Dynamic-SCFlip Decoding of Polar Codes

Ludovic Chandesris, Valentin Savin, David Declercq

Dynamic-SCFlip Decoding of 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 proposes a generalization of the recently introduced Successive Cancellation Flip (SCFlip) decoding of polar codes, characterized by a number of extra decoding attempts, where one or several positions are flipped from the standard Successive Cancellation (SC) decoding. To make such an approach effective, we first introduce the concept of higher-order bit-flips, and propose a new metric to determine the bit-flips that are more likely to correct the trajectory of the SC decoding. We then propose a generalized SCFlip decoding algorithm, referred to as Dynamic-SCFlip (D-SCFlip), which dynamically builds a list of candidate bit-flips, while guaranteeing that extra decoding attempts are performed by decreasing probability of success. Simulation results show that D-SCFlip is an effective alternative to SC-List decoding of polar codes, by providing very good error correcting performance, with an average computation complexity close to the one of the SC decoder.

Index Terms

Polar Codes, successive cancellation decoding, order statistic decoding, SCFlip decoding

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". However, for short to moderate code-lengths the incomplete polarization of the virtual channels may drastically penalize the error correction performance of the SC decoder. The main approaches proposed in the literature to address this issue rely on either modified kernels for the recursive encoding procedure, aimed at increasing the rate of polarization [2], [3], or enhanced versions of the SC decoder [4], [5], [6], aimed at increasing its ability to deal with incompletely polarized channels.

The SC-List (SCL) decoder proposed in [4] significantly improves the error correction performance for short to moderate block lengths, and is also known to approach the Maximum-Likelihood (ML) decoding performance at high Signal to Noise Ratio (SNR). Moreover, to advantageously exploit the potential of SCL decoding, especially when the size of the decoded list is large, the concatenation of an outer Cyclic Redundancy Check (CRC) code has also been proposed in [4], to help identifying the correct message within the decoded list. Concatenated CRC-Polar codes under under SCL decoding is the best polar-coding system proposed so far, and has been shown to compete with other families of modern error correcting codes, such as Low Density Parity Check (LDPC) and Turbo 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 SC-Stack decoding (SCS) [6], but at a cost of an increasing storage complexity.

A different approach has been proposed with SC-Flip (SCFlip) decoder, introduced in [7] for the BEC channel, and later generalized to concatenated CRC-Polar codes over the Binary-Input Additive White Gaussian Noise (BI-AWGN) channel in [5]. The concept of SCFLip decoding is

related to the ordered statistics decoding proposed in [8], whose applicability to decoding short Polar and concatenated CRC-Polar codes has been recently investigated in [9]. The principle is to allow a given number of new decoding attempts, in case that a failure of the initial SC decoding is detected by the CRC. Each new decoding attempt consists in flipping one single hard decision bit – starting with the least reliable one, according to the absolute value of the corresponding Log-Likelihood Ratio (LLR) – of the initial SC decoding attempt, then decoding the subsequent positions by using the standard SC decoding. The above procedure is repeated until the CRC is verified or a predetermined maximum number of decoding attempts is reached. The SCFlip decoder provides a tunable trade-off between decoding performance and decoding complexity, since each new decoding attempt is only performed if the previous one failed. In particular, the average computational complexity of the SCFlip decoder tends to the one of the SC decoder at medium to high SNR, while competing with the CRC-aided SCL with list size $L = 2$, in terms of error correction performance [5].

In this work, we propose two improvements to the SCFlip decoding, based on refining and expanding some of the concepts we previously introduced in [10]. First, a new metric is proposed, aimed at determining the flipping positions that are more likely to *correct the trajectory* of the SCFlip decoding, *i.e.*, those positions that, once flipped, are more likely to lead to a successful decoding attempt. The proposed metric takes into account the sequential aspect of the SC decoder, and is shown to yield an improved error correction performance and a reduced computational complexity, as compared to the conventional LLR-based metric from [5]. Secondly, we introduce a generalization of the SCFlip decoder by considering not only one single bit-flip per new decoding attempt, but a number of $\omega \geq 1$ nested bit-flips. These two improvements are materialized in a *Dynamic SCFlip decoder* (D-SCFlip), in which the flipping positions are chosen dynamically by taking into consideration all the previous attempts, so that the next attempt is guaranteed to be the one with the best probability of success according to the optimized metric. The D-SCFlip decoder is shown to compete with the CRC-aided SCL

decoder with list size up to $L = 16$ in terms of decoding performance, while having an average computational complexity similar to that of the standard SC decoding at medium to high SNR. Moreover, we derive lower bounds on the Word Error Rate (WER) performance of any SCFlip decoder with the number of bit-flips per decoding attempt bounded by a maximum value ω, and show that the D-SCFlip tightly approaches the WER lower bounds for $\omega \in \{1, 2\}$.

The remainder of the paper is organized as follows. Section II provides a short background on polar codes and main SC-based decoding algorithms. Section III introduces the concept of bit-flips of order $\omega \geq 1$, and defines the general structure of a SCFlip decoder relying on higher-order bit-flips. Theoretical lower bounds on the WER performance of such a decoder are derived in Section IV. Section V presents the proposed bit-flip metric, and investigates its efficiency in determining bit-flips leading to successful decoding attempts. The proposed D-SCFlip algorithm is finally described in Section VI, where Monte-Carlo simulation results are also provided for performance evaluation and comparison with other state of the art decoding techniques.

II. Preliminaries

A. Polar Codes and Successive Cancellation Decoding

A Polar Code [1] is characterized by a three-tuple $(N, K, \mathcal{I})$, where $N = 2^n$ is the code-length, K is the number of information bits, and $\mathcal{I} \subset \{1, ..., N\}$ is a set indicating the positions of the K information bits. 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 by $\mathbf{U} = u_1^N$ the *data vector*, of length N, containing K information bits at positions $i \in \mathcal{I}$, and $N - K$ frozen bits at positions $i \notin \mathcal{I}$, which are assumed to be 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 [1]. We further denote by $\mathbf{Y}$ the data received from the channel

and used at the decoder input. $\hat{\mathbf{U}} = \hat{u}_1^N$ denotes the decoder's output, with $\hat{u}_i$ being the hard decision estimate of the bit u_i.

In SC decoding, each hard decision estimate $\hat{u}_i$ depends on both $\mathbf{Y}$ and the previous estimates $\hat{u}_1^{i-1}$, and is computed according to the sign of the LLR:

$$\mathrm{L}_i = \log\left(\frac{\Pr(u_i = 0|\mathbf{Y}, \hat{u}_1^{i-1})}{\Pr(u_i = 1|\mathbf{Y}, \hat{u}_1^{i-1})}\right) \tag{1}$$

by using the hard decision function h:

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

where by convention $\mathrm{sign}(0) = \pm 1$ with equal probability.

B. List decoding of Polar Codes

Due to its sequential nature, early errors occurring during the SC decoding process cannot be reversed. To overcome this problem, SCL decoding [4] duplicates the SC decoding at each position $i \in \mathcal{I}$ in two parallel decoding threads, continuing in either possible direction. In order to avoid an exponentially growing complexity, the number of parallel decoding paths is limited to a chosen, usually small, parameter L. The L *surviving* decoding paths are determined according to a path metric, as discussed below. SCL decoder has a computational complexity growing as $\mathcal{O}(L \cdot N \log(N))$ and a space (memory) complexity of $\mathcal{O}(L \cdot N)$ [4]. It has also been shown to closely approach the ML decoding performance if the size of the list L is large enough. Moreover, in [4] it has been observed that the SCL performance can be significantly improved, by concatenating an outer CRC code, to facilitate the identification of the correct decoding path among the list of L candidates.

The SCL decoding computes a likelihood metric for each explored path, which can be alternatively expressed in the log-likelihood [4], or the log-likelihood ratio (LLR) [11] domain. In the LLR domain, the path metric is defined as follows:

Definition 1: For a path l of length $i \leq N$, the path metric is defined by:

$$PM[l]_i = \sum_{j=1}^{i} \log(1 + \exp(-(1 - 2 \cdot \hat{u}[l]_j) \cdot \mathrm{L}[l]_j)) \tag{3}$$

where

$$\mathrm{L}[l]_j = \log\left(\frac{\Pr(u_j = 0|\mathbf{Y}, \hat{u}[l]_1^j)}{\Pr(u_j = 1|\mathbf{Y}, \hat{u}[l]_1^i)}\right) \tag{4}$$

is the log-likelihood ratio of bit u_j given the channel output $\mathbf{Y}$ and the past trajectory of the path $\hat{u}[l]_1^j$.

Note that the sum in Eq. (3) is taken over all $j = 1, \ldots, i$, including both frozen and non-frozen positions. However, for a frozen position $j \notin \mathcal{I}$ the decoding path is not duplicated, thus $\hat{u}[l]_j = u_j$, irrespective of the $\mathrm{L}[l]_j$ value.

Several practical simplifications, aimed at reducing the computational complexity and/or the latency of the SCL decoding, as well as hardware implementations have been also proposed in the literature [12], [13], [14]. An alternative to SCL decoding is the SCS decoding proposed in [6], aimed at reducing the computational complexity, at a cost of a small loss in the error correction performance. Instead of exploring parallel decoding paths of the same length, the SCS uses an ordered stack of depth D, in which paths may have different lengths, and only the path with the largest path metric is extended. SCS decoding stops when the top path is of length N. The advantage of this decoder is that it is able to limit the number of operations compared to SCL decoder, especially when SC decoder is already able to decode correctly. The worst-case complexity of the SCS decoder is $\mathcal{O}(D \cdot N \log(N))$, but simulations show that the actual computational complexity is much lower, especially in moderate to high SNR regime.

III. Generalized SCFlip decoders

Let $\mathcal{C}(N, K + r, \mathcal{I})$ denote the serial concatenation of an outer $(K + r, K)$ CRC code and an inner $(N, K + r, \mathcal{I})$ polar code. Note that the subset $\mathcal{I} \subset \{1, ..., N\}$ contains $K + r$ positions, for the K information bits and the CRC of r bits.

The SCFlip decoder [5] consists of a standard SC decoding, possibly followed by a maximum number of T new decoding attempts, until no errors are detected by the CRC check. Each new decoding consists of flipping one decision of the initial SC attempt, and decoding the subsequent positions by using the standard SC decoding. The position to be flipped is determined according to a given metric based on the LLRs obtained after the SC decoding.

In this paper, we propose a generalization of the SCFlip decoder, by allowing a more than a single bit-flip for each decoding attempt. Therefore, we defined the notion of *bit-flip of order* ω as follows.

Definition 2: A *(bit-)flip of order* ω ($0 \leq \omega \leq K+r$) is a set of ω indices $\mathcal{E} = \{i_1, \dots i_\omega\} \subset \mathcal{I}$, such that $i_1 < \cdots < i_\omega$. The associated decoding attempt, denoted by SC($\mathcal{E}$), corresponds to the SC decoding with the hard decision function h, defined in Eq. (2), replaced by $h_{\mathcal{E}}$, defined below:

$$\forall i \in \mathcal{I}, \quad \hat{u}[\mathcal{E}]_i = h_{\mathcal{E}}(\mathrm{L}_i) \stackrel{\text{def}}{=} \begin{cases} h(\mathrm{L}_i) & \text{if } i \notin \mathcal{E} \\ 1 - h(\mathrm{L}_i) & \text{if } i \in \mathcal{E} \end{cases} \tag{5}$$

This decoding attempt outputs a vector $\hat{u}[\mathcal{E}]_1^N$, the estimation of the codeword u_1^N. To simplify the notation, when no confusion is possible, $\hat{u}[\mathcal{E}]_1^N$ will be simply denoted by $\hat{u}_1^N$.

Hence, in the *Generalized SCFlip* (described in Algorithm 1), the decoding attempt associated to a bit-flip of order ω corresponds to a standard SC decision for each position, except for the the ω positions in $\mathcal{E}$, for which the decision is flipped. Note that if $\mathcal{E} = \emptyset$ (bit-flip of order $\omega = 0$), the decoding attempt is exactly the same as the standard SC decoding.

The exhaustive exploration of all the bit-flips of order $\omega \in \{0, ..., K+r\}$ would require a total number of $\sum_{\omega=0}^{K+r} \binom{K+r}{\omega} = 2^{K+r}$ decoding attempts, which is obviously too complex for a practical decoding solution. Therefore, we further equip the Generalized SCFlip decoder with a list $\mathcal{L}_{\text{flip}} = \{\mathcal{E}_1, \mathcal{E}_2, \dots \mathcal{E}_T\}$ of T bit-flips of order $\omega_t, t \in \{1, \cdots T\}$.

The Generalized SCFlip algorithm proceeds to at most $T+1$ decoding attempts, starting with

Algorithm 1 Generalized SCFlip decoder

procedure GENERALIZED SCFLIP($\mathbf{Y}, \mathcal{I}, T, \mathcal{L}_{\text{flip}} = \{\mathcal{E}_1, \ldots, \mathcal{E}_T\}$)
  $\hat{u}_1^N \leftarrow$SC($\emptyset$)
  if CRC($\hat{u}_1^N$) = success **then** return $\hat{u}_1^N$; **end if**
  for $t = 1, \ldots, T$ **do**
    $\hat{u}_1^N \leftarrow$SC($\mathcal{E}_t$);
    if CRC($\hat{u}_1^N$) = success **then** return $\hat{u}_1^N$; **end if**
  end for
  return $\hat{u}_1^N$;
end procedure

the standard SC decoding and, followed by the decoding attempts SC($\mathcal{E}_t$), with $\mathcal{E}_t \in \mathcal{L}_{\text{flip}}$. The decoding process stops if:

- one of the decoding attempt verifies the CRC
- all T bit-flips from the list have been tested

The Generalized SCFlip decoder may recover the correct codeword, only if $\mathcal{L}_{\text{flip}}$ contains the unique bit-flip of order $\omega \geq 1$ that corrects the SC decoding trajectory (assuming that the initial SC decoding attempt failed). However, having the correct bit-flip in $\mathcal{L}_{\text{flip}}$ does not guarantee successful decoding, since an earlier, erroneous decoding attempt might verify the CRC (undetected error), so that the decoding process stops and the following bit-flips in $\mathcal{L}_{\text{flip}}$ are not tested. The probability this happens depends on both the probability of undetected error of the CRC and the position of the correct bit-flip within $\mathcal{L}_{\text{flip}}$.

In view of the previous discussion, the effectiveness of the Generalized SCFlip decoder depends directly on the way the list of tested bit-flips $\mathcal{L}_{\text{flip}}$ is determined. It also appears that rather than a predetermined list, $\mathcal{L}_{\text{flip}}$ should actually depend on the current noise realization, so as to increase the probability of including the correct bit-flip (*i.e.*, correcting the SC decoding trajectory) in front positions. Moreover, the information gathered during the decoding process

(e.g., LLR values computed during the initial SC decoding or the following decoding attempts) can also be used to determine those bit-flips that are most likely to correct a given noise realization, and thus to dynamically update the list $\mathcal{L}_{\text{flip}}$. To do so, the candidate bit-flips have to be evaluated by a metric that estimates their likelihood to correct a given noise realization, which will be discussed in Section V.

Before discussing the optimization of such a metric and the method to dynamically generate the list of bit-flips, in the next section we derive lower bounds on the WER performance of the Generalized SCFlip decoder using bit-flips of order $\leq \omega$. These lower bounds will also serve as a reference for assessing the effectiveness of the proposed D-SCFlip decoder in Section VI, and implicitly of the bit-flip metric proposed in Section V.

IV. Word Error Rate Lower Bound for Generalized SCFlip Decoders

A. Order of a noise realization

In the following, we shall use the expression *noise realization* to refer to the channel noise that corrupted the actually observed signal $\mathbf{Y}$. We say that a *noise realization is of order* ω, if there exits a bit-flip $\mathcal{E}$ of order ω, such that the observed signal $\mathbf{Y}$ is corrected by the SC($\mathcal{E}$) decoding attempt (see Definition 2). The order of a noise realization can be efficiently computed by using the Oracle-Assisted SC (OA-SC) decoder proposed in [5]. OA-SC performs the same operations as the standard SC decoder, but instead of propagating the hard decision estimates of the previous decoded bits, and thus risking to propagate an erroneous decision, it is helped by an oracle to propagate the correct decisions. Hence, the oracle-assisted LLR of the bit u_i, denoted by $\mathrm{L}_i^{\mathrm{OA}}$, can be expressed as:

$$\mathrm{L}_i^{\mathrm{OA}} = \log\left(\frac{\Pr(u_i = 0|\mathbf{Y}, u_1^i)}{\Pr(u_i = 1|\mathbf{Y}, u_1^i)}\right) \tag{6}$$

and the hard decision estimate of u_i is given by $\hat{u}_i^{\mathrm{OA}} = h(\mathrm{L}_i^{\mathrm{OA}})$. Let $\mathcal{E}_{\mathbf{Y}} = \{\ i \in \mathcal{I} \mid \hat{u}_i^{\mathrm{OA}} \neq u_i\}$ and $\omega_{\mathbf{Y}} = |\mathcal{E}_{\mathbf{Y}}|$ be the order (*i.e.* number of elements) of $\mathcal{E}_{\mathbf{Y}}$. Then the order of the noise

realization is equal to $\omega_{\mathbf{Y}}$, and the observed signal $\mathbf{Y}$ is successfully corrected by the SC($\mathcal{E}_{\mathbf{Y}}$) decoding.

B. WER Lower Bound

Let SCFlip-ω denote a Generalized SCFlip decoder (Algorithm 1) whose maximum bit-flip order is equal to ω. Hence, using the notation from Section III, $\omega = \max_{\mathcal{E} \in \mathcal{L}_{\text{flip}}} |\mathcal{E}|$. Such a decoder successfully corrects a noise realization of order $\omega_{\mathbf{Y}} \leq \omega$ if and only if (i) the corresponding bit-flip $\mathcal{E}_{\mathbf{Y}} \in \mathcal{L}_{\text{flip}}$ and (ii) no previous decoding attempt SC($\mathcal{E}$) satisfies the CRC check before SC($\mathcal{E}_{\mathbf{Y}}$). We further denote by iSCFlip-ω the *ideal* SCFlip-ω decoder that successfully corrects any noise realization of order less than or equal to ω. The ideal iSCFlip-ω decoder can be seen as an SCFlip-ω decoder such that (i) $\mathcal{L}_{\text{flip}}$ contains all the bit-flips of order less than or equal to ω, hence the list size is given by $T = \sum_{\omega'=1}^{\omega} \binom{K+r}{\omega'}$, and (ii) the CRC error detection is replaced by an ideal detector, which is satisfied only for the correct word.

The WER of any SCFlip-ω is lower-bounded by the WER of the iSCFlip-ω decoder. The latter can be efficiently determined by running the OA-SC decoder to compute the order $\omega_{\mathbf{Y}}$ of the actual noise realization, then declaring a decoding failure if and only if $\omega_{\mathbf{Y}} > \omega$. It is worth noticing that this lower bound, further referred to as the *ideal WER of order* ω (iWER-ω), is not necessarily achievable by the SCFlip-ω decoder and the iWER-ω lower bound can even be better than the ML performance in some cases (an iSCFlip-ω decoder with $\omega = K + r$ would be able to correct any noise realization). However, for small ω values, the ideal WER can be closely approached by practical SCFlip-ω decoders, provided that the CRC is reliable enough, as it will be shown in Sections V-VI.

Figure 1 presents the lower-bounds of SCFlip-ω decoders with $\omega = \{0, 1, 2, 3, 4\}$ for a CRC-concatenated polar code with parameters $(N, K + r) = (1024, 512 + 16)$. For $\omega = 0$, iWER-0 corresponds to the WER of the SC decoder for a polar code of length N, with $K+r$ information bits. Moreover, we also plot the performance of the SC decoder with $(N, K) = (1024, 512)$,

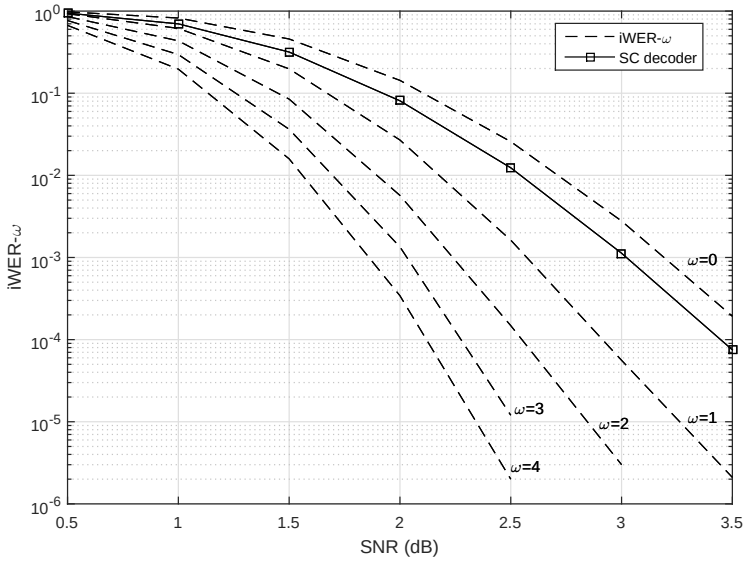


Fig. 1. Performance of ideal SCFlip-ω decoder for a code $(N, K + r) = (1024, 512 + 16)$

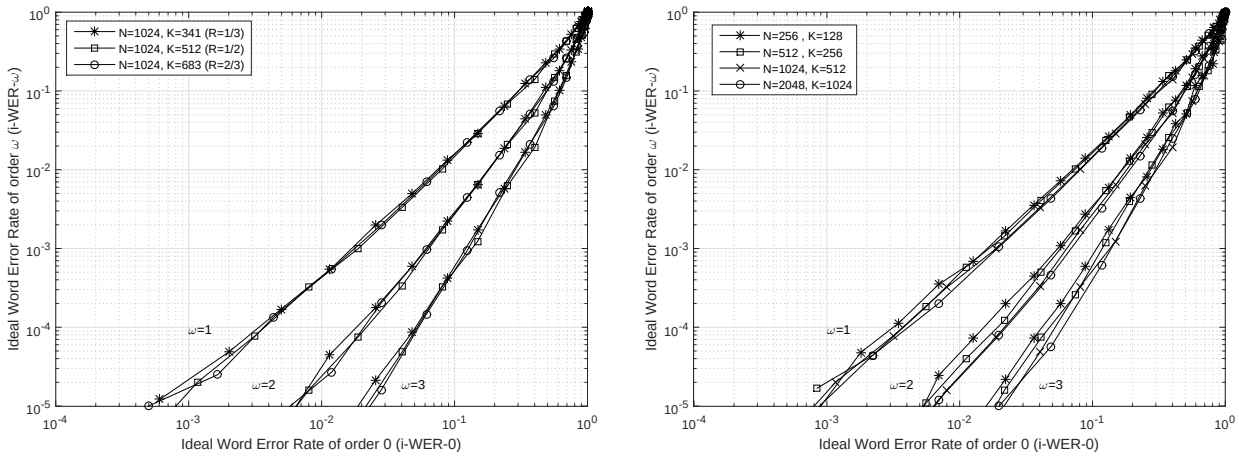
which is better than the iSCFlip-0 performance, due to the higher number of frozen bits. It can be seen that iSCFlip-ω decoders exhibit significant SNR gains compared to the SC decoder, from 0.5 dB for the iSCFlip-1, to about 1 dB for the iSCFlip-2 decoder, at WER $= 10^{-4}$.

C. Impact of the code-length and coding-rate on the ideal WER

This section investigates the ideal decoding performance of iSCFlip-ω decoders, for various code-lengths and coding rates, and small values of ω. More precisely, we investigate the relation between iWERω for $\omega = \{1, 2, 3\}$ and iWER-0, as function of the coding rate $R = \frac{K}{N}$ and code-length N:

$$\texttt{iWER}\omega = f^{(\omega)}_{N,R}(\texttt{iWER-}0) \tag{7}$$

The study is divided into two parts. (a) First, for a given code-length N, we observe this function for different coding rates R. Figure 2(a) plots iWERω, for $\omega \in \{1, 2, 3\}$, as a function of iWER-0 (assuming BI-AWGN channel), for a code-length $N = 1024$ and coding rates $R \in \{1/3, 1/2, 2/3\}$. It can be observed that for a given value of ω, the iWERω depends only on iWER-0 and is practically independent of the coding rate R. (b) Second, as shown in Fig. 2(b), a similar observation can be made if one considers a fixed coding rate $R = 1/2$, and



(a) $N = 1024$, varying $R \in \{1/3, 1/2, 2/3\}$ (b) $R = 1/2$ and varying $N \in \{512, 1024, 2048\}$

Fig. 2. iWER-ω as function of iWER-0 for varying coding rate R or varying code length N

variable code-length N. Therefore, we conclude that iWERω essentially depends on iWER-0, and thus Eq. (7) can be approximated to:

$$\texttt{iWER}\omega \simeq f^{(\omega)}(\texttt{iWER-0}) \tag{8}$$

Since iWER$\omega = 1 - \Pr(\omega_{\mathbf{Y}} \leq \omega)$, this analysis translates into the following interesting property: consider two polar codes $C_1(N_1, K_1)$ and $C_2(N_2, K_2)$, with the same WER performance under SC decoding, at SNR_1 and SNR_2, respectively. Then the noise realization orders $\omega_{\mathbf{Y}_1}$ and $\omega_{\mathbf{Y}_2}$ are expected to follow nearly the same probability distribution.

From a practical point of view, this analysis can also be used to determine different sets of code and decoder parameters that would be able to achieve a target WER performance (assuming a given SNR). Indeed, iWER-0 can be easily estimated, *e.g.*, by using the density evolution technique [15], [16]. Hence, in order to achieve a target WER, for example of 10^{-4}, the code parameters $(N, K + r)$ must be chosen such that iWER-$0 \approx 3 \cdot 10^{-3}$ for an SCFlip-1 decoder, or such that iWER-$0 \approx 3 \cdot 10^{-2}$ for an SCFlip-2 decoder. Of course, the SCFlip-1 and SCFlip-2 decoders under use should be able to closely approach the corresponding lower bounds, iWER-1 and iWER-2. In the following section, we will show that these lower bounds can be indeed tightly approached by practical decoders.

V. Optimized Metric for Generalized SCFlip Decoders

In order to build practical SCFlip-ω decoders that closely approach the ideal performance of iSCFlip-ω, we first introduce an optimized bit-flip metric, adapted to bit-flips of any order $\omega \geq 1$, then we propose an efficient strategy to build the bit-flips list $\mathcal{L}_{\text{flip}}$. In this section we describe the proposed metric, while the construction of $\mathcal{L}_{\text{flip}}$ will be discussed in the next section.

A. Proposed Metric for Generalized SCFlip Decoder

The SCFlip decoder from [5] considers only bit-flips of order 1, which are chosen according to the absolute value of the corresponding LLR. Thus, in case the initial SC decoding fails, the selected bit-flips of order 1 correspond to the T positions $i \in \mathcal{I}$ with the lowest $|L_i|$ values. However, using the absolute value of the LLR as likelihood metric for a bit-flip is sub-optimal, since it does not take into account the sequential aspect of the SC decoder. Indeed, while a lower absolute value of the LLR indicates that the corresponding hard decision bit has a higher error probability, it does not provide any information about the probability of being the first error that occurred during the sequential decoding process. In other words, such a metric does not distinguish the very first error from the subsequent ones.

We propose a new metric, aimed at evaluating the likelihood of a bit-flip $\mathcal{E}_\omega = \{i_1, \ldots, i_\omega\} \subset \mathcal{I}$, of order ω, to correct the trajectory of the SC decoding. By *correcting the trajectory of the SC decoding*, we mean that SC($\mathcal{E}_\omega$) successfully decodes all the bits u_i with $i \leq i_\omega$ (recall that indices $i_1, \ldots, i_\omega$ are assumed to be in increasing order). Note that this does not mean that the SC($\mathcal{E}_\omega$) decoding is successful, since there is no guarantee that it will successfully decode the subsequent bits, *i.e.*, bits u_i with $i > i_\omega$. For instance, for $\omega = 1, \mathcal{E}_1 = \{i_1\}$ corrects the trajectory of the SC decoding if and only if i_1 is the first erroneous position of the SC decoding attempt, but this does not guarantee that the SC($\mathcal{E}_1$) is successful.

For any $1 \leq \omega' \leq \omega$, let $\mathcal{E}_{\omega'} = \{i_1, \ldots, i_{\omega'}\}$ be the bit-flip of order ω' determined by the first ω' indices in $\mathcal{E}_\omega$. Let $\mathrm{L}[\mathcal{E}_{\omega'}]_i$, $\hat{u}[\mathcal{E}_{\omega'}]_i$ denote respectively the LLR and the hard decision estimate

computed by SC($\mathcal{E}_{\omega'}$), corresponding to bit u_i. According to Definition 2, SC($\mathcal{E}_{\omega'}$) and SC($\mathcal{E}_{\omega'-1}$) are identical for positions $i < i_{\omega'}$, while for $i = i_{\omega'}$, SC($\mathcal{E}_{\omega'}$) flips the hard decision estimate computed by SC($\mathcal{E}_{\omega'-1}$). Hence, for any $\omega' \leq \omega$, one has:

$$\mathrm{L}[\mathcal{E}_{\omega'}]_i = \mathrm{L}[\mathcal{E}_{\omega'-1}]_i \qquad \forall i \leq i_{\omega'} \tag{9}$$

$$\hat{u}[\mathcal{E}_{\omega'}]_i = \hat{u}[\mathcal{E}_{\omega'-1}]_i, \ \forall i < i_{\omega'} \ \text{ and } \ \hat{u}[\mathcal{E}_{\omega'}]_{i_{\omega'}} = 1 - \hat{u}[\mathcal{E}_{\omega'-1}]_{i_{\omega'}} \tag{10}$$

Let $P(\mathcal{E}_\omega)$ denote the probability of $\mathcal{E}_\omega$ correcting the trajectory of SC. It follows that:

$$\begin{aligned} P(\mathcal{E}_\omega) &= \Pr(\hat{u}[\mathcal{E}_\omega]_1^{i_\omega} = u_1^{i_\omega} | \mathbf{Y}) \\ &= \Pr(\hat{u}[\mathcal{E}_{\omega-1}]_{i_\omega} \neq u_{i_\omega}, \hat{u}[\mathcal{E}_{\omega-1}]_1^{i_{\omega-1}} = u_1^{i_{\omega-1}} | \mathbf{Y}) \\ &= p_e(\hat{u}[\mathcal{E}_{\omega-1}]_{i_\omega}) \cdot \prod_{j=i_{\omega-1}+1}^{i_\omega - 1} (1 - p_e(\hat{u}[\mathcal{E}_{\omega-1}]_j)) \cdot P(\mathcal{E}_{\omega-1}) \end{aligned} \tag{11}$$

where $p_e(\hat{u}[\mathcal{E}_{\omega-1}]_j) \overset{\text{def}}{=} \Pr\left(\hat{u}[\mathcal{E}_{\omega-1}]_j \neq u_j | \mathbf{Y}, \hat{u}[\mathcal{E}_{\omega-1}]_1^j = u_1^j\right)$. By taking into account Eq. (10), the above recursion can be unfolded to the following expression:

$$P(\mathcal{E}_\omega) = \prod_{j \in \mathcal{E}_\omega} p_e(\hat{u}[\mathcal{E}_{\omega-1}]_j) \ \cdot \prod_{\substack{j < i_\omega \\ j \in \mathcal{I} \setminus \mathcal{E}_\omega}} (1 - p_e(\hat{u}[\mathcal{E}_{\omega-1}]_j)) \tag{12}$$

Note that the second product on the right-hand side term of Eq. (12) is taken only over indexes $j \in \mathcal{I}$, since $p_e(\hat{u}[\mathcal{E}_{\omega-1}]_j) = 0$ for $j \notin \mathcal{I}$. Computing $p_e(\hat{u}[\mathcal{E}_{\omega-1}]_j)$ is an arduous task, since this probability is conditional on the fact that the previous bits have been correctly decoded by SC($\mathcal{E}_{\omega-1}$). Instead, one can compute the probability $q_e(\hat{u}[\mathcal{E}_{\omega-1}]_j) \overset{\text{def}}{=} \Pr(\hat{u}[\mathcal{E}_{\omega-1}]_j \neq u_j | \mathbf{Y}, \hat{u}[\mathcal{E}_{\omega-1}]_1^j)$, which is conditional on the previously decoded bits, irrespective of whether they have been correctly decoded or not, and is given by (this follows directly from the definition of $\mathrm{L}[\mathcal{E}_{\omega-1}]_j$ and $\hat{u}[\mathcal{E}_{\omega-1}]_j$):

$$q_e(\hat{u}[\mathcal{E}_{\omega-1}]_j) = \frac{1}{1 + \exp\left(|\mathrm{L}[\mathcal{E}_{\omega-1}]_j|\right)}, \qquad \forall j \in \mathcal{I} \tag{13}$$

Hence, we propose to use $q_e(\hat{u}[\mathcal{E}_{\omega-1}]_j)$ as an approximation of $p_e(\hat{u}[\mathcal{E}_{\omega-1}]_j)$, and we further introduce a parameter α (see below) as a mean to compensate this approximation. In practice,

the value of α can be optimized by Monte-Carlo simulation, as shown in Section V-C. Using $p_e(\hat{u}[\mathcal{E}_{\omega-1}]_j) \approx \frac{1}{1+\exp(\alpha|\mathrm{L}[\mathcal{E}_{\omega-1}]_j|)}$ in Eq. (12), we obtain the following metric, denoted by $M_\alpha(\mathcal{E}_\omega)$, which will be used to approximate the probability of $\mathcal{E}_\omega$ correcting the trajectory of SC:

Definition 3: The metric associated with a bit-flip $\mathcal{E}_\omega = \{i_1, \ldots, i_\omega\} \subset \mathcal{I}$, of order ω, is defined by:

$$M_\alpha(\mathcal{E}_\omega) = \prod_{j \in \mathcal{E}_\omega} \left(\frac{1}{1+\exp\left(\alpha|\mathrm{L}[\mathcal{E}_{\omega-1}]_j|\right)} \right) \cdot \prod_{\substack{j < i_\omega \\ j \in \mathcal{I} \setminus \mathcal{E}_\omega}} \left(\frac{1}{1+\exp\left(-\alpha|\mathrm{L}[\mathcal{E}_{\omega-1}]_j|\right)} \right) \tag{14}$$

Note that for a bit flip $\mathcal{E}_1 = \{i_1\}$ of order 1, the above metric can be written as:

$$M_\alpha(\mathcal{E}_1) = \frac{1}{1+\exp\left(\alpha|\mathrm{L}_{i_1}|\right)} \cdot \prod_{\substack{j < i_1 \\ j \in \mathcal{I}}} \left(\frac{1}{1+\exp\left(-\alpha|\mathrm{L}_j|\right)} \right), \tag{15}$$

where L_j are the LLR values computed by the initial SC decoding attempt. Moreover, the metric of the bit-flip $\mathcal{E}_\omega$ can be computed recursively, using the following equation:

$$M_\alpha(\mathcal{E}_\omega) = \frac{1}{1+\exp\left(\alpha|\mathrm{L}[\mathcal{E}_{\omega-1}]_{i_\omega}|\right)} \cdot \prod_{\substack{j=i_{\omega-1}+1 \\ j \in \mathcal{I}}}^{i_\omega - 1} \left(\frac{1}{1+\exp\left(-\alpha|\mathrm{L}[\mathcal{E}_{\omega-1}]_{i_\omega}|\right)} \right) \cdot M_\alpha(\mathcal{E}_{\omega-1}) \tag{16}$$

Indeed, by taking into account Eq. (9), it can be easily seen that the above recursion unfolds to the expression from Eq. (14).

Using the fact that $\frac{1}{1+\exp(x)} = \frac{\exp(-x)}{1+\exp(-x)}$, Eq. (14) can be rewritten:

$$M_\alpha(\mathcal{E}_\omega) = \prod_{j \in \mathcal{E}_\omega} \exp\left(-\alpha|\mathrm{L}[\mathcal{E}_{\omega-1}]_j|\right) \cdot \prod_{\substack{j \le i_\omega \\ j \in \mathcal{I}}} \left(\frac{1}{1+\exp\left(-\alpha|\mathrm{L}[\mathcal{E}_{\omega-1}]_j|\right)} \right) \tag{17}$$

By taking the logarithm of this formula, and denoting $M'_\alpha(\mathcal{E}_\omega) = -\frac{1}{\alpha} \cdot \log(M_\alpha(\mathcal{E}_\omega))$, one gets the following equivalent metric in the logarithmic domain:

$$\begin{aligned} M'_\alpha(\mathcal{E}_\omega) &= \sum_{j \in \mathcal{E}_\omega} |\mathrm{L}[\mathcal{E}_{\omega-1}]_j| + S_\alpha(\mathcal{E}_\omega) \\ \text{where } S_\alpha(\mathcal{E}_\omega) &= \frac{1}{\alpha} \sum_{\substack{j \le i_\omega \\ j \in \mathcal{I}}} \log(1+\exp(-\alpha \cdot |\mathrm{L}[\mathcal{E}_{\omega-1}]_j|)) \end{aligned} \tag{18}$$

On the basis of the above considerations, the list $\mathcal{L}_{\text{flip}}$ used within a generalized SCFlip-ω decoder should be constituted of bit-flips with the highest probability-domain metric M_α, or

equivalently with the lowest logarithmic domain metric M'_α, since they are the most likely to correct the trajectory of the SC decoding. For the sake of simplicity, the algorithms proposed in the next sections will be defined by using the metric M_α, but it is worth mentioning that the logarithm domain metric M'_α is more suitable for practical implementations, due to its better numerical stability.

B. Impact of the α parameter

In order to understand the impact of the parameter α on the proposed metric, we start by considering two limiting cases, namely $\alpha = 0$ and $\alpha \to +\infty$.

For $\alpha = 0$, using Eq. (17), it can be seen that $M_0(\mathcal{E}_\omega) = \left(\frac{1}{2}\right)^{k_\mathcal{I}}$, where $k_\mathcal{I}$ is the number of positions in $\mathcal{I}$ less than or equal to i_ω. Therefore, if $\mathcal{E}_\omega = \{i_1, \ldots, i_\omega\}$ and $\mathcal{E}'_{\omega'} = \{i'_1, \ldots, i'_{\omega'}\}$ are two bit-flips of order ω and ω', $M_0(\mathcal{E}_\omega) \geq M_0(\mathcal{E'}_{\omega'}) \Leftrightarrow i_\omega \leq i'_{\omega'}$. In other words, bit-flips are ordered by M_0 according to the index of their last flipped position.

For $\alpha \to +\infty$, we consider the equivalent logarithmic-domain metric defined in Eq. (18). It can be seen that $\lim\limits_{\alpha \to +\infty} S_\alpha(\mathcal{E}_\omega) = 0$, thus $M'_\infty(\mathcal{E}_\omega) = \sum_{j \in \mathcal{E}_\omega} |\mathrm{L}[\mathcal{E}_{\omega-1}]_j|$ is the sum of *reliabilities* (*i.e.* absolute value of the LLR) of the flipped positions. In the particular case of bit-flips of order 1, this metric is exactly the same as the one in [5].

In general, for $0 < \alpha < +\infty$, $S_\alpha(\mathcal{E}_\omega)$ can be seen as a penalty added to $\sum_{j \in \mathcal{E}_\omega} |\mathrm{L}[\mathcal{E}_{\omega-1}]_j|$, which takes into consideration the sequential aspect of the SC decoding, providing and intermediate and tunable solution between prioritizing bit-flips according to either the index of their last flipped position or the sum of reliabilities of the flipped positions.

The value of the trade-off parameter α can be optimized by Monte-Carlo simulation. It is expected that the optimal α value decreases with the SNR. Indeed, considering Eq. (18) for a fixed α value, and taking the limit as the SNR goes to infinity, the term $S_\alpha(\mathcal{E}_\omega)$ tends to 0 and becomes negligible compared to $\sum_{j \in \mathcal{E}_\omega} |\mathrm{L}[\mathcal{E}_{\omega-1}]_j|$, 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 α will decrease with the SNR, so that to rebalance the contribution of the $S_\alpha(\mathcal{E}_\omega)$ term to the value of the considered metric. This is confirmed by the Monte-Carlo simulations presented in section V-C.

Finally, it is worth underlining the strong similarity between the derived bit-flip metric and the path metric used by the SCL decoder (Eq. (3)). However, unlike the SCL path metric, frozen bits do not contribute to our proposed bit-flip metric.

C. Optimization of the α parameter

This section investigates the optimization of the α parameter, so that to increase the probability that the bit-flip $\mathcal{E}_{\mathbf{Y}}$ is ranked high by the metric M_α, where $\mathcal{E}_{\mathbf{Y}}$ is bit-flip correcting the SC decoding trajectory, for the the current noise realization $\mathbf{Y}$ (see Section IV-A).

Let $\bar{\mathcal{L}}_{\alpha,\mathbf{Y}}$ denote the list of all the bit-flips $\mathcal{E}$, of any order $\omega = 1, \ldots, K+r$, ordered according to decreasing values of $M_\alpha(\mathcal{E})$. Note that the bit-flips ordering depends on both the value of α and the current noise realization $\mathbf{Y}$. We denote by $\text{rk}_\alpha(\mathcal{E}_{\mathbf{Y}})$ the rank (position) of $\mathcal{E}_{\mathbf{Y}}$ within the ordered list $\bar{\mathcal{L}}_{\alpha,\mathbf{Y}}$. Let $\mathcal{E}_{\mathbf{Y}} = \{i_1, \ldots, i_{\omega_{\mathbf{Y}}}\}$, where $\omega_{\mathbf{Y}} \geq 1$ is the order of $\mathcal{E}_{\mathbf{Y}}$. Using the recursion from Eq. (16), it follows that:

$$M_\alpha(\{i_1\}) > M_\alpha(\{i_1, i_2\}) > \cdots > M_\alpha(\{i_1, \ldots, i_{\omega_{\mathbf{Y}}}\}) \tag{19}$$

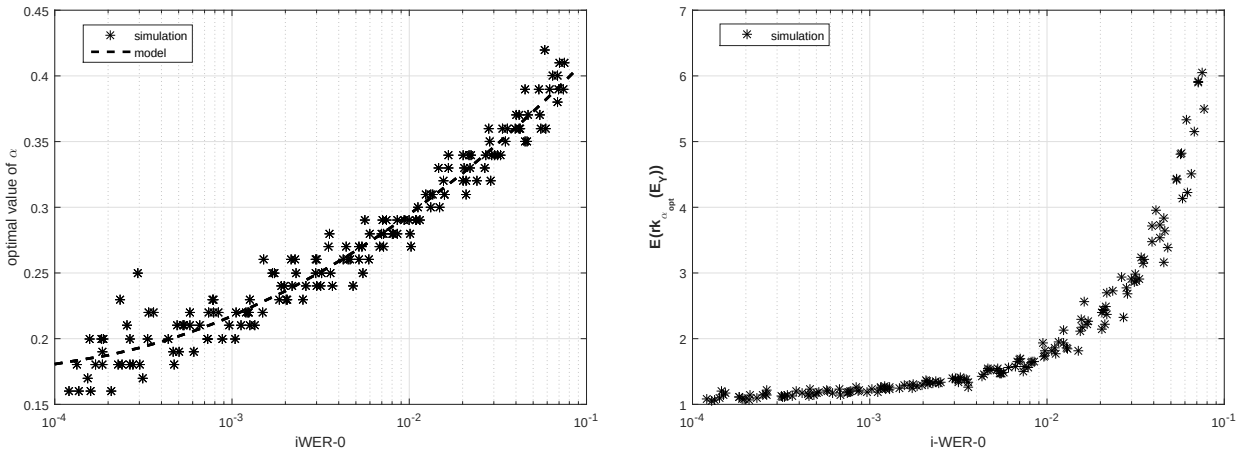
and therefore:

$$\text{rk}_\alpha(\mathcal{E}_{\mathbf{Y}}) \geq \omega_{\mathbf{Y}} \tag{20}$$

Finally, we define the optimal α value, denote by α_{opt}, as:

$$\alpha_{\text{opt}} = \underset{\alpha}{\text{argmin}}\, \mathbb{E}\left(\text{rk}_\alpha(\mathcal{E}_{\mathbf{Y}})\right), \tag{21}$$

where $\mathbb{E}\left(\text{rk}_\alpha(\mathcal{E}_{\mathbf{Y}})\right)$ denotes the expected value of the random variable $\text{rk}_\alpha(\mathcal{E}_{\mathbf{Y}})$, assuming that $\omega_{\mathbf{Y}} \geq 1$ (*i.e.*, SC fails to decode the current noise realization $\mathbf{Y}$).



(a) α_{opt} as function of iWER-0

(b) $\mathbb{E}\left(\text{rk}_{\alpha_{\text{opt}}}(\mathcal{E}_{\mathbf{Y}})\right)$ as function of iWER-0

Fig. 3. Optimal values as function of iWER-0 for various code-lengths N, coding-rates R, and SNR values. Each point corresponds to a different triplet (N, R, SNR).

We have determined the α_{opt} value by Monte-Carlo simulation, for various code parameters $(N, K+r)$ and SNR values. For each pair $(N, K+r)$ and SNR value, we also determined the corresponding iWER-0 value, *i.e.*, the WER of the SC decoder for a polar code with parameters $(N, K+r)$, as explained in Section IV-B. Precisely, we have considered parameters $(N, K+r) = (256, \{96, 128, 160\}), (512, \{192, 288, 256, 320\}), (1024, \{384, 512, 640\})$, while the SNR values have been chosen such that iWER-0 varies from 10^{-4} to 10^{-1}. Fig. 3(a) shows the scatter plot of α_{opt} as a function of iWER-0, while Fig. 3(b) shows the scatter plot of $\mathbb{E}\left(\text{rk}_{\alpha_{\text{opt}}}(\mathcal{E}_{\mathbf{Y}})\right)$ as a function of iWER-0.

Fig. 3(a) clearly indicates a correlation between α_{opt} and iWER-0 values. Hence, we propose to approximate the α_{opt} value, by using a quadratic model (in semilog scale):

$$\alpha_{\text{model}}(\texttt{iWER-0}) = a_1 \cdot \log(\texttt{iWER-0})^2 + a_2 \cdot \log(\texttt{iWER-0}) + a_3, \tag{22}$$

with the following coefficients providing the best fit to the simulation data:

$$a_1 = 0.0038, \quad a_2 = 0.0779, \quad a_3 = 0.5716$$

It is worth noticing that the band of the $\mathbb{E}\left(\text{rk}_{\alpha_{\text{opt}}}(\mathcal{E}_{\mathbf{Y}})\right)$ scatter plot in Fig. 3(b) is very narrow,

therefore approximating α_{opt} by α_{model} should result in a negligible difference in terms of rank expectation. Fig. 3(b) also demonstrates the effectiveness of the proposed metric in ranking in top positions the bit-flips $\mathcal{E}_{\mathbf{Y}}$.

Finally, we describe below an efficient algorithm to compute $\text{rk}_\alpha(\mathcal{E}_{\mathbf{Y}})$. For this, one needs to determine the number of bit-flips $\mathcal{E}$, such that $M_\alpha(\mathcal{E}) \geq M_\alpha(\mathcal{E}_{\mathbf{Y}})$.

Let $\mathcal{L}_m = \{\mathcal{E} \mid M_\alpha(\mathcal{E}) \geq m\}$, where $m \in [0, 1]$. To determine $\mathcal{L}_m$, we proceed as follows:

- First, we evaluate $M_\alpha(\mathcal{E}_1)$ for all the bit-flips $\mathcal{E}_1$ of order 1, and add to $\mathcal{L}_m$ those bit-flips such that $M_\alpha(\mathcal{E}_1) \geq m$.
- For $\omega > 1$, we evaluate $M_\alpha(\mathcal{E}_\omega)$ for all the bit-flips $\mathcal{E}_\omega$ of order ω, such that $\mathcal{E}_{\omega-1} \in \mathcal{L}_m$, where $\mathcal{E}_{\omega-1}$ denotes the bit-flip determined by the first $\omega - 1$ elements of $\mathcal{E}_\omega$. We add to $\mathcal{L}_m$ the bit-flips of order ω, such that $M_\alpha(\mathcal{E}_\omega) \geq m$.
- The algorithm stops if, at the previous step, no bit-flip of order ω is added to $\mathcal{L}_m$.

Clearly, $\text{rk}_\alpha(\mathcal{E}_{\mathbf{Y}})$ is equal to the number of bit-flips in $\mathcal{L}_m$, for $m = M_\alpha(\mathcal{E}_{\mathbf{Y}})$.

VI. Dynamic SCFlip Decoder

A. D-SCFlip and D-SCFlip-ω Decoders

In this section we introduce a generalized SCFlip decoding algorithm, characterized in that the bit-flip list $\mathcal{L}_{\text{flip}}$ contains the T bit-flips $\mathcal{E}$ with highest $M_\alpha(\mathcal{E})$ values, according to the current noise realization $\mathbf{Y}$. In order to avoid the evaluation of $M_\alpha(\mathcal{E})$ for all possible bit-flips $\mathcal{E}$, the proposed algorithm builds the list $\mathcal{L}_{\text{flip}}$ on-the-fly, concurrently with the initial SC decoding attempt, and then with each new decoding attempt $\text{SC}(\mathcal{E})$.

The proposed algorithm, referred to as *Dynamic SC-Flip* (D-SCFlip), is described in Algorithm 2. The description is similar to the one in Algorithm 1, except of the functions $\text{Init}()$ and $\text{Update}()$, used to initialize and update the list of bit-flips $\mathcal{L}_{\text{flip}}$ and the list of corresponding metric values $\mathcal{M}_{\text{flip}} \stackrel{\text{def}}{=} \{M_\alpha(\mathcal{E}) \mid \mathcal{E} \in \mathcal{L}_{\text{flip}}\}$.

Algorithm 2 D-SCFlip decoder

procedure D-SCFLIP($\mathbf{Y}, \mathcal{I}, T$)
  $\left(\hat{u}_1^N, \{\mathrm{L}_i\}_{i\in\mathcal{I}}\right) \leftarrow$SC($\emptyset$)
  if CRC($\hat{u}_1^N$) = success **then** return $\hat{u}_1^N$;
  else Init($\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{\mathrm{L}_i\}_{i\in\mathcal{I}}$); **end if**
  for $t = 1, \ldots, T$ **do**
    $\left(\hat{u}_1^N, \{\mathrm{L}[\mathcal{E}_t]_i\}_{i\in\mathcal{I}}\right) \leftarrow$SC($\mathcal{E}_t$)
    if CRC($\hat{u}_1^N$) = success **then** return $\hat{u}_1^N$;
    else Update($\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{\mathrm{L}[\mathcal{E}_t]_i\}_{i\in\mathcal{I}}, \mathcal{E}_t$); **end if**
  end for
  return $\hat{u}_1^N$;
end procedure

Init($\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{\mathrm{L}_i\}_{i\in\mathcal{I}}$): this function evaluates $M_\alpha(\mathcal{E})$ for all the bit-flips of order 1, $\mathcal{E} = \{i\}$, $i \in \mathcal{I}$, and orders them according to decreasing value of $M_\alpha(\mathcal{E})$. $\mathcal{L}_{\text{flip}}$ is initialized with the T bit-flips of order 1 with highest metric values, and the ordered metric values are stored in $\mathcal{M}_{\text{flip}}$. Note that the $M_\alpha(\mathcal{E})$ values computed at this step make use of the LLR values $\{\mathrm{L}_i\}_{i\in\mathcal{I}}$ computed during the initial SC decoding attempt (see Eq. (15)).

Update($\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{\mathrm{L}[\mathcal{E}_t]_i\}_{i\in\mathcal{I}}, \mathcal{E}_t$): After each unsuccessful decoding attempt $\mathcal{E}_t$, $\mathcal{L}_{\text{flip}}$ and $\mathcal{M}_{\text{flip}}$ are updated, as described in Algorithm 3. Let $\mathcal{E}_t = \{i_1, \ldots, i_{\omega_t}\}$, where ω_t is the order of $\mathcal{E}_t$. The function evaluates $M_\alpha(\mathcal{E})$, for all the all the bit-flips $\mathcal{E} = \mathcal{E}_t \cup \{i\}$, where $i \in \mathcal{I}$ and $i > i_{\omega_t}$. In case that $M_\alpha(\mathcal{E}) > \mathcal{M}_{\text{flip}}(T)$, $\mathcal{L}_{\text{flip}}$ and $\mathcal{M}_{\text{flip}}$ are updated by inserting $\mathcal{E}$ and $M_\alpha(\mathcal{E})$ into appropriate positions. Since $M_\alpha(\mathcal{E}_t) > M_\alpha(\mathcal{E})$, $\mathcal{E}$ is necessarily inserted into a position $t' > t$. Note that the $M_\alpha(\mathcal{E})$ values computed at this step make use of the LLR values $\{\mathrm{L}[\mathcal{E}_t]_i\}_{i\in\mathcal{I}}$ computed during the SC($\mathcal{E}_t$) decoding attempt (see Eq. (16)). We also note that the initialization of $\mathcal{L}_{\text{flip}}$ and $\mathcal{M}_{\text{flip}}$ can also result from the update procedure of Algorithm 3, by taking $\mathcal{E}_t = \emptyset$.

Algorithm 3 Bit-flips update

procedure UPDATE($\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{\mathrm{L}[\mathcal{E}_t]_i\}_{i\in\mathcal{I}}, \mathcal{E}_t$)
  for $i = \text{last}(\mathcal{E}_t) + 1, \ldots, N$ and $i \in \mathcal{I}$ **do**
    $\mathcal{E} = \mathcal{E}_t \cup \{i\}$; $m = M_\alpha(\mathcal{E})$;
    if $m > \mathcal{M}_{\text{flip}}(T)$ **then**
      $\text{Insert\_flip}(\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \mathcal{E}, m)$;
    end if
  end for
  return $(\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}})$;
end procedure

We now substantiate the ability of the D-SCFlip decoder to explore the bit-flips with highest metric values. We denote by $\mathcal{L}_{\mathbf{Y}} = \{\mathcal{E}_1, \ldots, \mathcal{E}_{T_{\mathbf{Y}}}\} \subset \mathcal{L}_{\text{flip}}$ the ordered list of bit-flips corresponding to the decoding attempts performed by the D-SCFlip decoder for the current noise realization $\mathbf{Y}$ (not including the initial SC decoding attempt). Hence, $T_{\mathbf{Y}} \leq T$, since the D-SCFlip decoder stops as soon as a decodig attempt satisfies the CRC. Put differently, $\mathcal{L}_{\mathbf{Y}}$ is determined by the first $T_{\mathbf{Y}}$ bit-flips in $\mathcal{L}_{\text{flip}}$, at the moment when the D-SCFlip decoder stops.

Proposition 1: $\mathcal{L}_{\mathbf{Y}}$ contains the $T_{\mathbf{Y}}$ bit-flips with the highest $M_\alpha(\mathcal{E})$ values among all the possible bit-flips $\mathcal{E}$.

Proof. We have to prove that for any bit-flip $\mathcal{E} = \{i_1, \ldots, i_\omega\}$ of order ω, such that $M_\alpha(\mathcal{E}) > M_\alpha(\mathcal{E}_{T_{\mathbf{Y}}})$, then $\mathcal{E} \in \mathcal{L}_{\mathbf{Y}}$. We proceed by induction on ω. For $\omega = 1$, the assertion follows from the fact that $\mathcal{L}_{\text{flip}}$ is initialized with the T bit-flips of order 1 with the highest metric values. For $\omega > 1$, let $\mathcal{E}' = \{i_1, \ldots, i_{\omega-1}\}$. Since $M_\alpha(\mathcal{E}') > M_\alpha(\mathcal{E}) > M_\alpha(\mathcal{E}_{T_{\mathbf{Y}}})$, it follows from the induction hypothesis that $\mathcal{E}' \in \mathcal{L}_{\mathbf{Y}}$. Hence, the decoding attempt SC($\mathcal{E}'$) is necessarily performed before SC($\mathcal{E}_{T_{\mathbf{Y}}}$), and $\mathcal{L}_{\text{flip}}$ is updated after SC($\mathcal{E}'$) by evaluating the bit-flips of order ω that contains $\mathcal{E}'$. During this update, $\mathcal{E}$ is added to $\mathcal{L}_{\text{flip}}$, in a position that necessarily precedes that of $\mathcal{E}_{T_{\mathbf{Y}}}$. Therefore, $\mathcal{E} \in \mathcal{L}_{\mathbf{Y}}$, which completes the proof. □

Finally, we denote by D-SCFlip-ω the decoder obtained by restricting $\mathcal{L}_{\text{flip}}$ to bit-flips of order less than or equal to ω. It has a similar description to the one provided in Algorithm 2, but the update procedure $\mathrm{Update}(\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{\mathrm{L}[\mathcal{E}_t]_i\}_{i\in\mathcal{I}}, \mathcal{E}_t)$ is only executed if the order of $\mathcal{E}_t$ is less than ω (thus, no bit-flip of order greater than ω is added to $\mathcal{L}_{\text{flip}}$). Similarly to Proposition 1, it can be seen that the D-SCFlip-ω decoder explores the bit-flips $\mathcal{E}$ of order less than or equal to ω, with the $T_{\mathbf{Y}}$ highest $M_\alpha(\mathcal{E})$ values. The purpose of the D-SCFlip-ω decoder is to assess the effectiveness of the proposed metric in approaching the performance of the ideal `i`SCFlip-ω decoder, defined in Section IV-B. We also note that limiting the maximum bit-flip order may have some practical payoffs, or be imposed by some practical constraints (*e.g.*, related to hardware implementation), but such considerations are beyond the scope of this paper.

B. Practical implementation

This section discusses two practical simplifications, which allow reducing the computational cost of implementing the proposed D-SCFlip decoder. First, to reduce the computational cost associated with new decoding attempts, the following proposition determines the position from which the SC decoding need to be restarted.

Proposition 2: Let $\mathcal{E}_1 = \{i_1, i_2 \ldots, i_{\omega_1}\}$ and $\mathcal{E}_2 = \{j_1, j_2 \ldots, j_{\omega_2}\}$ be two bit-flips, and $1 \leq \omega \leq \min(\omega_1, \omega_2)$ be such that $i_{\omega'} = j_{\omega'}$ for any $\omega' < \omega$, and $i_\omega \neq j_\omega$. Let $k = \min(i_\omega, j_\omega)$. Then SC($\mathcal{E}_1$) and SC($\mathcal{E}_2$) decoding attempts are strictly identical in terms of LLRs and hard-decision estimates until index k, where they differ only by the hard-decision estimate of the bit u_k.

As a consequence, assuming that SC($\mathcal{E}_1$) and SC($\mathcal{E}_2$) are two successive decoding attempts, the latter may start from the index $k+1$, after the hard-decision estimate of u_k has been flipped.

The following proposition, which follows from Eq. (16), allows reducing the computational complexity of the Update procedure. It allows avoiding the computation of the metric values $m = M_\alpha(\mathcal{E})$ (see Algorithm 3), for bit-flips $\mathcal{E} = \mathcal{E}_t \cup \{i\}$ which would not be inserted in the list anyway.

Proposition 3: Consider the update procedure $\mathrm{Update}(\mathcal{L}_{\text{flip}}, \mathcal{M}_{\text{flip}}, \{\mathrm{L}[\mathcal{E}_t]_i\}_{i\in\mathcal{I}}, \mathcal{E}_t)$, after some decoding attempt $\mathrm{SC}(\mathcal{E}_t)$, with $\mathcal{E}_t = \{i_1, \ldots, i_{\omega_t}\}$. For any $i > i_{\omega_t}$, let

$$\Pi(\mathcal{E}_t, i) = \prod_{\substack{j=i_{\omega_t}+1 \\ j\in\mathcal{I}}}^{i-1} \left(\frac{1}{1+\exp\left(-\alpha|\mathrm{L}[\mathcal{E}_t]_j|\right)} \right) \tag{23}$$

Then:

(i) $M_\alpha(\mathcal{E}_t \cup \{i\}) = M_\alpha(\mathcal{E}_t) \cdot \Pi(\mathcal{E}_t, i) \cdot \dfrac{1}{1+\exp\left(\alpha|\mathrm{L}[\mathcal{E}_t]_i)|\right)}$

(ii) Let $i_{\omega_t} < k \leq N-1$ be the last (highest) value such that $M_\alpha(\mathcal{E}_t) \cdot \Pi(\mathcal{E}_t, k) \geq \mathcal{M}_{\text{flip}}(T)$ (note that for $k = i_{\omega_t}+1$, $M_\alpha(\mathcal{E}_t) \cdot \Pi(\mathcal{E}_t, k) = M_\alpha(\mathcal{E}_t) \geq \mathcal{M}_{\text{flip}}(T)$). Then, $\mathcal{M}_{\text{flip}}(T) > M_\alpha(\mathcal{E}_t) \cdot \Pi(\mathcal{E}_t, i) > M_\alpha(\mathcal{E}_t \cup \{i\})$, for any $i = k+1, \ldots, N-1$. In particular, the **for** loop in Algorithm 3 (line 2) can be restricted to values $i = i_{\omega_t}+1, \ldots, k$.

C. Numerical Results

All the simulation results presented in this section assume a BI-AWGN channel. Concatenated CRC-polar codes use $(r = 16)$-bits CRC, with generator polynomial $g(x) = x^{16} + x^{15} + x^2 + 1$. The set $\mathcal{I}$ is optimized for each SNR value, according to the Gaussian Approximation method presented in [16].

We start by investigating the impact of the parameter α on the decoding performance of the D-SCFlip decoder. Fig. 4 shows the WER performance of the D-SCFlip decoder for a concatenated CRC-Polar code with parameters $(N, K+r) = (1024, 512+16)$, and several *fixed* α parameters, where *fixed* means that the same α parameter is used for all the SNR values. Each α parameter corresponds to the optimal α_{opt} value for a particular SNR (Section V-C), which is indicated in the legend. The maximum number of extra decoding attempts (*i.e.*, not including the initial SC decoding attempt) is set to $T = 20$. For comparison purposes, the WER performance of the SC decoder for the $(N, K) = (1024, 512)$ polar code is also shown. The dashed curve plots the WER performance using the $\alpha_{\text{model}}(\texttt{iWER-0})$ value, which varies with the SNR. Precisely, for each SNR value we first determine *offline* the corresponding $\texttt{iWER-0}$ value, by using the

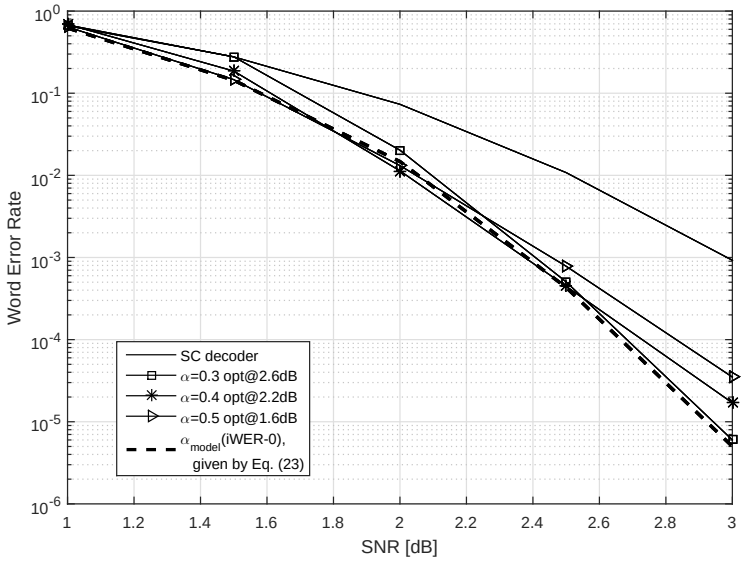


Fig. 4. Impact of α on the performance of D-SCFlip for $T = 20$ for a code $(N, K + r) = (1024, 512 + 16)$

OA-SC decoder, then the value of $\alpha_{\text{model}}(\texttt{iWER-}0)$, according to Eq. (22). The figure highlights the performance loss – in the low, medium or high SNR regime – when a fixed α value is used throughout the whole range of SNR values. It also demonstrates the effectiveness of the proposed model, since the α_{model} curve matches the envelope of the curves with a fixed α. For all the simulation results presented in the remaining of this section, we shall assume that $\alpha = \alpha_{\text{model}}(\texttt{iWER-}0)$.

Fig. 5 shows the WER performance of the D-SCFlip for $T \in \{10, 50, 400\}$, for the concatenated CRC-Polar code with parameters $(N, K+r) = (1024, 512+16)$. The values of T have been chosen such that the D-SCFlip performance is close to or outperforms the ideal performance $\texttt{iWER-}\omega$ for $\omega = 1, 2$ and 3 respectively, thus proving the ability of the proposed both metric and decoder to correct higher-order noise realizations. The impact of saturating to a low value of ω is also shown by considering a D-SCFlip-($\omega = 1$) decoder with $T = 10$, for which the performance tightly approaches the ideal performance $\texttt{iWER-}1$.

Fig. 6 provides a comparison of the D-SCFlip, SCFlip [5], and SCL decoders for concatenated CRC-polar codes, with $(N, K + r) = (1024, 512 + 16)$. The D-SCFlip decoder has a maximum

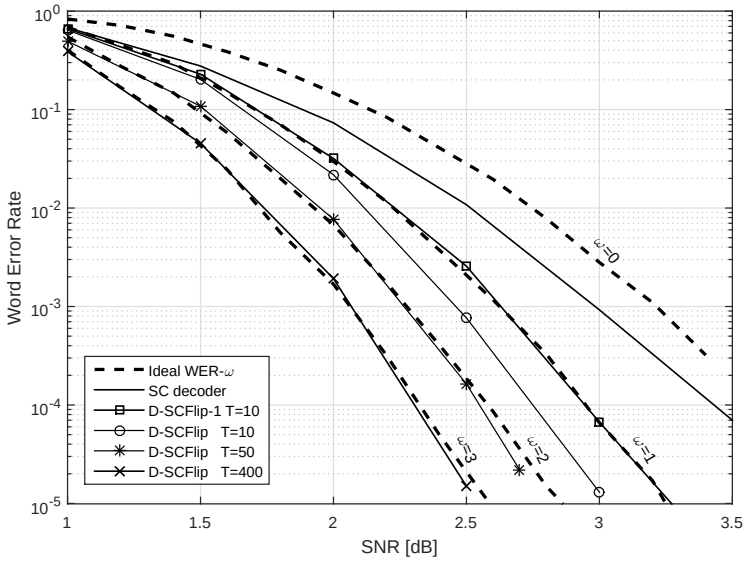


Fig. 5. Performance of D-SCFlip decoder for several values of T and a code $(N, K + r) = (1024, 512 + 16)$

number of extra decoding attempts $T = \{10, 50, 400\}$. The performance of the state-of-the-art SCFlip [5] is given for $T = 10$. However, as this decoder is actually a D-SCFlip-1 with $\alpha = +\infty$ (see Section V-B), even for higher values of T, its performance is still lower bounded by the ideal performance iWER-1. On the contrary, the D-SCFlip decoder performance significantly improves with increasing T values, outperforming the SCFlip decoder by $0.4\,$dB for $T = 10$, and $0.8\,$dB for $T = 400$ (at WER $= 10^{-4}$).

For $T = 400$, the proposed D-SCFlip decoder closely approaches the performance of the SCL decoder with $L = 16$. Even though the maximum number of additional attempts ($T = 400$) used by the D-SCFlip decoder is considerably higher than the size of the list ($L = 16$) used by the SCL decoder, it should be understood that the trade-off of the D-SCFlip decoder is different: it offers a low computational complexity, especially in moderate to high SNR regime, since additional decoding attempts are performed only in case the SC decoding fails.

For comparison purposes, we have also included in Fig. 6 the WER performance of a $(3, 6)$-regular LDPC code, with $(N, K) = (1024, 512)$, under Belief Propagation (BP) decoding. The LDPC code is constructed by using the Progressive Edge Growth (PEG) algorithm [17], and has

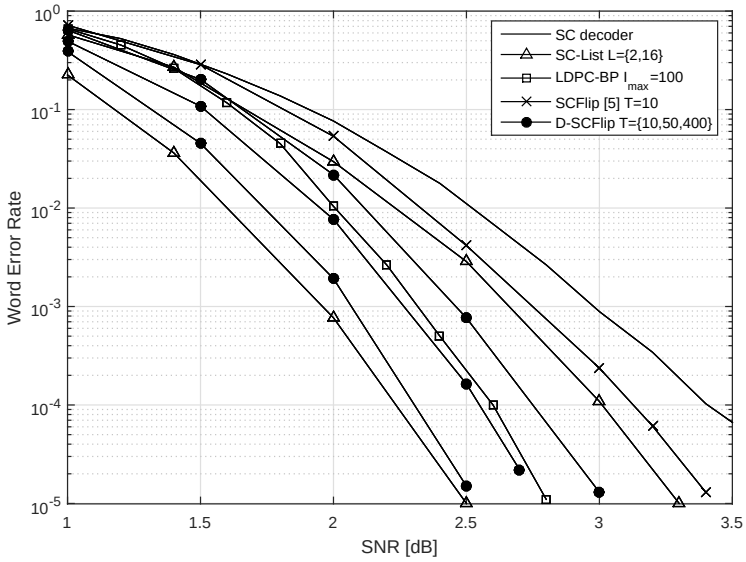


Fig. 6. Comparison between D-SCFlip decoder and BP decoder of LDPC codes and SC-List decoders of polar codes at length $N = 1024$ and rate $R = 0.5$

girth $g = 8$. The maximum number of iterations for the BP decoding is set to 100, since for higher values the performance improvement is actually negligible. It can be observed that the BP decoder is outperformed by the D-SCFlip decoder for $T \geq 50$.

The average number of extra decoding attempts performed by the D-SCFlip in case the SC decoder fails, denoted by T'_{ave}, is shown in Fig. 7. One can observe that T'_{ave} quickly drops and approaches 1 for high SNR values, demonstrating the effectiveness of the proposed D-SCFlip decoder, and implicitly of the proposed bit-flip metric, in finding the higher-order bit-flips that correct the actual noise realization. Comparing with the SCFlip from [5], the proposed D-SCFlip decoder requires a smaller number of extra decoding attempts at high SNR, thus resulting in a lower computational complexity, while providing a significant gain in terms of WER performance.

Finally, let T_{ave} denote the *overall average* number of extra decoding attempts, *i.e.*, averaged over all the cases, irrespective of the SC decoder status (successful or not). It follows that $T_{\text{ave}} = T'_{\text{ave}}\text{WER}_{\text{SC}}$, where WER_{SC} denotes the WER of the SC decoder. Compared to the SC decoding,

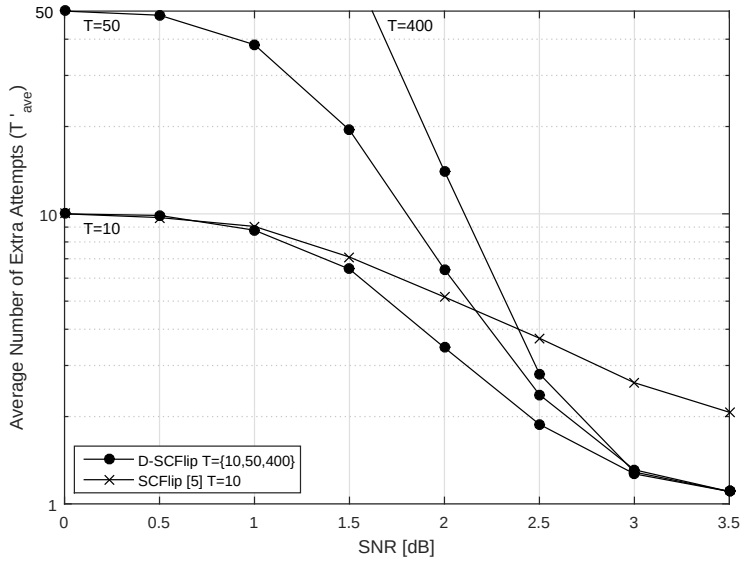


Fig. 7. Average number of extra attempts (T'_{ave}) for D-SCFlip decoders and $N = 1024$ and rate $R = 0.5$

D-SCFlip decoding results in an increase of both the average computational complexity[1] and average decoding latency by a factor of only $(1 + T_{\text{ave}})$, where the term 1 in the parenthesis accounts for the initial SC decoding attempt. Note also that the contribution of T_{ave} actually becomes negligible in the waterfall region of the SC decoder. As a matter of comparison, the computational complexity of the SCL decoder, with a list of size L, is L times higher than the one of the SC decoding, while they both have the same decoding latency. D-SCFlip considerably reduces the computational complexity, by relying on successive – rather than parallel – decoding attempts, coupled with a judicious choice of the latter ones. While this results in a variable decoding latency, with worst case latency given by the maximum number of decoding attempts T, the average decoding latency is nearly the same as the latency of the SC decoding.

[1]We do not take into account the practical simplifications proposed in Section VI-B, and assume that the computational complexity of each new decoding attempt is the same as the one of the initial SC decoding. The computational complexity of the Update procedure is not taken into account, since it is linear in N, and thus negligible with respect to the computational complexity of SC.

VII. Conclusion

In this paper, we investigated a Generalized SCFlip decoding for polar codes, characterized by T new decoding attempts, where one or several positions are flipped from the standard SC decoding. First, we studied the WER performance of an ideal Generalized SCFlip decoder, with maximum bit-flip order ω, which revealed potential for significant improvements, enabled by the use of higher-order bit-flips. Subsequently, we concentrated on proposing a practical method to take advantage of the benefits offered by the use of higher-order bit-flips, which led to two complementary improvements. First, a new metric was proposed, suited to bit-flips of any order, and optimized such that the sequential aspect of successive cancellation decoder is accurately taken into consideration. We also provided an analysis of the impact of the parameter α used within the proposed metric, and proposed an empirical model to estimate its optimal value as a function of $\texttt{iWER-}0$, which can be easily evaluated by using the density evolution technique. Secondly, we investigated a method to dynamically build the bit-flips list $\mathcal{L}_{\text{flip}}$, so that to guarantee that new decoding attempts are performed by decreasing probability of success, according to the proposed metric. The resulting D-SCFlip algorithm was shown to offer a substantial gain in terms of WER performance, as compared to the state-of-the-art SCFlip decoder, while having a lower computational complexity. Finally, we showed that the D-SCFlip decoder is an interesting variable-latency approach, which provides a different trade-off compared to SCL decoding of polar codes, by keeping the computational complexity close to the one of the SC decoder, while providing decoding performance close to SCL decoding with list size $L = 16$.

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

[3] V. Miloslavskaya and P. Trifonov, "Design of binary polar codes with arbitrary kernel," in *Information Theory Workshop (ITW)*. IEEE, 2012, pp. 119–123.

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

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

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

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

[8] 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.

[9] 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.

[10] L. Chandesris, V. Savin, and D. Declercq, "An improved scflip for polar codes," in *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2016, pp. 1–6.

[11] A. Balatsoukas-Stimming, M. B. Parizi, and A. Burg, "LLR-based successive cancellation list decoding of polar codes," *CoRR*, vol. abs/1401.3753, 2014. [Online]. Available: http://arxiv.org/abs/1401.3753

[12] G. Sarkis and W. J. Gross, "Increasing the throughput of polar decoders," *IEEE Communications Letters*, vol. 17, no. 4, pp. 725–728, 2013.

[13] A. Balatsoukas-Stimming, A. J. Raymond, W. J. Gross, and A. Burg, "Hardware architecture for list successive cancellation decoding of polar codes," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 61, no. 8, pp. 609–613, 2014.

[14] Y. Fan, J. Chen, C. Xia, C.-y. Tsui, J. Jin, H. Shen, and B. Li, "Low-latency list decoding of polar codes with double thresholding," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 1042–1046.

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

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

[17] D. A. Xiao-Yu Hu, E. Eleftheriou, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Transactions on Information Theory*, vol. 52, no. 51, pp. 386–398, 2005.