Belief Propagation Decoder With Multiple Bit-Flipping Sets and Stopping Criteria for Polar Codes

JIANYONG ZHANG\(^\text{D}\), (Member, IEEE), AND MUGUANG WANG\(^\text{D}\), (Member, IEEE)

Institute of Lightwave Technology, Beijing Jiaotong University, Beijing 100044, China
Key Laboratory of All-Optical Network and Advanced Telecommunication Network of EMC, Beijing Jiaotong University, Beijing 100044, China

Corresponding author: Jianyong Zhang (zhang.jianyong.bjtu09@gmail.com)

This work was supported by the National Natural Science Foundation of China under Grant 61671047 and Grant 61775015.

ABSTRACT

Compared with successive cancellation list (SCL) decoders, belief propagation (BP)-based decoders suffer performance loss in middle- and high-signal-to-noise ratio (SNR) regions. By analyzing the behavior of the incorrect decoding results of the bit-flipping BP decoder with a critical set of order \(\omega\) (BFBP-CS\(^\omega\)), we found that undetected errors mainly contribute to the error floor. Based on this observation, we proposed a belief propagation decoder with multiple bit-flipping sets (BFSs) and stopping criteria (BP-MF-MC) in this work. We use multiple stopping criteria to identify undetected errors and a small BFS to find an additional estimated codeword given by the bit-flipping BP (BFBP) function. For uncorrected errors, we use multiple BFSs to find estimated codewords with the BFBP function. Furthermore, we propose a method to dynamically generate a BFS based on the submatrix check. This method can remove unnecessary bit-flipping positions and increase the order of the critical set. Then, the best codeword is selected from all estimated codewords according to the maximum likelihood principle. Numerical results show that BP-MF-MC performs similarly to the cyclic redundancy check-aided SCL (CA-SCL) decoder with list size 16 and is slightly worse than CA-SCL with list size 32.

INDEX TERMS

Polar codes, belief propagation, bit-flipping, multiple stopping criteria.

I. INTRODUCTION

Polar codes are well known for their ability to achieve Shannon capacity and their low encoding and decoding complexity [1]. The successive-cancellation (SC) decoding algorithm proposed by Arıkan [1] is one of the common decoding methods for polar codes. To improve the performance of SC, Tal and Vardy [2] introduced a successive cancellation list (SCL) decoding algorithm whose performance is very close to that of maximum-likelihood decoding.

On the other hand, the belief propagation (BP) algorithm [1], [3] is theoretically more parallel than SC-based algorithms. However, the BP algorithm often has a much higher computational complexity than SC. To lower the complexity, an early-stopping criterion [4]–[6] was proposed to reduce the number of iterations of the BP algorithm. The authors of [7] proposed a subfactor graph-freezing technique to reduce the average number of computations as well as the average number of iterations required by the BP algorithm. To reduce the required memory, a stage-combined BP decoding algorithm [8] was introduced to reduce the decoding latency and memory requirement. Moreover, a generalized BP algorithm [9], [10] based on these modifying factor graphs was proposed to further improve the performance of the BP algorithm.

Additionally, the error-correction performance of the BP algorithm is worse than that of the SCL decoder. Several BP-based algorithms have been proposed, and they outperform the conventional BP algorithm. The BP list decoder (BPL) was proposed based on a permuted factor graph [11], [12]. The performance of the BPL decoder is close to that of the SCL decoder, whereas it is inferior to that of the cyclic redundancy check (CRC)-aided SCL (CA-SCL) decoder. In [13], the proposed BP bit-strengthening (BPBS) decoder performs similarly to the SCL decoder in medium- and high-SNR regions. However, BPL and BPBS benefit only slightly or not at all from an outer CRC code.

In [14], a parity-check matrix was introduced to improve the performance of the BP decoder. In [15], CRC-polar BP (CPBP) and neural CPBP (NCPBP) decoders, which
achieve significant error-correction performance improvement compared to conventional CRC-aided BP decoders, were proposed. Inspired by the SC-based bit-flipping decoders, a bit-flipping BP decoder using a critical set of order \( \omega \) (BFBP-CS\( ^{\omega} \)) was proposed in [16]. BFBP-CS\( ^{\omega} \) uses a CRC check to detect errors and exhaustively sets the a priori knowledge of the unreliable bits to find a codeword passed the CRC check. However, the BFBP-CS\( ^{\omega} \) decoder exhibits an error floor in high signal-to-noise ratio (SNR) regions.

In this paper, we aim to lower the error floor of the bit-flipping BP (BFBP) decoder and design a BP decoder that has a similar performance to that of CA-SCL decoders. The main contributions of this paper are summarized as follows:

1) In this work, we propose a belief propagation decoder with multiple bit-flipping sets (BFSs) and stopping criteria (BP-MF-MC), which is a generalization of BFBP-CS\( ^{\omega} \). We analyze the behavior of the incorrect decoding results of the BFBP-CS\( ^{\omega} \) decoder. We find that undetected errors contribute to the error floor in high-SNR regions. Therefore, we use multiple stopping criteria to identify undetected errors and multiple BFSs to find more estimated codewords with the proposed decoder. Finally, the best codeword is selected from the estimated codewords according to the maximum likelihood principle.

2) We present many types of BFSs in this work. We propose a method to dynamically generate a BFS based on the submatrix check. This method can use the submatrix check to remove unnecessary bit-flipping positions. Thus, it can increase the order of the critical set. It is shown in the numerical results that the configurations with a dynamically generated BFS can improve performance.

The rest of this paper is organized as follows: In Section II, we introduce the BP and BFBP-CS\( ^{\omega} \) algorithms. Section III analyzes the decoding error properties of BFBP-CS\( ^{\omega} \). Section IV proposes the BP-MF-MC algorithm. In Section V, we present the numerical results of BP-MF-MC for the polar codes (2048,1024+24), (1024,256+24), and (1024,768+24). We also evaluate the computational complexity of the BP-MF-MC in this section. Concluding remarks are given in Section VI.

II. PRELIMINARIES

A. NOTATION

In this work, we use letters \( W \) in standard font to denote scalars and boldface letters \( \mathbf{W} \) to denote vectors and matrices.

B. POLAR CODES

Polar codes are linear block codes based on the phenomenon of channel polarization, in which individual channels are recursively combined and split such that their mutual information tends toward either 1 or 0. In other words, some of these channels become completely noise-free, while the others become completely noisy. Furthermore, the fraction of noiseless channels tends toward the capacity of the underlying binary symmetric channels [1]. Polar codes are specified by a generator tensor \( \mathbf{G}_N \), where \( N = 2^n \) is the code length. A polar code \((N, r)\) can be generated in two steps. Let \( A \) and \( A' \) be reliable and unreliable positions for information and frozen bits. First, an \( N \)-bit message \( \mathbf{u} \) is constructed by assigning \( r \) information bits to \( A \) and 0 to \( A' \). Then, the \( N \)-bit \( \mathbf{u} \) is multiplied with the generator \( \mathbf{G}_N \) to generate an \( N \)-bit transmitted codeword \( \mathbf{x} = \mathbf{uG}_N \).

C. BELIEF PROPAGATION DECODER

The process of polar coding (encoding and decoding) can be represented by a factor graph [1]. Figure 1 shows the factor graph of polar codes with \( N = 8 \), which is divided into \( n = \log_2 N \) stages. Each stage has \( N/2 \) processing elements (PEs), and each PE has two input and two output variable nodes. The BP decoding of polar codes is the process of passing the log-likelihood ratio (LLR) iteratively through the factor graph. Node \((i,j)\) is associated with two types of LLR: left-to-right \( R_{i,j} \) and right-to-left \( L_{i,j} \), where \( i \) is the row index on the factor graph at stage \( j \). Each PE computes the \( \mathbf{R} \) and \( \mathbf{L} \) messages as follows:

\[
\begin{align*}
R_{i,j+1} &= g(R_{i,j}, R_{i+2,j} + L_{i+2,j+1}), \\
R_{i+2,j+1} &= g(R_{i,j}, L_{i,j+1}) + R_{i+2,j}, \\
L_{i,j} &= g(L_{i,j+1}, L_{i+2,j+1} + R_{i+2,j}), \\
L_{i+2,j} &= g(R_{i,j}, L_{i,j+1}) + L_{i+2,j+1},
\end{align*}
\]

where \( g(x, y) = 0.9375 \cdot \text{sign}(x)\text{sign}(y)\min(|x|, |y|) \).

The messages \( \mathbf{R} \) and \( \mathbf{L} \) are initialized by

\[
\begin{align*}
R_{0,i}^0 &= \infty, & i \in A^c \\
0, & i \in A \\
L_{i,n+1}^0 &= \text{llr}_i,
\end{align*}
\]

where \( \text{llr}_i \) is the LLR of the \( i \)-th received bit.

In this work, BP decoding uses the CRC check as the early stopping criterion to reduce the number of iterations.
D. BIT-FLIPPING BP DECODER USING CS\(^{\omega}\)

The critical set (CS) contains the bit positions that tend to be unreliable, and its construction is given in [16]. BFBP-CS\(^{\omega}\) is summarized in Algorithm 1. First, the conventional BP algorithm is performed. If the BP decoder fails the CRC test, BFBP-CS\(^{\omega}\) exhaustively enumerates all the possible values \(b^{\omega}_1 = (b_1, \ldots, b_\omega)\) of target positions \(j^{\omega}_1 = (j_1, \ldots, j_\omega)\) in CS\(^{\omega}\). Finally, BFBP-CS\(^{\omega}\) terminates the iteration once the estimated codeword passes the CRC check.

Algorithm 1 BP Algorithm Using CS\(^{\omega}\)

1: Input: \(llr^N\), \(A\), CS\(^{\omega}\)
2: Output: \(\hat{u}^N\)
3: Initialize \(L\) and \(R\) by using (5) and (6)
4: \(\hat{u}^N_1 \leftarrow \text{BP}(llr^N_1, A, L, R)\)
5: if \(\hat{u}^N_1\) does not satisfy CRC then
6: for all \((j^{\omega}_1, b^{\omega}_1)\) in CS\(^{\omega}\) do
7: Initialize \(R\) by (5)
8: for \(l = 1\) to \(\omega\) do
9: \(R_{j_l,1} \leftarrow (1 - 2 b_l) \times \infty\)
10: end for
11: \(\hat{u}^N_1 \leftarrow \text{BP}(llr^N_1, A, L, R)\)
12: if \(\hat{u}^N_1\) satisfy CRC then
13: return \(\hat{u}^N_1\)
14: end if
15: end for
16: end if

III. ANALYSIS OF THE PERFORMANCE OF BFBP-CS\(^{\omega}\)

In the first part of this section, we present the simulation conditions for this work and analyze the performance limits of BFBP-CS\(^{\omega}\) using oracle-assisted BP-CS\(^{\omega}\) (OABP-CS\(^{\omega}\)). We find that OABP-CS\(^{\omega}\) has an error floor in the high-SNR region, and it is difficult to further improve the performance simply by increasing \(\omega\). In the second part, we analyze the error types of OABP-CS\(^{\omega}\). We find that undetected errors mainly contributed to the error floor in the high-SNR region.

A. SIMULATION CONDITION

In this work, the modulation format is binary phase-shift keying, and the channel noise model is additive white Gaussian noise. The maximum number of iterations for the BP decoders is 100. The CRC checks used in this work have lengths of 11, 16, and 24 bits. Their polynomials [17] are \(x^{11} + x^{10} + x^9 + x^5 + x^{10} + x^9 + x^5 + 1\) and \(x^{24} + x^{23} + x^6 + x^5 + x^1 + 1\), respectively.

B. FER LOWER BOUNDS OF THE BFBP-CS\(^{\omega}\)

According to [16], the performance and complexity of BFBP-CS\(^{\omega}\) increase with \(\omega\). If \(\omega \geq 6\), the size of CS\(^{\omega}\), which is approximately \(2^{\omega} \times |\text{CS}|\), is too large to simulate its performance. Therefore, we use OABP-CS\(^{\omega}\) to predict the theoretical optimal performance of BFBP-CS\(^{\omega}\), which serves as a lower bound on the frame error (FER) results. OABP-CS\(^{\omega}\) is almost the same as BFBP-CS\(^{\omega}\) except that line 9 of Algorithm 1 is replaced by

\[
IR_{j_l,1} \leftarrow (1 - u_{j_l}) \times \infty, \quad (7)
\]

where \(j_l\) is the \(l\)-th position of CS\(^{\omega}\) and \(u_{j_l}\) is the \(j_l\)-th message bit.

The performance of the OABP-CS\(^{\omega}\) decoder for the polar code (1024, 768+24) is shown in Figure 2. The code is constructed from the reliability table of [17] and has 24 CRC bits. As \(\omega\) increases, the FER of OABP-CS\(^{\omega}\) decreases. However, the performance difference between the two adjacent curves decreases as \(\omega\) increases. Compared with CA-SCL, OABP-CS\(^{\omega}\) has the error floor at a high SNR. For example, OABP-CS\(^{9}\) performs similarly to or better than CA-SCL with list size 8 (CA-SCL8) in a range of [3, 4]dB. However, the performance of OABP-CS\(^{9}\) is worse than that of CA-SCL8 if SNR>4 dB.

![FER comparison of the CA-SCL and OABP-CS\(^{\omega}\) decoders. The polar code is (1024, 768+24).](image)

C. ANALYSIS OF THE ERRORS OF OABP-CS\(^{\omega}\)

We divide the errors of OABP-CS\(^{\omega}\) into three types. The first type, error type I, includes all undetected errors that satisfy the CRC check. The second type, error type II, passes the CRC check during conventional BP decoding. The third type, error type III, is an error that is a decoding failure of the CRC check. The second type, error type II, passes the CRC check during conventional BP decoding. The third type, error type III, is an error that is a decoding failure of OABP-CS\(^{\omega}\).

The percentages of the three error types are shown in Figure 3. As shown in the figure, the percentage of type III decreases, whereas the percentages of types I and II increase, with the SNR. The percentage of type II is almost the same as that of type I. Moreover, the percentage of type II does not decrease as \(\omega\) increases. Therefore, we can conclude that error type II is the dominant factor accounting for the error floor of OABP-CS\(^{\omega}\) and BFBP-CS\(^{\omega}\) in the high-SNR region. Furthermore, the performance of OABP-CS\(^{\omega}\) in the high-SNR region cannot be further improved by simply increasing \(\omega\) due to error type II. For the low-SNR region, error type III is the main factor in the failure of the decoder.
Based on the above observations, we generalize BFBP-CSω and propose the BP-MF-MC decoder. We use multiple BFSs and stopping criteria to lower the rate of error types II and III simultaneously. The scheme of BP-MF-MC is given in Algorithms 2 and 3.

Let $S$ be an independent stopping criterion other than the CRC check, such as the $G$-matrix [5], the worst of information bits (WIB) [4] or the best frozen bits (BFB) [6]. If the estimated codeword $\hat{u}^N_1$ satisfies $S$, it is more likely a correct codeword and is output by the algorithm.

Let $\Phi = \{\Phi_1, \ldots, \Phi_{n_\Phi}\}$ be a sequence containing $n_\Phi$ BFSs. The BFSs in $\Phi$ can be constructed by various methods, such as the critical set or all the information bits. On line 8 of Algorithm 2, a function $\text{BFBP}(\cdot)$ outputs the decoding results $U_j$ for each BFS $\Phi_i$, where $U_j$ is a row vector of length $N$.

The scheme of BFBP$(\cdot)$ is shown in Algorithm 3. It is a generalized BFBP-CSω decoder over the flipping sets in $\Omega$. The function BFBP$(\cdot)$ differs from the BFBP-CSω decoder in two ways. The first is in line 5 of Algorithm 3. The BFBP decoder reinitializes both messages $R$ and $L$ in the beginning of its flipping process. However, BFBP-CSω only re-initializes $R$ in line 7 of Algorithm 1. Therefore, BFBP$(\cdot)$ can be implemented in parallel. Second, the BFSs in $\Omega$ do not require all the bit-flipping positions $j^m_i$ to be of the same length. Thus, $\Omega$ can consist of several disjoint BFSs.

Let $\Psi = \{\Psi_1, \ldots, \Psi_{n_\Psi}\}$ be a sequence containing $n_\Psi$ BFSs. If $\hat{u}^N_1$ does not satisfy the CRC check, BFBP$(\cdot)$ is also used to find the decoding results $U_j$ for each $\Psi_i$.

On line 18 of Algorithm 2, the best codeword $\hat{u}^N_1$ is selected in the set $U = \{U_1; \ldots; U_{n_\Psi}\}$ according to the maximum likelihood principle [11]

$$\hat{u}^N_1 = \text{argmin}_{U_j \in U} ||y^N_j - (1 - 2U_jG)||.$$  \hspace{1cm} (8)

Of course, the sequences $\Phi$ and $\Psi$ should be carefully chosen to balance the performance and complexity of BP-MF-MC.

To efficiently find the error bit positions, we propose a method $D(\lambda_{th}, F, A^c)$ to dynamically generate the BFS with a submatrix check. This method is inspired by the BP bit-strengthening method [13]. The scheme of $D(\lambda_{th}, F, A^c)$ is shown in Algorithm 4. Let $G_{2m}$ be a generator matrix of size $2^m \times 2^m$. Let $\mathbf{u}_k$ be a row vector of length $2^m$, which is the $k$-th subfactor graph of $\mathbf{u}_1^N$. Let $\lambda_{th}$ be the last stage that performs the submatrix check. Let $\mathbf{x}_k$ be the row vector corresponding to $\mathbf{u}_k$, which is given by

$$\mathbf{x}_k = (1 - \text{sign}(R_k + L_k))/2,$$  \hspace{1cm} (9)
where $R_k$ and $L_k$ are the $k$-th row vectors of length $2^m$ at the $m$-th stage.

First, the set of frozen bit positions $A^i$ is initialized by $A^c$. If the submatrix check in line 7 of Algorithm 4 is satisfied, we take these information bits to be correct and update $A'$. Then, we use the updated $A'$ to generate the CS with the algorithm given in [16]. Finally, Algorithm 4 outputs $CS^ω$ using $A^i$. In this way, we increase the order $ω$ by eliminating the unnecessary bit-flipping positions. According to the performance of OABP-CS$^ω$, this procedure is helpful in decreasing the number of errors of type III.

Remarks:
1. If $Φ = ∅$ and $Ψ = \{CS^ω\}$, BP-MF-MC is reduced to BFBP-CS$^ω$.
2. If $F = ∞$, $D(λ_{th}, F, A^c) = ∅$.

V. SIMULATION RESULTS

In this section, we introduce several configurations for BP-MF-MC and evaluate the FER performance for three polar codes (1024, 256+24), (1024, 768+24) and (2048, 1024+24). The code (2048, 1024+24) is designed by the Gaussian approximation (GA) method [18]. The design SNR is 2.5 dB, which is optimized according to the simulation results of CA-SCL decoders with various list sizes over a wide range. The polar codes (1024,256+24) and (1024,768+24) are designed by the reliability table given in [17].

A. CONFIGURATIONS

According to Algorithm 2, there are two sequences used by the function BFBP(·) of Algorithm 3. The sequence $Φ$ is used on line 8 to provide an additional estimated codeword that passes the CRC check and lowers the number of errors of type II. Thus, we expect that $Φ$ is as small as possible. In contrast, $Ψ$ is used to find a codeword that can pass the CRC check. We require $Ψ$ to be as large as possible because otherwise, the conventional BP fails.

In this work, $Φ$ and $Ψ$ are mainly constructed by five types of BFSs. The first two are the critical sets $CS^1$ and $CS^3$. According to the BPBS decoder, bit strengthening is also helpful to improve the performance of the BP decoder. Thus, the size of $CS^1$ for BP-MF-MC is twice the number of critical bit-flipping positions, which is given by $|CS^1| = 2|CS^3$. The third $A \setminus CS^3$ is the information set excluding $CS^3$. Interestingly, we find that the frozen bits $A^c$ can be used as the flipping set. The last set is the BFS dynamically generated by Algorithm 4. For the polar codes (2048,1024+24) and (1024,768+24), $λ_{th}$ is 9 and $F = 3000$, while $λ_{th} = 6$ for the polar code (1024,256+24).

In this work, the configuration of BP-MF-MC refers to the way $Φ$ and $Ψ$ are constructed. In the following, we use the tuple $(|Φ|, |Ψ|, D(λ_{th}, F, A^c)', F)$ to denote a configuration of $Φ$ and $Ψ$, where $|·|$ is the number of elements of a sequence. The configurations used in this work for BP-MF-MC are summarized in Table 1.

Finally, there are two stopping criteria used in this work. The first is the CRC check used in BFBP (-) and on line 5 of Algorithm 2. The second $S$ is the G-matrix used on line 6 of Algorithm 2.

B. PERFORMANCE

The FER performance of BP-MF-MC with configurations of $(0, 2760, ∞)$, $(0, 4412, ∞)$, $(440, 1760, ∞)$ and $(440, 4412, ∞)$ is shown in Figure 4. The polar code is (2048,1024+24). It is shown in the figure that the performance of $(0, 2760, ∞)$ and $(0, 4412, ∞)$ is better than that of BFBP-CS$^3$ when $2 ≤ E_b/N_0 ≤ 3$. Moreover, $(0, 4412, ∞)$ has an approximately 0.25 dB gain over BFBP-CS$^3$ at $FER = 10^{-6}$. Although the performance gain of $(0, 2760, ∞)$ may contribute to the simulation error, it is clear that a larger BFS always has a better performance than BFBP-CS$^ω$. Therefore, we want to make $Ψ$ as large as possible without significantly increasing complexity.

However, the configurations $(0, 2760, ∞)$ and $(0, 4412, ∞)$ perform similarly to BFBP-CS$^3$ with increasing $E_b/N_0$, especially at $E_b/N_0 = 3$ dB. The main reason is that these two configurations cannot reduce the number of errors of type II if $|Φ| = 0$. Therefore, configurations with $|Φ| > 0$
can lower the error floor at high SNRs. The figure shows that there is a performance improvement of \((440,1760,\infty)\) over \(\text{BFBF-CS}^3\) if \(E_b/N_0 \geq 2.75\) dB. Furthermore, \((440,4412,\infty)\) also outperforms \((0,4412,\infty)\) if \(E_b/N_0 \geq 2.75\) dB.

The performance of the configurations, which are \((440,4412,\infty)\), \((440,4412,9)\), \((1760,4412,9)\), and \((0,3412,9)\), is shown in Figure 5.

The figure shows that \((440,4412,9)\) and \((1760,4412,9)\) have the best performance among all the configurations of \(\text{BP-MF-MC}\). These two configurations outperform \(\text{CA-SCL}\) with list size 16 \((\text{CA-SCL16})\) and are even close to \(\text{CA-SCL}\) with list size 32 \((\text{CA-SCL32})\) when \(E_b/N_0 > 1.75\) dB and 24 CRC bits are used. Finally, the performance of \((440,4412,9)\) and \((1760,4412,9)\) does not show any error floor, and they are similar to each other. Thus, we can select the smaller \(\Phi\) to reduce the complexity of \(\text{BP-MF-MC}\).

As shown in Figure 5, \((440,4412,\infty)\) performs worse than \((440,4412,9)\) if \(E_b/N_0 > 2\) dB. Thus, the dynamic BFS \(D(A_{th}, F, A^c)\) can further improve the performance of \(\text{BP-MF-MC}\) at high SNR. The configuration \((0,3412,9)\) performs better than \((440,4412,\infty)\) if \(E_b/N_0 \leq 2.5\) dB but shows error floors at FER = 10^{-6}.

Based on the observation above, we found that configurations with \(\Phi > 0\) and \([\text{CS}^3 \cup (A \setminus \text{CS}^3) \cup A^c, D(F, 3000, A^c)]\) have the best performance of all configurations in which \(F \leq n - 1\). The configuration with \(\Phi = 0\) and \([\text{CS}^3 \cup (A \setminus \text{CS}^3), D(F, 3000, A^c)]\) has performance second only to the best configuration.

For the polar codes \((1024,256+24)\) and \((1024,768+24)\), the performance of \(\text{BP-MF-MC}\) is shown in Figure 6 and 7. The configurations are \((178,1834,6)\) and \((0,1090,6)\) for \((1024,256+24)\), and \((198,2406,9)\) and \((0,2174,9)\) for \((1024,768+24)\). As shown in Figure 6 and 7, configurations \((178,1834,6)\) and \((198,2406,9)\) outperform \(\text{BFBP-CS}^3\) and are close to \(\text{CA-SCL8}\) with 24 CRC bits. The configurations \((0,2174,9)\) and \((0,1090,6)\) also have better performance than \(\text{BFBP-CS}^3\). However, \((0,1090,6)\) shows an error floor at an FER of 10^{-6}. Therefore, the numerical results for the codes \((1024,256+24)\) and \((1024,768+24)\) show similar characteristics to those of the configurations of \((2048,1024+24)\).
FIGURE 7. FER comparison of the CA-SCL, BFBP-CS\(^3\), and BP-MF-MC decoders. The polar codes are (1024,768+24) and (1024,768+16). The code (1024,768+16) is designed from the reliability table in [17]. The configurations for (1024,768+24) are (198,2406,9) and (0,2174,9).

that of (440,4412,9) for code (2048,1024+24), as shown in Figure 5. The polar codes (1024,256+11) and (1024,768+16) are designed from the reliability table in [17]. These two codes have the best performance for the CA-SCL\(^8\) decoder.

As shown in Figure 6, CA-SCL\(^8\) with 11 CRC bits slightly outperforms CA-SCL\(^8\) with 24 CRC bits and has an approximately 0.5 dB performance gain over (178,1834,6) at FER = 10\(^{-6}\). For the code (1024,768+24), CA-SCL\(^8\) with 16 CRC bits is 0.3 dB better than (198,2406,9) at FER = 10\(^{-6}\), as shown in Figure 7.

C. COMPLEXITY

The average number of iterations for various decoders is shown in Figure 8. The target code is (2048,1024+24). It is observed that the configuration (1760,4412,9) uses the highest average number of iterations among all the configurations. The configurations (440,4412,9), (440,4412,\(\infty\)) and (440,1760,\(\infty\)) use more iterations than BFBP-CS\(^3\) while using fewer iterations than (1760,4412,9). These three configurations overlap each other in high-SNR regions. (0,3412,9) and (0,4412,\(\infty\)) converge to BFBP-CS\(^3\) and require slightly more iterations than the conventional BP decoder. Furthermore, the complexity of (0,3412,9) is much higher than that of (0,4412,\(\infty\)) if \(E_b/N_0 \leq 2.5\) dB. Therefore, although the dynamic flipping set can improve performance, it increases computational complexity, which may decrease the throughput of hardware implementation due to the large resource requirement.

Next, we perform complexity comparisons between the proposed BP-MF-MC and conventional decoders based on the simulation results. Let \(I\) be the average number of iterations for BP-based decoders, such as BP-MF-MC and BFBP-CS\(^\omega\). Here, for the BP decoder, one computation by a PE given in (1)-(4) is taken to have unit complexity. Thus, the computational complexity of the BP-based decoder is calculated using 2\(Nn\bar{I}\). The complexity of the CA-SCL decoder is taken from [16]. The comparison results are given in Table 2. Please note that the complexity for the configurations given in Table 2 is the average number of PEs when implementing all the decoders in software.

TABLE 2. Computational complexity of various decoders. The target code is (2048,1024+24).

| Decoders   | \(E_b/N_0\) (dB) |
|------------|------------------|
|            | 2.25             | 2.5               | 2.75            | 3               |
| BFBP-CS\(^3\) | 2.229,250        | 1.095,291        | 617,710         | 401,473         |
| (440,4412,9) | 6.066,911        | 3.364,334        | 1.719,490       | 1.368,233       |
| (0,3412,9)   | 4.054,084        | 1.817,950        | 654,915         | 462,730         |
| CA-SCL\(^16\) | 284,976         |                  |                  |                 |
| CA-SCL\(^32\) | 563,504         |                  |                  |                 |

The table shows that the complexity of BP-MF-MC with the configuration (440,4412,9) decreases with SNR. At \(E_b/N_0 = 3\) dB, the complexity of (440,4412,9) is approximately two times as much as that of CA-SCL\(^32\) or three times as much as that of BFBP-CS\(^3\). The complexity of (0,3412,9) is close to that of CA-SCL\(^32\) or BFBP-CS\(^3\) if \(E_b/N_0 \geq 2.75\) dB. Furthermore, the average complexity of the configurations given in Table 2 is much higher than that of the CA-SCL decoders if \(E_b/N_0 \leq 2.5\) dB.

VI. CONCLUSION

In this paper, we propose the BP-MF-MC algorithm for polar codes and present the numerical results for two polar codes. The simulation results show that BP-MF-MC performs similarly to the CA-SCL\(^16\) algorithm. Although the complexity of the proposed algorithm can approach that of the CA-SCL\(^32\) decoder in high-SNR regions, it is important for future studies to lower the complexity of BP-MF-MC in low-SNR regions so that it can be used in practice.
REFERENCES

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

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

[3] E. Arikan, “Polar codes: A pipelined implementation,” in *Proc. 4th Int. Symp. Broadband Commun. (ISBC)*, Jul. 2010, pp. 413–416.

[4] C. Simsek and K. Turk, “Simplified early stopping criterion for belief-propagation polar code decoders,” *IEEE Commun. Lett.*, vol. 20, no. 8, pp. 1515–1518, Aug. 2016.

[5] 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.

[6] Y. Yan, X. Zhang, and B. Wu, “Simplified early stopping criterion for belief-propagation polar code decoder based on frozen bits,” *IEEE Access*, vol. 7, pp. 134691–134696, 2019.

[7] S. M. Abbas, Y. Fan, J. Chen, and C.-Y. Tsui, “High-throughput and energy-efficient belief propagation polar code decoder,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 3, pp. 1098–1111, Mar. 2017.

[8] J. Sha, X. Liu, Z. Wang, and X. Zeng, “A memory efficient belief-propagation decoder for polar codes,” *China Commun.*, vol. 12, no. 5, pp. 34–41, May 2015.

[9] Y. Zhang, A. Liu, X. Pan, S. He, and C. Gong, “A generalization belief propagation decoding algorithm for polar codes based on particle swarm optimization,” *Math. Problems Eng.*, vol. 2014, 2014, Art. no. 606913.

[10] Y. Zhang, A. Liu, X. Pan, Z. Ye, and C. Gong, “A modified belief propagation polar decoder,” *IEEE Commun. Lett.*, vol. 18, no. 7, pp. 1091–1094, Jul. 2014.

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

[12] N. Doan, S. A. Hashemi, M. Mondelli, and W. J. Gross, “On the decoding of polar codes on permuted factor graphs,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.

[13] X. Wang, Z. Zheng, J. Li, L. Shan, and Z. Li, “Belief propagation bit-strengthening decoder for polar codes,” *IEEE Commun. Lett.*, vol. 23, no. 11, pp. 1958–1961, Nov. 2019.

[14] G. Li, J. Mu, X. Jiao, J. Guo, and X. Liu, “Enhanced belief propagation decoding of polar codes by adapting the parity-check matrix,” *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 1, p. 40, Dec. 2017.

[15] N. Doan, S. A. Hashemi, E. N. Mambou, T. Tonnellier, and W. J. Gross, “Neural belief propagation decoding of CRC-polar concatenated codes,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[16] Y. Yu, Z. Pan, N. Liu, and X. You, “Belief propagation bit-flip decoder for polar codes,” *IEEE Access*, vol. 7, pp. 10937–10946, 2019.

[17] *NR: Multiplexing and Channel Coding (Release 15)*, document TS 38.212, 3GPP, 2019. [Online]. Available: https://www.3gpp.org/ftp/Specs/2019-12/Rel-15/38.212

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

JIANYONG ZHANG (Member, IEEE) received the B.S. and Ph.D. degrees in optics from the Beijing Institute of Technology, Beijing, China, in 2000 and 2004, respectively.

In 2006, he joined the Institute of Lightwave Technology, Beijing Jiaotong University, Beijing, where he is currently an Associate Professor and a Ph.D. Supervisor. He has authored or coauthored more than 40 academic articles in peer-reviewed international journals and conferences. His research interests include classical and modern coding theory and information theory and their applications in optical and wireless communications.

MUGUANG WANG (Member, IEEE) received the B.S. degree in optics from Shandong University, Jinan, China, in 1999, and the Ph.D. degree in electrical engineering from Beijing Jiaotong University, Beijing, China, in 2004.

In 2004, he joined the Key Laboratory of All-Optical Network and Advanced Telecommunication Network, Institute of Lightwave Technology, Ministry of Education, Beijing Jiaotong University, Beijing, where he is currently a Full Professor and an Associate Director with the Institute of Lightwave Technology. He has authored or coauthored more than 100 academic articles in peer-reviewed international journals and conferences. His main current research interests include optical fiber sensing, microwave photonics, optical fiber components, and communications. He was a recipient of the New Century Excellent Talents in University, Ministry of Education, in 2010, the Beijing Higher Education Young Elite Teacher, in 2014, and the Beijing Nova of Science and Technology, in 2008.