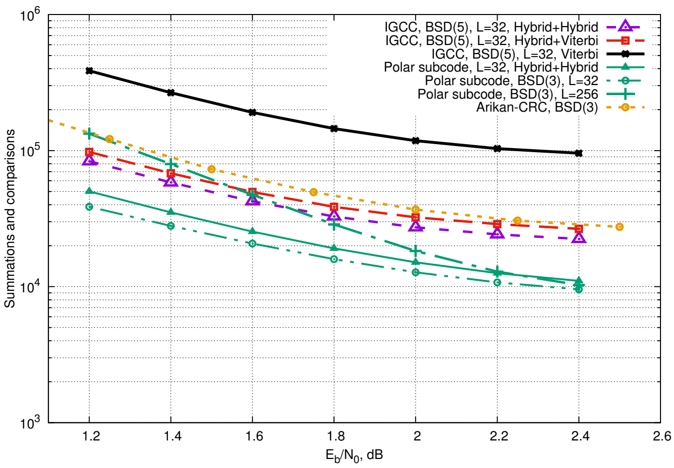

Figure 5. Decoding complexity of (1024, 512) codes

terms of the average number of summation and comparison operations. It can be seen that the hybrid decoding algorithm provides almost the same performance as tree-trellis Viterbi algorithm, but has much lower complexity. Both algorithms allow IGCC to outperform polar subcodes and polar codes with CRC. It can be also seen that the IGCC with decoder list size $L = 32$ outperforms the polar subcode with $L = 256$. However, this comes at the expense of two times more arithmetic operations in the high SNR region.

V. Conclusion

In this paper a low-complexity algorithm for decoding of interlinked generalized concatenated codes is proposed. The algorithm is based on the generalized Plotkin decomposition of the considered codes. It allows one to achieve significant performance gain with respect to polar subcodes and polar codes with CRC.

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] P. Trifonov and V. Miloslavskaya, “Polar codes with dynamic frozen symbols and their decoding by directed search,” in *Proceedings of IEEE Information Theory Workshop*, September 2013, pp. 1 – 5.

[3] ——, “Polar subcodes,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 2, pp. 254–266, February 2016.

[4] G. Trofimiuk and P. Trifonov, “Block sequential decoding of polar codes,” in *Proceedings of International Symposium on Wireless Communication Systems*, 2015, pp. 326–330.

[5] U. Wachsmann, R. F. H. Fischer, and J. B. Huber, “Multilevel codes: Theoretical concepts and practical design rules,” *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1361–1391, July 1999.

[6] V. Miloslavskaya and P. Trifonov, “Sequential decoding of polar codes,” *IEEE Communications Letters*, vol. 18, no. 7, pp. 1127–1130, 2014.

[7] I. Tal and A. Vardy, “List decoding of polar codes,” in *Proceedings of IEEE International Symposium on Information Theory*, 2011, pp. 1–5.

[8] M. Roder and R. Hamzaoui, “Fast tree-trellis list Viterbi decoding,” *IEEE Transactions on Communications*, vol. 54, no. 3, pp. 453–461, March 2006.

[9] T. Kasami, S. Lin, and W. Peterson, “New generalizations of the Reed-Muller codes part I: Primitive codes,” *IEEE Transactions on Information Theory*, vol. 14, no. 2, pp. 189–199, March 1968.

[10] P. A. Martin, D. Taylor, and M. P. Fossorier, “Soft-input soft-output list-based decoding algorithm,” *IEEE Transactions on Communications*, vol. 52, no. 2, pp. 252–262, February 2004.

[11] R. R. Green, “A serial orthogonal decoder,” *JPL Space Program Summary*, vol. 4, no. 31-39, pp. 241–253, 1966.

[12] J. Ha, J. Kim, D. Klinc, and S. W. McLaughlin, “Rate-compatible punctured low-density parity-check codes with short block lengths,” *IEEE Transactions On Information Theory*, vol. 52, no. 2, February 2006.