IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 18, 2018, accepted December 27, 2018, date of publication January 10, 2019, date of current version February 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2891951

Belief Propagation Bit-Flip Decoder for Polar Codes

YONGRUN YU, ZHIWEN PAN, NAN LIU, AND XIAOHU YOU, (Fellow, IEEE)
National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China

Corresponding author: Zhiwen Pan (pzw@seu.edu.cn)

This work was supported in part by the national major project under Grant 2017ZX03001002-004 and in part by the 333 Program of Jiangsu under Grant BRA2017366.

ABSTRACT The bit-flip method has been successfully applied to the successive cancellation (SC) decoder to improve the block error rate (BLER) performance for polar codes in the finite code length region. However, due to the sequential decoding, the SC decoder inherently suffers from longer decoding latency than that of the belief propagation (BP) decoder with efficient early stopping criterion. It is natural to ask how to perform bit-flip in a polar BP decoder. In this paper, bit-flip is introduced into the BP decoder for polar codes. The idea of critical set (CS), that is, originally proposed by Zhang *et al.* for identifying unreliable bits in a SC bit-flip decoder, is extended to the BP decoder here. After revealing the relationship between CS and the incorrect BP decoding results, critical set with order ω (CS-ω) is constructed to identify unreliable bit decisions in polar BP decoding. The simulation results demonstrate that compared with the conventional BP decoder, the BLER of the proposed bit-flip decoder can achieve significant signal-to-noise ratio (SNR) gain which is comparable to that of a cyclic redundancy check-aided SC list decoder with a moderate list size. In addition, the decoding latency of the proposed BP bit-flip decoder is only slightly higher than that of the conventional BP decoder in the medium and high SNR regions.

INDEX TERMS Polar codes, belief propagation, critical set, bit-flip.

I. INTRODUCTION

With infinite code length, polar codes achieve the capacity of binary-input memoryless output-symmetric channels under the successive cancellation (SC) decoding [3]. However, the block error rate (BLER) performance of polar codes under the SC decoding with finite length is inferior to that of low-density parity-check (LDPC) codes and Turbo codes [4], [5]. Therefore, cyclic redundancy check (CRC) aided SC list (CA-SCL) decoder [6] is introduced to improve the BLER of polar codes in practical code length region. The CA-SCL decoder that almost approaches the maximum likelihood decoding performance has long been considered as the decoding scheme that has the best BLER among practical polar decoding algorithms.

To reduce the complexity of the SCL decoder, in recent works [1], [2], [7], [8], it is shown that the SC bit-flip (SCF) decoder is able to yield the same BLER as the CA-SCL decoder, and the complexity of SCF decoding approaches that of the SC decoding in high signal-to-noise ratio (SNR) region. Works in [9] further reduce the complexity of the SCF decoder based on fast decoding for some constituent nodes. Despite their superior BLER, above SC-based decoding algorithms are still sequential in nature, which causes difficulty in parallel decoding. As contrary, the belief propagation (BP) decoding of polar codes is inherently parallel [10], [11]. It is known that for polar codes, the BLER of the conventional BP decoding is similar to that of the SC decoding [4], [5]. Therefore, the BLER of the BP decoding is not satisfying and needs to be improved. Yuan and Parhi [12] and Cammerer *et al.* [13] propose hybrid BP-SC decoding schemes, where the BP decoding is followed by the SC decoding. If the result of BP decoding is invalid, the denoised log-likelihood ratios (LLRs) are passed to the SC decoder. Post-processing algorithms are proposed in [14], where the error types of the BP decoding are classified and the perturbation-based method is employed to deal with different type of BP error. Permuted factor graph is introduced in [15] and [16], based on which BP list (BPL) decoder is proposed [17]. The idea behind BPL decoder is that when the standard polar code factor graph fails to produce the correct decoding result, the permuted version of the standard graph may yield the correct estimate because the message passing process is altered. However, even with above enhanced methods, the BLER of BP decoder is still not comparable to that

2169-3536 © 2019 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

of the CA-SCL decoder. This work aims to design a polar BP decoder that has the similar BLER to that of the CA-SCL decoder with moderate list size. The main contributions of this paper are summarized as follows.

(I) Inspired by the SCF decoder, we introduce bit-flip into polar BP decoder. In this paper, CRC-polar concatenated codes are used. If the CRC is not satisfied when the maximum number of BP iteration is reached, then the bit-flip decoding is activated. Bit-flip means that the priori knowledge of the flipped information bit is set to infinity, i.e., if u_i is flipped to 0, then the so-called R message of u_i is set to plus infinity, and vice versa. Note that in LDPC codes, there exist bit saturation methods [18]–[20] that set the initial channel LLRs of variable nodes to infinity for better BLER performance. However, in this paper, we flip the priori knowledge of information bits, not that of variable nodes.

(II) The behavior of the incorrect BP decoding results is analyzed. It is found that in the BP decoding, although information bits are treated parallel, i.e., they are estimated simultaneously, when the BP decoding fails, they exhibit similar behavior to that of the SC decoding: flipping the first incorrect decision of information bit may yield the correct decoding result.

(III) Based on the critical set that is originally proposed for SCF decoder [1], [2], critical set with order ω (CS-ω) is introduced in this paper to indicate which bits should be flipped if the original BP decoding fails. CS-ω is a truncated version of the modified critical set (MCS) [1], [2] in the sense that CS-ω consists of the most error-prone elements in MCS (the details are in Section IV). Compared with the MCS, CS-ω results in less attempt bit-flip decoding and therefore achieves lower latency. In addition, based on the observations in [1] and [2], we further explain why critical set is powerful in identifying unreliable bits.

Simulation results demonstrate that the BLER of the proposed BP bit-flip (BPF) decoder can achieve significant SNR gains compared with that of the standard BP decoding. For example, there is 1.1 dB gain at BLER $= 10^{-4}$ when decoding polar codes with length 2048 and rate 0.5. In addition, the decoding latency is analyzed by both upper bound and simulations.

The rest of this paper is organized as follows. Section II provides preliminaries of polar codes, critical set, and the BP decoding. Key observations when the BP decoding fails and a genie-aided BPF decoder are illustrated in Section III. The bit-flip BP decoder using CS-ω is provided in Section IV. Simulation results are presented in Section V. Conclusions are drawn in Section VI.

II. PRELIMINARIES

A. POLAR CODES

Polar codes are linear block codes with generator matrix $\boldsymbol{G}_N = \boldsymbol{B}_N\boldsymbol{F}^{\otimes n}$, where $N = 2^n$ denotes code length, $\boldsymbol{B}_N$ denotes the bit-reversal permutation matrix [3], and $\boldsymbol{F} = \left[\begin{smallmatrix}1 & 0\\ 1 & 1\end{smallmatrix}\right]$. The encoding process can be expressed as

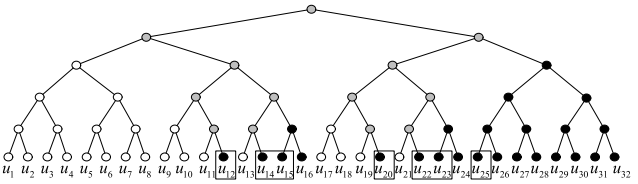


FIGURE 1. An example of CS with $P(32, 16)$ under GA construction at $E_b/N_0 = 2.5$ dB.

$\boldsymbol{x}_1^N = \boldsymbol{u}_1^N\boldsymbol{G}_N$, where $\boldsymbol{x}_1^N = (x_1, x_2, \ldots, x_N)$ represents the codeword and $\boldsymbol{u}_1^N = (u_1, u_2, \ldots, u_N)$ denotes the source vector. $\boldsymbol{u}_1^N$ includes both information and frozen bits. The index sets of information and frozen bits are $\mathcal{A}$ and $\mathcal{A}^c$, respectively. $\mathcal{A}\cap\mathcal{A}^c = \oslash$ and $\mathcal{A}\cup\mathcal{A}^c = \{1, 2, \ldots, N\}$. $P(N, K)$ represents polar codes with length N and K information bits. $P(N, K+r)$ denotes that the $K + r$ unfrozen bits of polar codes include r bits CRC.

In this paper, frozen bits are fixed to zero, and $\mathcal{A}$ is constructed by Gaussian approximation (GA) [21]. In GA, under the assumption that all-zero codeword is transmitted in the additive white Gaussian noise (AWGN) channel, the expectation of LLRs evolves according to equations (5) and (6) in [21]. Denote $E\left\{L_N^{(i)}\right\}$ the expectation of LLR of the i-th polarized channel, and the error rate of the i-th polarized channel can be expressed as follows:

$$P_e(u_i) = Q\left(\sqrt{E\left\{L_N^{(i)}\right\}/2}\right), \tag{1}$$

where $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^{+\infty} e^{-\frac{\alpha^2}{2}}d\alpha$. The indices of K polarized channels with the K largest $E\left\{L_N^{(i)}\right\}$ (corresponding to the K smallest $P_e(u_i)$) are selected to form $\mathcal{A}$, where K is the number of information bits. Note that although so far there is no analytical code construction method for the BP decoder, polar codes constructed for the SC decoding also work well under the BP decoding.

B. CRITICAL SET

Polar code structure can be represented by a full binary tree. As shown in Figure 1, black node means that all its leaf nodes are information bits, while white node means that all its leaf nodes are frozen bits. Grey node denotes that its leaf nodes include both information and frozen bits. Black nodes are referred to as rate 1 nodes [22]. The critical set (CS) takes the index of the first bit in each rate 1 node [1], [2]:

$$\mathrm{CS} = \bigcup_i \mathcal{R}_i[1], \tag{2}$$

where $\mathcal{R}_i$ denotes the i-th rate 1 node, and $\mathcal{R}_i[1]$ denotes the index of the first bit in $\mathcal{R}_i$. For example in Figure 1, there are seven rate 1 nodes and CS $= \{12, 14, 15, 20, 22, 23, 25\}$.

Information bits in the CS tend to be unreliable [1], [2]. To facilitate clear understanding, numerical results of P_e^{SC} and P_e^{CS} are provided in Figure 2. $P_e^{\mathrm{SC}} = 1 - \prod_{i\in\mathcal{A}}(1 - P_e(u_i))$ denotes the BLER of SC decoding under GA [23], while $P_e^{\mathrm{CS}} = 1 - \prod_{i\in\mathrm{CS}}(1 - P_e(u_i))$ only considers information bits

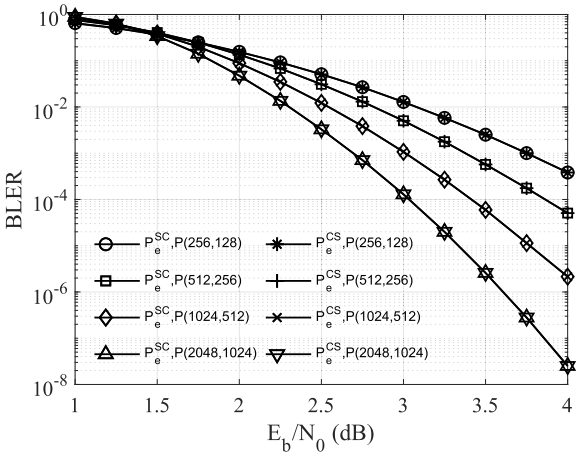


FIGURE 2. Numerical results of P_e^{SC} and P_e^{CS} under GA.

in the CS. Obviously $P_e^{\mathrm{SC}} > P_e^{\mathrm{CS}}$. However, it can be observed that the values of P_e^{SC} and P_e^{CS} coincide, which indicates that bits in the CS dominate the error probability.

In the next section, we also provide further explanations on the CS, and reveal that CS can be employed to identify unreliable bits in polar BP decoding. For more details about CS, readers may refer to [1] and [2].

C. BELIEF PROPAGATION DECODING FOR POLAR CODES

Polar code factor graph is shown in Figure 3, where $N = 2^3$ and pair (a, b) denotes the row and column number. There are three stages in Figure 3 and each stage consists of four processing elements (PEs). The structure of PE is shown in Figure 4. One PE has four nodes and each node is associated with two types of messages, i.e., the right-to-left message L and left-to-right message R. In this paper, L and R are in the form of LLR. The message propagation rules are as follows:

$$\begin{aligned}
L_{2j-1,i} &= g(R_{2j,i} + L_{\gamma(i,j)+2^{n-i},i+1}, L_{\gamma(i,j),i+1}),\\
L_{2j,i} &= g(R_{2j-1,i}, L_{\gamma(i,j),i+1}) + L_{\gamma(i,j)+2^{n-i},i+1},\\
R_{\gamma(i,j),i+1} &= g(R_{2j,i} + L_{\gamma(i,j)+2^{n-i},i+1}, R_{2j-1,i}),\\
R_{\gamma(i,j)+2^{n-i},i+1} &= g(R_{2j-1,i}, L_{\gamma(i,j),i+1}) + R_{2j,i},
\end{aligned} \tag{3}$$

where $g(x, y) = 0.9375\mathrm{sign}(x)\mathrm{sign}(y)\min\{|x|, |y|\}$, and 0.9375 follows from the scaling factor used in [24]. $\gamma(i, j)$ in fact represents the connections of PEs shown in Figure 3, and $\gamma(i, j) = 1 + \mathrm{mod}(j - 1, 2^{n-i}) + \lfloor \frac{j-1}{2^{n-i}} \rfloor 2^{n-i+1}$,[1] where i denotes the stage number and j denotes the number of PE in a stage counting from top to bottom.

Denote $\boldsymbol{L}_{N\times(n+1)}$ and $\boldsymbol{R}_{N\times(n+1)}$ the two matrices that store L and R messages. Before the iterative decoding, $\boldsymbol{L}$ and $\boldsymbol{R}$ are initialized as follows:

$$L_{i,j} = \begin{cases} 0, & j \neq n+1 \\ LLR_i & j = n+1, \end{cases} \tag{4}$$

$$R_{i,j} = \begin{cases} 0, & j \neq 1 \text{ or } j = 1, \quad i \in \mathcal{A} \\ +\infty, & j = 1, \quad i \in \mathcal{A}^c, \end{cases} \tag{5}$$

[1]In this work, polar code factor graph is strictly the same as $\boldsymbol{G}_N = \boldsymbol{B}_N\boldsymbol{F}^{\otimes n}$ [3], so the decoding index is different from the pipelined form [11].

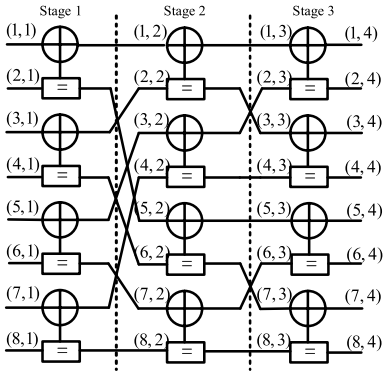


FIGURE 3. Factor graph of polar codes with length N=8.

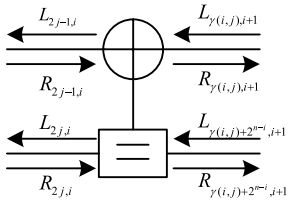


FIGURE 4. PE in BP decoding of polar codes.

where LLR_i denotes the LLR of the i-th received bit. The $+\infty$ in the first column of $\boldsymbol{R}$ indicates the priori knowledge carried by frozen bits. The estimation of the source vector $\hat{\boldsymbol{u}}_1^N = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_N)$ and codeword $\hat{\boldsymbol{x}}_1^N = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_N)$ can be obtained through (6) and (7), respectively.

$$\hat{u}_i = \begin{cases} 0, & L_{i,1} + R_{i,1} \geq 0 \\ 1, & L_{i,1} + R_{i,1} < 0, \end{cases} \tag{6}$$

$$\hat{x}_i = \begin{cases} 0, & L_{i,n+1} + R_{i,n+1} \geq 0 \\ 1, & L_{i,n+1} + R_{i,n+1} < 0. \end{cases} \tag{7}$$

In this paper, the bit-flip of information bit u_i means that the priori knowledge of u_i is set to infinity, i.e., if u_i is flipped to 1, then we set $R_{i,1} = -\infty$ as if u_i is frozen. With this method, the priori knowledge of flipped bits will propagate in the factor graph, which is expected to correct the wrongly propagated messages in the original yet failed BP decoding.

III. ANALYSIS OF THE INCORRECT BP DECODING RESULTS AND CRITICAL SET

In the first part of this section, we analyze the behavior of the incorrect BP decoding results, and then a genie-aided bit-flip decoder is designed to confirm that if we flip certain information bit, the BLER of the BP decoder will improve. In the second part, we provide further explanations on why the critical set can identify unreliable information bits. Then, by simulations, it is revealed that the critical set includes the first error bit with high probability under the BP decoding.

TABLE 1. Simulation configurations.

Code construction	GA at $E_b/N_0 = 2.5$ dB
Channel	Additive white Gaussian noise
Modulation	Binary phase shift keying
Code rate	$R = 0.5$
Early stop criterion	G-matrix method [24]
Maximum iteration	100

A. BEHAVIOR OF THE INCORRECT BP DECODING RESULTS

The behavior of the incorrect BP decoding results is analyzed by Monte-Carlo simulations. When the conventional BP decoding of polar codes fails,[2] the first index and whole indices where $\hat{\boldsymbol{u}}_1^N \neq \boldsymbol{u}_1^N$ are represented by i^* and $\mathcal{E}$, respectively:

$$i^* = \arg\min_{i\in\mathcal{A}} \left\{ i | \hat{u}_i \neq u_i \right\}, \tag{8}$$

$$\mathcal{E} = \left\{ i | \hat{u}_i \neq u_i, i \in \mathcal{A} \right\}. \tag{9}$$

Note that although $P_e(u_i)$ in (1) under GA is derived from the SC perspective, polar codes constructed by $P_e(u_i)$ also work well under the BP decoder. This implies some similarity between the SC and BP decoding, i.e., the first error bit u_{i^*} tends to be error-prone. We hence observe the following two behaviors of the incorrect BP decoding results.

Behavior 1: $P_e(u_{i^*})$ tends to be the largest one among $P_e(u_i), i \in \mathcal{E}$. Here, $P_e(u_i), i \in \mathcal{E}$ follows from (1).

This can be observed by the following simulation results. Denote $\mathcal{B}$ the event that $P_e(u_{i^*})$ is the largest one among $P_e(u_i), i \in \mathcal{E}$:

$$\mathcal{B} : \arg\max_{i\in\mathcal{E}} \{P_e(u_i)\} \text{ is } i^*. \tag{10}$$

The frequency of event $\mathcal{B}$ is provided in Table 2. The simulation configurations used in Table 2 are summarized in Table 1. In a wide code length range, it can be seen that with the increase of SNR, the frequency of event $\mathcal{B}$ also increases and eventually approaches 90%. Such results clearly show that $P_e(u_{i^*})$ tends to be the largest one among $P_e(u_i), i \in \mathcal{E}$, especially in medium and high SNR range.

Behavior 2: After the original BP decoding fails with the received signal $\boldsymbol{y}_1^N = (y_1, \ldots, y_N)$, if u_{i^*} can be flipped to the true value and an additional BP decoding is then performed, the BP decoder can yield the correct decoding result using the same $\boldsymbol{y}_1^N = (y_1, \ldots, y_N)$.

We use a genie-aided manner to show above behavior. The genie-aided manner means that the decoder knows which information bit is u_{i^*} when the original BP decoding fails with the received signal $\boldsymbol{y}_1^N$. Then, bit-flip BP decoding is activated using the same $\boldsymbol{y}_1^N$. In the bit-flip decoding, the genie-aided decoder sets the priori information of u_{i^*} to infinity as if u_{i^*} is frozen:

$$R_{i^*,1} = \begin{cases} +\infty, & \text{the true value of } u_{i^*} \text{ is 0,} \\ -\infty, & \text{the true value of } u_{i^*} \text{ is 1.} \end{cases} \tag{11}$$

The BLER of such genie-aided decoder is presented in Figure 5. With a wide range of code length, it can be observed that compared with the conventional BP decoding, the BLER can be improved by 0.5dB in almost the whole simulated SNR region. In addition, the BLER improvement increases as SNR increases. Such phenomenon coincides with data in Table 2, i.e., in low SNR region, the frequency of event $\mathcal{B}$ is relatively small, which causes inefficient bit-flip. In medium and high SNR regime, the frequency of event $\mathcal{B}$ is large, yielding efficient bit-flip. The BLER improvement in Figure 5 shows that even if only one information bit is flipped, the BLER improvement can be significant.

[2]The failure of the BP decoding can be detected by some early stopping criterion, such as G-matrix method [24] or CRC check. G-matrix method tests if $\hat{\boldsymbol{x}}_1^N$ equals $\hat{\boldsymbol{u}}_1^N \boldsymbol{G}_N$. If $\hat{\boldsymbol{x}}_1^N$ equals $\hat{\boldsymbol{u}}_1^N \boldsymbol{G}_N$, the BP decoding stops.

TABLE 2. The frequency (%) of event $\mathcal{B}$ under different SNRs and code lengths with simulation runs 10^6.

SNR(dB)	1.5	2	2.5	3	3.5
$P(256, 128)$	32.88	41.23	51.17	66.54	76.07
$P(512, 256)$	44.39	63.55	82.14	86.92	92.52
$P(1024, 512)$	33.18	67.76	75.70	85.51	91.59
$P(2048, 1024)$	21.03	48.60	61.68	79.91	84.11

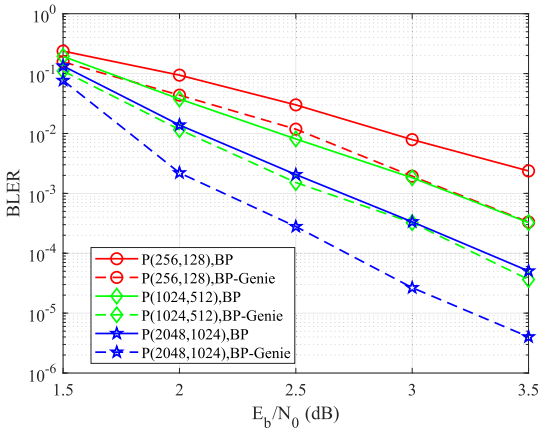


FIGURE 5. The BLER performance comparison between the conventional BP decoder and the genie-aided BP bit-flip decoder.

B. FURTHER EXPLANATIONS ON CRITICAL SET

In [1] and [2], information bits in CS are shown to be error-prone under binary erasure channel and AWGN channel. In fact, there exist more general theoretical explanations. As a result of the recursive structure of polar codes, each rate 1 node consisting of N_{r1} information bits can be considered as a sub-polar code, where N_{r1} is power of 2. Inside such sub-polar code there are N_{r1} polarized channels, counting from 0 to $N_{\text{r1}} - 1$. According to the partial order of polarized channels [25], [26], the i-th polarized channel is degraded to the j-th polarized channel if the binary expansions of i and j

TABLE 3. The values of α and |CS| under different codes, all codes are constructed by GA at $E_b/N_0 = 2.5$ dB.

Codes	α	|CS|	α/|CS|
$P(1024, 512)$	120	124	96.77%
$P(2048, 1024)$	214	220	97.27%
$P(4096, 2048)$	404	416	97.12%

TABLE 4. The frequency (%) of event $\mathcal{Q}$ under different SNRs and code lengths with simulation runs 10^6.

SNR(dB)	1.5	2	2.5	3	3.5
$P(256, 128)$	96.89	99.03	97.08	97.47	98.44
$P(512, 256)$	97.20	98.21	99.53	97.66	99.53
$P(1024, 512)$	98.13	100	100	100	100
$P(2048, 1024)$	98.13	98.93	99.07	98.60	100

satisfy:

$$i = (i_{n_{\mathrm{r1}}}, \ldots, i_p, \ldots, i_1)_2, \quad j = (j_{n_{\mathrm{r1}}}, \ldots, j_p, \ldots, j_1)_2,$$
$$i_p = 0, j_p = 1, \quad i_k = j_k, k \neq p, \tag{12}$$

where $(i_{n_{\mathrm{r1}}}, \ldots, i_p, \ldots, i_1)_2$ and $(j_{n_{\mathrm{r1}}}, \ldots, j_p, \ldots, j_1)_2$ are the binary expansions of i and j, respectively, and $n_{\mathrm{r1}} = \log_2 N_{\mathrm{r1}}$. The first bit inside the rate 1 sub-polar code corresponds to the 0-th polarized channel whose binary expansion consists of all zeros. Obviously through (12) we can see that all the other polarized channels is upgraded to the 0-th one. In other words, polar codes convey information using rate 1 sub-codes, and CS selects the most degraded polarized channel in each rate 1 node.

It should be noted that although CS does not necessarily include all the |CS| bits in $\mathcal{A}$ with |CS| largest $P_e(u_i)$, where |CS| is the number of bits in the CS, CS can cover most of them. Assume that all the |CS| information bits with largest $P_e(u_i)$ form the set $\mathcal{D}$, and α represents that the CS includes α bits in $\mathcal{D}$. Table 3 presents the values of α and |CS| under different code configurations. It can be observed that α almost approaches |CS|, which indicates the CS is a powerful tool to identify unreliable bits. Note that there exists no explicit relationship between the first bit in one rate 1 node and the second bit in another rate 1 node.

In the end of this section, the relationship between the first error index i^* and CS is revealed. It is shown by simulations that the CS includes i^* with high probability under the BP decoding. Let event $\mathcal{Q}$ denote that i^* is included by the CS: $i^* \in$CS. The frequency of event $\mathcal{Q}$ is large, and the corresponding results are provided in Table 4, where simulation parameters are the same as in Table 1. The results in Table 4 imply that without the help of a genie, the CS can serve as the set that includes i^* with high probability under the BP decoding. Note that the results in Table 4 coincide well with [1, Table 1] and [2, Tables 1 and 2] for SC decoding. Therefore, in the next section, the bit-flip method for polar BP decoder is proposed based on flipping bits in the CS.

IV. BIT-FLIP BP DECODER

In this section, critical set with order ω (CS-ω) is proposed to identify ω error-prone bits that need to be flipped in one BP decoding turn. We first describe a general case where ω bits can be identified and corrected in one attempt BP decoding, and then take the one-bit-flip decoder as a special case. The decoding latency of the proposed decoder with CS-ω is analyzed by upper bound.

A. THE CONSTRUCTION OF CS-ω

CS-1 with one-bit-flip is one-dimensional vector that is the same as CS. Therefore, as shown in Table 4, CS-1 includes i^* with high probability and can be used to indicate which single bit should be flipped. When $\omega \geq 2$, we must choose ω indices that include ω error-prone bits. This can be achieved by constructing new critical sets progressively as proposed in [1] and [2], and then the MCS is obtained. However, the MCS includes massive (approximately $|\mathrm{CS}|^{\omega}$) error-prone indices that are required to be tested by the bit-flip decoding [1], [2]. Therefore, to avoid the exponential attempt bit-flip decoding caused by the MCS and obtain lower decoding latency, we propose CS-ω, a truncated version of the MCS, which includes the most error-prone elements in the MCS.

CS-ω is a $|\mathrm{CS}|\times\omega$ matrix, and one row of CS-ω represents the ω indices of the ω flipped bits in one attempt BP decoding. CS-ω has |CS| rows, which indicates that CS-ω has the same number of elements as the CS. Each row in the CS-ω can be expressed as $(j_1, j_2, \ldots, j_\omega)$, where $j_k \in \mathcal{A}$, $1 \leq k \leq \omega$. Each row of the CS-ω is obtained as follows.

First, sort elements in the CS in descending order according to the error rate $P_e(u_i)$, $i \in$ CS:

$$\begin{aligned} &\left\{\mathrm{CS}(k_1), \mathrm{CS}(k_2), \ldots, \mathrm{CS}(k_{|\mathrm{CS}|})\right\}, \\ &\text{s.t. } P_e(u_{\mathrm{CS}(k_i)}) \geq P_e(u_{\mathrm{CS}(k_{i+1})}). \end{aligned} \tag{13}$$

Next, the i-th row in the CS-ω is expressed as follows:

$$\begin{aligned} &\text{CS-}\omega(i) = (j_1, j_2, \ldots, j_\omega), \quad 1 \leq i \leq |\mathrm{CS}| \\ &j_1 = \mathrm{CS}(k_i), \\ &j_2, j_3, \ldots, j_\omega \in \mathcal{A}, \\ &j_1 < j_2 < \ldots < j_\omega. \end{aligned} \tag{14}$$

$j_p, 2 \leq p \leq \omega$ in (14) is obtained through the following three steps.

(a) Freeze bit indices from 1 to j_{p-1} and get a new frozen set $\mathcal{A}^c_{p-1}$. The reason of this step is that we assume $\left\{u_{j_1}, \ldots, u_{j_{p-1}}\right\}$ are flipped to the correct values, and hence there are no errors in $\left\{u_1, \ldots, u_{j_{p-1}}\right\}$. Since $\left\{u_1, \ldots, u_{j_{p-1}}\right\}$ are assumed correct, they are considered as frozen bits. This step is the same as the progressively frozen method [1], [2].

(b) Use $\mathcal{A}^c_{p-1}$ to construct a new critical set CS_{p-1}. According to the analysis in Table 4, assuming the former bits $\left\{u_1, \ldots, u_{j_{p-1}}\right\}$ are correct, CS_{p-1} includes the first error index i^* after $u_{j_{p-1}}$ with high probability. Find the index $k \in \mathrm{CS}_{p-1}$ such that:

$$k = \underset{i \in \mathrm{CS}_{p-1}}{\arg\max} \left\{P_e(u_i)\right\}. \tag{15}$$

(c) Set $j_p = k$, i.e., select the most error-prone bit in CS_{p-1} as the p-th flip index and discard the other indices in CS_{p-1}. Note that the MCS [1], [2] selects every bit in CS_{p-1},

Algorithm 1 Construct CS-ω

Input: $\mathcal{A}^c$, ω, $\{P_e(u_i)\}_{i\in\mathcal{A}}$
Output: CS-ω

1: construct CS by (2) using $\mathcal{A}^c$
2: $M \leftarrow$ the number of elements in the CS
3: CS-$\omega \leftarrow$ an $M \times \omega$ matrix. //the row denotes flip index
4: **for** $q = 1 : M$ **do**
5: CS-$\omega(q, 1) \leftarrow$ CS(k_q) //CS is sorted in (13) off-line
6: **for** $p = 2 : \omega$ **do**
7: $\mathcal{A}^c_{p-1} \leftarrow$ freeze indices from 1 to CS-$\omega(q, p-1)$
8: $\mathrm{CS}_{p-1} \leftarrow$ construct critical set by (2) using $\mathcal{A}^c_{p-1}$
9: CS-$\omega(q, p) \leftarrow \arg\max_{i\in\mathrm{CS}_{p-1}} \{P_e(u_i)\}$
10: **end for**
11: **end for**

which makes the number of elements in the MCS increase exponentially with the increase of ω. Therefore, CS-ω can be considered as a truncated version of the MCS.

Note that above operations do not require online data, while in [1] and [2], online LLRs are needed to sort elements in the MCS to find the currently most unreliable bit. The construction of the CS-ω is summarized in Algorithm 1.

B. BIT-FLIP BP DECODER USING CS-ω

In this subsection, the bit-flip BP decoder using CS-ω is proposed. In general, multi-bit-flip in BP decoding is more difficult than that in SC decoding. The reason is as follows. In the SC decoding with ω flipped bits, once $(u_{j_1}, u_{j_2}, \ldots, u_{j_\omega})$ is selected as the target bits, the flipped values of $(u_{j_1}, u_{j_2}, \ldots, u_{j_\omega})$ are determined by the opposite direction of their LLRs $(L_N^{(j_1)}, L_N^{(j_2)}, \ldots, L_N^{(j_\omega)})$ that are successively calculated in the SC process. In other words, we do not need to set the priori knowledge for $(u_{j_1}, u_{j_2}, \ldots, u_{j_\omega})$ in the SC bit-flip decoding.

However, in the BP decoding with the proposed bit-flip method, it is necessary to set the priori knowledge for the target bits. Once the wrong priori knowledge is used, the BP decoder will never yield the correct decoding result. Therefore, we exhaustively enumerate all the possible values of the target bits. Specifically, for a given row $(j_1, j_2, \ldots, j_\omega)$ in CS-ω, we exhaustively enumerate all the possible values of the priori knowledge $(R_{j_1,1}, R_{j_2,1}, \ldots, R_{j_\omega,1}) \in \{+\infty, -\infty\}^\omega$ for $(u_{j_1}, u_{j_2}, \ldots, u_{j_\omega})$, i.e., each row in CS-$\omega$ may involve 2^ω times of attempt bit-flip decoding. Thus, the worst case number of bit-flip decoding is $T_{\max} = 2^\omega \times |\mathrm{CS}|$. Although at first sight $T_{\max}$ is quite large, simulation results demonstrate that the average decoding latency of the above flipping method is similar to that of the conventional BP decoding in medium and high SNR regions.

The proposed bit-flip BP decoder using CS-ω is summarized in Algorithm 2, where CRC-polar concatenated codes are employed. The **while** in line 4 means that each row in CS-ω is tested. The **for** in line 6 means that all the possible values of the priori knowledge

Algorithm 2 Bit-flip BP Decoder Using CS-ω

Input: $\boldsymbol{llr}_1^N$, $\mathcal{A}$, CS-ω, $T_{\max}$
Output: $\hat{\boldsymbol{u}}_1^N$

1: Initialize the BP decoder using (4) and (5).
2: $\hat{\boldsymbol{u}}_1^N \leftarrow$ conventional_BP_decoder($\boldsymbol{llr}_1^N$, $\mathcal{A}$, $\boldsymbol{L}$, $\boldsymbol{R}$)
3: $t \leftarrow 1$ //the number of attempt bit-flip decoding
4: **while** $\hat{\boldsymbol{u}}_1^N$ does not satisfy CRC && $t \le T_{\max}$ **do**
5: $(j_1, j_2, \ldots, j_w) \leftarrow$ CS-$\omega(t)$ //the t-th row of CS-ω
6: **for** $k = 0 : 2^\omega - 1$ **do**
7: $b_1^\omega \leftarrow$ dec2bin(k, ω) //extend k to ω digits
8: $\boldsymbol{R} \leftarrow \boldsymbol{O}$, $\boldsymbol{R}(\mathcal{A}^c, 1) \leftarrow +\infty$ //refresh $\boldsymbol{R}$
9: **for** $l = 1 : \omega$ **do**
10: $R_{j_l,1} \leftarrow (1 - 2b_l) \times \infty$
11: **end for**
12: $\hat{\boldsymbol{u}}_1^N \leftarrow$ conventional_BP_decoder($\boldsymbol{llr}_1^N$, $\mathcal{A}$, $\boldsymbol{L}$, $\boldsymbol{R}$)
13: **if** $\hat{\boldsymbol{u}}_1^N$ satisfies CRC **then**
14: **return** $\hat{\boldsymbol{u}}_1^N$
15: **else**
16: **if** $t = T_{\max}$&&$k = 2^\omega - 1$ **then**
17: The ω bits-flip BP decoding fails.
18: **end if**
19: **end if**
20: **end for**
21: $t \leftarrow t + 1$
22: **end while**

$(R_{j_1,1}, R_{j_2,1}, \ldots, R_{j_\omega,1}) \in \{+\infty, -\infty\}^\omega$ of the current index $(j_1, j_2, \ldots, j_\omega)$ are enumerated. Lines 9-11 set the priori knowledge for the target bits.

Further, we comment on the special case where $\omega = 1$. Since there is only one bit to be flipped, it is much easier to guess the value of the target bit and therefore exhaustive enumeration is not necessary. When $\omega = 1$, the original CS [1], [2] is used to indicate unreliable bits.[3] Assume that $u_{\mathrm{CS}(i)}$ is selected to be flipped. We can simply set the value of $u_{\mathrm{CS}(i)}$ to the opposite direction of $L_{\mathrm{CS}(i),1} + R_{\mathrm{CS}(i),1}$ that is obtained from the original yet failed BP decoding, e.g., if $L_{\mathrm{CS}(i),1} + R_{\mathrm{CS}(i),1} > 0$, then in the bit-flip decoding we set $R_{\mathrm{CS}(i),1} = -\infty$ and vice versa. Such simplified operation will reduce half of latency in bit-flipping process compared with the exhaustive enumeration.

The upper bound of the average decoding latency of the bit-flip BP decoder using CS-ω is provided as follows.

Proposition 1: Without the CRC early termination, the average iteration number $\bar{I}_\omega$ and decoding latency $\bar{L}_\omega$ (in clock cycle) of the bit-flip BP decoder using CS-ω ($\omega \neq 1$) can be upper bounded by (16) and (17), respectively.

$$\bar{I}_\omega \le [1 + P_e(N, K, r, \gamma)2^\omega|\mathrm{CS}|]I, \tag{16}$$

$$\bar{L}_\omega \le [1 + P_e(N, K, r, \gamma)2^\omega|\mathrm{CS}|]I2\log_2 N, \tag{17}$$

[3] Instead of the CS, we can also select unreliable bits directly based on bits in $\mathcal{A}$ with the |CS| largest $P_e(u_i)$ in (1). However, simulation results show that the BLER of this scheme is inferior to that of the CS. Thus, in the next section we only present simulation results based on the CS.

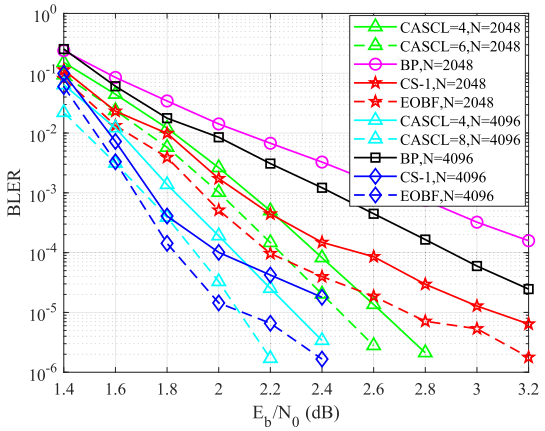


FIGURE 6. The BLER of the BPF decoder with CS-1. For $P(2048,1024+24)$, $T_{\max} = |\mathrm{CS}| = 220$, and for $P(4096,2048+24)$, $T_{\max} = |\mathrm{CS}| = 411$.

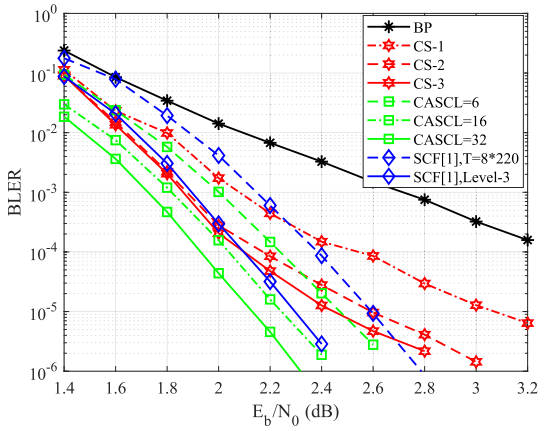


FIGURE 7. The BLER comparisons between the proposed BP bit-flip decoder and some existing SC-based decoders. CS-ω decoder has the maximum bit-flip number $T = 2^{\omega} \times |\mathrm{CS}|$ with $|\mathrm{CS}| = 220$ under $P(2048,1024+24)$.

where $P_e(N, K, r, \gamma)$ is the BLER of the conventional BP decoder when polar code $P(N, K + r)$ is used at $E_b/N_0 = \gamma$, and I represents the fixed iteration number in each BP decoding without the CRC early termination.

The proof follows by noting that the term 1 in (16) and (17) denotes the latency of the conventional BP decoding, and $P_e(N, K, r, \gamma)2^{\omega}|\mathrm{CS}|$ represents the extra latency caused by the bit-flip decoding in the worst case.

V. SIMULATION RESULTS

In this section, the BLER and average decoding complexity of the proposed BPF decoder using CS-ω are presented.

A. BLER OF BIT-FLIP DECODER USING CS-ω

In this subsection, the BLER of the proposed BPF decoder using CS-ω is provided. Simulation configurations are the same as in Table 1, except that the early stopping criterion is replaced by 24-bit CRC with $g(x) = x^{24}+x^{23}+x^6+x^5+x+1$. We first present the BLER of CS-1 in Figure 6.

In Figure 6, there is a benchmark called exhaustive one-bit-flip (EOBF) decoder. In the EOBF decoder, all bits in $\mathcal{A}$ are to be flipped until the decoding result satisfies the CRC. The BLER gap between the EOBF decoder and CS-1 can be considered as a metric that measures the accuracy of CS-1 in finding the bit flipping whose value will correct the already failed BP decoding. It can be observed that the BLER of CS-1 is significantly superior to that of the conventional BP decoder, e.g., at BLER$= 10^{-4}$, there is 0.9dB gain under $P(4096, 2048 + 24)$. Compared with the CA-SCL decoder, for $P(2048, 1024+24)$ and $P(4096, 2048+24)$, the BLER of CS-1 approaches that of the CA-SCL with $L = 4$. Compared with the EOBF decoder, the BLER gap is round 0.2-0.4 dB in all simulated SNR region. Such BLER gap can be narrowed down by decoders using CS-2 and CS-3.

The BLER performances of decoders using CS-2/CS-3 are provided in Figure 7 under $P(2048,1024 + 24)$. In Figure 7, at BLER$= 10^{-5}$, CS-2 outperforms CS-1 0.4dB, while CS-3 outperforms CS-2 0.2dB. In addition, the BLER of CS-3 is comparable to CA-SCL with $L = 16$ when SNR $<$2.2 dB.

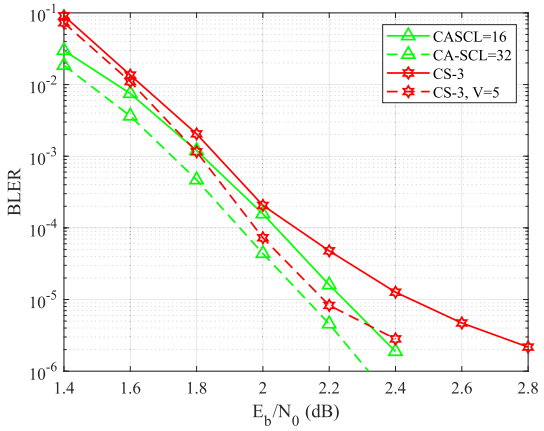


FIGURE 8. BLER of BPF CS-3 decoder with $V = 5$. The permuted factor graphs are optimized by methods in [16].

The BLER of the SCF decoder based on the critical set [1] is provided. SCF with $T = 8 \times 220$ means that the SCF decoder has a limited number T of attempt decoding. SCF level-3 represents that the SCF decoder uses level-3 MCS without the limitation on bit-flip number [1]. With the same number T of attempt bit-flip, the proposed CS-3 BP decoder outperforms the SCF decoder [1] when SNR $<$ 2.7dB, e.g., at BLER $= 10^{-4}$, the CS-3 decoder has 0.3 dB gain over SCF with the same T. SCF level-3 decoder has similar BLER to CA-SCL with $L = 16$. The BLER of the proposed CS-3 decoder can compete with SCF level-3 when SNR $\leq$ 2.2 dB, but falls short at higher SNR points, which may result from the error floor of the BP decoding. However, in the next subsection we will see that the decoding latency of the CS-3 decoder is significantly lower than that of SC-based decoders.

For the rest of this subsection, we provide a possible way to further improve the BLER of BPF decoder based on permuted

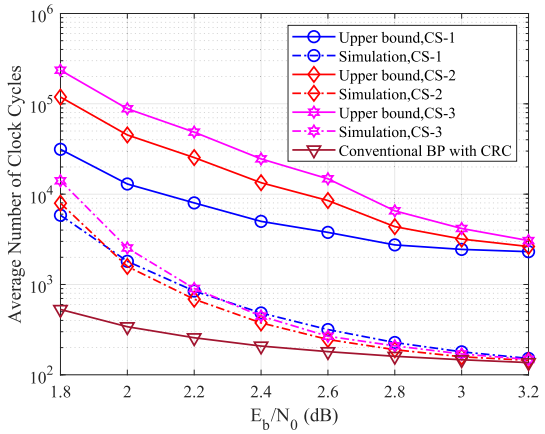


FIGURE 9. Average number of decoding clock cycles for P(2048, 1024+24). The maximum iteration number is 100 for one BP decoding.

factor graph [15], [16]. The idea is simple: (a) First, perform Algorithm 2. (b) If Algorithm 2 fails, use a permuted factor graph to re-execute Algorithm 2. This process continues until the decoding result satisfies the CRC or all the permuted graphs are tested. Denote V the number of permuted graphs (including the standard graph), and the BLER of BPF CS-3 with $V = 5$ is provided in Figure 8 for P(2048,1024 + 24). It can be seen that compared with CS-3 decoder, there is 0.2 dB gain at BLER $= 10^{-5}$, and the BLER of CS-3 with $V = 5$ is comparable to that of CASCL with $L = 32$. However, in such scheme, the number of attempt bit-flip decoding is V times larger than that of the CS-3 decoder. Therefore, the permuted factor graph is only a potential way to improve the BLER of the BPF decoder.

B. AVERAGE DECODING COMPLEXITY OF BIT-FLIP DECODER USING CS-ω

For the average decoding latency $\bar{L}_\omega$, simulation results are given in Figure 9 for $P(2048, 1024 + 24)$. Upper bound curves are derived from (17) without early stopping, while simulation curves use the CRC as early termination criterion. In Figure 9, it can be observed that the value of upper bound increases as ω increases, while the simulation curves almost overlap with each other in medium and high SNR regions. It is also observed that the average decoding clock cycle of the proposed BPF decoder can approach that of the conventional BP decoder when $E_b/N_0 \geq 2.6$dB.

Next, we make the decoding latency comparison between the proposed BPF decoder and SC-based decoders. Recent works [27], [28] focus on reducing the decoding latency of SCL decoder by identifying constitute nodes, such as rate 0, rate 1, repetition (Rep), and single parity check (SPC) nodes [27], and five new nodes called Type I-V [28]. When there is no limitation on hardware resources, the decoding clock cycles required by rate 0, rate 1, Rep, SPC, Type I, Type II, Type III, Type IV, and Type V nodes are 1, $\min\{L-1, N_s\}$, 2, $1+\min\{L, N_s\}$, 2, 2, $1+\min\{L-1, N_s-2\}$, $1+\min\{L-1, N_s-4\}$, and 2, respectively [27], [28], where N_s is the number of bits that a node includes. The decoding clock cycle comparisons are shown in Table 5, where SCL denotes the conventional SCL scheme [6], Fast SCL represents the decoding method in [27], and new node denotes the decoding method in [28]. The target polar code is $P(2048, 1024 + 24)$. It can be seen that the decoding clock cycle of the proposed BPF decoder decreases fast with the increase of SNR. CS-1 has slightly higher latency than that of CS-2/CS-3 when $E_b/N_0 > 2.4$dB because the error correction ability of CS-1 is relatively weaker, which results in more attempt bit-flip decoding. When SNR ≥ 2.4dB, the average decoding clock cycle of the proposed CS-3 decoder is the lowest one compared with that of all SC-based decoders in Table 5, which confirms that the proposed decoder is especially suitable for medium and high SNR points. The SCF decoders in [1] use the standard SC that does not involve fast decoding based on constituent nodes, so the decoding clock cycles of such decoders converge to $2N - 2$.

TABLE 5. Decoding clock cycle comparisons between the proposed BPF decoder and SC-based decoders. SCF-T denotes SCF in [1] with limited bit-flip number $T = 8 \times 220$, and SCF-3 represents SCF in [1] with level-3 MCS.

Decoders	E_b/N_0 (dB)						
	2	2.2	2.4	2.6	2.8	3	3.2
CS-1	1805	847	484	317	228	180	152
CS-2	1584	687	376	246	190	158	145
CS-3	2536	902	447	267	209	170	148
SCF-T	33816	10027	5267	4308	4135	4103	4096
SCF-3	48349	11911	5428	4310	4141	4103	4096
	List Size						
	6		16			32	
SCL	5142		5142			5142	
FastSCL	805		1074			1269	
NewNode	523		765			941	

After latency comparisons, we make complexity comparisons between the proposed BPF decoder and the conventional CA-SCL decoder. First, we focus on the computational complexity. Here, for the BP decoder, one computation of one intermediate LLR shown in (3) takes unit complexity, while for the CA-SCL decoder, one computation of $\text{sign}(x)\text{sign}(y)\min(|x|, |y|)$ or $(1-2b)x + y$ takes unit complexity, where x, y denotes the LLR and b denotes the bit value. The comparison results are given in Table 6. It is not surprising that the SCL decoder has lower computational complexity than that of the proposed decoder because SC-based algorithms are famous for their low computational complexity. However, note that at high SNR points such as 3 or 3.2 dB, the proposed decoder can yield similar computational complexity compared to that of the SCL decoder.

Next, we focus on the memory requirement (MR). The MR of the conventional SCL decoder and the proposed BPF decoder can be expressed as follows:

$$MR_{\text{SCL}} = [N+(N-1)L]Q_{\text{LLR}}+LQ_{\text{PM}}+(2N-1)L, \quad (18)$$

$$MR_{\text{BPF}} = 2N(n+1)Q_{\text{LLR}} + |\mathcal{A}| + |\text{CS}|\omega n. \quad (19)$$

where Q_{LLR} and Q_{PM} denote the number of quantization bits for LLR and path metric, respectively. The three terms in (19)

TABLE 6. Computational complexity comparisons between the proposed BPF decoder and the conventional SCL decoder for decoding one received signal y_1^N. The computational complexity of the BP decoder is calculated using $2N\bar{n}\bar{l}_\omega$. The target code is $P(2048, 1024 + 24)$.

Decoders	E_b/N_0 (dB)				
	2.4	2.6	2.8	3	3.2
CS-3	914821	547016	428539	348436	303101
	List Size				
	16		32		
SCL	284976		563504		

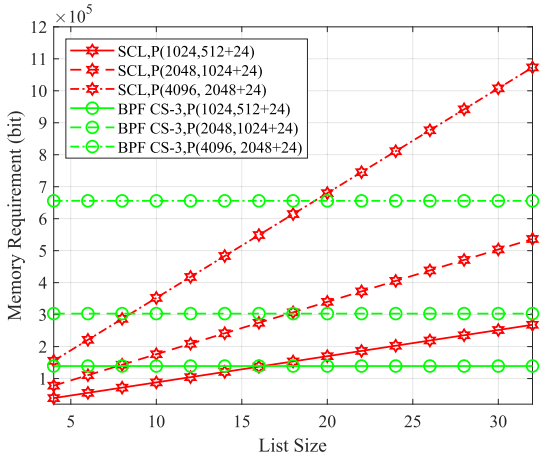


FIGURE 10. The MR comparisons between the proposed CS-3 decoder and the SCL decoder, $\omega = 3$, $Q_{\text{LLR}} = Q_{\text{PM}} = 6$.

denote the memory required by $\boldsymbol{L}/\boldsymbol{R}$ matrices, information bit output, and CS-ω, respectively. The MR of the proposed CS-3 decoder and the SCL decoder is shown in Figure 10. When $L = 16$, the MR of the proposed decoder and SCL decoder almost coincides under different code lengths. Since $L = 16$ is of our interest, we conclude that the proposed decoder has the similar MR to that of the SCL decoder.

Therefore, the proposed BPF decoder achieves the lower latency at the cost of higher computational complexity. In latency-sensitive scenarios, such additional cost is still worthwhile because the latency of the proposed decoder is much lower than that of the SCL decoder as shown in Table 5.

VI. CONCLUSION

In this paper, bit-flip is extended to the BP decoding for polar codes. We analyze the behavior of the wrongly estimated codewords, and propose the bit-flip decoder based on critical set with order ω. The BLER and decoding complexity of the proposed decoder are presented. Simulation results demonstrate that the BLER of the proposed BPF decoder can approach that of CA-SCL with $L = 16$. The decoding latency of the proposed decoder can approach that of the conventional BP decoder in medium and high SNR regions.

REFERENCES

[1] Z. Zhang, K. Qin, L. Zhang, H. Zhang, and G. T. Chen, "Progressive bit-flipping decoding of polar codes over layered critical sets," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[2] Z. Zhang, K. Qin, L. Zhang, and G. T. Chen, "Progressive bit-flipping decoding of polar codes: A critical-set based tree search approach," *IEEE Access*, vol. 6, pp. 57738–57750, 2018.

[3] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.

[4] K. Niu, K. Chen, J. Lin, and Q. T. Zhang, "Polar codes: Primary concepts and practical decoding algorithms," *IEEE Commun. Mag.*, vol. 52, no. 7, pp. 192–203, Jul. 2014.

[5] A. Balatsoukas-Stimming, P. Giard, and A. Burg, "Comparison of polar decoders with existing low-density parity-check and turbo decoders," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Mar. 2017, pp. 1–6.

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

[7] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, "A low-complexity improved successive cancellation decoder for polar codes," in *Proc. 48th Asilomar Conf. Signals, Syst. Comput.*, Nov. 2014, pp. 2116–2120.

[8] L. Chandesris, V. Savin, and D. Declercq, "Dynamic-SCFlip decoding of polar codes," *IEEE Trans. Commun.*, vol. 66, no. 6, pp. 2333–2345, Jun. 2018.

[9] P. Giard and A. Burg, "Fast-SSC-flip decoding of polar codes," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Apr. 2018, pp. 73–77.

[10] E. Arıkan, "A performance comparison of polar codes and Reed-Muller codes," *IEEE Commun. Lett.*, vol. 12, no. 6, pp. 447–449, Jun. 2008.

[11] E. Arýkan, "Polar codes: A pipelined implementation," in *Proc. 4th Int. Symp. Broadband Commun. (ISBC)*, Jul. 2010, pp. 11–14.

[12] B. Yuan and K. K. Parhi, "Algorithm and architecture for hybrid decoding of polar codes," in *Proc. 48th Asilomar Conf. Signals, Syst. Comput.*, Nov. 2014, pp. 2050–2053.

[13] S. Cammerer, B. Leible, M. Stahl, J. Hoydis, and S. T. Brink, "Combining belief propagation and successive cancellation list decoding of polar codes on a GPU platform," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 3664–3668.

[14] S. Sun, S.-G. Cho, and Z. Zhang, "Post-processing methods for improving coding gain in belief propagation decoding of polar codes," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[15] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Belief propagation decoding of polar codes on permuted factor graphs," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6.

[16] N. Doan, S. A. Hashemi, M. Mondelli, and W. J. Gross. (2018). "On the decoding of polar codes on permuted factor graphs." [Online]. Available: https://arxiv.org/abs/1806.11195

[17] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Belief propagation list decoding of polar codes," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1536–1539, Aug. 2018.

[18] N. Varnica, M. P. C. Fossorier, and A. Kavcic, "Augmented belief propagation decoding of low-density parity check codes," *IEEE Trans. Commun.*, vol. 55, no. 7, pp. 1308–1317, Jul. 2007.

[19] S. Scholl, P. Schläfer, and N. Wehn, "Saturated min-sum decoding: An 'afterburner' for LDPC decoder hardware," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2016, pp. 1219–1224.

[20] P. Kang, Y. Xie, L. Yang, C. Zheng, J. Yuan, and Y. Wei. (2018). "Enhanced quasi-maximum likelihood decoding of short LDPC codes based on saturation." [Online]. Available: https://arxiv.org/abs/1810.13111

[21] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3221–3227, Nov. 2012.

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

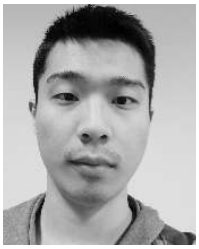
[23] D. Wu, Y. Li, and Y. Sun, "Construction and block error rate analysis of polar codes over AWGN channel based on Gaussian approximation," *IEEE Commun. Lett.*, vol. 18, no. 7, pp. 1099–1102, Jul. 2014.

[24] B. Yuan and K. K. Parhi, "Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders," *IEEE Trans. Signal Process.*, vol. 62, no. 24, pp. 6496–6506, Dec. 2014.

[25] R. Mori and T. Tanaka, "Performance of polar codes with the construction using density evolution," *IEEE Commun. Lett.*, vol. 13, no. 7, pp. 519–521, Jul. 2009.

[26] C. Schürch, "A partial order for the synthesized channels of a polar code," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2016, pp. 220–224.

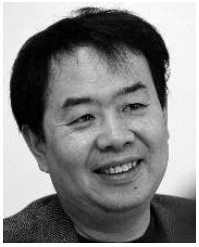
[27] S. A. Hashemi, C. Condo, and W. J. Gross, "Fast and flexible successive-cancellation list decoders for polar codes," *IEEE Trans. Signal Process.*, vol. 65, no. 21, pp. 5756–5769, Nov. 2017.
[28] M. Hanif, M. H. Ardakani, and M. Ardakani, "Fast list decoding of polar codes: Decoders for additional nodes," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Apr. 2018, pp. 37–42.



YONGRUN YU received the bachelor's degree in communication engineering from Tianjin University, Tianjin, China, in 2016. He is currently pursuing the master's degree with Southeast University. His research interests include channel coding, especially polar coding.

ZHIWEN PAN has been with the National Mobile Communications Research Laboratory, Southeast University, as an Associate Professor, since 2000, and a Professor, since 2004. From 2000 to 2001, he was involved in the research and standardization of 3G, and since 2002, he has been involved in the investigations on key technologies for IMT-A and 5G. He has published over 50 papers recently, and holds over 50 patents. His research interests include self-organizing networks, wireless networking, and radio transmission technology for wireless communications.

NAN LIU received the B.Eng. degree in electrical engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2001, and the Ph.D. degree in electrical and computer engineering from the University of Maryland at College Park, College Park, MD, USA, in 2007. From 2007 to 2008, she was a Postdoctoral Scholar with the Wireless Systems Lab, Department of Electrical Engineering, Stanford University. In 2009, she became a Professor with the National Mobile Communications Research Laboratory, School of Information Science and Engineering, Southeast University, Nanjing, China. Her research interests include network information theory for wireless networks, SON algorithms for next-generation cellular networks, and energy-efficient communications. She is an Associate Editor of *China Communications*.



XIAOHU YOU has been with the National Mobile Communications Research Laboratory, Southeast University, since 1990, where he held the rank of Professor, was a Chair Professor of the Cheung Kong Scholars Program, and served as the Director of the Laboratory. He has contributed over 50 IEEE journal papers and two books in the areas of adaptive signal processing, neural networks, and their applications to communication systems, and holds over 80 patents. He was a Premier Foundation Investigator of the China National Science Foundation. From 1999 to 2002, he was a Principal Expert of the C3G Project, responsible for organizing China's 3G Mobile Communications R&D Activities. From 2001 to 2006, he was a Principal Expert of the National 863 FuTURE Project. His research interests include mobile communication systems, signal processing, and its applications. His current research interests include wireless and mobile communication systems and modern digital signal processing. He has been an IEEE Fellow since 2011. He has been a Section Chair of the IEEE Nanjing Section, since 2010. He served as a General Co-Chair for IEEE WCNC 2014.