An Improved SCFlip Decoder for Polar Codes

Ludovic Chandesris, Valentin Savin, D Declercq

HAL Id: cea-01573437

https://cea.hal.science/cea-01573437

Submitted on 9 Aug 2017

An Improved SCFlip Decoder for Polar Codes

L.Chandesris[†‡], V.Savin[†], D.Declercq[‡]

ludovic.chandesris@cea.fr, valentin.savin@cea.fr, declercq@ensea.fr

[†]CEA-LETI / Minatec, Grenoble, France   [‡]ETIS, ENSEA/UCP/CNRS, Cergy-Pontoise, France

Abstract—**This paper focuses on the recently introduced Successive Cancellation Flip (SCFlip) decoder of polar codes. Our contribution is twofold. First, we propose the use of an optimized metric to determine the flipping positions within the SCFlip decoder, which improves its ability to find the first error that occurred during the initial SC decoding attempt. We also show that the proposed metric allows closely approaching the performance of an ideal SCFlip decoder. Second, we introduce a generalisation of the SCFlip decoder to a number of ω nested flips, denoted by SCFlip-ω, using a similar optimized metric to determine the positions of the nested flips. We show that the SCFlip-2 decoder yields significant gains in terms of decoding performance and competes with the performance of the CRC-aided SC-List decoder with list size $L = 4$, while having an average decoding complexity similar to that of the standard SC decoding, at medium to high signal to noise ratio.**

Keywords—*Polar Codes, SCFlip decoding, order statistic decoding.*

I. INTRODUCTION

Polar codes are a new class of error-correcting codes, proposed by Arikan in [1], which provably achieve the capacity of any symmetric binary-input memoryless channel under successive cancellation (SC) decoding. However, for short to moderate blocklengths, the frame error rate (FER) performance of polar codes under successive cancellation decoding does not compete with other families of codes such as LDPC or Turbo-Codes.

In [2], a Successive Cancellation list-decoding (SCL) has been proposed, which significantly outperforms the simple SC decoding, and approaches the Maximum-Likelihood (ML) performance at high signal to noise ratio (SNR). Moreover, when applied to polar codes concatenated with an outer cyclic redundancy check (CRC) code -used to identify the correct message from the decoded list- it has been shown that the SCL decoder may successfully compete with other families of capacity approaching codes, like Low Density Parity Check (LDPC) codes. However, SCL decoder suffers from high storage and computational complexity, which grows linearly with the size of the list. Several improvements have been proposed to reduce its computational complexity, such as Stack decoding (SCS) in [3], but at a cost of an increasing storage complexity.

Successive Cancellation Flip decoder has been introduced in [4] for the BEC channel and generalised to binary-input additive white Gaussian noise (BI-AWGN) channel by using a CRC in [5]. It is close to the order statistic decoding proposed in [6] and which has been specifically used for polar codes in [7]. The idea is to allow a given number of new decoding attempts, in case that a failure of the initial SC decoding attempt is detected by the CRC. Each new

attempt consists in flipping one single decision - starting with the least reliable one, according to the absolute value of the log-likelihood ratio (LLR) - of the initial SC attempt, then decoding the subsequent positions by using standard SC decoding. The above procedure is iterated until the CRC is verified or a predetermined maximum number of flips is reached. The SCFlip decoder provides an interesting trade-off between decoding performance and decoding complexity, since each new decoding attempt is only performed if the previous one failed. Consequently, the average computational complexity of the SCFlip decoder approaches the one of SC decoder at medium to high SNR, while competing with the CRC-aided SCL with $L = 2$, in terms of error correction performance.

In this work we propose two improvements to the SCFlip decoder, aimed at both increasing the error correction performance and reducing the computational complexity. First, we propose the use of a new metric to determine the flipping positions within the SCFlip decoder. The proposed metric takes into account the sequential aspect of the SC decoder, and we show it yields an improved FER performance and a reduced computational complexity compared to LLR-based metric used in [5]. Second, we introduce a generalization of the SCFlip decoder to a number of ω nested flips, denoted by SCFlip-ω. We show that the SCFlip-2 decoder with the proposed metric to select the two flipping positions competes with the CRC-aided SCL decoder with $L = 4$, in terms of decoding performance, while having an average decoding complexity similar to that of the standard SC decoding at medium to high SNR. Furthermore, we also use an *Oracle-assisted* decoder as in [5] to determine the lower bound of these SCFlip decoders and shows that both proposed algorithms for $\omega = 1$ and $\omega = 2$ can closely approach the optimal performance.

The paper is organised as follows. In Section II, a review of Polar Codes is presented. Section III describes the SCFlip-ω decoder. Section IV presents the proposed metric for the selection of the flipping positions. Simulation results are presented in section V.

II. PRELIMINARIES

A. Polar Codes

A Polar Code is characterized by the three-tuple $(N, K, \mathcal{I})$, where $N = 2^n$ is the blocklength, K the number of information bits and $\mathcal{I}$ is the set of K indices indicating the position of the information bits inside the block of size N. Bits corresponding to positions $i \notin \mathcal{I}$ are referred to as *frozen bits* and are fixed to pre-determined values known at both the encoder and the decoder.

We denote $\mathbf{U} = u_0^{N-1}$ the data vector, of length N, containing K information bits at the positions $i \in I$, and

$N - K$ frozen bits that are set to zero. The encoded vector, denoted by $\mathbf{X}$, is obtained by:

$$\mathbf{X} = \mathbf{U} \cdot \mathbf{G}_N$$

where $\mathbf{G}_N$ is the generator matrix defined as in [1]. We further denote by $\mathbf{Y}$ the data received from the channel and used at the decoder input. $\hat{\mathbf{U}} = \hat{u}_0^{N-1}$ denotes the decoder's output, with $\hat{u}_i$ being the estimation of the bit u_i.

B. Successive Cancellation Decoder

SC decoder is the standard low complexity decoder of polar codes given in [1]. The decoding process consists in taking a decision on bit u_i, denoted $\hat{u}_i$, according to the sign of the LLR:

$$L(u_i) = \log \left(\frac{\Pr(u_i = 0 | \mathbf{Y}, \hat{u}_0^{i-1})}{\Pr(u_i = 1 | \mathbf{Y}, \hat{u}_0^{i-1})} \right)$$

using the decision function h:

$$\hat{u}_i = h(L(u_i)) \overset{\text{def}}{=} \begin{cases} u_i & \text{if } i \notin \mathcal{I} \\ \frac{1 - \text{sign}(L(u_i))}{2} & \text{if } i \in \mathcal{I} \end{cases} \quad (1)$$

where by convention $\text{sign}(0) = \pm 1$ with equal probability. Note that the computations and decisions are performed sequentially in SC decoder, as the estimation of the current bit u_i depends on the previous decoded bits $\hat{u}_0^{i-1}$.

III. DEFINITION AND ANALYSIS OF SCFLIP-ω DECODERS

A. Definition of SCFlip-ω Decoders

Let $\mathcal{C}(N, K + r, \mathcal{I})$ denotes the serial concatenation of an outer $(K + r, K)$ CRC code and an inner $(N, K + r, \mathcal{I})$ Polar code. Note that the number of unfrozen positions of the Polar code is $K + r$, where K is the number of information bits, and r is the size of the CRC.

The SCFlip decoder [5] consists of a standard SC decoding, possibly followed by maximum number T of new decoding attempts, until no errors are detected by the CRC check. Each new decoding consists of (i) flipping only one decision of the initial SC attempt, then (ii) decoding the subsequent positions by using standard SC decoding. The T *flipping positions* are those corresponding to the lowest absolute values of the LLRs computed during the initial SC attempt.

Algorithm 1 SCFlip decoder with $\omega = 1$

1: **procedure** SCFLIP-1($\mathbf{Y}, \mathcal{I}, T^{(1)}$)
2:     $(\hat{u}_0^{N-1}, \{L(u_i)\}_{i \in \mathcal{I}}) \leftarrow \text{SC}(\mathbf{Y}, \mathcal{I}, \varnothing)$
3:     **if** CRC($\hat{u}_0^{N-1}$)=success **then** Return $\hat{u}_0^{N-1}$ **end if**
4:     $\mathcal{L}_{\text{flip}} = \text{FlipDetermine}(\{L(u_i)\}_{i \in \mathcal{I}}, -1, T^{(1)})$
5:     **for** $j = 0, .., T^{(1)} - 1$ **do**
6:         $\hat{u}_0^{N-1} \leftarrow \text{SC}(\mathbf{Y}, \mathcal{I}, \mathcal{L}_{\text{flip}}(j))$
7:         **if** CRC($\hat{u}_0^{N-1}$)=success **then** Return $\hat{u}_0^{N-1}$ **end if**
8:     **end for**
9:     Return $\hat{u}_0^{N-1}$
10: **end procedure**

However, the success of the SCFlip decoding depends on (i) the ability to find the very first error that occurred during the initial SC attempt, and (ii) the ability of SC to successfully

---

Algorithm 2 SCFlip decoder with $\omega = 2$

1: **procedure** SCFLIP-2($\mathbf{Y}, \mathcal{I}, T^{(1)}, \{T^{(2,1)}, T^{(2,2)}\}$)
2:     $(\hat{u}_0^{N-1}, \{L(u_i)\}_{i \in \mathcal{I}}) \leftarrow \text{SC}(\mathbf{Y}, \mathcal{I}, \varnothing)$
3:     **if** CRC($\hat{u}_0^{N-1}$)=success **then** Return $\hat{u}_0^{N-1}$ **end if**
4:     $\mathcal{L}_{\text{flip}}^{(1)} = \text{FlipDetermine}(\{L(u_i)\}_{i \in \mathcal{I}}, -1, T^{(1)})$
5:     **for** $j = 0, .., T^{(1)} - 1$ **do**
6:         $(\hat{u}_0^{N-1}, \{L(u_i)\}_{\mathcal{I}}) \leftarrow \text{SC}(\mathbf{Y}, \mathcal{I}, \mathcal{L}_{\text{flip}}^{(1)}(j))$
7:         **if** CRC($\hat{u}_0^{N-1}$)=success **then** Return $\hat{u}_0^{N-1}$ **end if**
8:         **if** $j < T^{(2,1)}$ **then**
9:             $\mathcal{L}_{\text{flip}}^{(2)} = (\mathcal{L}_{\text{flip}}^{(2)}, \text{FlipDetermine}($
                 $\{L(u_i)\}_{\mathcal{I}}, j, \mathcal{L}_{\text{flip}}^{(1)}(j), T^{(2,2)})$
10:        **end if**
11:    **end for**
12:    **for** $i = 0, .., T^{(2,1)} - 1$ **do**
13:        **for** $j = 0, .., T^{(2,2)} - 1$ **do**
14:            $\hat{u}_0^{N-1} \leftarrow \text{SC}(\mathbf{Y}, \mathcal{I},$
                 $\{\mathcal{L}_{\text{flip}}^{(1)}(i), \mathcal{L}_{\text{flip}}^{(2)}(j + i \cdot T^{(2,2)})\})$
15:            **if** CRC($\hat{u}_0^{N-1}$)=success **then**
16:                Return $\hat{u}_0^{N-1}$
17:            **end if**
18:        **end for**
19:    **end for**
20:    Return $\hat{u}_0^{N-1}$
21: **end procedure**

---

decode the subsequent positions, once the first position in error has been flipped. In this work we introduce two new enhancements to the SCFlip decoder, aimed at improving the two above-mentioned characteristics. We propose the use of an optimized metric to determine the flipping positions, building upon the probability of a given position being first error that occurred in the initial SC attempt (see section IV). We note that the global structure of the SCFlip decoding stays the same, the only difference being on the generation of the ordered list of flipping positions, denoted by $\mathcal{L}_{\text{flip}}$. We also introduce a generalisation of the SCFlip decoder to a number of ω nested flips: the first flip is performed on one decision of the initial SC attempt, while the i-th flip ($2 \le i \le \omega$) is performed on a bit position belonging to the new decoding trajectory determined by the previous flips (1 to $i - 1$). Such a sequence of ω nested flips will also be referred to as an *order-ω flip*. The SCFlip-ω decoding for $\omega = 1$ and $\omega = 2$ are presented in Algorithm 1 and Algorithm 2, respectively.

A simple and efficient implementation of SCFlip-ω decoder consists in incrementing ω recursively. As long as no errors are detected by the CRC code, we proceed step by step from SC decoder, SCFlip with 1 flip for a number of $T^{(1)}$ attempts, then SCFlip with 2 flips for a number of $T^{(2)}$ attempts. New decoding attempts in SCFlip decoder are similar to the standard SC decoder, the only difference being on the hard decision function. Thus, we use the notation SC($\mathbf{Y}, \mathcal{I}, \mathcal{E}$), where $\mathcal{E}$ is a set of ω indices corresponding to the flipping positions. The hard decision function h' can be defined as follow:

$$\hat{u}_i = h'(L(u_i)) \overset{\text{def}}{=} \begin{cases} h(L(u_i)) & \text{if } i \notin \mathcal{E} \\ 1 - h(L(u_i)) & \text{if } i \in \mathcal{E} \end{cases}$$

Note that we have a standard SC decoder if $\mathcal{E} = \varnothing$. In

Algorithm 1 and Algorithm 2, candidate positions for the ω-flips are stored in an ordered list denoted by $\mathcal{L}_{\text{flip}}^{(\omega)}$ of size $T^{(\omega)}$ and generated by the function *FlipDetermine*, described in Algorithm 3, which is performed using a metric denoted M. The calculation of this metric will be discussed in section IV.

Algorithm 3 is used in SCFlip-1 as well as SCFlip-2 to determine T indexes strictly larger than k_1 of the least reliable decisions according to the proposed metric. To do so, we first calculate the metric vector m, then generate an index vector J, such that $m(J)$ is sorted in descending order (this operation is performed by the function denoted *sort_index*). For SCFlip-1, $\mathcal{L}_{\text{flip}}^{(1)}$ is generated by calling this function once with $T = T^{(1)}$ and $k_1 = -1$. For the SCFlip-2, we have two degrees of freedom concerning the choice of the first and second flipping position. Therefore, we use two parameters $T^{(2,1)}$ and $T^{(2,2)}$ to characterize the flips of order 2. $T^{(2,1)}$ is the number of positions from the $\mathcal{L}_{\text{flip}}^{(1)}$ list, for which flips of order-2 will be explored. The number of order-2 flips explored for each of these positions is given by $T^{(2,2)}$. The ordered list $\mathcal{L}_{\text{flip}}^{(2)}$ of size $T^{(2)}$ is obtained by concatenating the vectors of size $T^{(2,2)}$ returned by $T^{(2,1)}$ successive calls to this function. For each call, k_1 corresponds to the position of the first bit flipped. The maximum number of order-2 attempts is $T^{(2)} = T^{(2,1)} \cdot T^{(2,2)}$.

Algorithm 3 FlipDetermine

1: **procedure** FLIPDETERMINE($\{L(u_i)\}_{i \in \mathcal{I}}, k_1, T$)
2:     **for** $i = 0, \ldots, N - 1$ **do**
3:         **if** $i \in \mathcal{I}$ and $i > k_1$ **then** $m(i) = M_\alpha(u_i)$
4:         **else** $m(i) = 0$
5:         **end if**
6:     **end for**
7:     $J \leftarrow$ sort_index(m)
8:     Return $J(0 : T - 1)$
9: **end procedure**

B. Oracle-Assisted Decoder and Order of a noise Realization

Following [5], we distinguish between *channel-generated errors* (CGE) and *propagation errors* (PE) in the SC decoding. *Propagation errors* are generated by an erroneous decision, which propagates in the decoding process, while *channel-generated errors* correspond to erroneous decisions which are only generated by the noise realization at the decoder's input. From these definitions, the first error in SC decoding is necessary a CGE.

In [5], an Oracle-assisted decoder (OA-SC) has been introduced to count the number ω of *channel-generated errors* which occur during a SC decoding. OA-SC performs a standard SC decoder with a hard decision function $h^{(OA)}$ modified to ensure that the decision is correct and no error will propagate during the process: $h^{(OA)}(L(u_i)) = u_i$. Hence, ω is defined by:

$$\omega = \underset{i \in \mathcal{I}}{\#}\{h^{(OA)}(L(u_i)) \neq h(L(u_i))\},$$

where the symbol $\#\{A\}$ denotes the number of times the condition A is verified. In this paper, the parameter ω is referred to as *order of a noise realization*.

Note that we use the same notation ω for the flip order in the SCFlip-ω decoder and the *order of a noise realization* as they are directly related. Indeed, the SCFlip-ω is able to decode a noise realization of order $\omega' \leq \omega$, provided that (i) the corresponding order-ω' flip has been selected in the corresponding ordered list $\mathcal{L}_{\text{flip}}^{(\omega')}$ and (ii) the CRC is not verified by one of the previous decoding attempts. As in [5], we use the OA-SC decoder to predict the optimal performance of a SCFlip-ω decoder, regardless of the choice of the metric M and the complexity $(T^{(1)}, T^{(2)}, \ldots T^{(\omega)})$, by declaring a decoding failure if and only if the order of the noise realization is greater than ω. These optimal performance serve as lower bounds on the FER results for practical SCFlip-ω decoders. We will further denote $\text{FER}_{\text{OA}\omega}$ the lower bound of the SCFlip-ω decoder.

Fig. 1 presents the lower-bounds of SCFlip-ω decoders with $\omega = \{1, 2, 3\}$ over BI-AWGN channel for a Polar code with parameters $(N, K + r) = (1024, 512+16)$. We also plot the performance of the SC decoder with $(N, K) = (1024, 512)$. It can be seen that ideal SCFlip-ω decoders exhibit significant SNR gains compared to the SC decoder, from 0.5 dB for the ideal SCFlip-1, to about 1 dB for the ideal SCFlip-2 decoder, at FER$= 10^{-4}$. This ideal performance can be achieved with $T^{(\omega)} = \binom{K + r}{\omega}$ for any order ω, assuming a perfect CRC, i.e. collision probability equal to 0. In practice, due to non perfect CRC, the probability of getting a CRC collision (hence an erroneous decoded message) increases with the number of decoding attempts. Therefore, optimizing the choice of flipping positions allows improving simultaneously the latency and the FER performance.

C. Importance of SCFlip-1

We define $P_M(\omega)$, the probability of not correcting a noise realization of order ω for a SCFlip-ω using a chosen metric M, a number $T^{(\omega)}$ of attempts, and assuming a perfect CRC. As a consequence, FER of SCFlip-ω can be lower bounded by:
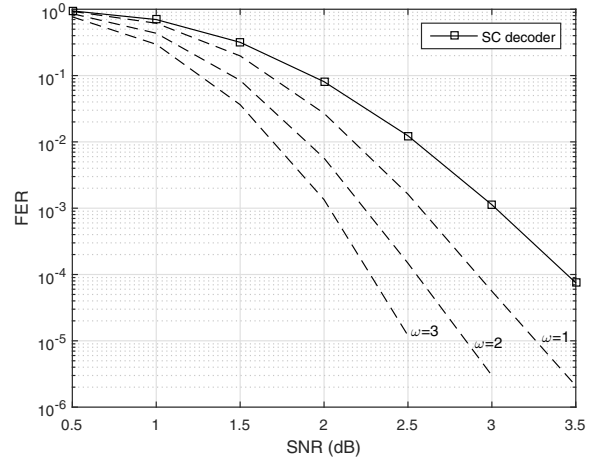

Fig. 1. Lower bound of SCFlip-ω for a polar code $(N, K) = (1024, 512)$

$$\text{FER}_{\text{SCFlip}-\omega} \geq \text{FER}_{\text{OA}\omega} + \sum_{\omega'=1}^{\omega} P_M(\omega') \cdot D(\omega') \quad (2)$$

where $D(\omega')$ denote the probability of the noise realization being of order ω' and $\text{FER}_{\text{OA}\omega}$ the lower bound determined by the OA-SC defined above. This is an inequality because of non perfect CRC. The sum in the right hand side of the above inequality is referred to as the *loss of order* ω. It can be seen that this loss is incremental, so the loss of order ω will propagate at order $\omega + 1$. In particular, to construct an SCFlip-2 decoder that closely approaches its theoretical lower bound $\text{FER}_{\text{OA}2}$, we would like loss of order 2 to be of same order of magnitude as $\text{FER}_{\text{OA}2}$. This implies that the loss of order 1 should also be of same order of magnitude as $\text{FER}_{\text{OA}2}$, and therefore order of magnitude smaller than $\text{FER}_{\text{OA}1}$. This condition is concretely materialised by a SCFlip-1 decoder matching its lower bound predicted by the OA-SC decoder.

D. Complexity of SCFlip-ω

In addition to the FER performance, the performance of SCFlip-ω decoder is also characterized by its computational complexity. This computational complexity depends on the number of decoding attempts performed by the SCFlip-ω decoder in order to decode a given noise realization. Therefore, we denote by $N_c^{(ave)}$ and call the *normalized computational complexity* the average number of attempts. It is given by:

$$N_c^{(\text{ave})} = 1 + \text{FER}_{\text{SC}} \cdot T^{(\text{ave})} \xrightarrow[\text{SNR}\to+\infty]{} 1,$$

where $T^{(\text{ave})}$ is the average number of decoding attempts and depends on the metric and the chosen values $T^{(\omega)}$. It is worth noticing that complexity of SCFlip tends to the one of the SC decoder when SNR tends to infinity.

IV. A NEW METRIC FOR EFFICIENT SCFLIP-ω DECODERS

In [5], the SCFlip decoder uses flipping positions which are ordered according to the absolute value of their LLRs. The criterion can be described as a metric $M(u_i), i \in \mathcal{I}$, defined by:

$$M(u_i) = |\text{L}(u_i)| \quad (3)$$

The flipping positions are those corresponding to the $T^{(1)}$ positions with the lowest $M(u_i)$. However, we point out that this metric is sub-optimal, because it does not take into account the sequential aspect of the SC decoder. Indeed, while a lower LLR absolute value indicates that the corresponding hard decision has a higher probability of being in error, it does not provide any information about the probability of being the first error that occurred during the sequential decoding process. To address this issue, we propose a new metric, which is aimed at identifying the first error that occurred during the sequential decoding process. The probability of $\hat{u}_k$ being the first error is given by:

$$\Pr(\hat{u}_k \neq u_k, \hat{u}_0^{k-1} = u_0^{k-1}) = p_{\text{err}}(\hat{u}_k) \cdot \prod_{i=0}^{k-1}(1 - p_{\text{err}}(\hat{u}_i))$$

where $p_{\text{err}}(\hat{u}_i) = \Pr(\hat{u}_i \neq u_i|Y, \hat{u}_0^{i-1} = u_0^{i-1})$. This probability cannot be computed in practice, as we have no guarantee

that previous bits have been correctly decoded. Instead, we can compute the probability $\Pr(\hat{u}_i \neq u_i|Y, \hat{u}_0^{i-1})$ given by (this follows from the definition of $L(u_i)$):

$$\Pr(\hat{u}_i \neq u_i|Y, \hat{u}_0^{i-1}) = \frac{1}{1 + \exp\left(|\text{L}(u_i)|\right)}$$

Note that if u_i is a frozen bit, it cannot be in error, as the decoder always takes the right decision. Therefore, for frozen bits, the above probability is set to zero.

To compute the probability of being the first error, we consider $\Pr(\hat{u}_i \neq u_i|\hat{u}_0^{i-1})$ as an approximation of $p_{\text{err}}(\hat{u}_i)$, and introduce a parameter α to compensate the approximation and which can be optimized by simulation. Thus, the proposed metric is given by:

Definition 1. *Given a bit* $u_k, k \in \mathcal{I}$, *the metric associated to* u_k *is defined by:*

$$M_\alpha(u_k) = \frac{1}{1 + \exp\left(\alpha|L(u_k)|\right)} \cdot \prod_{\substack{i=0 \\ i\in\mathcal{I}}}^{k-1}\left(\frac{1}{1 + \exp\left(-\alpha|L(u_i)|\right)}\right) \quad (4)$$

where α *is a parameter to be optimized by simulation.*

We further define the equivalent logarithmic domain metric $M'_\alpha(u_k) = -\frac{1}{\alpha} \cdot \log(M_\alpha(u_k))$. It follows that:

$$M'_\alpha(u_k) = |L(u_k)| + \frac{1}{\alpha}\sum_{\substack{i=0 \\ i\in\mathcal{I}}}^{k-1}(1 + \exp(-\alpha \cdot |L(u_i)\,)) \quad (5)$$
$$= |L(u_k)| + S_\alpha(u_0^k)$$

The $S_\alpha(u_0^k)$ sum can be seen as a penalty added to $|L(u_k)|$, which take into consideration the sequential aspect of the SC decoding. Indeed, this term increases with increasing number and decreasing reliability of previously decoded bits, so that the last decoded bits are penalized compared to the metric (3). To understand the impact of the parameter α, we consider the following limit cases. For $\alpha = 0$, the above metric becomes $M_0(u_k) = \frac{1}{2^{k_\mathcal{I}}}$, where $k_\mathcal{I}$ is the number of positions in $\mathcal{I}$ less than or equal to k. Hence, the M_0 induced ordering corresponds to the usual decoding order. For $\alpha \longrightarrow +\infty$ it can be easily seen that $\lim_{\alpha\to+\infty} S_\alpha(u_0^k) = 0$, so that M_∞ is equivalent to (3). In general, the use of the M_α metric can be seen an intermediate trade-off between the decoding order and the one given by the LLRs reliability (3). To find the best possible trade-off, we optimize the value of α by Monte-Carlo simulation. Note also that the optimized value should depend on the code used and the SNR. We provide an intuitive explanation of the behavior of the optimized alpha value as function of the SNR. Consider equation (5) for some fixed value of alpha. When the SNR goes to infinity, the term $S_\alpha(u_0^k)$ tends to 0 and becomes negligible compared to $|L(u_k)|$, and therefore the sequential characteristic of the decoder is no longer accounted for by the considered metric. Consequently, it is expected that the optimal value of alpha will increase with the SNR, so that to rebalance the contribution of the $S_\alpha(u_0^k)$ term to the value of the considered metric. To confirm this intuition, Table I shows the optimized alpha values for a code $(N, K + r) = (1024, 512 + 16)$ and several SNR values.

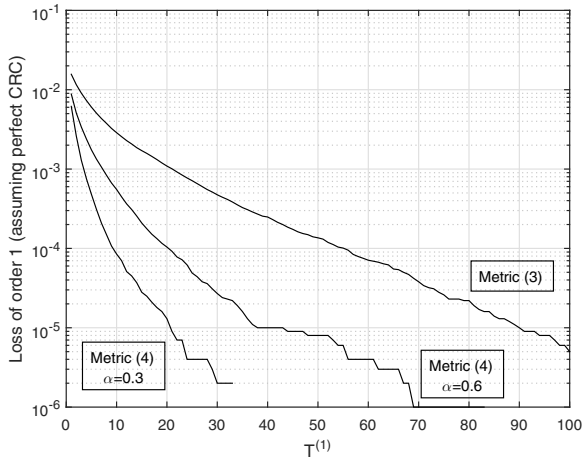| SNR (dB)	1.5	2.5	3
α_{opt}	0.4	0.3	0.25

Fig. 2. Loss of order 1 in function of $T^{(1)}$ for SNR=2.5dB

As expected, it can be observed that the optimal alpha value increases with the SNR.

We further consider a $(1024, 528)$ Polar code and consider only random noise realizations of order 1 at SNR= 2.5dB. Fig. 2 plots the loss of order 1 ($P_M(1) \cdot D(1)$) assuming perfect CRC. It is calculated as the probability of the only *channel-generated* error not being in the $\mathcal{L}_{\text{flip}}^{(1)}$ list generated by the *FlipDetermine* procedure, as a function of the list size $T^{(1)}$, by using either the metric (3), or our proposed metric (4) with $\alpha = 0.3$ (optimized value) and $\alpha = 0.6$. According to Section III-C, we can determine the value of $T^{(1)}$ such that the loss of order 1 has the same order of magnitude as the theoretical lower bound of the SCFlip-2, FER$_{OA2}$. At SNR=2.5db, the SCFlip-2 lower bound is about 10^{-4}, therefore, we need $D(1) \cdot P_M(1) < 10^{-4}$. This condition can easily be satisfied with the proposed metric (4), while we need $T^{(1)}$ much higher with the metric (3), which means in practice higher computational complexity. For next simulations, we choose $T^{(1)} = 20$.

Let us now consider $\mathbf{Y}$ be a noise realization of order $\omega = 2$ for a given code $\mathcal{C}(N, K, \mathcal{I})$ with the first CGE in position u_{k1}. We define the set $\mathcal{I}' = \{k \in \mathcal{I}, k > k1\}$ of cardinality $K' < K$. Consider now the code $\mathcal{C}_2(N, K', \mathcal{I}')$. The order of the noise realization $\mathbf{Y}$ is only $\omega = 1$ for the code $\mathcal{C}_2$ and therefore finding the second CGE is equivalent to decoding a noise realization of order 1 for the code $\mathcal{C}_2$. As a consequence, the metric M_α can be used also for SCFlip-2 by considering the set $\mathcal{I}'$ defined above instead of the set $\mathcal{I}$. However, the optimum value for α may be different. Numerical optimisation for α_2 for a code $(1024, 512 + 16)$ shows that the optimum value is $\alpha_2 = 0.5 > \alpha_1 = 0.3$ at SNR=2.5dB.

V. SIMULATION RESULTS

Throughout this section we consider transmission over a BI-AWGN channel, using a CRC-Polar code concatenation
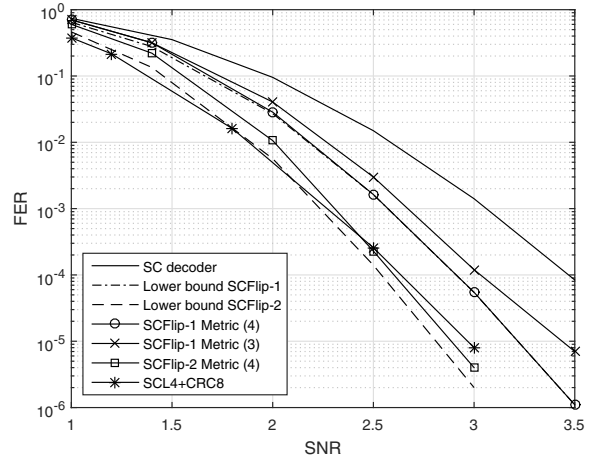

Fig. 3. Performance of SCFlip-1 and SCFlip-2 decoders

with parameters $N = 1024$, $K = 512$, and $r = 16$. The $K + r$ positions for information and CRC bits, given by the set $\mathcal{I}$, are optimized for the SC decoder by Gaussian approximation as in [8]. Also, this set is updated for each different value of the SNR. Speaking of CRC, we systematically use a 16-CRC with generator polynomial $g(x) = x^{16} + x^{15} + x^2 + 1$.

The FER performance of SCFlip-1 and SCFlip-2 decoders is shown in Fig. 3. For SCFlip-1, the maximum number of flips is set to $T^{(1)} = 20$. We compare the SCFlip-1 decoder using the metric (3), with the one using our proposed metric (4) with α=0.3, and the theoretical lower bound corresponding to an ideal SCFlip-1 decoder. While the performance gain compared to [5] is not impressive, it can be seen that our proposed metric closely approaches the theoretical lower bound. As the metric (4) exhibits a negligible loss of order 1, it can further be used to correct second order noise realizations (note that SCFlip-1 and SCFlip-2 using the metric (3) have nearly the same FER performance, since in this case the loss of order 1 is dominant). We further plot the FER performance of the proposed SCFlip-2 with $T^{(1)} = 20, T^{(2,1)} = T^{(2,2)} = 5, \alpha_1 = 0.3$, and $\alpha_2 = 0.5$, and compare with the CRC-aided SCL with $L = 4$ and 16-bit CRC. The theoretical lower bound of SCFlip-2 decoder is also shown. It can be seen that the proposed SCFlip-2 closely approaches the theoretical lower bound, and exhibits nearly the same performance as the CRC-aided SCL decoder with $L = 4$ at medium/high SNR.

In Fig. 4, we plot the average normalized complexity for SCFlip-1 with metric (3) and $T^{(1)}$=40, for SCFlip-1 with the proposed metric (4) with $T^{(1)} = 20$, and SCFlip-2 with same parameters as in previous paragraph. Moreover, we also plot the normalized complexity of SC and SCL decoders with $L = \{2, 4\}$. We note that the SCFlip-1 decoder exhibits similar FER performance when using our proposed metric (4) with $T^{(1)} = 20$, or the metric (3) with $T^{(1)} = 40$. We observe that the complexity of SCFlip is very high at low SNR, but converges quickly to the one of the SC decoder. For a SNR of 2.2dB, we have already a complexity lower than the one of the SCL with $L = 2$. Moreover, one can see that the proposed metric allows reducing the average normalized complexity of the SCFlip-1 decoder by a factor of 2, as compared to the metric (3). Also,
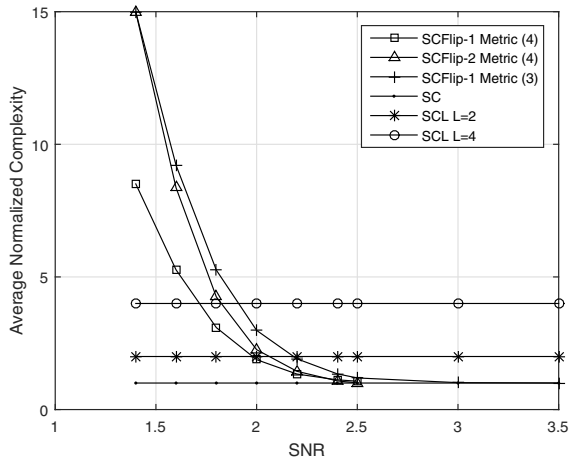
Fig. 4. Average normalized computational complexity for SCFlip-1

computational complexity of our proposed SCFlip-2 is even slightly better than SCFlip-1 with metric (3) and $T^{(1)} = 40$, while it improves performance by 0.4dB at FER of 10^{-4}.

VI. CONCLUSION

In this paper, we first proposed an improvement of the SCFlip decoder of order-1, by introducing a new metric to determine the flipping positions, which takes into account the sequential aspect of the SC decoder. The proposed metric increases the ability of the SCFlip decoder to find the first error that occurred during the initial SC attempt, thus improving both decoding performance and computational complexity. Moreover, we have shown that the proposed metric allows closely approaching the theoretical lower bound corresponding to an ideal SCFlip decoder, and explained that this a necessary condition for building an effective SCFlip decoder of order 2.

We have further investigated an SCFlip decoder of order-2, which uses an analogous metric to determine the order-2 flip positions. We have shown that the SCFlip-2 decoder yields significant gains in terms of decoding performance, closely approaching the performance of the CRC-aided SC-List decoder with list size $L = 4$, while having an average decoding complexity similar to that of the standard SC decoding at medium to high SNR.

ACKNOWLEDGMENT

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, 2009.

[2] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[3] K. Niu and K. Chen, "Stack decoding of polar codes," *Electronics letters*, vol. 48, no. 12, pp. 695–697, 2012.

[4] M. Bastani Parizi, "Polar codes: Finite length implementation, error correlations and multilevel modulation," Master's thesis, Swiss Federal Institute of Technology, 2012.

[5] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, "A low-complexity improved successive cancellation decoder for polar codes," in *IEEE 48th Asilomar Conference on Signals, Systems and Computers*, 2014, pp. 2116–2120.

[6] M. P. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Transactions on Information Theory*, vol. 41, no. 5, pp. 1379–1396, 1995.

[7] D. Wu, Y. Li, X. Guo, and Y. Sun, "Ordered statistic decoding for short polar codes," *IEEE Communications Letters*, vol. 20, no. 6, pp. 1064–1067, 2016.

[8] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Transactions on Communications*, vol. 60, no. 11, pp. 3221–3227, 2012.