SIMPLIFIED DYNAMIC SC-FLIP POLAR DECODING

Furkan Ercan *Thibaud Tonnellier* *Nghia Doan* *Warren J. Gross*

Department of Electrical and Computer Engineering, McGill University, Montréal, Québec, Canada

ABSTRACT

SC-Flip (SCF) decoding is a low-complexity polar code decoding algorithm alternative to SC-List (SCL) algorithm with small list sizes. To achieve the performance of the SCL algorithm with large list sizes, the Dynamic SC-Flip (DSCF) algorithm was proposed. However, DSCF involves logarithmic and exponential computations that are not suitable for practical hardware implementations. In this work, we propose a simple approximation that replaces the transcendental computations of DSCF decoding. Moreover, we show how to incorporate fast decoding techniques with the DSCF algorithm. With proposed approaches, the computational complexity of DSCF decoding is remarkably reduced while maintaining equivalent decoding performance.

***Index Terms*—** Polar codes, 5G, SC-Flip decoding

1. INTRODUCTION

Polar codes, introduced by Arıkan in [1], are a class of forward error-correcting codes that asymptotically achieve the capacity of binary-input discrete memoryless channels. They have been selected as a coding scheme within the 5th generation wireless communication (5G) standard [2]. Although the successive cancellation (SC) decoding algorithm enables to prove the capacity achieving property of polar codes, its error-correction performance is mediocre at practical code lengths.

Various SC-based decoding algorithms have been proposed to enhance the error-correction performance, such as SC-List (SCL) [3], SC-Flip (SCF) [4] and SC-Stack (SCS) [5] decoding. SC-List decoding uses several SC decoders in parallel to maintain a list of candidate codewords. Error-correction performance of SCL decoding improves with the list size, which adversely impacts its implementation complexity [6]. Similarly, SCS can be parametrized to match the SCL decoding performance but exhibits a high memory complexity [7]. SCF decoding uses several SC decoding attempts when an initial SC decoding fails due to a single channel-induced error. It has improved error-correction performance and has an average computational complexity similar to that of SC at medium-to-high signal-to-noise ratio (SNR) regions. However, SCF has variable decoding latency, and its error-correction performance can only compare with SCL decoding with small list sizes. Existing improvements on SCF decoding [8–10] are limited with correction of a single channel-induced error. To improve the error-correction performance, Dynamic SC-Flip (DSCF) decoding was proposed in [11], which can find more than one channel-induced errors. However, operations involved in DSCF decoding require logarithmic and exponential calculations, which make it challenging for practical hardware implementations.

To improve the throughput of SC-based decoders, the identification of easily-decoded sub-codes found in polar codes – called special nodes – was first introduced in [12, 13]. Decoding techniques of these special nodes are extended for SCF decoding in [14, 15]. However, how to consider such special nodes under DSCF decoding remains unexplored.

In this paper, we first show that the transcendental computations in the DSCF algorithm can be replaced by a simple approximation, that allows DSCF to be considered for practical hardware implementations. We show that the proposed approximation does not incur any significant loss in error-correction performance. Then, we propose novel methods to consider special nodes under DSCF decoding. We show that the considered decoding techniques can reduce the average number of decoding steps by up to a factor of 6.4, while the error-correction performance remains similar to that of the original DSCF algorithm. These two improvements will allow to consider hardware implementations targeting high average throughput of DSCF-based decoding algorithms.

The organization of this paper is as follows. Encoding and decoding of polar codes are described in Section 2. The approximation that replaces the transcendental computations in DSCF decoding is explained in Section 3. The fast decoding techniques for DSCF decoding are proposed in Section 4. The proposed simplifications are combined in Section 5 and a complexity reduction analysis and frame error rate (FER) performance comparisons are provided. Finally, conclusions and future works are drawn in Section 6.

2. BACKGROUND

2.1. Polar Codes

A polar code $PC(N, K)$ splits N channels into K reliable ones that are used to transmit the information bits, and $N-K$ unreliable ones, which are frozen to a known value (usually to 0). The set of frozen and non-frozen indices are denoted with $\mathcal{A}^C$ and $\mathcal{A}$, respectively. The encoding of a polar code is a linear transformation, such that $\boldsymbol{x} = \boldsymbol{u}\boldsymbol{G}^{\otimes n}$, where $\boldsymbol{x}$

is the encoded vector, $\boldsymbol{u}$ is the message vector, and the generator matrix $\boldsymbol{G}^{\otimes n}$ is the n-th Kronecker product ($\otimes$) of the polar code kernel $\boldsymbol{G} = \left[\begin{smallmatrix}1 & 0\\ 1 & 1\end{smallmatrix}\right]$ and $n = \log_2 N$, $n \in \mathbb{Z}^+$. The bits are estimated sequentially in SC decoding of polar codes, starting from the leftmost index. Estimation of each bit $\hat{u}_i$ depends on the channel observation $\boldsymbol{y}$ and previously decoded bits $\hat{\boldsymbol{u}}_{\boldsymbol{0:i-1}}$, such that

$$\hat{u}_i = \begin{cases} 0, & \text{if } \Pr[\boldsymbol{y}, \hat{\boldsymbol{u}}_{\boldsymbol{0:i-1}} | u_i = 0] \geq \Pr[\boldsymbol{y}, \hat{\boldsymbol{u}}_{\boldsymbol{0:i-1}} | u_i = 1]; \\ 0, & \text{if } i \in \mathcal{A}^C; \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

The decoding schedule of the SC algorithm can be interpreted as a binary tree search that starts from the root node (located at the stage $S = n$), and with priority given to the left branch. Each node contains $N_v = 2^S$ soft information, interpreted in log-likelihood ratio (LLR) form ($\boldsymbol{L^S}$) and N_v hard information ($\boldsymbol{\beta^S}$), called partial sums. The bit estimations are performed at leaf node stage $S = 0$.

It was shown in [12] and [13] that nodes in the SC decoding tree with special frozen bit patterns are not needed to be explicitly traversed; dedicated fast decoding techniques for such special nodes improves the throughput of the decoding substantially. Among these special nodes, decoding of Rate-0 (where all indices are frozen) Rate-1 (where no indices are frozen), and Rep (where only the rightmost index is non-frozen) are within the scope of this work.

2.2. SC-Flip and Dynamic SC-Flip Decoding

When the SC decoding fails, the incorrect bit estimations are either due to errors that are caused by the noisy channel or due to propagated errors as a result of the sequential decoding schedule. If a channel-induced error in a failed estimated codeword is corrected, then its associated propagated errors – if any – also disappear.

When the observation above was made in [4], it was also observed that most of the decoding failures are due to a single channel-induced error. Hence, if a single channel-induced error was avoided, then the error-correction performance would improve. Aided by an outer cyclic redundancy check (CRC) code for detecting whether an initial SC decoding has failed, SCF decoding first creates a list of bit-flipping positions sorted accordingly to their LLR magnitudes. Then, the SC decoding process is relaunched but the hard decision at the index that holds the next lowest LLR magnitude is flipped. This is repeated until no errors are detected or until all positions in the list have been considered.

There are two main problems associated with SCF decoding. The first problem is that SCF algorithm can only correct at most one single channel-induced error, so its performance improvement is limited. The second problem is that SCF relies solely on the LLR magnitudes to decide on the bit-flipping positions, which is shown to be suboptimal in [16].

DSCF decoding [11] tackles the two problems associated with the SCF decoding. It updates the set of flipping indices progressively over the course of each decoding attempt. Let $\mathcal{E}_\omega = \{i_1, \ldots, i_\omega\}$ denote the set of bit-flipping indices at an additional decoding attempt, where $i_1 < \cdots < i_\omega$ and $0 \leq \omega \leq K + C$. Note that C is the CRC remainder length. In this sense, ω is the number of attempted channel-induced errors, which is addressed as *the decoding order*. $\mathcal{E}_\omega$ is built progressively over a prior additional decoding attempt with $\mathcal{E}_{\omega-1} = \{i_1, \ldots, i_{\omega-1}\}$. The metric associated with $\mathcal{E}_\omega$ is

$$M_\alpha(\mathcal{E}_\omega) = \sum_{j \in \mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + S_\alpha(\mathcal{E}_\omega), \tag{2}$$

where

$$S_\alpha(\mathcal{E}_\omega) = \frac{1}{\alpha} \sum_{\substack{j \leq i_\omega \\ \forall j \in \mathcal{A}}} \log(1 + \exp(-\alpha |L^0[\mathcal{E}_{\omega-1}]_j|)). \tag{3}$$

In (2) and (3), α is an approximation factor that can be optimized via Monte-Carlo simulations [11] or machine learning [17], and $L^0[\mathcal{E}_{\omega-1}]_j$ is the LLR at index j of the current decoding attempt. Similar to SCF decoding, the bit-flipping set that has the lowest $M_\alpha(\mathcal{E}_\omega)$ value is used at each extra decoding attempt. DSCF demonstrates superior error-correction performance compared to SCF decoding, and its associated metric calculation is shown to be more effective in finding the erroneous indices. However, the logarithmic and exponential operations in (3) make DSCF inconvenient for efficient hardware implementations. We refer to [11, 16, 17] for acquiring in-depth knowledge on how $M_\alpha(\mathcal{E}_\omega)$ (2) is derived for the DSCF decoding. Note that the SC-Oracle (SCO) algorithm is a genie-aided algorithm (*i.e.* the message is known to the decoder) that can evaluate the ideal performance of SCF and DSCF algorithms if all the first ω channel-induced errors are successfully corrected.

3. APPROXIMATIONS FOR DSCF DECODING

When compared to SCF, DSCF decoding has significantly improved error-correction performance not only because it can correct more than one single channel-induced error, but also because it can find erroneous indices more effectively. In fact, without $S_\alpha(\mathcal{E}_\omega)$ in (2) and with $\omega = 1$, DSCF reverts to SCF decoding. Therefore, $S_\alpha(\mathcal{E}_\omega)$ can be interpreted as an adjustment function for $M_\alpha(\mathcal{E}_\omega)$ towards efficient identification of the correct bit-flipping indices. We reformulate $S_\alpha(\mathcal{E}_\omega)$ in (3) as

$$S_\alpha(\mathcal{E}_\omega) = \sum_{\substack{j \leq i_\omega \\ \forall j \in \mathcal{A}}} f_\alpha(|L^0[\mathcal{E}_{\omega-1}]_j|), \tag{4}$$

where

$$f_\alpha(x) = \frac{1}{\alpha} \log(1 + \exp(-\alpha x)). \tag{5}$$

Following the Monte-Carlo optimizations from [11], $\alpha = 0.3$ is used throughout this paper. Interestingly, it was shown that a similar expression, $f(x) = \log(1 + \exp(-x))$, used in the soft-input soft-output decoding algorithm of turbo codes, can be approximated in different ways without adversely affecting the decoding performance [18, 19]. Inspired from the

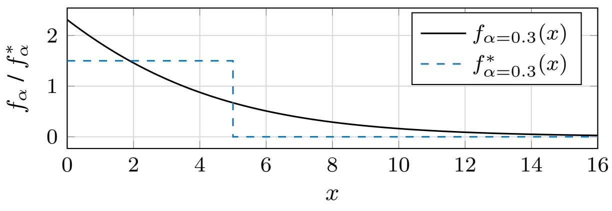

Fig. 1. $f_\alpha(x)$ with $\alpha = 0.3$, and its constant approximation $f^*_\alpha(x)$.

constant log-MAP approximation in [18], we use a similar approximation to simplify $f_\alpha(x)$ as:

$$f^*_{\alpha=0.3}(x) = \begin{cases} \frac{3}{2}, & \text{if } |x| \leq 5 \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

For illustration purpose, Fig. 1 plots the original function f_α and its proposed approximation f^*_α, with $\alpha = 0.3$. Note that in [20], a *linear approximation* to $f_\alpha(x)$ was used following [19] to reduce its complexity. However, our approach to approximate $f_\alpha(x)$ is simpler as it only involves a constant value. Fig. 2 compares the FER performance of DSCF decoding with the constant approximation (6) against its original approach from [11], using length-1024 polar codes with 3 different rates. The polar codes are constructed using the reliability sequence from [2]. The length-16 CRC defined in [2] is serially concatenated with the polar code. A BPSK modulation and an AWGN channel are considered. Note that the same settings are used for all the following Monte-Carlo simulations. Three error orders $\omega \in \{1, 2, 3\}$ are targeted, each of which corresponding to $T_{max} \in \{10, 40, 200\}$, respectively. The decoding performance with SC-Oracle for each ω value is also shown. Observe that in the considered cases, the approximated DSCF achieves similar decoding performance as the original approach but without transcendental computations, since they are replaced by a constant.

4. DECODING OF SPECIAL NODES UNDER DSCF

The purpose of identifying special nodes in the frozen bit sequence is to decode them without traversing their child nodes. Consequently, the leaf LLR magnitudes that are required to evaluate the metric for DSCF decoding in (2)-(6) are not computed. Hence, we show how to perform the metric computations for special nodes without considering them. Note that the following approaches for special node decoding can be applied to DSCF decoding with or without the approximation presented in Section 3.

Let us split the metric calculation of DSCF:

$$M_\alpha(\mathcal{E}_\omega) = \underbrace{|L^0[\mathcal{E}_{\omega-1}]_{i_w}|}_{M'_\alpha(\mathcal{E}_\omega)} + \underbrace{\sum_{j\in\mathcal{E}_{\omega-1}} |L^0[\mathcal{E}_{\omega-1}]_j| + S_\alpha(\mathcal{E}_\omega)}_{M''_\alpha(\mathcal{E}_\omega)}. \tag{7}$$

Observe that $M'_\alpha(\mathcal{E}_\omega)$ takes a value only if the index i_w is a candidate for bit-flipping during a future decoding iteration. On the other hand, $M''_\alpha(\mathcal{E}_\omega)$ is set to 0 at the beginning of any extra decoding attempt and accumulated for each leaf index j as follows:

$$\begin{aligned} M''_\alpha(\mathcal{E}_\omega)_j = M''_\alpha(\mathcal{E}_\omega)_{j-1} & \\ + |L^0[\mathcal{E}_{\omega-1}]_j| \text{ if } j \in \mathcal{E}_{\omega-1} & \\ + f_\alpha(|L^0[\mathcal{E}_{\omega-1}]_j|) \text{ if } j \in \mathcal{A}. & \end{aligned} \tag{8}$$

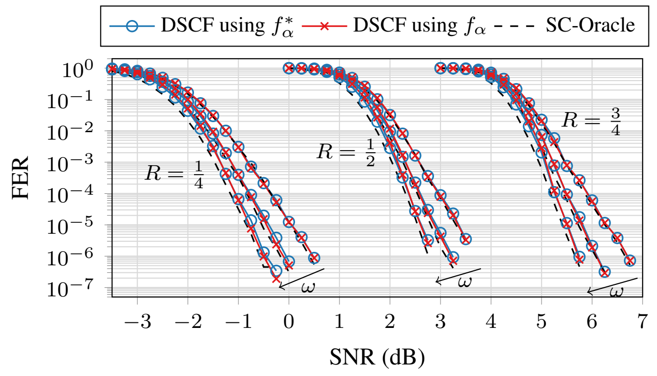

Fig. 2. FER performance comparison of DSCF decoding with and without the proposed approximation. Polar codes with $N = 1024$ and $R \in \{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}\}$, $C = 16$, $\omega \in \{1, 2, 3\}$ with $T_{max} \in \{10, 40, 200\}$.

We now provide a way to compute $M'_\alpha(\mathcal{E}_\omega)$ and $M''_\alpha(\mathcal{E}_\omega)$ when a Rep or a Rate-1 nodes are encountered during DSCF decoding. Note that the decoding of Rate-0 nodes for DSCF decoding is the same as [12] since it does not have any non-frozen indices; this has been addressed previously in [14] for SCF decoding.

Let consider the tree representation of a Rep node. By definition, only the last leaf index is non-frozen. Thus, the LLR in the last leaf node obtained through SC decoding is equal to the sum of all LLRs in the root node; and only one flipping-event is possible. Therefore, for a Rep node of size N_v, at decoding tree stage S with its first leaf index being j, we can write

$$M'_\alpha(\mathcal{E}_\omega)_{j_{\text{Rep}}} = \left|\sum_{i\in N_v} L^S[\mathcal{E}_{\omega-1}]_i\right|, \tag{9}$$

$$\begin{aligned} M''_\alpha(\mathcal{E}_\omega)_{j_{\text{Rep}}} = M''_\alpha(\mathcal{E}_\omega)_{j-1} + f_\alpha\left(\left|\sum_{i\in N_v} L^S[\mathcal{E}_{\omega-1}]_i\right|\right) & \\ + \left|\sum_{i\in N_v} L^S[\mathcal{E}_{\omega-1}]_i\right| \text{ if } j_{\text{Rep}} \in \mathcal{E}_{\omega-1}. & \end{aligned} \tag{10}$$

Thus, if the corresponding flipping event is selected during an extra decoding attempt, all the N_v partial sums at level S have to be flipped. Note that the proposed metric calculation and update for Rep nodes are exact, meaning they produce the same output under DSCF decoding.

By definition, Rate-1 nodes do not involve any frozen bits. Thus, they correspond to an uncoded sequence and all the indices at the top of the node are considered for bit-flipping. We therefore use the top-level LLRs directly in metric calculations for the prospective flipping indices. For a Rate-1 node of size N_v, at decoding tree stage S, with its first leaf index

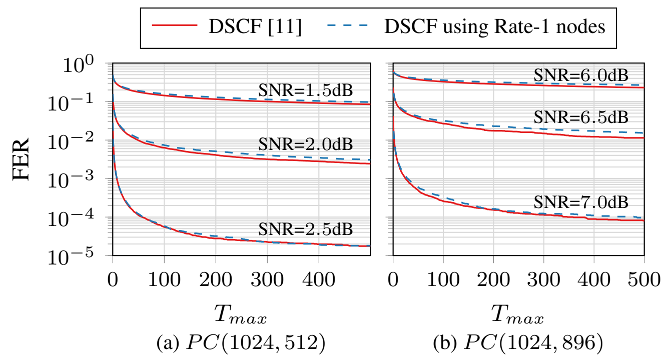

Fig. 3. FER performance of DSCF decoding with and without using Rate-1 nodes, with respect to a wide range of T_{max} values. $C = 16, \omega = 3$. $f_{\alpha=0.3}(x)$ is not approximated.

being j, $M'_\alpha(\mathcal{E}_\omega)$ is calculated for each index i $(0 \leq i < N_v)$ within the node, such that

$$M'_\alpha(\mathcal{E}_\omega)_{j+i} = |L^S[\mathcal{E}_{\omega-1}]_i| \,, \tag{11}$$

and $M''_\alpha(\mathcal{E}_\omega)$ is updated only once per node, such that

$$\begin{aligned} M''_\alpha(\mathcal{E}_\omega)_{j_{\text{Rate1}}} &= M''_\alpha(\mathcal{E}_\omega)_{j-1} \\ &+ \sum_{\substack{i \in N_v \\ \{j+i\} \in \mathcal{E}_{\omega-1}}} |L^S[\mathcal{E}_{\omega-1}]_{j+i}| + \sum_{i \in N_v} f_\alpha(|L^S[\mathcal{E}_{\omega-1}]_{j+i}|) \end{aligned} \tag{12}$$

Unlike in Rep nodes, the proposed metric for Rate-1 nodes is not exact. Indeed, flipping one hard decision at the leaf side of a Rate-1 node may correspond to multiple flips in the partial sums found at its root. In order to validate our approach for Rate-1 nodes in DSCF decoding, Fig. 3 depicts how the FER evolves with T_{max} and three different SNR values for $PC(1024, 512)$ and $PC(1024, 896)$, with $\omega = 3$. The frozen set associated with these polar codes exhibit 46% and 87% indices that fall under Rate-1 nodes. It can be seen that the error-correction performance with Rate-1 nodes is similar to that of the original DSCF algorithm, while computations are saved since the decoding tree is pruned.

5. COMBINATION OF THE SIMPLIFICATIONS

We now combine the simplifications proposed in the two previous sections and discuss some numerical analysis.

Fig. 4 presents the error correction performance of DSCF decoding using the constant approximation from (6) and the special node decoding techniques from (9)-(12), against baseline DSCF decoding from [11] and CRC-aided SCL (CA-SCL) decoding with $L \in \{4, 8, 16\}$. DSCF decoders are simulated with $\omega \in \{2, 3\}$ with T_{max} of 40 and 200, respectively. According to Fig. 4, DSCF decoding with the proposed simplifications has a similar FER performance to the original DSCF algorithm, and the loss is negligibly small even at $\omega = 3$. Proposed DSCF with $\omega = 2$ and $\omega = 3$ is able to outperform CA-SCL with $L = 4$ and $L = 8$, respectively.

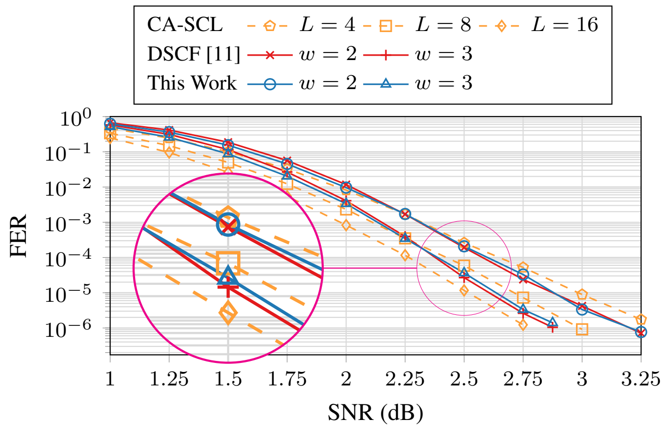

Fig. 4. FER comparison of the proposed simplified DSCF decoding against baseline DSCF and CA-SCL decoders. $PC(1024, 512), C = 16$.

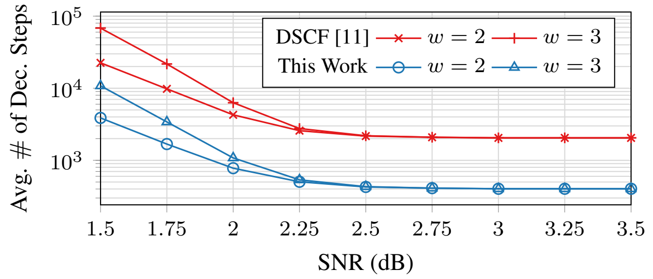

Fig. 5. Comparison of average number of decoding steps for proposed simplified DSCF decoding and baseline DSCF decoding. $PC(1024, 512), C = 16$.

Fig. 5 shows the average number of decoding steps under DSCF decoding, with and without the proposed simplifications. One decoding step follows the definition in [1]. The proposed special node decoding techniques reduce the average number of steps for DSCF decoding significantly, by up to 6.4×. Moreover the transcendental operations are not required anymore.

6. CONCLUSION AND FUTURE WORK

We proposed a simple approximation to replace the logarithmic and exponential computations of the DSCF decoding. The proposed approximation allows to reduce the computational effort of DSCF tremendously and its hardware implementation can be considered. In addition, we showed how to incorporate fast decoding techniques for DSCF decoding. We showed that all the proposed approximations and simplifications can be applied to DSCF decoding independently, and they do not alter its error-correction performance significantly. The proposed fast decoding techniques for DSCF has reduced the computational complexity by up to 6.4×. Future research directions involve decoding of more sophisticated special node patterns under DSCF decoding and hardware implementations with the proposed simplifications.

7. REFERENCES

[1] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, July 2009.

[2] 3GPP, "NR; Multiplexing and Channel Coding," http://www.3gpp.org/DynaReport/38-series.htm, Tech. Rep. TS 38.212, January 2018, Release 15.

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

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

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

[6] F. Ercan, C. Condo, S. A. Hashemi, and W. J. Gross, "On error-correction performance and implementation of polar code list decoders for 5G," in *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Oct 2017, pp. 443–449.

[7] K. Chen, K. Niu, and J. Lin, "Improved successive cancellation decoding of polar codes," *IEEE Transactions on Communications*, vol. 61, no. 8, pp. 3100–3107, August 2013.

[8] F. Ercan, C. Condo, S. A. Hashemi, and W. J. Gross, "Partitioned successive-cancellation flip decoding of polar codes," in *2018 IEEE International Conference on Communications (ICC)*, May 2018, pp. 1–6.

[9] C. Condo, F. Ercan, and W. J. Gross, "Improved successive cancellation flip decoding of polar codes based on error distribution," in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, April 2018, pp. 19–24.

[10] F. Ercan, C. Condo, and W. J. Gross, "Improved bit-flipping algorithm for successive cancellation decoding of polar codes," *IEEE Transactions on Communications*, vol. 67, no. 1, pp. 61–72, Jan 2019.

[11] L. Chandesris, V. Savin, and D. Declercq, "Dynamic-SCFlip decoding of polar codes," *IEEE Transactions on Communications*, vol. 66, no. 6, pp. 2333–2345, June 2018.

[12] A. Alamdar-Yazdi and F. R. Kschischang, "A simplified successive-cancellation decoder for polar codes," *IEEE Communications Letters*, vol. 15, no. 12, pp. 1378–1380, December 2011.

[13] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, "Fast polar decoders: Algorithm and implementation," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 946–957, May 2014.

[14] P. Giard and A. Burg, "Fast-SSC-flip decoding of polar codes," in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, April 2018, pp. 73–77.

[15] F. Ercan, T. Tonnellier, and W. J. Gross, "Energy-efficient hardware architectures for fast polar decoders," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 1, pp. 322–335, Jan 2020.

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

[17] N. Doan, S. A. Hashemi, F. Ercan, T. Tonnellier, and W. J. Gross, "Neural dynamic successive cancellation flip decoding of polar codes," *arXiv e-prints*, p. arXiv:1907.11563, Jul 2019.

[18] W. J. Gross and P. G. Gulak, "Simplified MAP algorithm suitable for implementation of turbo decoders," *Electronics Letters*, vol. 34, no. 16, pp. 1577–1578, Aug 1998.

[19] Jung-Fu Cheng and T. Ottosson, "Linearly approximated log-MAP algorithms for turbo decoding," in *VTC2000-Spring. 2000 IEEE 51st Vehicular Technology Conference Proceedings*, vol. 3, May 2000, pp. 2252–2256.

[20] Y. Zhou, J. Lin, and Z. Wang, "Improved fast-SSC-flip decoding of polar codes," *IEEE Communications Letters*, vol. 23, no. 6, pp. 950–953, June 2019.