# On puncturing strategies for polar codes

Ludovic Chandesris, Valentin Savin, David Declercq

# On Puncturing Strategies for Polar Codes

Ludovic Chandesris[†‡], Valentin Savin[†], David Declercq[‡]
ludovic.chandesris@cea.fr, valentin.savin@cea.fr, declercq@ensea.fr
[†]CEA-LETI, Minatec campus, Grenoble, France [‡]ETIS, ENSEA/UCP/CNRS, Cergy-Pontoise, France

***Abstract*—This paper introduces a class of specific puncturing patterns, called symmetric puncturing patterns, which can be characterized and generated from the rows of the generator matrix $G_N$. They are first shown to be non-equivalent, then a low-complexity method to generate symmetric puncturing patterns is proposed, which performs a search tree algorithm with limited depth, over the rows of $G_N$. Symmetric patterns are further optimized by density evolution, and shown to yield better performance than state-of-the-art rate compatible code constructions, relying on either puncturing or shortening techniques.**

***Index Terms*—Punctured polar codes, equivalent puncturing patterns, symmetric puncturing patterns.**

## I. Introduction

Polar codes are a recently discovered family of error correcting codes [1], known to achieve the capacity of any binary-input memoryless output-symmetric channel. Their construction relies on a specific recursive encoding procedure that synthesizes a set of $N$ virtual channels from $N$ instances of the transmission channel, where $N$ denotes the code-length. The recursive encoding procedure is reversed at the receiver end, by applying a Successive Cancellation (SC) decoder. The asymptotic effectiveness of the SC decoder derives from the fact that the synthesized channels tend to become either noiseless or completely noisy, as the code-length goes to infinity, phenomenon which is known as "channel polarization".

A Polar codeword $x_0^{N-1} \in \{0,1\}^N$ is obtained by applying a linear transformation on a *data vector* $u_0^{N-1} \in \{0,1\}^N$ that contains both *information* and *frozen bits* (*i.e.*, bits whose values are known at both the transmitter and the receiver). The positions of information and frozen bits are specified by a binary *information pattern* $I \subset \{0,1\}^N$, with 1s corresponding to information bits, and 0s corresponding to frozen bits. The linear transformation is defined by the square matrix $\mathbf{G}_N$, of size $N \times N$, defined as the $n$-th Kronecker product of the *kernel matrix* $\mathbf{G}_2$ [1]:

$$G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{1}$$

Hence, $N = 2^n$ and $x_0^{N-1} = u_0^{N-1} \cdot \mathbf{G}_N$, with $\mathbf{G}_N = \mathbf{G}_2^{\otimes n}$. However, in practical communication systems the length of the coded block may not be a power of 2. In this case, punctured Polar codes have to be considered, in which only a number $N - N_p$ of the encoded bits $x_0^{N-1}$ are transmitted over the channel. The positions of the $N_p$ punctured bits are indicated by a *puncturing pattern* $P \subset \{0,1\}^N$. The optimization of the puncturing pattern amounts to determining the $N_p$ positions yielding the punctured code with the best possible decoding performance. It is worth noticing that the optimal solution requires a joint optimization of both $P$ and $I$, since different puncturing patterns may lead to different information patterns. Hence, a brute-force search algorithm would require testing all the possible puncturing patterns, *i.e.* $\binom{N}{N_p}$ possibilities, and for each puncturing pattern finding the optimal pattern of information bits, *e.g.*, by using the density evolution technique, as explained in Section II. In practice, this is not feasible even for relatively small values of $N$ and $N_p$. The optimization problem is slightly different in case of communication systems employing retransmission techniques, such as Hybrid Automatic Repeat reQuest (HARQ) [2–5], since the overall system performance depends on the performance of both the punctured and the unpunctured (mother) code, which are further constrained by a dependency relation, as they both have to use the same information pattern $I$.

While the problem of finding the optimal puncturing pattern is still open, several puncturing techniques have been proposed in the literature, aimed at enhancing the punctured Polar code performance, by either making use of different properties of the punctured code (in terms of minimal distance or exponent bound) [5–7], or relying on the density evolution analysis [8, 9]. However, as no efficient method has been proposed to evaluate the optimal solution for moderate to long code-length, the gap between any method and the optimality is unpredictable.

In this paper we introduce a number of theoretical tools aimed at analyzing and classifying puncturing patterns. To this end, we first introduce *equivalent puncturing patterns*, and show they yield punctured codes with the same error correction performance. Then, we give a constructive solution to the problem of finding a unique representative for each equivalence class of puncturing patterns. Such a representative is referred to as a *primitive puncturing pattern*, and we further distinguish between primitive patterns that are *symmetric* from those that are not. The symmetry property means that $P$ is equal to the *erasure pattern* $E$ that is generated on the data vector $u_0^{N-1}$, corresponding to virtual channels with symmetric capacity equal to zero (see Section II-B). Then, we show that symmetric patterns can be generated by the rows of the generator matrix $\mathbf{G}_N$, and propose an algorithm able to generate symmetric patterns of a given *order* (the order of a symmetric pattern is defined as the minimum number of rows of $\mathbf{G}_N$ that contribute to generating it). This makes possible to implement a brute-force optimization algorithm on symmetric patterns of predetermined maximum order. Finally, the performance of the optimized puncturing patterns is evaluated and compared with state of the art solutions.

## II. Preliminaries

### A. Notation

- *Vectors* are denoted by either $a_0^{N-1} = (a_0, \ldots, a_{N-1})$ or $A = (A(0), \ldots, A(N-1))$. *Matrices* are denoted by boldface uppercase letters, *e.g.*, $\mathbf{M}$.
- $B_N : \{0, \ldots, N-1\} \to \{0, \ldots, N-1\}$ denotes the *bit-reversal* permutation, and $\mathbf{B}_N$ the corresponding binary permutation matrix, where $\mathbf{B}_N(i,j) = 1 \Leftrightarrow j = B_N(i)$.
- For any permutation $\phi : \{0, \ldots, N-1\} \to \{0, \ldots, N-1\}$ and any vector $A = (A(0), \ldots, A(N-1))$, $\phi(A)$ is the vector defined by $\phi(A)(i) = A(\phi(i))$.
- $|A|$ denotes the number of of 1s of a binary vector $A$.
- $A = \vee_{\ell=1}^{L} A_\ell$ denotes the *bit-wise* OR *operation* of a set of *binary vectors* $A_1, \ldots, A_L$, where $A(i) = 1 \Leftrightarrow \exists \ell \in \{1, \ldots, L\}$, such that $A_\ell(i) = 1$. By a slight abuse of language, we shall also say that $A$ *is the union of the vectors* $A_1, \ldots, A_L$.
- For two binary vectors $A, B$, we say that $A$ *is included in* $B$, and write $A \subseteq B$, if $A \vee B = B$.
- $A \leq_{lex} B$ denotes the lexicographic ordering of $A$ and $B$. Thus, assuming $A \neq B$, $A <_{lex} B \Leftrightarrow A(k) < B(k)$, where $k = \min\{i \mid A(i) \neq B(i)\}$

### B. Punctured Polar Codes

Let $u_0^{N-1}$ be a data vector, $x_0^{N-1} = u_0^{N-1} \cdot \mathbf{G}_N$ the corresponding encoded vector[1], and $P \subset \{0,1\}^N$ a puncturing pattern (hence, $x_i$ is punctured if and only if $P(i) = 1$). Assume that that transmission takes place over a Binary-Input Discrete Memoryless Channel (B-DMC) $W$, with output alphabet $\mathcal{Y}$. We denote by $y[P]_0^{N-1}$ the punctured received sequence. For the sake of simplicity, we shall assume that $y[P]_0^{N-1}$ is a sequence of length $N$, with $y[P]_0^{N-1} \in (\mathcal{Y} \cup \{\star\})^N$ and $y[P]_i = \star \Leftrightarrow P(i) = 1$ (the output alphabet is extended with the symbol $\star$ to account for punctured positions).

Let $I \subset \{0,1\}^N$ be an information pattern, with $|I| = K < N - N_p$. We denote by $(N, P, I)$ the corresponding punctured Polar code of dimension $K$ and length $N - N_p$. Frozen bits are assumed to be set to zero, that is $u_i = 0$, for $I(i) = 0$. Using similar notation to that of [1], we denote by $W[P]_N^{(i)}$ the virtual channel with input $u_i$ and output $(y[P]_0^{N-1}, u_0^{i-1})$, and by $\mathcal{I}(W[P]_N^{(i)})$ the symmetric capacity of $W[P]_N^{(i)}$. For a given $P$, the optimal information set $I$ is given by the $K$ positions with highest $\mathcal{I}(W[P]_N^{(i)})$ values.

The symmetric capacity $\mathcal{I}(W[P]_N^{(i)})$ can be determined from the distribution of the log-likelihood ratio (LLR) values $\gamma_i$, computed by the *genie-aided*[2] SC decoder:

$$\gamma_i = \log\left(\frac{\Pr(u_i = 0 \mid y[P]_0^{N-1}, u_0^{i-1})}{\Pr(u_i = 1 \mid y[P]_0^{N-1}, u_0^{i-1})}\right), \tag{2}$$

[1]Note that in [1], encoding is defined by $x_0^{N-1} = u_0^{N-1} \cdot \widetilde{\mathbf{G}}_N$, where $\widetilde{\mathbf{G}}_N = \mathbf{B}_N \cdot \mathbf{G}_N$. Hence, $\widetilde{\mathbf{G}}_N$ differs from $\mathbf{G}_N$ by a column permutation, but the two constructions are known to be equivalent.

[2]Following [1], we refer to such a decoder as *genie-aided*, since it assumes perfect knowledge of the previous data bits $u_0^{i-1}$.

Let $\Gamma(W[P]_N^{(i)})$ denote the probability distribution function of $\gamma_i$. In practice, $\Gamma(W[P]_N^{(i)})$ can be estimated, under the all-zero codeword assumption, by relying on the density evolution (DE) technique [10]. It further allows determining the *genie-aided* error probability:

$$p_{i|}^{\text{GA}} = \Pr(\gamma_i < 0) + \frac{1}{2}\Pr(\gamma_i = 0), \tag{3}$$

which can be used as an alternative metric for the choice of the information pattern $I$, by choosing by the $K$ positions with lowest $p_{i|}^{\text{GA}}$ values.

**Definition 1.** *The data bit erasure pattern is a binary vector* $E[P] \in \{0,1\}^N$*, such that* $E[P](i) = 1$ *iff* $\mathcal{I}(W[P]_N^{(i)}) = 0$.

Assuming that $\mathcal{I}(W) > 0$, it can be shown that $E[P]$ does actually not depend on $W$, but only on the puncturing pattern $P$, as explained in Section IV-A. When no confusion is possible, we shall simply denote $E[P]$ by $E$. It follows from the above definition that the erasure pattern indicates data bit positions $i \in \{0, \ldots, N-1\}$ such that $\Gamma(W[P]_N^{(i)})$ is the Dirac distribution supported at zero. It is clear that the information pattern $I$ should be chosen such that $E \wedge I = \underline{0}$, where $\wedge$ denotes the bit-wise AND operation, and $\underline{0}$ the all-zero vector. The following theorem is proved in [9].

**Theorem 2.** $|E[P]| = |P|$*, for any puncturing pattern* $P$.

### C. Successive Cancellation Decoder

The SC decoder takes advantage of the code construction, by estimating the value of a data bit $u_i$, denoted by $\hat{u}_i$, based on the previous estimates $\hat{u}_0^{i-1}$. To this end, the following LLR value is computed:

$$\hat{\gamma}_i = \log\left(\frac{\Pr(u_i = 0 \mid y[P]_0^{N-1}, \hat{u}_0^{i-1})}{\Pr(u_i = 1 \mid y[P]_0^{N-1}, \hat{u}_0^{i-1})}\right), \tag{4}$$

then the data bit estimate is defined by $\hat{u}_i = \dfrac{1 - \text{sign}(\hat{\gamma}_i)}{2}$, if $I(i) = 1$, and $\hat{u}_i = u_i$, if $I(i) = 0$ (note that $\text{sign}(0) = \pm 1$ with equal probability). Since the SC decoder has to rely on the previous estimates of the data bits, $\hat{\gamma}_i$ and $\gamma_i$ values need not have the same probability distribution. However, in practice, the Word Error Rate (WER) performance of the SC decoder can be approximated by the following formula, which is known to be tight, especially in the high Signal to Noise Ratio (SNR) regime [11]:

$$\text{WER} \approx \text{WER}^{\text{GA}} = 1 - \prod_{i \in \mathcal{I}} (1 - p_{i|}^{\text{GA}}) \tag{5}$$

## III. Equivalent and Primitive Puncturing Patterns

### A. Equivalent Puncturing Patterns

For $j = 0, ..., n-1$ and $k = 0, 2^{j+1}, \ldots, (2^{n-j-1} - 1)2^{j+1}$, let $\phi_{k,j} : \{0, \ldots, N-1\} \to \{0, \ldots, N-1\}$ be the permutation swapping the elements within the blocks $[k, \ldots, k + 2^j - 1]$ and $[k + 2^j, \ldots, k + 2^{j+1} - 1]$. Hence:

$$\phi_{k,j}(i) = \begin{cases} i, & \text{if } i < k \text{ or } i \geq k + 2^{j+1} \\ i + 2^j, & \text{if } k \leq i < k + 2^j \\ i - 2^j, & \text{if } k + 2^j \leq i < k + 2^{j+1} \end{cases} \tag{6}$$

TABLE I
NON-EQUIVALENT PATTERNS WITH SAME ERASURE SET

| punct. pattern | $\mathcal{I} = \{\mathcal{I}(W[P]_7^{(i)}) \mid i = 0, \ldots, 7\}$<br>$p = \{p_{i\vert}^{\mathrm{GA}} \mid i = 0, \ldots, 7\}$ $[W = \mathrm{BEC}(\varepsilon = 0.5)]$ |
|---|---|
| $P$ | $\mathcal{I} = \{\ 0,\ 0,\ 0,\ \mathbf{0.25},\ 0,\ \mathbf{0.375},\ \mathbf{0.4375}, \mathbf{0.9375}\ \}$<br>$p = \{\, 0.5, 0.5, 0.5, \mathbf{0.375}, 0.5, \mathbf{0.3125}, \mathbf{0.2813}, \mathbf{0.0313}\,\}$ |
| $P'$ | $\mathcal{I}' = \{\ 0,\ 0,\ 0,\ \mathbf{0.25},\ 0,\ \mathbf{0.250},\ \mathbf{0.5625}, \mathbf{0.9375}\ \}$<br>$p' = \{\, 0.5, 0.5, 0.5, \mathbf{0.375}, 0.5,\ \mathbf{0.375},\ \mathbf{0.2188}, \mathbf{0.0313}\,\}$ |

$P = [11101000],\ P' = [11011000],\ E[P] = E[P'] = P$

In the following, $\phi_{k,j}$ will be referred to as an *elementary permutation*. Clearly, $\phi_{k,j} \circ \phi_{k,j}$ is the identity permutation.

**Definition 3.** *Two puncturing patterns $P$ and $P'$ are said to be equivalent, denoted by $P \approx P'$, if there exists a sequence of elementary permutations $\phi_{k_1,j_1}, \ldots, \phi_{k_t,j_t}$ that transform $B_N(P)$ into $B_N(P')$. Hence:*

$$P' = B_N(\phi_{k_t,j_t}(\cdots(\phi_{k_1,j_1}(B_N(P)))\cdots)) \tag{7}$$

**Proposition 4.** *Let $P' \approx P$, $y[P]_0^{N-1} \in (\mathcal{Y} \cup \{\star\})^N$, and $y'[P']_0^{N-1}$ be the permuted version of $y[P]_0^{N-1}$, defined by the permutation $B_N \circ \phi_{k_t,j_t} \circ \cdots \circ \phi_{k_1,j_1} \circ B_N$, from Eq. (7). Let $\gamma_0^{N-1}$ and ${\gamma'}_0^{N-1}$ be the LLR values computed by the genie-aided SC decoder, under the all-zero codeword assumption, when submitted with $y[P]_0^{N-1}$ and $y'[P']_0^{N-1}$ inputs, respectively. Then $\gamma_0^{N-1} = {\gamma'}_0^{N-1}$.*

*Proof*: It is enough to prove the proposition for $t = 1$, since $B_N \circ \phi_{k_t,j_t} \circ \cdots \circ \phi_{k_1,j_1} \circ B_N = (B_N \circ \phi_{k_t,j_t} \circ B_N) \circ \cdots \circ (B_N \circ \phi_{k_1,j_1} \circ B_N)$. Assuming $t = 1$, the proposition is equivalent to the fact that the genie-aided SC decoder on $\widetilde{\mathbf{G}}_N = \mathbf{B}_N \cdot \mathbf{G}_N$ is invariant by any elementary permutation $\phi_{k,j}$ of $y[P]_0^{N-1}$. This can be proved by induction, starting from the observation that permuting the $y_0$ and $y_1$ values on a kernel block (see Fig. 1) does not change the values of $\gamma_0$ and $\gamma_1$, since under the all-zero codeword assumption $\gamma_0 = 2\,\mathrm{atanh}(\tanh(\ell_0/2)\tanh(\ell_1/2))$ and $\gamma_1 = \ell_0 + \ell_1$, where $\ell_i = \log(\Pr(x_i = 0 \mid y_i)/\Pr(x_i = 1 \mid y_i))$, $i = 0, 1$. The induction step follows from the fact that $\phi_{k,j}$ permutes the half-top and half-bottom outputs of a $\widetilde{\mathbf{G}}_{2^{j+1}}$ sub-block of $\widetilde{\mathbf{G}}_N$. □

**Theorem 5.** *Let $P$ and $P'$ be two equivalent patterns. Then, the following properties hold:*

$(i)$ $\Gamma(W[P]_N^{(i)}) = \Gamma(W[P']_N^{(i)}),\ \forall W, \forall i = 0, \ldots, N-1$
$(ii)$ $\mathcal{I}(W[P]_N^{(i)}) = \mathcal{I}(W[P']_N^{(i)}),\ \forall W, \forall i = 0, \ldots, N-1$
$(iii)$ $E[P] = E[P']$

*Proof*: $(i)$ follows from Proposition 4, $(ii)$ follows from $(i)$, and $(iii)$ from $(ii)$. □

In particular, it follows that the genie-aided SC decoder has the same error probability ($p_{i\vert}^{\mathrm{GA}}, i = 0, \ldots, N-1$), and therefore the same $\mathrm{WER}^{\mathrm{GA}}$ (Eq. (5)), irrespective of which of $P$ or $P'$ is being used.

It is worth noticing that there exist puncturing patterns with $E[P] = E[P']$, but which are not equivalent. An example is provided in Table I for the Polar code of length $N = 8$, where $P = [1, 1, 1, 0, 1, 0, 0, 0]$ and $P' = [1, 1, 0, 1, 1, 0, 0, 0]$. It can be seen that $E[P] = E[P'] = P$, corresponding to indexes $i = 0, 1, 2, 4$ with $\mathcal{I}(W[P]_N^{(i)}) = \mathcal{I}(W[P']_N^{(i)}) = 0$. However, $\mathcal{I}(W[P]_N^{(i)}) \neq \mathcal{I}(W[P']_N^{(i)})$ for $i = 5, 6$, hence $P$ and $P'$ cannot be equivalent[3]. Table I also provides the error probability values $p_{i\vert}^{\mathrm{GA}}$. Using Eq. (5), it can be verified that, depending on the number of information bits, better $\mathrm{WER}^{\mathrm{GA}}$ performance is provided by $P'$ if $K = 2$ (with information bits $\{u_6, u_7\}$), or $P$ if $K = 3$ (with information bits $\{u_5, u_6, u_7\}$).

Considering equivalence classes of puncturing patterns is particularly relevant when brute-force search is used to find the puncturing pattern (together with the corresponding information set) providing the best WER performance. In the following section we introduce a constructive method to determine a unique representative in each equivalence class, which allows reducing the complexity of the brute-force search.

**Algorithm 1** Function $\phi$

```
procedure φ(P)
    for j = 0 : 1 : n − 1 do
        for k = 0 : 2^{j+1} : N − 1 do
            if P_k^{k+2^j−1} <_lex P_{k+2^j}^{k+2^{j+1}−1} then
                P ← φ_{k,j}(P)
            end if
        end for
    end for
    return P
end procedure
```

### B. Primitive Pattern of an Equivalence Class

For a puncturing pattern $P$, let $\phi(P)$ be the puncturing pattern defined by Algorithm 1. Algorithm 1 visits all the elementary permutations in a specific order, but an elementary permutation $\phi_{k,j}$ is performed only if $[P(k), \ldots, P(k + 2^j - 1)] <_{lex} [P(k + 2^j), \ldots, P(k + 2^{j+1} - 1)]$. Note that in such a case, $\phi_{k,j}$ actually permutes the above two blocks. We further define:

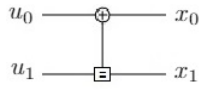
$$\Phi(P) = B_N(\phi(B_N(P)) \tag{8}$$

**Theorem 6.** *The following properties hold:*

$(i)$ $\Phi(P) \approx P$
$(ii)$ $P \approx P' \Leftrightarrow \Phi(P) = \Phi(P')$
$(iii)$ $\Phi(P) = \underset{P' \approx P}{\mathrm{argmax}_{lex}}\, B_N(P')$

From the above theorem, it follows that $\Phi(P)$ only depends on the equivalence class of $P$. In the following, we say that $P$ is a *primitive pattern* if $\Phi(P) = P$. Hence, each equivalence class contains one and only one primitive pattern. The proofs of the above and the following theorem will be provided in an extended version of this paper.

[3] In [8, 9], it is stated without proof that $E[P] = E[P']$ if and only if $p_{i\vert}^{\mathrm{GA}} = {p'}_{i\vert}^{\mathrm{GA}},\ \forall i = 0, \ldots, N-1$. Our example also shows that the statement in [8, 9] is actually false.




| $P$ | [00] | [10] | [01] | [11] |
|---|---|---|---|---|
| $E$ | [00] | [10] | [10] | [11] |

Fig. 1. Kernel bloc $\mathbf{G}_2$, and list of puncturing and erasure patterns

**Theorem 7.** *Let $P$ be a puncturing pattern, $\bar{P} = B_N(P)$, $\bar{P}_0 = [\bar{P}(0), \ldots, \bar{P}(N/2-1)]$ and $\bar{P}_1 = [\bar{P}(N/2), \ldots, \bar{P}(N-1)]$. Then $P$ is primitive if and only if $B_{N/2}(\bar{P}_0)$ and $B_{N/2}(\bar{P}_1)$ are primitive, and $\bar{P}_1 \leq_{lex} \bar{P}_0$.*

The above theorem states that primitive patterns can be determined recursively. However, the recursive construction method requires storing all of the primitive patterns of length $N$, in order to derive those of length $2N$, thus becoming unfeasible for large $N$ values. In Section IV, we restrict to a specific class of puncturing patterns, referred to as *symmetric*. We show that any symmetric pattern is primitive, and propose a search-tree algorithm that allows generating symmetric patterns even for large values of $N$.

## IV. Symmetric Puncturing Patterns

### A. Symmetric Patterns

**Definition 8.** *A puncturing pattern $P$ is said to be symmetric*[4] *if $E[P] = P$.*

The following theorem provides a simple characterization of symmetric patterns, as the union of a subset of rows of $\mathbf{G}_N$.

**Theorem 9.** *Let $P \subset \{0,1\}^N$ be a puncturing pattern. Then $E[P] = P$ if and only if $P = \underline{0}$ (the all-zero vector) or $P$ is the union of a subset of rows of $\mathbf{G}_N$, that is, $\exists \mathcal{L} \subset \{0, \ldots, N-1\}$, $P = \bigvee_{i \in \mathcal{L}} G_N^{(i)}$.*

*Proof*: First we prove that if $P = \vee_{i \in \mathcal{L}} G_N^{(i)}$, then $E[P] = P$. Indeed, let $\mathcal{L}_+ = \{0 \leq i \leq N/2-1 \mid i \in \mathcal{L} \text{ or } N/2+i \in \mathcal{L}\}$, and $\mathcal{L}_- = \{0 \leq i \leq N/2-1 \mid N/2+i \in \mathcal{L}\}$. Define $P_+ = \vee_{i \in \mathcal{L}_+} G_{N/2}^{(i)}$ and $P_- = \vee_{i \in \mathcal{L}_-} G_{N/2}^{(i)}$. Since $G_N^{(i)} \subset G_N^{(k)}$, for any $k, i$, such that $G_N(k,i) = 1$, it follows easily that $P = (P_+, P_-)$. Then the equality $E[P] = P$ follows by using induction arguments.

To prove the direct implication, we prove that for any $P$, $E[P]$ is the the union of a subset of rows of $\mathbf{G}_N$. The proof goes by induction on $n$. The statement can be easily verified for $n = 1$ ($N = 2$), according to the list of erasure patterns provided in Fig. 1. Let $N = 2^n$, with $n > 1$, and $E = E[P]$. Considering the recursive structure from Fig. 2, the puncturing pattern $P$ generates an erasure pattern on each of the $N/2$ kernel blocks on the right side of the figure. Putting them together, they produce two puncturing patterns on the top and bottom $\mathbf{G}_{N/2}$ blocks, indicated respectively by $P_+$ and $P_-$. Hence, $E = (E_+, E_-)$ is the concatenation

[4]Note that using Arikan's generator matrix $\widetilde{\mathbf{G}}_N = \mathbf{B}_N \cdot \mathbf{G}_N$, symmetric patterns correspond to patterns $P$, such that $E[P] = B_N(P)$. Such patterns have been considered in [8], but the link with the rows of $\mathbf{G}_N$ (Theorem 9) has not been not pointed out.

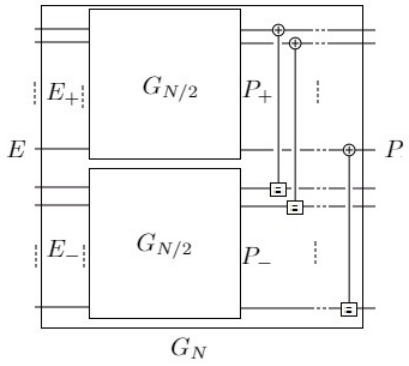



Fig. 2. Recursive structure of a Polar code of length $N$

of $E_+ = E[P_+]$ and $E_- = E[P_-]$. By construction, and considering the possible erasure patterns for kernel blocks, it follows that $P_- \subseteq P_+$, and therefore it can be easily verified that $E_- \subseteq E_+$. By induction, $E_- = \vee_{i \in \mathcal{L}_-} G_{N/2}^{(i)}$ and $E_+ = \vee_{i \in \mathcal{L}_+} G_{N/2}^{(i)}$, with $\mathcal{L}_- \subseteq \mathcal{L}_+ \subseteq \{0, \ldots, N/2-1\}$. It follows that $E = \vee_{i \in \mathcal{L}} G_N^{(i)}$, where $\mathcal{L} = \mathcal{L}_+ \cup \{N/2 + \mathcal{L}_-\}$, and also using the fact that $G_N^{(N/2+i)} = (G_{N/2}^{(i)}, G_{N/2}^{(i)})$, for any $0 \leq i \leq N/2-1$. □

Note that the proof of the above theorem actually demonstrates that an erasure pattern is necessarily the the union of a subset of rows of $\mathbf{G}_N$. It also indicates how the erasure pattern $E[P]$ can be determined in practice: one has to propagate $P$, from right to the left, throughout the recursive structure shown in Fig. 2. At each propagation step, the pattern is propagated through a number of kernel blocks, according to the $P \mapsto E[P]$ rule shown in Fig. 1. In particular, this shows that the erasure pattern does not depend on the channel $W$, but only on the puncturing pattern $P$.

**Theorem 10.** *If $P$ is a symmetric pattern, then $\Phi(P) = P$, i.e. $P$ is the primitive pattern of its equivalence class.*

*Proof*: First, it can be proved that the bit-reversal permutation of any row of $G_N$ is also a row of $G_N$. Therefore, if $P$ is a symmetric pattern, then so is $B_N(P)$. Secondly, one can easily check that any row of $\mathbf{G}_N$, and therefore any symmetric pattern, is invariant by $\phi$. From the above observations, it follows that for any symmetric pattern $P$, one has: $\Phi(P) = B_N(\phi(B_N(P))) = B_N(B_N(P)) = P$. □

Due to space limitations, the proof of the following theorem will be included in an extended version of the paper.

**Theorem 11.** *Let $P$ be a puncturing pattern, $\bar{P} = B_N(P)$, $\bar{P}_0 = [\bar{P}(0), \ldots, \bar{P}(N/2-1)]$ and $\bar{P}_1 = [\bar{P}(N/2), \ldots, \bar{P}(N-1)]$. Then $P$ is symmetric if and only if $B_{N/2}(\bar{P}_0)$ and $B_{N/2}(\bar{P}_1)$ are symmetric, and $\bar{P}_1 \subseteq \bar{P}_0$.*

While the previous theorem indicates that symmetric patterns can be determined recursively (similarly to primitive patterns), the search-tree algorithm introduced in the next section is aimed at generating symmetric patterns, by exploiting their description as a union of a subset of rows of $\mathbf{G}_N$ (Theorem 9).

### B. Search-Tree Algorithm

The *order* $\lambda$ of a symmetric puncturing pattern $\mathcal{P}$ is defined as the minimum number of rows of $\mathbf{G}_N$ that generate $\mathcal{P}$. Hence $\lambda = \min\left\{|\mathcal{L}| \mid P = \cup_{i\in\mathcal{L}} G_N^{(i)}\right\}$. It can be easily verified that a symmetric puncturing pattern $P$, of order $\lambda$ and weight $|P| = N_p$, is generated by a set $\mathcal{L}$ of rows of $\mathbf{G}_N$, such that:

- $|\mathcal{L}| = \lambda$ and $\forall (i,j) \in \mathcal{L}^2, G_N^{(i)} \not\subseteq G_N^{(j)}$,
- $|\vee_{i\in\mathcal{L}} G_N^{(i)}| = N_p$

The search-tree algorithm proposed in this section generates symmetric puncturing patterns of order $\lambda \leq \lambda_{max}$, for some fixed $\lambda_{max}$ value. Note that for small $\lambda_{max}$ values, it might not be possible to get any value of $N_p$, thus resulting in a loss in flexibility with respect to the punctured code-length. However, the flexibility quickly increases with the considered maximum order.

We denote by $L = (L(1), \ldots, L(\lambda_{max}))$ a vector such that $0 \leq L(\tau) \leq N$, for any $\tau = 1, \ldots, \lambda_{max}$. For $1 \leq \lambda \leq \lambda_{max}$, we further denote $P_\lambda = \vee_{1\leq\tau\leq\lambda} G_N^{(L(\tau))}$. Hence, $L(1), \ldots, L(\lambda)$ are the $\lambda$ indexes of the $\mathbf{G}_N$-rows contributing to the symmetric puncturing pattern $P_\lambda$. Since the rows of $\mathbf{G}_N$ are numbered from 0 to $N-1$, we define $G_N^{(N)} = \underline{0}$, the all-zero vector. We also define $P_0 = \underline{0}$.

For given $N_p$ and $\lambda_{max}$ values, the search-tree algorithm works as follows:

0) Let $\lambda = 1$ and $L(\tau) = N$, for any $\tau = 1, \ldots, \lambda_{max}$.
1) Calculate the complementary weight values, defined by $W_{cp}(i) = |G_N^{(i)}| - |G_N^{(i)} \wedge P_{\lambda-1}|$, for $i = 0, \ldots, N-1$.
2) Determine the largest index $i$ strictly less than $L(\lambda)$, such that:
$$\begin{cases} 0 < W_{cp}(i) \leq N_p - |P_\lambda|, & \text{if } \lambda < \lambda_{max} \\ W_{cp}(i) = N_p - |P_\lambda|, & \text{if } \lambda = \lambda_{max} \end{cases} \quad (9)$$
If no index $i$ verifies Eq. (9), then set $\lambda = \lambda - 1$ (upper tree level) and go to Step 4).
3) Set $L(\lambda) = i$;
   - If $|P_\lambda| = |N_p|$, then store $P_\lambda$ (symmetric puncturing pattern of order $\lambda$ and weight $N_p$), and go back to Step 2) (continue the search on the same tree level);
   - Otherwise, set $\lambda = \lambda + 1$, then go back to Step 1) (continue search on the lower tree level).
4) If $\lambda = 0$ then all the puncturing patterns have been found and the process stops; otherwise, go to Step 1).

It can be noticed that the complexity of this algorithm is $\mathcal{O}(N^{\lambda_{max}})$, which allows generating symmetric patterns even for high $N$ values, assuming the $\lambda_{max}$ is relatively small.

## V. Numerical Results

Throughout this section, we consider a Binary-Input Additive While Gaussian Noise (BI-AWGN) channel, with noise variance $\sigma^2$. For a given puncturing pattern $P$, the information pattern minimizing the $\text{WER}^{\text{GA}}$ (Eq. (5)) depends on both $P$ and $\sigma^2$, and can be determined by DE, as explained in Section II. We denote this information pattern by $I(P, \sigma^2)$, and the corresponding word error rate by $\text{WER}^{\text{GA}}(P, \sigma^2)$. Given a *target word error rate* $\eta > 0$, we define the *noise (variance) threshold* $\sigma^2_{\text{th}}(P)$, as follows:

$$\sigma^2_{\text{th}}(P) = \sup\left\{\sigma^2 \mid \text{WER}^{\text{GA}}(P, \sigma^2) \leq \eta\right\} \quad (10)$$

It corresponds to the maximum $\sigma^2$ for which the punctured code provides a word error rate less than or equal to $\eta$, assuming an appropriate choice of the information pattern.

TABLE II
Number of primitive patterns for $N = 64$

| $N_p$ | 6 | 8 | 10 | 12 | 14 |
|---|---|---|---|---|---|
| primitive | 381 | 2005 | 10599 | 42894 | 150502 |
| symmetric | 156 | 605 | 2045 | 5913 | 14345 |
| non-symmetric | 225 | 1600 | 8554 | 37281 | 136157 |

### A. Primitive vs. Symmetric Patterns

For $N = 64$, we generate all the primitive puncturing patterns, by using the recursive construction method from Theorem 7. We further class them in two categories, namely symmetric patterns and non-symetric ones. The number of patterns in each category is provided in Table II. It can be seen that the number of non-symmetric patterns increases much faster than the number of symmetric ones. For all the puncturing patterns, we use the DE technique[5] to determine the noise threshold for a target word error rate $\eta = 10^{-4}$, assuming $K = 20$ information bits. Fig. 3 shows the noise threshold (in dB) of the best puncturing patterns in each one of the two categories, for various $N_p$ values. We observe that the best symmetric and non-symmetric patterns have virtually the same noise threshold for any $N_p$ value. Therefore, one may restrict the search of good puncturing patterns to symmetric patterns only. Moreover, Fig. 3 also shows that by considering only symmetric patterns of order $\lambda \leq 3$, the corresponding noise threshold values are very close to the optimal ones. This justifies the optimization method considered in the next section, which consists in exploring only symmetric patterns of relatively low order.

### B. Symmetric vs. SotA Patterns

We consider the punctured codes with parameters $(N, K, N_p) = (256, 64, 85)$ and $(1024, 256, 336)$. For each $\sigma^2$ value, we rely on DE to find the symmetric puncturing pattern $P$ (and corresponding information pattern $I(P, \sigma^2)$) that minimizes $\text{WER}^{\text{GA}}(P, \sigma^2)$. Due to the very large number of symmetric patterns, evaluation $\text{WER}^{\text{GA}}(P, \sigma^2)$ for all of them is not feasible. Thus, we restrict our search to symmetric patterns of order $\lambda \leq \lambda_{max}$, by using the search-tree algorithm from Section IV-B. The WER performance of the punctured Polar code under SC decoding is then evaluated by Monte-Carlo simulation. Simulation results, are shown in Fig. 4. For comparison purposes, we have also included the WER performance of rate compatible codes (with same $(N, N_p, K)$

[5]We use DE under the Gaussian Approximation hypothesis [11]

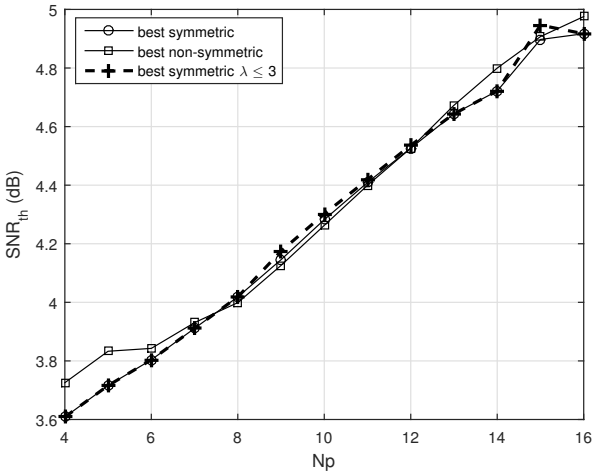



Fig. 3. Comparisons of the best symmetric and non-symmetric patterns for $N = 64$, $K = 20$ and $\eta = 10^{-4}$

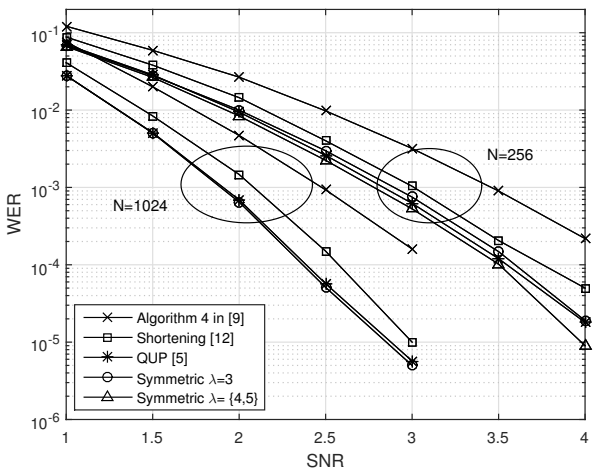



Fig. 4. WER performance of rate-compatible codes with parameters $(N, K, N_p) = (256, 64, 85)$ and $(1024, 256, 336)$

parameters), obtained by using either puncturing [5, 9] or shortening [12] techniques from the state of the art.

The number of symmetric patterns of order less than or equal to $\lambda_{max}$ is given in Table III, for $\lambda_{max} = 3, 4, 5$. For $N = 256$, we note that the best symmetric puncturing patterns of order less than or equal to $\lambda_{max}$, are the same for both $\lambda_{max} = 4$ and $\lambda_{max} = 5$. Hence, Fig. 4 contains only one corresponding plot, indicated by $\lambda_{max} = \{4, 5\}$. This further confirms the observation from the previous section, concerning the relatively small order of the best puncturing patterns.

Concerning the comparison with the state of the art, we observe that the proposed method outperforms the methods from [9, 12], and provides slightly better performance than the Quasi-uniform Puncturing (QUP) method [5], in the high SNR regime ($N = 256$). It is worth noticing that, although the approach in [5] is different from ours, it actually turns out that the QUP pattern is a symmetric pattern of order $\lambda = 4$ ($N = 256$) or $\lambda = 3$ ($N = 1024$).

TABLE III
NUMBER OF SYMMETRIC PATTERNS OF ORDER $\leq \lambda_{max}$

| $\lambda_{max}$ | 3 | 4 | 5 |
|---|---|---|---|
| $N = 256$ | 2940 | $3.5 \cdot 10^5$ | $13.5 \cdot 10^6$ |
| $N = 1024$ | $3 \cdot 10^5$ | – | – |

## VI. CONCLUSION

In this paper, we studied equivalence classes of puncturing patterns and proposed a constructive solution to find a unique representative of each class, called primitive pattern. Primitive patterns have been further classified into symmetric patterns, equal to their erasure pattern, and non-symmetric ones. As the complexity of an exhaustive search through all the primitive patterns is impracticable even for relatively small $N$, we concentrated on the first category and indicated a method to generate symmetric patterns, by using a search-tree algorithm that exploits their characterization as union of rows of the $\mathbf{G}_N$ matrix. Simulations showed that the proposed method performs better than the main methods of the state-of-the-art.

## ACKNOWLEDGMENT

The research leading to these results received funding from the European Commission H2020 Programme, under grant agreement 671650 (mmMagic Project).

## 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, 2009.

[2] S.-N. Hong, D. Hui, and I. Marić, "Capacity-achieving rate-compatible polar codes," *arXiv preprint arXiv:1510.01776*, 2015.

[3] H. Saber and I. Marsland, "An incremental redundancy Hybrid ARQ scheme via puncturing and extending of polar codes," *IEEE Transactions on Communications*, vol. 63, no. 11, pp. 3964–3973, 2015.

[4] K. Chen, K. Niu, and J. Lin, "A hybrid ARQ scheme based on polar codes," *IEEE Communications Letters*, vol. 17, no. 10, pp. 1996–1999, 2013.

[5] K. Niu, K. Chen, and J.-R. Lin, "Beyond turbo codes: rate-compatible punctured polar codes," in *International Conference on Communications (ICC)*, pp. 3423–3427, IEEE, 2013.

[6] D.-M. Shin, S.-C. Lim, and K. Yang, "Design of length-compatible polar codes based on the reduction of polarizing matrices," *IEEE Transactions on Communications*, vol. 61, no. 7, pp. 2593–2599, 2013.

[7] A. Eslami and H. Pishro-Nik, "A practical approach to polar codes," in *IEEE International Symposium on Information Theory Proceedings (ISIT)*, pp. 16–20, IEEE, 2011.

[8] J. Kim, J.-H. Kim, and S.-H. Kim, "An efficient search on puncturing patterns for short polar codes," in *International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 182–184, IEEE, 2015.

[9] L. Zhang, Z. Zhang, X. Wang, Q. Yu, and Y. Chen, "On the puncturing patterns for punctured polar codes," in *International Symposium on Information Theory*, pp. 121–125, IEEE, 2014.

[10] R. Mori and T. Tanaka, "Performance and construction of polar codes on symmetric binary-input memoryless channels," in *International Symposium on Information Theory*, pp. 1496–1500, IEEE, 2009.

[11] D. Wu, Y. Li, and Y. Sun, "Construction and block error rate analysis of polar codes over AWGN channel based on gaussian approximation," *IEEE Communications Letters*, vol. 18, no. 7, pp. 1099–1102, 2014.

[12] R. Wang and R. Liu, "A novel puncturing scheme for polar codes," *IEEE Communications Letters*, vol. 18, no. 12, pp. 2081–2084, 2014.