Genetic Algorithm-based Polar Code Construction for the AWGN Channel

Ahmed Elkelesh, Moustafa Ebada, Sebastian Cammerer and Stephan ten Brink
Institute of Telecommunications, Pfaffenwaldring 47, University of Stuttgart, 70569 Stuttgart, Germany
{elkelesh,ebada,cammerer,tenbrink}@inue.uni-stuttgart.de

***Abstract*—We propose a new polar code construction framework (i.e., selecting the frozen bit positions) for the additive white Gaussian noise (AWGN) channel, tailored to a given decoding algorithm, rather than based on the (not necessarily optimal) assumption of successive cancellation (SC) decoding. The proposed framework is based on the Genetic Algorithm (GenAlg), where populations (i.e., collections) of information sets evolve successively via evolutionary transformations based on their individual error-rate performance. These populations converge towards an information set that fits the decoding behavior. Using our proposed algorithm, we construct a polar code of length 2048 with code rate 0.5, without the CRC-aid, tailored to *plain* successive cancellation list (SCL) decoding, achieving the same error-rate performance as the CRC-aided SCL decoding, and leading to a coding gain of $1\,\mathrm{dB}$ at BER of 10^{-6}. Further, a belief propagation (BP)-tailored polar code approaches the SCL error-rate performance without any modifications in the decoding algorithm itself.**

I. Introduction

Polar codes [1] are the first family of codes proven to be capacity achieving for any Binary Input Discrete Memoryless Channel (BI-DMC) and with an explicit construction method under low complexity successive cancellation (SC) decoding. However, for finite lengths, both the SC decoder and the polar code itself (i.e., its maximum likelihood (ML) performance) are shown to be sub-optimal when compared to other state-of-the-art coding schemes such as low-density parity-check (LDPC) codes. Later, Tal and Vardy introduced a successive cancellation list (SCL) decoder [2] enabling a decoding performance close to the ML bound for sufficiently large list sizes. The concatenation with an additional high-rate cyclic redundancy check (CRC) code [2] or parity-check (PC) code [3] further improves the code performance itself, as it increases the minimum distance and, thus, improves the weight spectrum of the code. This simple concatenation renders polar codes into a powerful coding scheme. Later, an extension of polar codes, namely Polar Subcodes, were proposed in [4], outperforming the above-mentioned code constructions. Polar codes were recently selected by the 3GPP group as the channel codes for the upcoming *5th generation mobile communication standard (5G)* uplink/downlink control channel [5].

Although several other decoding schemes such as the iterative belief propagation (BP) decoding [6] and the iterative belief propagation list (BPL) decoder [7] exist, their error-rate performance is currently not competitive with the CRC-aided SCL decoding. Therefore, a good polar code design tailored to an explicit decoder may change this situation as it promises either an improved error-rate performance or savings in terms of decoding complexity for a specific type of decoder. In this work, we consider the decoding algorithm throughout the code construction phase instead of constructing the code based on the typical assumption of SC decoding. An intuitive example why a design for SC is sub-optimal for other decoding schemes can be given by considering the girth of the graph under BP decoding, which strongly depends on the frozen positions of the code. Thus, freezing additional nodes can even result in a degraded decoding performance under BP decoding although the ML performance indeed gets better (see Fig. 9 in [8]).

In a strict sense, one may argue that polar codes are inherently connected with SC decoding and only a design based on the concept of channel polarization results in a *true* "polar" code. However, from a more practical point of view, we seek to find the most efficient coding scheme for finite length constraints and, with slight abuse of notation, we regard the proposed codes as polar codes. Thus, a design method that considers the decoder and improves the overall performance is an important step for future polar code applications.

Polar code construction (or design), throughout this paper, refers to selecting an appropriate frozen bit position pattern. Different polar code construction algorithms exist assuming SC decoding. However, an explicit design tailored to SCL or BP decoding turns out to be cumbersome due to the many dependencies in the decoding graph and the high dimensionality of the optimization problem. In this work, we propose a new framework for polar code design matched to a specific decoding algorithm embedded in the well-understood Genetic Algorithm (GenAlg) context. As a result, the optimization algorithm works on a specific error-rate simulation setup, i.e., it inherently takes into account the actual decoder and channel. This renders the GenAlg-based polar code optimization into a solid and powerful design method. To the best of our knowledge, the resulting polar codes in this work outperform any known design method for *plain* polar codes under SCL decoding *without* the aid of an additional CRC (i.e., CRC-aided SCL performance could be achieved *without* the aid of a CRC). Additionally, the BP decoder of the *proposed* code achieves (and slightly outperforms) the SCL decoding performance of *conventional* codes without any decoder modifications. The interested reader can refer to [9] for an extended version.

II. Polar codes

Polar codes [1] are based on the concept of channel polarization, in which $N = 2^n$ identical copies of a channel W are combined and N synthesized bit-channels are generated.

This work has been supported by DFG, Germany, under grant BR 3205/5-1. The authors would like to thank Alexander Vardy and Ido Tal for providing their set of frozen/non-frozen bit positions as a baseline for comparison.

DOI: 10.30420/454862007

These synthesized bit-channels show a polarization behavior, in the sense that some bit-channels are purely noiseless and the rest are completely noisy. A recursive channel combination provides the polarization matrix

$$\mathbf{G}_N = \mathbf{B}_N \cdot \mathbf{F}^{\otimes n}, \qquad \mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

where $\mathbf{B}_N$ is a bit-reversal permutation matrix and $\mathbf{F}^{\otimes n}$ denotes the n-th Kronecker power of $\mathbf{F}$. The polar codewords $\mathbf{x}$ are given by $\mathbf{x} = \mathbf{u} \cdot \mathbf{G}_N$, where $\mathbf{u}$ contains k information bits and $N-k$ frozen bits, w.l.o.g. we set the frozen positions to "0". The information set $\mathbb{A}$ contains the k most reliable positions of $\mathbf{u}$ in which the k information bits are transmitted and $\bar{\mathbb{A}}$ denotes the frozen positions (i.e., the complementary set to $\mathbb{A}$). The *conventional* generator matrix, denoted by $\mathbf{G}$, is constructed as the rows $\{\mathbf{r}_i\}$ of $\mathbf{G}_N$ with $i \in \mathbb{A}$. The task of the polar code construction, in its original form, is to find the information set $\mathbb{A}$ which maximizes the code performance (under SC decoding) for a specific channel condition. More details on the problem of polar code construction is provided in Section III.

Throughout this work, we use non-systematic polar encoding. However, it is straightforward to use the GenAlg to construct systematic polar codes. In this work, a polar code with codeword length N and k information bits is denoted by $\mathscr{P}(N,k)$, i.e., the information set has the cardinality $|\mathbb{A}| = k$ and the code rate $R_c = k/N$. In the following, we revise the basic polar decoding algorithms.

SC decoding [1] is the first polar decoder, where all information bits $\hat{u}_i$ are sequentially hard-decided based on the previously estimated bits $\{\hat{u}_1, \ldots, \hat{u}_{i-1}\}$ and the channel information $\mathbf{y}$, where $i \in \{1, \ldots, N\}$.

SCL decoding [2] denoted by SCL (L), utilizes a list of L most likely paths during SC decoding; at every decision the decoder branches into two paths ($\hat{u}_i = 0$ and $\hat{u}_i = 1$) instead of the hard decision in the SC decoder. To limit the exponential growth of complexity, only the L most reliable paths are kept sorted in the list according to a specific path metric.

CRC-aided SCL decoding [2] denoted by SCL+CRC-r (L), where an additional high-rate CRC of r bits is concatenated to the polar code, to help in selecting the final codeword from the L surviving candidates, yielding significant performance gains in competing with the state-of-the-art error correcting codes.

BP decoding [6] denoted by BP $(N_{it,max})$ is an iterative message passing decoder based on the *encoding* graph of the polar code. Log-likelihood ratio (LLR) messages are iteratively passed along the encoding graph until reaching a maximum number of iterations $(N_{it,max})$ or meeting an early stopping condition. The error-rate performance of polar codes under BP decoding is typically not competitive with SCL decoding. In this work, we use the **G**-*matrix-based* early stopping condition, where decoding terminates when $\hat{\mathbf{x}} = \hat{\mathbf{u}} \cdot \mathbf{G}_N$ [10].

III. Polar Code Construction

The polar code construction phase is about deciding the most reliable k bit positions that are set as the information bit positions, while the remaining $N-k$ bit positions are set as frozen bit positions. Thus, ranking the bit-channels according to their reliabilities is of major significance in the polar code construction phase. The information set $\mathbb{A}$ is the outcome from the polar code construction phase specifying the indices of the information bit positions. A corresponding logical $\mathbf{A}$-vector can be used such that $\mathbf{A} = [a_1, a_2, \ldots, a_N]$, where $a_i \in \{0,1\}$ and $1 \leq i \leq N$. Bit position i is frozen if $a_i = 0$, while bit position j is non-frozen (i.e., can be used for information transmission) if $a_j = 1$. For instance consider the $\mathscr{P}(8,4)$-code, the information set $\mathbb{A} = \{4,6,7,8\}$ can thus be represented by the logical vector $\mathbf{A} = [0\,0\,0\,1\,0\,1\,1\,1]$.

The code construction can be considered as an optimization problem, where the objective is to find the optimal set of k good positions in a set of indices $\{1, \ldots, N\}$, as shown in (1).

$$\begin{aligned} \mathbf{A}_{\text{opt}} = \arg \min_{\mathbf{A}} \quad & \text{BER}(\mathbf{A}) \text{ or } \text{BLER}(\mathbf{A}) \\ \text{subject to} \quad & \left(\sum_{i=1}^{N} a_i \right) = k, \\ & a_i \in \{0,1\}. \end{aligned} \tag{1}$$

where $\mathbf{A} = [a_1, a_2, \ldots, a_N]$, $R_c = k/N$ and $i = 1, 2, \ldots, N$.

The information set $\mathbb{A}$ (or, equivalently, the $\mathbf{A}$-vector) is channel dependent, meaning that it depends on the respective channel parameter (e.g., design SNR for additive white Gaussian noise (AWGN) channel, or design ε for Binary Erasure Channel (BEC)). Thus, polar codes are non-universal. The minimum distance of polar codes d_{min} depends on the polar code construction (i.e., $\mathbb{A}$). Polar codes d_{min} is equal to the minimum weight of the rows $\{\mathbf{r}_i\}$ in the $\mathbf{G}_N$-matrix with indices in $\mathbb{A}$ (i.e., $i \in \mathbb{A}$) [11, Lemma 3].

Choosing the best k bit positions for information transmission is even more crucial for short length polar codes. This can be attributed to the fact that the bit-channels of the short length polar codes are not fully polarized, and the portion of semi-polarized bit-channels (which would be normally unfrozen) leads to high error-rates. Although efficient polar code construction only exists for the BEC case [1], many algorithms were devised for the AWGN channel case. A survey on the effect of the design SNR and the effect of the specific polar code construction algorithm used on the error-rate performance of SC decoding is presented in [12].

Arıkan uses the symmetric capacity $I(W_i)$ or the Bhattacharyya parameter $Z(W_i)$ of the virtual channel W_i to assess its reliability [1]. However, the Bhattacharyya parameters are preferred because of being connected with an explicit bound on the block error rate (BLER) under SC decoding.

In [13], the polar code construction step is viewed as an instance of density evolution and, thus, a construction algorithm based on convolutions was proposed. A Gaussian approximation of the density evolution for polar code construction was proposed in [14].

In [11], picking the frozen bit positions according to the Reed–Muller (RM) rule was observed to enhance the error-rate performance significantly under maximum a posteriori (MAP) decoding due to the fact that the RM rule maximizes the

minimum distance d_{min}. An RM code can be viewed as a polar code with a different frozen/non-frozen bit selection strategy [11], where both codes are based on the same polarization matrix $\mathbf{G}_N$. However, the k information bit positions of RM codes are the positions corresponding to the k row indices with the maximum weights in the $\mathbf{G}_N$-matrix. Consequently, the RM code construction phase is channel independent as it merely depends on the row weights of $\mathbf{G}_N$. A hybrid polar and RM code construction [15] results in significant error-rate improvement gains under SCL decoding without the CRC-aid, by improving the minimum distance of the resultant code. This underlines the benefits of improving the minimum distance of the code and also the need of an improved construction algorithm tailored to the decoder. A similar family of codes was introduced in [16] providing significant error-rate performance gains under BP and SCL decoding.

In [17], an LLR-based polar code construction is proposed, in which the LLRs of the non-frozen bits are tracked during BP decoding to identify weak information bit-channels, and then the information set $\mathbb{A}$ is modified by swapping these weak information bit-channels with strong frozen bit-channels. The resulting code shows an enhanced error-rate performance under both BP and SCL decoding due to the resultant reduction in the number of low weight codewords. Some work has been done to construct polar codes which are tailored to a specific decoder, e.g., polar code construction assuming SCL decoding [18] and BP decoding [19], where a Monte-Carlo-based construction is proposed similar to [1].

As the output alphabet size grows exponentially with the codelength, it is computationally of high complexity to precisely calculate $I(W_i)$ or $Z(W_i)$ per bit-channel. However, a quantization can be used to closely approximate them [20]. A recent discovery which reduces the complexity of the polar code design is the partial order for synthesized channels [21]. Later an heuristic closed-form algorithm called polarization weight (PW) [22] was proposed to determine the reliability of bit-channels based on their indices and a carefully chosen parameter β [23], resulting in a significant complexity reduction in the polar code construction.

It is important to keep in mind that polar codes that are constructed based on the mutual information or the Bhattacharyya parameters of the bit-channels, as proposed by Arıkan, are tailored to hard-output SC decoders. Therefore, they are not necessarily optimum when using other decoders such as the soft-output BP decoder [11], [16], [17] or the SCL decoder [16], [17], [24]. To the best of our knowledge, no analytical polar code construction rule exists thus far which would be *optimized* for BP or SCL decoding and, thus, the nature of the iterative or list decoding is not usually taken into account while designing polar codes. Therefore, the problem of taking the type of decoding into consideration while constructing the polar code is, thus far, an open problem. In this work, we propose a method which always converges to a "good enough" solution, i.e., leads to a better error-rate performance when compared to the state-of-the-art polar code construction techniques.

IV. Genetic Algorithm-based polar code construction

GenAlg was first introduced by Holland in 1975 [25] as an efficient tool that helps in achieving (good) solutions for high-dimensional optimization problems which are computationally intractable. Beside finding (good) local minima, a well-parametrized GenAlg is known for converging to these minima very quickly [26]. Due to that merit, GenAlg has attracted a lot of research in the Artificial Intelligence (AI) field leading to improved and adaptive variants of it.

Furthermore, researchers from other fields (e.g., channel coding, signal processing) worked on adapting the GenAlg to some of their specific problems that lacked enough theoretical basis for solving, leading to improvements over the existent theoretical methods of approaching the same problems. GenAlg was applied in the channel coding field, e.g., during code construction [26], [27] and while decoding of linear block codes [28], LDPC codes [29] and convolutional codes [30].

The GenAlg is inspired by the natural evolution where *populations* of *offsprings*, resulting from *parents*, keep *evolving* and compete, while only the *fittest* offsprings *survive*. Therefore, the GenAlg starts with an initial population of individuals and the fittest of them survive and give birth to new offsprings which represent the new population. The criterion of fitness is, therefore, critical for the GenAlg so that the offsprings keep evolving towards the fittest. If the size of population and the number of evolution stages are sufficiently large, the GenAlg converges to an ultimate (sub-) optimal solution.

The task of polar code construction can be viewed as an optimization problem (see (1)) searching for the optimum information set $\mathbb{A}$ that has the minimum (possible) cost function. This optimization problem can be solved using GenAlg. The bit error rate (BER) has been selected as the cost function throughout this work for optimization conducted on AWGN channels. All presented results throughout this work are simulated on GPUs to accelerate our error-rate simulations [31]. The whole setup is depicted in Fig. 1. Next we briefly introduce the most important GenAlg fundamentals.

A. Population

In this work, the population $\mathbb{P} = \{\mathbf{A}_i\}$, for $i = 1, \ldots, S$, is a collection of S candidate information sets represented by their binary $\mathbf{A}$-vectors, each with its own fitness value (e.g., BER or BLER), that represent the search space.

B. Initialization

As it turns out, GenAlg converges faster, and probably to a better solution, if its population $\mathbb{P}_{init}$ is initialized with a sufficiently good collection of estimated $\mathbf{A}$-vectors. For that purpose, the population is initially filled with a collection of $\mathbf{A}$-vectors, all based on the Bhattacharyya construction [1] obtained for BECs with various erasure probabilities. Besides, we considered having the $\mathbf{A}$-vectors constructed according to [15] among the initial population, in the SCL-tailored polar code construction phase, which improved the acquired solution $\mathbf{A}_{GenAlg}$ remarkably.

ISBN 978-3-8007-4862-4

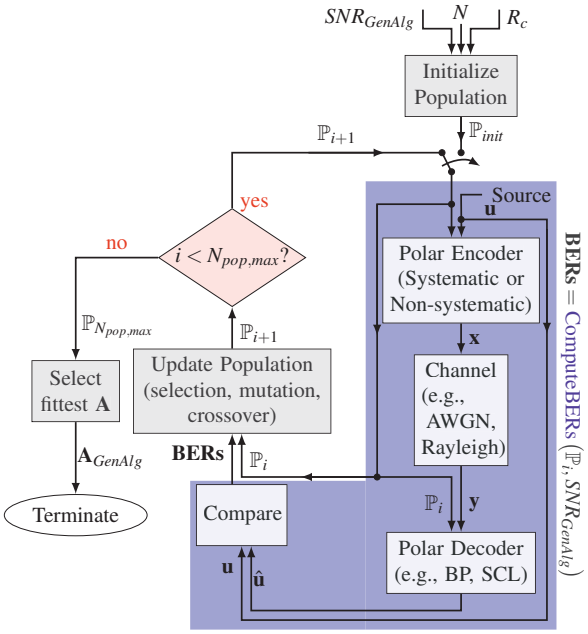


Fig. 1: An abstract view of the genetic algorithm (GenAlg)-based polar code construction.

C. Fitness function

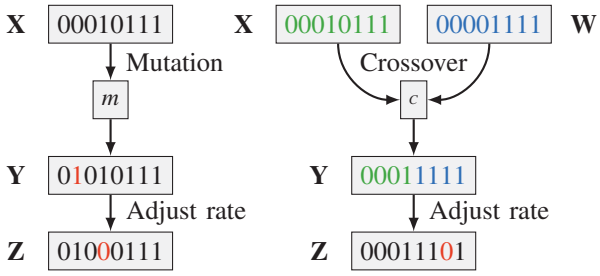
The cost function chosen in this work is the error-rate at a user-defined design SNR $\left(SNR_{GenAlg}\right)$. The fitness function that decides the rank of each of the individual **A**-vectors is, thus, selected to be the inverse of the error-rate. In other words, the **A** leading to the minimum error-rate is announced to be the optimum $\mathbf{A}_{GenAlg}$. Alternatively, one might consider using mutual information, LLR-reliability $\sum_{i=1}^{N} |LLR_i|$ or any other reasonable metric as the fitness function.

D. Mutation

Mutation guarantees more diversity and acts against premature convergence at a certain bit position. It is, straightforwardly, a bit flip of a random position in the **A**-vector representing a *frozen-to-non-frozen* (or a *non-frozen-to-frozen*) switch at that respective bit position. A mutation example is shown in Fig. 2a, where the offspring vector **Y** is the result of bit flipping the 2^{nd} bit of the parent vector **X**. However, the polar code construction problem has the constraint that the number of non-frozen bit positions (i.e., number of ones) is equal to k to maintain the code rate $R_c = k/N$. To restore R_c, one further mutation is applied to the resultant offspring vector **Y** yielding the vector **Z**. For our specific problem, one mutation always occurs for each parent per evolutionary step.

E. Crossover

The crossover applied throughout this work is a single midpoint crossover (see Fig. 2b), where the 1^{st} half of the first parent vector **X** is combined with the 2^{nd} half of the second parent vector **W** to generate the vector **Y**. This often leads to a change in the number of ones in the resulting vector **Y** (i.e., remember from (1) that the total number of ones in the **A**-vector should be equal to k and thus the code rate $R_c = k/N$). To restore R_c, sequential mutations are applied to the resultant vector until reaching the k ones in the binary **A**-vector (**Z** in



(a) Mutation (b) Crossover

Fig. 2: Examples of GenAlg's evolutionary transformations in the polar code construction context. Inputs (**X** and/or **W**) and output (**Z**) satisfy the constraints in (1).

Fig. 2b). Similar to mutation, one crossover always occurs for each pair of parents per evolutionary step.

F. Selection and population update

In this context, by the term *selection* we mean the way the new population is generated. We applied the following scheme in order to generate the new population:

- The fittest T **A**-vectors are always pushed forward as members of the new population (i.e., self-offsprings). This ensures convergence to the (local) optimum and guarantees a monotonic behaviour of the cost function through evolving populations which facilitates observing the candidate solutions.
- Crossovers are applied between each pair of the fittest T **A**-vectors, resulting in new $\binom{T}{2}$ offsprings.
- Mutations are applied on the fittest T **A**-vectors, resulting in new T mutated offsprings.

Consequently, the size of the new population S is

$$S = \underbrace{T}_{T \text{ fittest}} + \underbrace{\binom{T}{2}}_{\text{crossover}} + \underbrace{T}_{\text{mutation}} = \frac{T^2+3T}{2}$$

For all simulation results using the GenAlg as discussed next, we set $S = 20$ and $T = 5$. We make the source code public and also provide the best polar code designs from this work online[1].

V. Simulation Results over AWGN Channel

In this section, we show the results of designing polar codes using the GenAlg method over the AWGN channel. To be coherent with the results shown in [2], we use codes of length $N = 2048$ and code rate $R_c = 0.5$.

A. BP decoder

We use the GenAlg to design polar codes tailored to BP decoding over AWGN channel. Fig. 3 shows a BER comparison between a code constructed via [20] at a design SNR $(E_b/N_0) = 2\,\text{dB}$ and a code constructed using our proposed GenAlg at SNR_{GenAlg} $(E_b/N_0) = 2\,\text{dB}$ under BP $(N_{it,max} = 200)$ decoding. The GenAlg-based construction

[1] https://github.com/AhmedElkelesh/Genetic-Algorithm-based-Polar-Code-Construction

ISBN 978-3-8007-4862-4

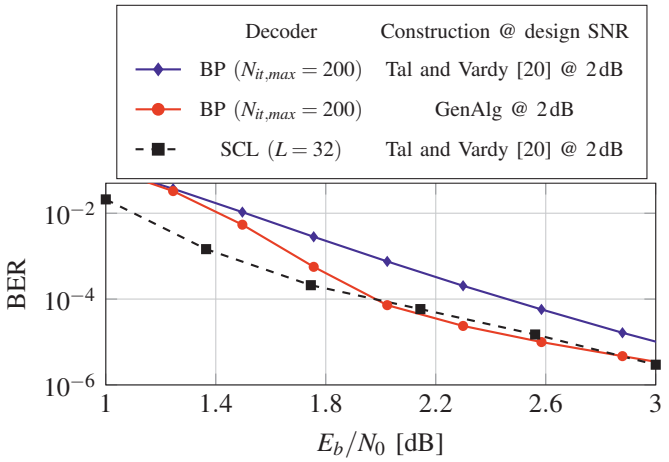


Fig. 3: BER of the GenAlg-based $\mathscr{P}(2048,1024)$-code under BP decoding over the AWGN channel.

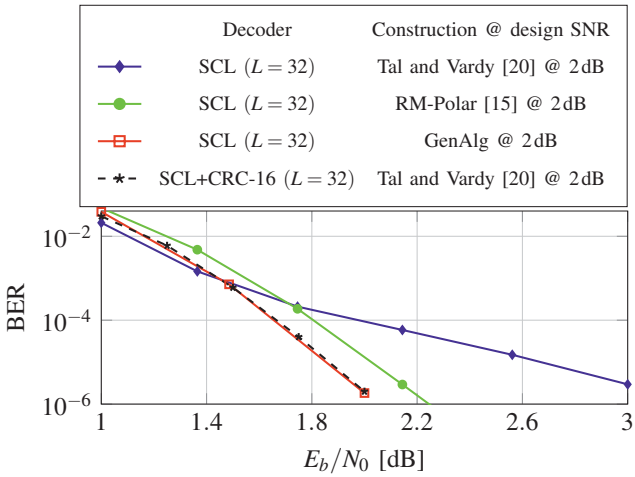


Fig. 4: BER of the GenAlg-based $\mathscr{P}(2048,1024)$-code under SCL decoding over the AWGN channel. The CRC-aided polar code (- ⋆ -) is a $\mathscr{P}(2048,1040)$-code: $N = 2048$, $k = 1024$ and $r = 16$.

yields a 0.5 dB net coding gain at BER of 10^{-4} when compared to the construction proposed in [20], while the only difference between the two codes is the selection of the frozen/non-frozen bit positions, i.e., both use exactly the same decoder.

The d_{min} of the BP-tailored polar code using GenAlg was found to be 8, while the d_{min} of the polar code constructed via [20] is 16. This is due to the fact that the performance of a linear code under iterative decoding is dominated by the structure of the stopping sets in the Tanner graph of the code and not its d_{min} [32]. Thus, d_{min} is not the only parameter to be maximized in order to design linear codes tailored to iterative decoding. A fact supporting this claim is that the RM code has a larger (maximum) d_{min} but worse error-rate performance under iterative decoding when compared to polar codes [6].

Note that the BER performance of the BP-tailored polar code (—●—) is very close to the performance of a polar code constructed via [20] under SCL ($L = 32$) (- ■ -), see Fig. 3.

B. SCL decoder

Next, the GenAlg is applied to construct polar codes tailored to SCL ($L = 32$) over the AWGN channel. Fig. 4 shows a BER comparison between a code constructed via the method proposed in [20] at $E_b/N_0 = 2$ dB and a code constructed using our proposed GenAlg at $E_b/N_0 = 2$ dB under SCL ($L = 32$) decoding. The GenAlg-based construction shows significant performance improvements, yielding a 1 dB net coding gain at BER of 10^{-6} when compared to the construction proposed in [20], where again the only difference between the two codes is the frozen/non-frozen bit positions.

The GenAlg-optimized polar code without CRC-aid under SCL decoding (—□—) performs equally well as the CRC-aided polar code under CRC-aided SCL decoding (- ⋆ -), with the same list size $L = 32$ and the same code rate $R_c = 0.5$.

The d_{min} of the SCL-tailored polar code using GenAlg is 16 which is exactly the same as the d_{min} of the polar code constructed via [20]. However, the RM-Polar code (—●—) has a $d_{min} = 32$. Thus, again d_{min} is not the only parameter to be maximized in order to design polar codes tailored to SCL decoding. Note that only maximizing d_{min} will lead to an RM code.

The distribution of frozen bit-channels over the N synthesized virtual channels, shown in the "Frozen channel chart" in Fig. 5, depends on the considered decoder while constructing the code. The conventional construction techniques (e.g., [1], [20]) assume that an SC decoder is used. Thus, these construction techniques yield very similar **A**-vectors having a very similar distribution as shown in Fig. 5a and 5b. Our proposed GenAlg-based construction tailors the **A**-vector (or the code) to a specific decoder (e.g., BP or SCL). The BP decoder has an iterative (i.e., non-sequential) decoding nature when compared to SC decoding. Thus, the **A**-vector tailored to BP decoding has a much different frozen bit-channel distribution as shown in Fig. 5c. Note that the **A**-vector tailored to BP decoding contains a significant number of non-frozen bits in the first half part of the code, which is not the case in the conventional code construction algorithms assuming SC decoding. Similarly, the GenAlg-based construction tailored to SCL decoding should take the list decoding nature into consideration and, thus, a different frozen bit-channel distribution as shown in Fig. 5d. This shows that the GenAlg-based polar code construction takes the decoding nature into consideration while constructing the code, which is the main advantage of the GenAlg. Furthermore, Table I shows that the GenAlg-optimized **A**-vectors are indeed tailored to the considered decoder.

TABLE I: Illustration of polar design and decoder architecture mismatch by evaluating the minimum E_b/N_0 required to achieve a target BER of 10^{-4} for a $\mathscr{P}(2048,1024)$-code over AWGN channel

Construction @ design SNR	Decoder		
	SC	BP ($N_{it,max} = 200$)	SCL ($L = 32$)
Bhattacharyya [1] @ 3.6dB	**2.7 dB**	2.45 dB	1.8 dB
Tal and Vardy [20] @ 2dB	**2.65 dB**	2.45 dB	2 dB
GenAlg BP-tailored @ 2dB	> 9 dB	**2 dB**	> 7 dB
GenAlg SCL-tailored @ 2dB	> 6 dB	2.55 dB	**1.65 dB**

VI. Conclusion

As stated in [2], the best error-rate performance of finite length polar codes (i.e., under ML decoding without CRC) is not competitive with the other state-of-the-art coding schemes. In this work, we focus on polar code construction (i.e.,

ISBN 978-3-8007-4862-4

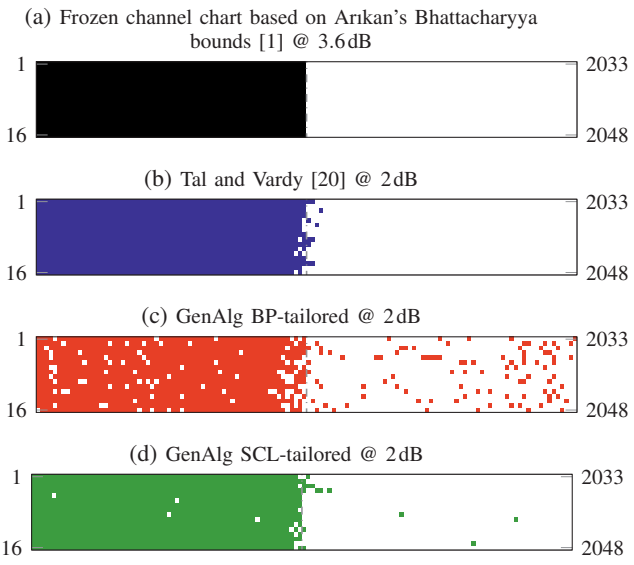


Fig. 5: Frozen channel chart (i.e., frozen bit position pattern) of a $\mathscr{P}$(2048,1024)-code over the AWGN channel with different polar code construction algorithms. The 2048 bit positions are plotted over a 16×128 matrix and sorted with decreasing Bhattacharyya parameter value. White: non-frozen; colored: frozen.

changing the code itself which is defined by the **A**-vector) in order to boost the error-rate performance under the state-of-the-art feasible practical decoders (e.g., BP, SCL). We propose a Genetic Algorithm-based polar code design where the decoding nature is taken into account, yielding significant performance gains in terms of error-rate. For a polar code of length 2048 and code rate 0.5 over the binary input AWGN channel under *plain* SCL decoding, approximately a 1 dB coding gain at BER of 10^{-6} is achieved when compared to the conventionally constructed polar codes. This enables achieving the same error-rate performance of polar codes under state-of-the-art CRC-aided SCL decoding with *plain* SCL decoding without the aid of a CRC. Furthermore, GenAlg is used to construct polar codes tailored to flooding BP decoding, finally closing the performance gap between conventional iterative BP decoding and conventional SCL decoding. Fortunately, the achieved gains come "for free", since optimizing the **A**-vector is done once and offline. The source code and the best polar code designs from this work can be found in [33].

References

[1] E. Arıkan, "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, July 2009.

[2] I. Tal and A. Vardy, "List Decoding of Polar Codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[3] T. Wang, D. Qu, and T. Jiang, "Parity-Check-Concatenated Polar Codes," *IEEE Commun. Lett.*, vol. 20, no. 12, pp. 2342–2345, Dec. 2016.

[4] P. Trifonov and V. Miloslavskaya, "Polar Subcodes," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 254–266, Feb. 2016.

[5] Huawei, "Evaluation of TBCC and Polar Codes for Small Block Lengths," *3GPP TSG RAN WG1 N.85, Tech. Rep.*, May 2016.

[6] E. Arıkan, "A Performance Comparison of Polar Codes and Reed-Muller Codes," *IEEE Commun. Lett.*, vol. 12, no. 6, pp. 447–449, June 2008.

[7] A. Elkelesh, M. Ebada, S. Cammerer, and S. ten Brink, "Belief Propagation List Decoding of Polar Codes," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1536–1539, Aug. 2018.

[8] A. Elkelesh, S. Cammerer, M. Ebada, and S. ten Brink, "Mitigating Clipping Effects on Error Floors under Belief Propagation Decoding of Polar Codes," in *Inter. Symp. Wireless Commun. Syst.*, Aug. 2017.

[9] A. Elkelesh, M. Ebada, S. Cammerer, and S. ten Brink, "Decoder-tailored Polar Code Design Using the Genetic Algorithm," *ArXiv e-prints*, Feb. 2019.

[10] B. Yuan and K. K. Parhi, "Early Stopping Criteria for Energy-Efficient Low-Latency Belief-Propagation Polar Code Decoders," *IEEE Trans. Sig. Process.*, vol. 62, no. 24, pp. 6496–6506, Dec. 2014.

[11] N. Hussami, S. B. Korada, and R. Urbanke, "Performance of Polar Codes for Channel and Source Coding," in *IEEE Inter. Symp. Inf. Theory (ISIT)*, June 2009, pp. 1488–1492.

[12] H. Vangala, E. Viterbo, and Y. Hong, "A Comparative Study of Polar Code Constructions for the AWGN Channel," *ArXiv e-prints*, Jan. 2015.

[13] R. Mori and T. Tanaka, "Performance of Polar Codes with the Construction using Density Evolution," *IEEE Commun. Lett.*, vol. 13, no. 7, pp. 519–521, July 2009.

[14] P. Trifonov, "Efficient Design and Decoding of Polar Codes," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3221–3227, Nov. 2012.

[15] B. Li, H. Shen, and D. Tse, "A RM-Polar Codes," *ArXiv e-prints*, July 2014.

[16] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "From Polar to Reed-Muller Codes: A Technique to Improve the Finite-Length Performance," *IEEE Trans. Commun.*, vol. 62, no. 9, pp. 3084–3091, Sep. 2014.

[17] M. Qin, J. Guo, A. Bhatia, A. G. i Fabregas, and P. Siegel, "Polar Code Constructions Based on LLR Evolution," *IEEE Commun. Lett.*, vol. 21, no. 6, pp. 1221–1224, June 2017.

[18] P. Yuan, T. Prinz, and G. Böcherer, "Polar Code Construction for List Decoding," *ArXiv e-prints*, July 2017.

[19] S. Sun and Z. Zhang, "Designing Practical Polar Codes Using Simulation-Based Bit Selection," *IEEE J. Emerging and Sel. Topics Circuits Syst.*, vol. 7, no. 4, pp. 594–603, Dec. 2017.

[20] I. Tal and A. Vardy, "How to Construct Polar Codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 10, pp. 6562–6582, Oct. 2013.

[21] C. Schürch, "A Partial Order For the Synthesized Channels of a Polar Code," in *IEEE Inter. Symp. Inf. Theory (ISIT)*, July 2016, pp. 220–224.

[22] "3GPP, R1-167209, Polar code design and rate matching, Huawei, HiSilicon."

[23] G. He, J. C. Belfiore, I. Land, G. Yang, X. Liu, Y. Chen, R. Li, J. Wang, Y. Ge, R. Zhang, and W. Tong, "β-expansion: A Theoretical Framework for Fast and Recursive Construction of Polar Codes," in *IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[24] V. Bioglio, C. Condo, and I. Land, "Design of Polar Codes in 5G New Radio," *ArXiv e-prints*, Apr. 2018.

[25] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[26] L. Hebbes, R. R. Malyan, and A. P. Lenaghan, "Genetic algorithms for turbo codes," in *EUROCON - The Inter. Conf. on Computer as a Tool*, vol. 1, Nov. 2005, pp. 478–481.

[27] K. Dontas and K. D. Jong, "Discovery of maximal distance codes using genetic algorithms," in *IEEE Inter. Conf. on Tools for Artificial Intelligence*, Nov. 1990, pp. 805–811.

[28] H. Maini, K. Mehrotra, C. Mohan, and S. Ranka, "Genetic Algorithms for Soft-decision Decoding of Linear Block Codes," *Evol. Comput.*, vol. 2, no. 2, pp. 145–164, June 1994.

[29] A. Scandurra et al., "A Genetic-Algorithm Based Decoder for Low Density Parity Check Codes," *Latin American Applied Research*, vol. 36, no. 3, pp. 169–172, July 2006.

[30] H. Berbia, M. Belkasmi, F. Elbouanani, and F. Ayoub, "On the decoding of convolutional codes using genetic algorithms," in *Inter. Conf. on Computer and Commun. Engineering*, May 2008, pp. 667–671.

[31] S. Cammerer, B. Leible, M. Stahl, J. Hoydis, and S. ten Brink, "Combining Belief Propagation and Successive Cancellation List Decoding of Polar Codes on a GPU Platform," in *IEEE Inter. Conf. on Acoustics, Speech, and Sig. Process. (ICASSP)*, Mar. 2017, pp. 3664–3668.

[32] M. Schwartz and A. Vardy, "On the Stopping Distance and the Stopping Redundancy of Codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 922–932, Mar. 2006.

[33] A. Elkelesh, "Genetic Algorithm-based Polar Code Construction," Dec. 2018. [Online]. Available: https://github.com/AhmedElkelesh/Genetic-Algorithm-based-Polar-Code-Construction

