Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 895782, 14 pages
http://dx.doi.org/10.1155/2014/895782

Research Article

A Parallel Decoding Algorithm for Short Polar Codes Based on Error Checking and Correcting

Yingxian Zhang, Xiaofei Pan, Kegang Pan, Zhan Ye, and Chao Gong

Laboratory of Satellite Communications, College of Communications Engineering, PLA University of Science and Technology, Nanjing 210007, China

Correspondence should be addressed to Yingxian Zhang; jenus.xyz@gmail.com

Received 4 May 2014; Accepted 8 July 2014; Published 23 July 2014

Academic Editor: Lei Cao

We propose a parallel decoding algorithm based on error checking and correcting to improve the performance of the short polar codes. In order to enhance the error-correcting capacity of the decoding algorithm, we first derive the *error-checking equations* generated on the basis of the frozen nodes, and then we introduce the method to check the errors in the input nodes of the decoder by the solutions of these equations. In order to further correct those checked errors, we adopt the method of modifying the probability messages of the error nodes with constant values according to the maximization principle. Due to the existence of multiple solutions of the *error-checking equations*, we formulate a CRC-aided optimization problem of finding the optimal solution with three different target functions, so as to improve the accuracy of error checking. Besides, in order to increase the throughput of decoding, we use a parallel method based on the decoding tree to calculate probability messages of all the nodes in the decoder. Numerical results show that the proposed decoding algorithm achieves better performance than that of some existing decoding algorithms with the same code length.

1. Introduction

Due to the ability of achieving Shannon capacity and its low encoding and decoding complexity, the polar codes have received much attention in recent years [1–20]. However, compared to some original coding schemes such as LDPC and Turbo codes, the polar codes have a remarkable drawback; that is, the performance of the codes in the finite length regime is limited [2, 3]. Hence, researchers have proposed many decoding algorithms to improve the performance of the codes [4–19].

In [4, 5], a list successive-cancelation (SCL) decoding algorithm was proposed with consideration of L successive-cancelation (SC) [1] decoding paths, and the results showed that performance of SCL was very close to that of maximum-likelihood (ML) decoding. Then, in [6], another decoding algorithm derived from SC called stack successive-cancelation (SCS) was introduced to decrease the time complexity of the SCL. In particular, with CRC aided, SCL yielded better performance than that of some Turbo codes, as shown in [7]. However, due to the serial processing nature of the SC, the algorithms in [4–7] suffered a low decoding throughput and high latency. Based on this observation, some improved versions of SC were proposed with the explicit aim to increase throughput and reduce the latency without sacrificing error-rate performance, such as simplified successive-cancellation (SSC) [8], maximum-likelihood SSC (ML-SSC) [9], and repetition single parity check ML-SSC (RSM-SSC) [10, 11]. Besides those SC based algorithms, researchers had also investigated some other algorithms. In [12, 13], the ML and maximum a posteriori (MAP) decoding were proposed for the short polar codes. And in [14], a linear programming decoder was introduced for the binary erase channels (BECs). With the factor graph representation of polar codes [15], authors in [16, 17] showed that belief propagation (BP) polar decoding had particular advantages with respect to the decoding throughput, while the performance was better than that of the SC and some improved SC decoding. What is more is that, with the minimal stopping set optimized, results of [18, 19] had shown that the error floor performance of polar codes was superior to that of LDPC codes.

Indeed, all the decoding algorithms in [4–19] can improve the performance of polar codes to a certain degree. However, as the capacity achieving coding scheme, the results of those algorithms are disappointing. Hence, we cannot help wondering why the performance of the polar codes with finite length is inferior to that of the existing coding schemes and how we can improve it. To answer the questions, we need to make a further analysis of those decoding algorithms in [4–19].

For the decoding algorithms with serial processing, there has been the problem of error propagation except the low decoding throughput and high latency [20, 21]. That is to say, errors which occurred in the previous node will lead to the error decoding of the later node. However, none of the existing serial processing algorithms has considered this observation. Furthermore, it is noticed from the factor graph of polar codes in [15] that the degree of the check or variable nodes in the decoder is 2 or 3, which will weaken the error-correcting capacity of the decoding, as compared to the LDPC codes with the average degree usually greater than 3 [22, 23]. Hence, the performance of the polar codes is inferior to that of LDPC codes with the same length [18, 19]. What is more is that BP polar decoding needs more iterations than that of LDPC codes, as shown in [16, 17, 22, 23]. Therefore, in order to improve the performance of a decoding algorithm for polar codes, it is important to enhance the error-correcting capacity of the algorithm.

Motivated by aforementioned observations, we propose a parallel decoding algorithm for short polar codes based on error checking and correcting in this paper. We first classify the nodes of the proposed decoder into two categories: *information nodes* and *frozen nodes*, values of which are determined and independent of decoding algorithms. Then, we introduce the method to check the errors in the input nodes of the decoder, with the solutions of the *error-checking equations* generated based on the frozen nodes. To correct those checked errors, we modify the probability messages of the error nodes with constant values according to the maximization principle. Through delving the error-checking equations solving problem, we find that there exist multiple solutions for those equations. Hence, as to check the errors as accurately as possible, we further formulate a CRC-aided optimization problem of finding the optimal solution of the error-checking equations with three different target functions. Besides, we also use a parallel method based on the *decoding tree representations of the nodes* to calculate probability messages in order to increase the throughput of decoding. The main contributions of this paper can be summarized as follows.

(i) An error-checking algorithm for polar decoding based on the error-checking equations solving is introduced; furthermore, as to enhance the accuracy of the error checking, a CRC-aided optimization problem of finding the optimal solution is formulated.

(ii) To correct the checked errors, we propose a method of modifying the probability messages of the error nodes according to the maximization principle.

(iii) In order to improve the throughput of the decoding, we propose a parallel probability messages calculating method based on the decoding tree representation of the nodes.

(iv) The whole procedure of the proposed decoding algorithm is described with the form of pseudocode, and the complexity of the algorithm is also analyzed.

The finding of this paper suggests that, with the error checking and correcting, the error-correcting capacity of the decoding algorithm can be enhanced, which will yield a better performance at cost of certain complexity. Specifically, with the parallel probability messages calculating, the throughput of decoding is higher than the serial process based decoding algorithms. All of these results are finally proved by our simulation work.

The remainder of this paper is organized as follows. In Section 2, we explain some notations and introduce certain preliminary concepts used in the subsequent sections. And in Section 3, the method of the error checking for decoding based on the error-checking equations is described in detail. In Section 4, we introduce the methods of probability messages calculating and error correcting, and after the formulation of the CRC-aided optimization problem of finding the optimal solution, the proposed decoding algorithm with the form of pseudocode is presented. Then, the complexity of our algorithm is analyzed. Section 5 provides the simulation results for the complexity and bit error performance. Finally, we make some conclusions in Section 6.

2. Preliminary

2.1. Notations. In this work, the blackboard bold letters, such as $\mathbb{X}$, denote the sets, and $|\mathbb{X}|$ denotes the number of elements in $\mathbb{X}$. The notation u_0^{N-1} denotes an N-dimensional vector $(u_0, u_1, \ldots, u_{N-1})$, and u_i^j indicates a subvector $(u_i, u_{i+1}, \ldots, u_{j-1}, u_j)$ of u_0^{N-1}, $0 \le i, j \le N-1$. When $i > j$, u_i^j is an empty vector. Further, given a vector set $\mathbb{U}$, vector $\vec{u}_i$ is the ith element of $\mathbb{U}$.

The matrixes in this work are denoted by bold letters. The subscript of a matrix indicates its size; for example, $\mathbf{A}_{N\times M}$ represents an $N \times M$ matrix $\mathbf{A}$. Specifically, the square matrixes are written as $\mathbf{A}_N$, size of which is $N \times N$, and $\mathbf{A}_N^{-1}$ is the inverse of $\mathbf{A}_N$. Furthermore, the Kronecker product of two matrixes $\mathbf{A}$ and $\mathbf{B}$ is written as $\mathbf{A} \otimes \mathbf{B}$, and the nth Kronecker power of $\mathbf{A}$ is $\mathbf{A}^{\otimes n}$.

During the procedure of the encoding and decoding, we denote the intermediate node as $v(i, j)$, $0 \le i \le n, 0 \le j \le N-1$, where $N = 2^n$ is the code length. Besides, we also indicate the probability values of the intermediate node $v(i, j)$ being equal to 0 or 1 as $p_{v(i,j)}(0)$ or $p_{v(i,j)}(1)$.

Throughout this Paper, "$\oplus$" denotes the Modulo-Two Sum, and "$\sum_{i=0}^{M} \oplus x_i$" means "$x_0 \oplus x_1 \oplus, \ldots, \oplus x_M$".

2.2. Polar Encoding and Decoding. A polar coding scheme can be uniquely defined by three parameters: block-length $N = 2^n$, code rate $R = K/N$, and an information set $\mathbb{I} \subset \mathbb{N} = \{0, 1, \ldots, N-1\}$, where $K = |\mathbb{I}|$. With these three parameters,

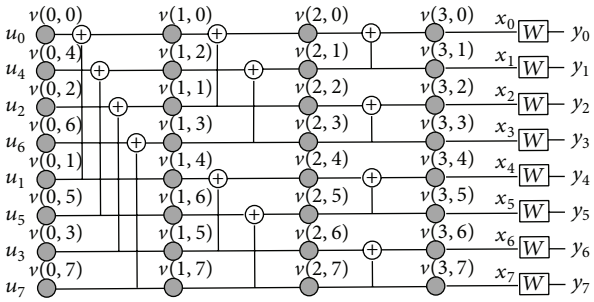

Figure 1: Construction of the polar encoding with length $N = 8$.

a source binary vector u_0^{N-1} consisting of K information bits and $N - K$ frozen bits can be mapped a codeword x_0^{N-1} by a linear matrix $\mathbf{G}_N = \mathbf{B}_N\mathbf{F}_2^{\otimes n}$, where $\mathbf{F}_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$, $\mathbf{B}_N$ is a bit-reversal permutation matrix defined in [1], and $x_0^{N-1} = u_0^{N-1}\mathbf{G}_N$.

In practice, the polar encoding can be completed with the construction shown in Figure 1, where the gray circle nodes are the intermediate nodes. And the nodes in the leftmost column are the input nodes of encoder, values of which are equal to binary source vector; that is, $v(0, i) = u_i$, while the nodes in the rightmost column are the output nodes of encoder, $v(n, i) = x_i$. Based on the construction, a codeword x_0^7 is generated by the recursively linear transformation of the nodes between adjacent columns.

After the procedure of the polar encoding, all the bits in the codeword x_0^{N-1} are passed to the N-channels, which are consisted of N independent channels of W, with a transition probability of $W(y_i \mid x_i)$, where y_i is ith element of the received vector y_0^{N-1}.

At the receiver, the decoder can output the estimated codeword $\widehat{x}_0^{N-1}$ and the estimated source binary vector $\widehat{u}_0^{N-1}$ with different decoding algorithms [1–19]. It is noticed from [1–19] that the construction of all the decoders is the same as that of the encoder; here, we make a strict proof for that with the mathematical formula in the following theorem.

Theorem 1. *For the generation matrix of the a polar code $\mathbf{G}_N$, there exists*

$$\mathbf{G}_N^{-1} = \mathbf{G}_N. \tag{1}$$

That is to say, for the decoding of the polar codes, one will have

$$\widehat{u}_0^{N-1} = \widehat{x}_0^{N-1}\mathbf{G}_N^{-1} = \widehat{x}_0^{N-1}\mathbf{G}_N, \tag{2}$$

where $\mathbf{G}_N^{-1}$ is construction matrix of the decoder.

Proof. The proof of Theorem 1 is based on the matrix transformation, which is shown detailedly in Appendix A. □

Hence, as for the polar encoder shown in Figure 1, there is

$$\mathbf{G}_8 = \mathbf{G}_8^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{3}$$

Furthermore, we have the construction of the decoder as shown in Figure 2(a), where nodes in the rightmost column are the input nodes of the decoder, and the output nodes are the nodes in the leftmost column. During the procedure of the decoding, the probability messages of the received vector are recursively propagated from the rightmost column nodes to the leftmost column nodes. Then, the estimated source binary vector $\widehat{u}_0^7$ can be decided by

$$\widehat{u}_i = \begin{cases} 0, & p_{v(0,i)}(0) > p_{v(0,i)}(1) \\ 1, & \text{otherwise.} \end{cases} \tag{4}$$

In fact, the input probability messages of the decoder depend on the transition probability $W(y_i \mid x_i)$ and the received vector y_0^7; hence, there is

$$\begin{aligned} p_{v(n,i)}(0) &= W\left(y_i \mid x_i = 0\right), \\ p_{v(n,i)}(1) &= W\left(y_i \mid x_i = 1\right). \end{aligned} \tag{5}$$

For convenience of expression, we will write the input probability messages $W(y_i \mid x_i = 0)$ and $W(y_i \mid x_i = 1)$ as $q_i(0)$ and $q_i(1)$, respectively, in the rest of this paper. Therefore, we further have

$$\begin{aligned} p_{v(n,i)}(0) &= q_i(0), \\ p_{v(n,i)}(1) &= q_i(1). \end{aligned} \tag{6}$$

2.3. Frozen and Information Nodes. In practice, due to the input of frozen bits [1], values of some nodes in the decoder are determined, which are independent of the decoding algorithm, as the red circle nodes illustrated in Figure 2(a) (code construction method is the same as [1]). Based on this observation, we classify the nodes in the decoder into two categories: the nodes with determined values are called *frozen nodes*, and the other nodes are called *information nodes*, as the gray circle nodes shown in Figure 2(a). In addition, with the basic process units of the polar decoder shown in Figure 2(b), we have the following lemma.

Lemma 2. *For the decoder of a polar code with rate $R < 1$, there must exist some frozen nodes, the number of which depends on the information set $\mathbb{I}$.*

Proof. The proof of Lemma 2 can be easily finished based on the process units of the polar decoder as shown in Figure 2(b), where $v(i, j_1)$, $v(i, j_2)$, $v(i + 1, j_3)$, and $v(i + 1, j_4)$ are the four nodes of the decoder. □

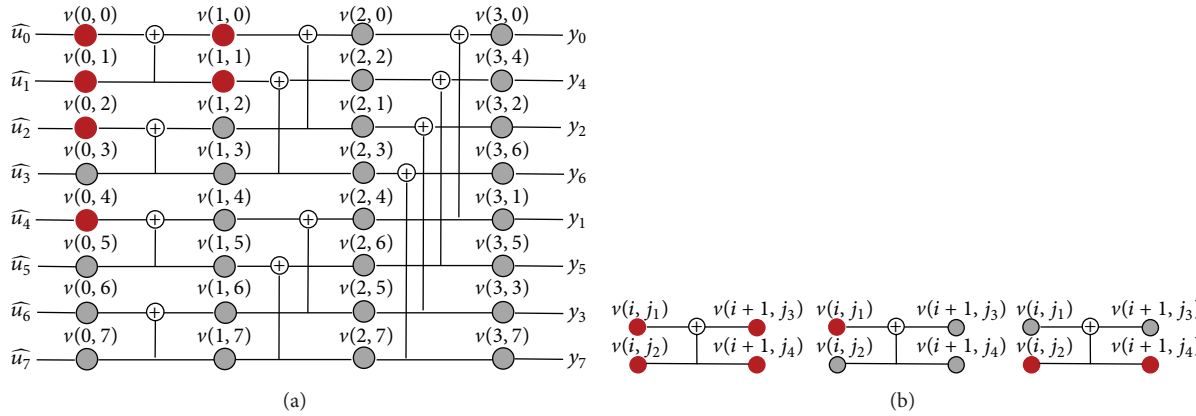

FIGURE 2: (a) Construction of polar decoding with code length $N = 8$. (b) Basic process units of the polar decoder.

Lemma 2 has shown that, for a polar code with rate $R < 1$, the frozen nodes are always existing; for example, the frozen nodes in Figure 2(a) are $v(0,0)$, $v(1,0)$, $v(0,1)$, $v(1,1)$, $v(0,2)$, and $v(0,4)$. For convenience, we denote the frozen node set of a polar code as $\mathbb{V}_F$, and we assume that the default value of each frozen node is 0 in the subsequent sections.

2.4. Decoding Tree Representation. It can be found from the construction of the decoder in Figure 2(a) that the decoding of a node $v(i, j)$ can be equivalently represented as a binary decoding tree with some input nodes, where $v(i, j)$ is the root node of that tree, and the input nodes are the leaf nodes. The height of a decoding tree is as most as $\log_2 N$, and each of the intermediate node has one or two son nodes. As illustrated in Figure 3, the decoding trees for frozen nodes $v(0,0)$, $v(0,1)$, $v(0,2)$, $v(0,4)$, $v(1,0)$, and $v(1,1)$ in Figure 2(a) are given.

During the decoding procedure, probability messages of $v(0,0)$, $v(0,1)$, $v(0,2)$, $v(0,4)$, $v(1,0)$, and $v(1,1)$ will strictly depend on the probability messages of the leaf nodes as the bottom nodes shown in Figure 3. In addition, based on the (2), we further have

$$\begin{aligned}
v(0,0) &= \sum_{i=0}^{7} \oplus v(3,i) \\
v(1,0) &= v(3,0) \oplus v(3,1) \oplus v(3,2) \oplus v(3,3) \\
v(0,1) &= v(3,4) \oplus v(3,5) \oplus v(3,6) \oplus v(3,7) \\
v(1,1) &= v(3,4) \oplus v(3,5) \oplus v(3,6) \oplus v(3,7) \\
v(0,2) &= v(3,2) \oplus v(3,3) \oplus v(3,6) \oplus v(3,7) \\
v(0,4) &= v(3,1) \oplus v(3,3) \oplus v(3,5) \oplus v(3,7).
\end{aligned} \tag{7}$$

To generalize the decoding tree representation for the decoding, we introduce the following Lemma.

Lemma 3. *In the decoder of a polar code with length $N = 2^n$, there is a unique decoding tree for each node $v(i, j)$, the leaf nodes set of which is indicated as $\mathbb{V}^L_{v(i,j)}$. And if $j \neq N-1$, the number of the leaf nodes is even; that is,*

$$\begin{aligned}
v(i,j) &= \sum_{k=0}^{M/2} \oplus v(n, j_{2k}), \\
0 \le j_{2k} \le N-1, &\qquad v(n, j_{2k}) \in \mathbb{V}^L_{v(i,j)},
\end{aligned} \tag{8}$$

where $M = |\mathbb{V}^L_{v(i,j)}|$ and $(M \bmod 2) = 0$. While if $j = N-1$, M is equal to 1, and it is true that

$$v(i, N-1) = v(n, N-1). \tag{9}$$

Proof. The proof of Lemma 3 is based on (2) and construction of the generation matrix. It is easily proved that, except the last column (only one "1" element), there is an even number of "1" elements in all the other columns of $\mathbf{F}_2^{\otimes n}$. As $\mathbf{B}_N$ is a bit-reversal permutation matrix, which is generated by permutation of rows in $\mathbf{I}_N$, hence, the generation matrix $\mathbf{G}_N$ has the same characteristic as $\mathbf{F}_2^{\otimes n}$ (see the proof of Theorem 1). Therefore, (8) and (9) can be easily proved by (2). □

Lemma 3 has clearly shown the relationship between the input nodes and other intermediate nodes of the decoder, which is useful for error checking and probability messages calculation introduced in the subsequent sections.

3. Error Checking for Decoding

As analyzed in Section 1, the key problem to improve the performance of polar codes is to enhance the error-correcting capacity of the decoding. In this section, we will show how to achieve the goal.

3.1. Error Checking by the Frozen Nodes. It is noticed from Section 2.3 that the values of the frozen nodes are determined. Hence, if the decoding is correct, the probability messages of any frozen node $v(i, j)$ must satisfy the condition of $p_{v(i,j)}(0) > p_{v(i,j)}(1)$ (the default value of frozen nodes is 0), which is called *reliability condition* throughout this

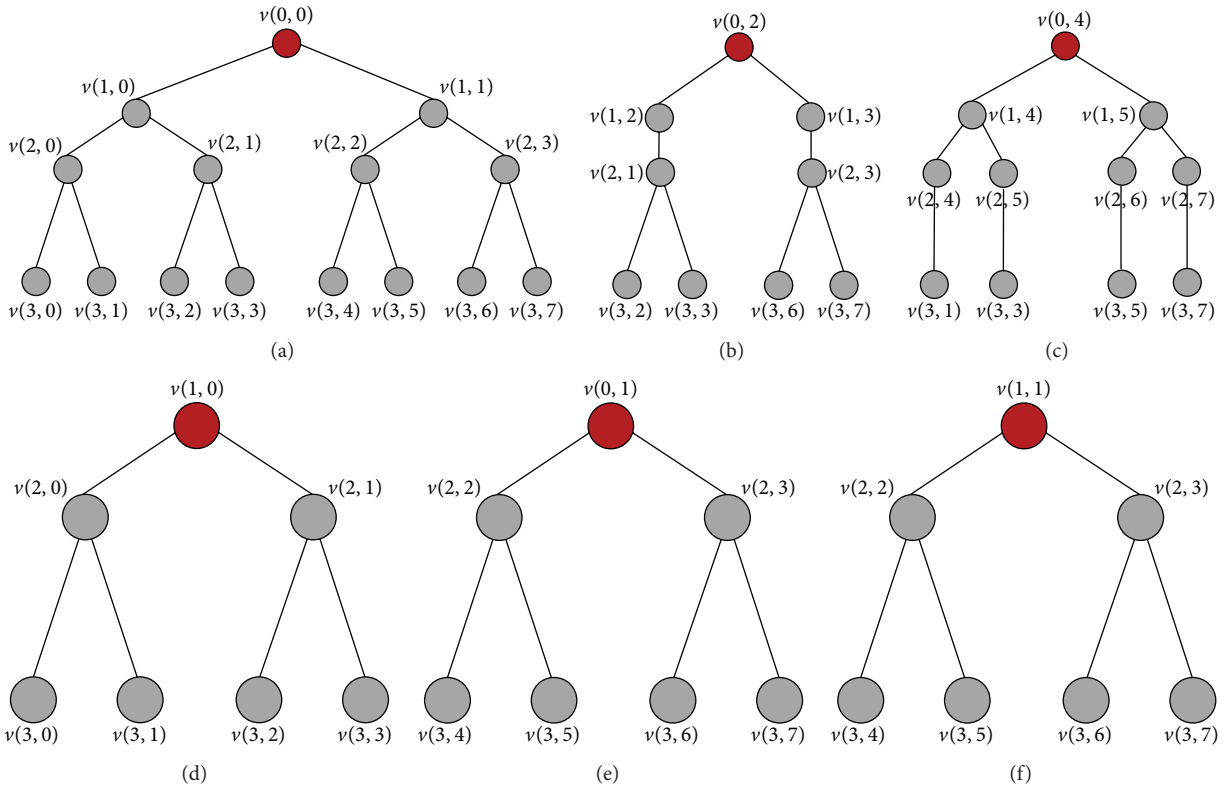

Figure 3: The decoding trees for the nodes $v(0,0)$, $v(0,1)$, $v(1,0)$, $v(1,1)$, $v(0,2)$, and $v(0,4)$.

paper. While in practice, due to the noise of received signal, there may exist some frozen nodes unsatisfying the reliability condition, which indicates that there must exist errors in the input nodes of the decoder. Hence, it is exactly based on this observation that we can check the errors during the decoding. To further describe detailedly, a theorem is introduced to show the relationship between the reliability condition of the frozen nodes and the errors in the input nodes of the decoder.

Theorem 4. *For any frozen node $v(i,j)$ with a leaf node set $\mathbb{V}^L_{v(i,j)}$, if the probability messages of $v(i,j)$ do not satisfy the reliability condition during the decoding procedure, there must exist an odd number of error nodes in $\mathbb{V}^L_{v(i,j)}$; otherwise, the error number will be even (including 0).*

Proof. For the proof of Theorem 4 see Appendix B for detail. □

Theorem 4 has provided us an effective method to detect the errors in the leaf nodes set of the frozen node. For example, if the probability messages of the frozen node $v(0,0)$ in Figure 2 do not satisfy the reliability condition, that is, $p_{v(0,0)}(0) \leq p_{v(0,0)}(1)$, it can be confirmed that there must exist errors in the set of $\{v(3,0), v(3,1), \ldots, v(3,7)\}$, and the number of these errors may be 1 or 3 or 5 or 7. That is to say, through checking the reliability condition of the frozen nodes, we can confirm existence of the errors in the input nodes of the decoder, which is further presented as a corollary.

Corollary 5. *For a polar code with the frozen node set $\mathbb{V}_F$, if $\exists v(i,j) \in \mathbb{V}_F$ and $v(i,j)$ does not satisfy the reliability condition, there must exist errors in the input nodes of decoder.*

Proof. The proof of Corollary 5 is easily completed based on Theorem 4. □

Corollary 5 has clearly shown that, through checking the probability messages of each frozen node, errors in the input nodes of decoder can be detected.

3.2. Error-Checking Equations. As aforementioned, errors in the input nodes can be found with probability messages of the frozen node, but there still is a problem, which is how to locate the exact position of each error. To solve the problem, a parameter called *error indicator* is defined for each input node of the decoder. And for the input node $v(n,i)$, the error indicator is denoted as c_i, value of which is given by

$$c_i = \begin{cases} 1, & v(n,i) \text{ is error} \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

That is to say, by the parameter of error indicator, we can determine whether an input node is error or not.

Hence, the above problem can be transformed into how to obtain the error indicator of each input node. Motivated by this observation, we introduce another corollary of Theorem 4.

Corollary 6. *For any frozen node $v(i,j)$ with a leaf node set $\mathbb{V}^L_{v(i,j)}$, there is*

$$\begin{aligned}&\left(\sum_{k=0}^{M-1} c_{i_k}\right) \bmod 2 = 1, \quad p_{v(i,j)}(0) \le p_{v(i,j)}(1)\\ &\left(\sum_{k=0}^{M-1} c_{i_k}\right) \bmod 2 = 0, \quad \textit{otherwise},\end{aligned} \tag{11}$$

where $M = |\mathbb{V}^L_{v(i,j)}|$, $v(n, i_k) \in \mathbb{V}^L_{v(i,j)}$, and $N = 2^n$ is code length. Furthermore, under the field of $GF(2)$, (11) can be written as

$$\begin{aligned}&\sum_{k=0}^{M-1} \oplus c_{i_k} = 1, \quad p_{v(i,j)}(0) \le p_{v(i,j)}(1)\\ &\sum_{k=0}^{M-1} \oplus c_{i_k} = 0, \quad \textit{otherwise}.\end{aligned} \tag{12}$$

Proof. The proof of Corollary 6 is based on Lemma 3 and Theorem 4, and here we ignore the detailed derivation process. □

Corollary 6 has shown that the problem of obtaining the error indicator can be transformed to find solutions of (12) under the condition of (11). In order to introduce more specifically, here, we will take an example based on the decoder in Figure 2(a).

Example 7. We assume that frozen nodes $v(0,0)$, $v(1,0)$, $v(0,2)$, and $v(0,4)$ do not satisfy the reliability condition; hence, based on Theorem 4 and Corollary 6, there are equations as

$$\begin{aligned}&\sum_{i=0}^{7} \oplus c_i = 1\\ &c_0 \oplus c_1 \oplus c_2 \oplus c_3 = 1\\ &c_4 \oplus c_5 \oplus c_6 \oplus c_7 = 0\\ &c_4 \oplus c_5 \oplus c_6 \oplus c_7 = 0\\ &c_2 \oplus c_3 \oplus c_6 \oplus c_7 = 1\\ &c_1 \oplus c_3 \oplus c_5 \oplus c_7 = 1.\end{aligned} \tag{13}$$

Furthermore, (13) can be written as matrix form, which is

$$\mathbf{E}_{68}\left(c_0^7\right)^T = \begin{bmatrix}1&1&1&1&1&1&1&1\\1&1&1&1&0&0&0&0\\0&0&0&0&1&1&1&1\\0&0&0&0&1&1&1&1\\0&0&1&1&0&0&1&1\\0&1&0&1&0&1&0&1\end{bmatrix}\begin{bmatrix}c_0\\c_1\\c_2\\c_3\\c_4\\c_5\\c_6\\c_7\end{bmatrix} = \left(\gamma_0^5\right)^T, \tag{14}$$

where $\gamma_0^5 = (1,\ 1,\ 0,\ 0,\ 1,\ 1,)$ and $\mathbf{E}_{68}$ is the coefficient matrix with size of 6×8. Therefore, by solving (14), we will get the error indicator vector of input nodes in Figure 2. In order to further generalize the above example, we provide a lemma.

Lemma 8. *For a polar code with the code length N, code rate $R = K/N$, and frozen node set $\mathbb{V}_F$, we have the error-checking equations as*

$$\mathbf{E}_{MN}\left(c_0^{N-1}\right)^T = \left(\gamma_0^{M-1}\right)^T, \tag{15}$$

where c_0^{N-1} is the error indicator vector and $M = |\mathbb{V}_F|$, $M \ge N - K$. $\mathbf{E}_{MN}$ is called error-checking matrix, elements of which are determined by the code construction method, and γ_0^{M-1} is called error-checking vector, elements of which depend on the probability messages of the frozen nodes in $\mathbb{V}_F$; that is, $\forall v_i \in \mathbb{V}_F$, $0 \le i \le M-1$, there is a unique $\gamma_i \in \gamma_0^{M-1}$ such that

$$\gamma_i = \begin{cases}1, & p_{v_i}(0) \le p_{v_i}(1)\\ 0, & p_{v_i}(0) > p_{v_i}(1).\end{cases} \tag{16}$$

Proof. The proof of the Lemma 8 is based on (10)–(14), Lemma 3, and Theorem 4, which will be ignored here. □

3.3. Solutions of Error-Checking Equations. Lemma 8 provides a general method to determine the position of errors in the input nodes by the error-checking equations. It is still needed to investigate the existence of solutions of the error-checking equations.

Theorem 9. *For a polar code with code length N and code rate $R = K/N$, there is*

$$\operatorname{rank}(\mathbf{E}_{MN}) = \operatorname{rank}\left(\left[\ \mathbf{E}_{MN}\ \middle|\ \left(\gamma_0^{M-1}\right)^T\ \right]\right) = N - K, \tag{17}$$

where $\left[\ \mathbf{E}_{MN}\ \middle|\ (\gamma_0^{M-1})^T\ \right]$ is the augmented matrix of (15) and $\operatorname{rank}(\mathbf{X})$ is the rank of matrix $\mathbf{X}$.

Proof. For the proof of Theorem 9 see Appendix C for detail. □

It is noticed from Theorem 9 that there must exist multiple solutions for error-checking equations; therefore, we further investigate the general expression of solutions of the error-checking equations as shown in the following corollary.

Corollary 10. *For a polar code with code length N and code rate $R = K/N$, there exists a transformation matrix $\mathbf{P}_{NM}$ in the field of $GF(2)$ such that*

$$\left[\begin{array}{c|c} \mathbf{E}_{MN} & \left(\gamma_0^{M-1}\right)^T \end{array}\right] \xrightarrow{\mathbf{P}_{NM}} \left[\begin{array}{c|c|c} \mathbf{I}_H & \mathbf{A}_{HK} & \left(\overline{\gamma}_0^{H-1}\right)^T \\ \hline \mathbf{0}_{(M-H)H} & \mathbf{0}_{(M-H)K} & \mathbf{0}_{(M-H)\times 1} \end{array}\right], \tag{18}$$

where $H = N - K$, $\mathbf{A}_{HK}$ is the submatrix of transformation result of $\mathbf{E}_{MN}$, and $\overline{\gamma}_0^{H-1}$ is the subvector of transformation result of γ_0^{M-1}. Based on (18)*, the general solutions of error-checking equations can be obtained by*

$$\left(\widehat{c}_0^{N-1}\right)^T = \left[\begin{array}{c} \left(\widehat{c}_K^{N-1}\right)^T \\ \hline \left(\widehat{c}_0^{K-1}\right)^T \end{array}\right] = \left[\begin{array}{c} \mathbf{A}_{HK}\left(\widehat{c}_0^{K-1}\right)^T \oplus \left(\overline{\gamma}_0^{H-1}\right)^T \\ \hline \left(\widehat{c}_0^{K-1}\right)^T \end{array}\right], \tag{19}$$

$$\left(c_0^{N-1}\right)^T = \widehat{\mathbf{B}}_N\left(\widehat{c}_0^{N-1}\right)^T, \tag{20}$$

where $\widehat{c}_i \in \{0, 1\}$ and $\widehat{\mathbf{B}}_N$ is an element-permutation matrix, which is determined by the matrix transformation of (18).

Proof. The proof of Corollary 10 is based on Theorem 9 and the linear equation solving theory, which are ignored here. □

It is noticed from (18) and (19) that solutions of the error-checking equations tightly depend on the two vectors: $\overline{\gamma}_0^{H-1}$ and $\widehat{c}_0^{K-1}$. Where $\overline{\gamma}_0^{H-1}$ is determined by the transformation matrix $\mathbf{P}_{NM}$ and the error-checking vector γ_0^{M-1}, and $\widehat{c}_0^{K-1}$ is a random vector. In general, based on $\widehat{c}_0^{K-1}$, the number of solutions for the error-checking equations may be up to 2^K, which is a terrible number for decoding. Although the solutions number can be reduced through the checking of (11), it still needs to reduce the solutions number in order to increase efficiency of error checking. To achieve the goal, we further introduce a theorem.

Theorem 11. *For a polar code with code length $N = 2^n$ and frozen node set $\mathbb{V}_F$, there exists a positive real number δ such that $\forall v(i, j) \in \mathbb{V}_F$; if $(p_{v(i,j)}(0)/p_{v(i,j)}(1)) \geq \delta$, there is*

$$\forall v\left(n, j_k\right) \in \mathbb{V}_{v(i,j)}^L \Longrightarrow c_{j_k} = 0, \tag{21}$$

where $\mathbb{V}_{v(i,j)}^L$ is the leaf nodes set of $v(i, j)$, $0 \leq j_k \leq N - 1$, $0 \leq k \leq |\mathbb{V}_{v(i,j)}^L| - 1$, and the value of δ is related to the transition probability of the channel and the signal power.

Proof. For the proof of Theorem 11 see Appendix D for detail. □

Theorem 11 has shown that, with the probability messages of the frozen node and δ, we can determine the values of some elements in $\widehat{c}_0^{K-1}$ quickly, by which the freedom degree of $\widehat{c}_0^{K-1}$ will be further reduced. Correspondingly, the number of solutions for the error-checking equations will be also reduced.

Based on above results, we take (14) as an example to show the detailed process of solving the error-checking equations. Through the linear transformation of (18), we have $\overline{\gamma}_0^3 = (1, 1, 1, 0)$,

$$\mathbf{A}_4 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}, \tag{22}$$

$$\widehat{\mathbf{B}}_8 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{23}$$

By the element permutation of $\widehat{\mathbf{B}}_8$, we further have $\widehat{c}_0^3 = (c_3, c_5, c_6, c_7)$ and $\widehat{c}_4^7 = (c_0, c_1, c_2, c_4)$. If $(p_{v(0,1)}(0)/p_{v(0,1)}(1)) \geq \delta$, with the checking of (21), there is $(c_3, c_5, c_6, c_7) = (c_3, 0, 0, 0)$, and $(c_0, c_1, c_2, c_4) = (c_3 \oplus 1, c_3 \oplus 1, c_3 \oplus 1, 0)$, which imply that the solutions number will be 2. Furthermore, with the checking of (11), we obtain the exact solution $c_0^7 = (0, 0, 0, 1, 0, 0, 0, 0)$; that is, the 4th input node is error.

It is noticed clearly from the above example that, with the checking of (11) and (21), the number of the solutions can be greatly reduced, which make the error checking more efficient. And, of course, the final number of the solutions will depend on the probability messages of the frozen nodes and δ.

As the summarization of this section, we given the complete process framework of error checking by solutions of the error-checking equations, which is shown in Algorithm 1.

4. Proposed Decoding Algorithm

In this section, we will introduce the proposed decoding algorithm in detail.

4.1. Probability Messages Calculating. Probability messages calculating is an important aspect of a decoding algorithm. Our proposed algorithm is different from the SC and BP algorithms, because the probability messages are calculated based on the decoding tree representation of the nodes in the decoder, and for an intermediate node $v(i, j)$ with only one son node $v(i + 1, j_o)$, $0 \leq j_o \leq N - 1$, there is

$$\begin{aligned} p_{v(i,j)}(0) &= p_{v(i+1,j_o)}(0), \\ p_{v(i,j)}(1) &= p_{v(i+1,j_o)}(1). \end{aligned} \tag{24}$$

Input:
The frozen nodes set, $\mathbb{V}_F$;
The probability messages set of the $\mathbb{V}_F$;
The matrixes, $\mathbf{P}_{NM}$, $\mathbf{A}_{HK}$ and $\widehat{\mathbf{B}}_N$;
Output:
The error indicator vectors set, $\mathbb{C}$;
(1) Getting γ_0^{M-1} with the probability messages set of $\mathbb{V}_F$;
(2) Getting $\overline{\gamma}_0^{H-1}$ with γ_0^{M-1} and $\mathbf{P}_{NM}$;
(3) **for** each $v(i, j) \in \mathbb{V}_F$ **do**
(4) **if** $p_{v(i,j)}(0)/p_{v(i,j)}(1) > \delta$ **then**
(5) Setting the error indicator for each leaf node of $v(i, j)$ to 0;
(6) **end if**
(7) **end for**
(8) **for** each valid of $\widehat{c}_0^{K-1}$ **do**
(9) Getting $\widehat{c}_K^{N-1}$ with $\mathbf{A}_{HK}$ and $\overline{\gamma}_0^{N-K-1}$;
(10) **if** (11) is satisfied **then**
(11) Getting $c_0^{N-1} \in \mathbb{C}$ with $\widehat{\mathbf{B}}_N$;
(12) **else**
(13) Dropping the solution and continuing;
(14) **end if**
(15) **end for**
(16) **return** $\mathbb{C}$;

Algorithm 1: Error checking for decoding.

While if $v(i, j)$ has two son nodes $v(i + 1, j_l)$ and $v(i + 1, j_r)$, $0 \le j_l, j_r \le N - 1$, we will have

$$\begin{aligned} p_{v(i,j)}(0) &= p_{v(i+1,j_l)}(0)\, p_{v(i+1,j_r)}(0) \\ &\quad + p_{v(i+1,j_l)}(1)\, p_{v(i+1,j_r)}(1), \\ p_{v(i,j)}(1) &= p_{v(i+1,j_l)}(0)\, p_{v(i+1,j_r)}(1) \\ &\quad + p_{v(i+1,j_l)}(1)\, p_{v(i+1,j_r)}(0). \end{aligned} \tag{25}$$

Based on (24) and (25), the probability messages of all the variable nodes can be calculated in parallel, which will be beneficial to the decoding throughput.

4.2. Error Correcting. Algorithm 1 in Section 3.3 has provided an effective method to detect errors in the input nodes of the decoder, and, now, we will consider how to correct these errors. To achieve the goal, we propose a method based on modifying the probability messages of the error nodes with constant values according to the maximization principle. Based on the method, the new probability messages of a error node will be given by

$$\begin{aligned} q_i'(0) &= \lambda_0, \qquad q_i(0) > q_i(1) \\ q_i'(0) &= 1 - \lambda_0, \quad \text{otherwise}, \end{aligned} \tag{26}$$

and $q_i'(1) = 1 - q_i'(0)$, where $q_i'(0)$, $q_i'(1)$ are the new probability messages of the error node $v(n, i)$, and λ_0 is a small nonnegative constant; that is, $0 \le \lambda_0 \ll 1$. Furthermore, we will get the new probability vector of the input nodes as

$$\begin{aligned} q_0^{N-1}(0)' &= \left(q_0(0)', q_1(0)', \ldots, q_{N-1}(0)'\right) \\ q_0^{N-1}(1)' &= \left(q_0(1)', q_1(1)', \ldots, q_{N-1}(1)'\right), \end{aligned} \tag{27}$$

where $q_i(0)' = q_i'(0)$ and $q_i(1)' = q_i'(1)$, if the input node $v(n, i)$ is error; otherwise, $q_i(0)' = q_i(0)$ and $q_i(1)' = q_i(1)$. Then, probability messages of all the nodes in the decoder will be recalculated.

In fact, when there is only one error indicator vector output from Algorithm 1, that is, $|\mathbb{C}| = 1$, after the error correcting and the probability messages recalculation, the estimated source binary vector $\widehat{u}_0^{N-1}$ can output directly by the hard decision of the output nodes. While if $|\mathbb{C}| > 1$, in order to minimize the decoding error probability, it needs further research about how to get the optimal error indicator vector.

4.3. Reliability Degree. To find the optimal error indicator vector, we will introduce a parameter called *reliability degree* for each node in the decoder. And for a node $v(i, j)$, the reliability degree $\zeta^{v(i,j)}$ is given by

$$\zeta^{v(i,j)} = \begin{cases} \dfrac{p_{v(i,j)}(0)}{p_{v(i,j)}(1)}, & p_{v(i,j)}(0) > p_{v(i,j)}(1) \\ \dfrac{p_{v(i,j)}(1)}{p_{v(i,j)}(0)}, & \text{otherwise}. \end{cases} \tag{28}$$

The reliability degree indicates the reliability of the node's decision value, and the larger the reliability degree, the higher

the reliability of that value. For example, if the probability messages of the node $v(0,0)$ in Figure 2 are $p_{v(0,0)}(0) = 0.95$ and $p_{v(0,0)}(1) = 0.05$, there is $\zeta^{v(i,j)} = 0.95/0.05 = 19$; that is, the reliability degree of $v(0,0) = 0$ is 19. And in fact, the reliability degree is an important reference parameter for the choice of the optimal error indicator vector, which will be introduced in the following subsection.

4.4. Optimal Error Indicator Vector. As aforementioned, due to the existence of $|\mathbb{C}| > 1$, correspondingly, one node in the decoder may have multiple reliability degrees. We denote the kth reliability degree of node $v(i, j)$ as $\zeta_k^{v(i,j)}$, value of which depends on the kth element of $\mathbb{C}$, that is, $\vec{c}_k$. Based on the definition of reliability degree, we introduce three methods to get the optimal error indicator vector.

The first method is based on the fact that, when there is no noise in the channel, the reliability degree of the node will approach to infinity; that is, $\zeta^{v(i,j)} \to \infty$. Hence, the main consideration is to maximize the reliability degree of all the nodes in decoder, and the target function can be written as

$$\widehat{k} = \underset{\vec{c}_k \in \mathbb{C}}{\operatorname{argmax}} \left\{ \sum_{i=0}^{\log_2 N} \sum_{j=0}^{N} \zeta_k^{v(i,j)} \right\}, \tag{29}$$

where $\vec{c}_{\widehat{k}}$ is the optimal error indicator vector.

To reduce the complexity, we have further introduced two simplified versions of the above method. On one hand, we just maximize the reliability degree of all the frozen nodes; hence, the target function can be written as

$$\widehat{k} = \underset{\vec{c}_k \in \mathbb{C}}{\operatorname{argmax}} \left\{ \sum_{v(i,j) \in \mathbb{V}_F} \zeta_k^{v(i,j)} \right\}. \tag{30}$$

On the other hand, we take the maximization of the output nodes' reliability degree as the optimization target, function of which will be given by

$$\widehat{k} = \underset{\vec{c}_k \in \mathbb{C}}{\operatorname{argmax}} \left\{ \sum_{j=0}^{N-1} \zeta_k^{v(0,j)} \right\}. \tag{31}$$

Hence, the problem of getting the optimal error indicator vector can be formulated as an optimization problem with the above three target functions. What is more is that, with the CRC aided, the accuracy of the optimal error indicator vector can be enhanced. Based on these observations, the finding of the optimal error indicator vector will be divided into the following steps.

(1) Initialization: we first get number L candidates of the optimal error indicator vector, $\vec{c}_{\widehat{k}_0}, \vec{c}_{\widehat{k}_1}, \ldots, \vec{c}_{\widehat{k}_{L-1}}$, by the formulas of (29) or (30) or (31).

(2) CRC-checking: in order to get the optimal error indicator vector correctly, we further exclude some candidates from $\vec{c}_{\widehat{k}_0}, \vec{c}_{\widehat{k}_1}, \ldots, \vec{c}_{\widehat{k}_{L-1}}$ by the CRC-checking. If there is only one valid candidate after the CRC-checking, the optimal error indicator vector will be output directly; otherwise, the remaining candidates will further be processed in step 3.

(3) Determination: if there are multiple candidates with a correct CRC-checking, we will further choose the optimal error indicator vector from the remaining candidates of step 2 with the formulas of (29) or (30) or (31).

Table 1: The space and time complexity of each step in Algorithm 2.

Step number in Algorithm 2	Space complexity	Time complexity
(1)	$O(1)$	$O(N)$
(2)	$O(N\log_2 N)$	$O(N\log_2 N)$
(3)	$O(X_0)$	$O(X_1)$
(4)–(7)	$O(T_0 N\log_2 N)$	$O(T_0 N\log_2 N)$
(8)	$O(1)$	$O(T_0 N\log_2 N)$ or $O(T_0 N)$
(9)	$O(1)$	$O(N)$

Table 2: The space and time complexity of each step in Algorithm 1.

Step number in Algorithm 1	Space complexity	Time complexity
(1)	$O(1)$	$O(M)$
(2)	$O(1)$	$O(M)$
(3)–(7)	$O(1)$	$O(MN)$
(8)–(15)	$O(1)$	$O\left(T_1(M-K)K\right) + O(T_1 M)$

So far, we have introduced the main steps of proposed decoding algorithm in detail, and, as the summarization of these results, we now provide the whole decoding procedure with the form of pseudocode, as shown in Algorithm 2.

4.5. Complexity Analysis. In this section, the complexity of the proposed decoding algorithm is considered. We first investigate the space and time complexity of each step in Algorithm 2, as shown in Table 1.

In Table 1, $O(X_0), O(X_1)$ are the space and time complexity of Algorithm 1, respectively, and T_0 is the element number of error indicator vectors output by Algorithm 1; that is, $T_0 = |\mathbb{C}|$. It is noticed that the complexity of the Algorithm 1 has a great influence on the complexity of the proposed decoding algorithm; hence, we further analyze the complexity of each step of Algorithm 1, and the results are shown in Table 2.

In Table 2, M is the number of the frozen nodes, and T_1 is the valid solution number of the error-checking equations after the checking of (21). Hence, we get the space and time complexity of Algorithm 1 as $O(X_0) = O(1)$ and $O(X_1) = 2O(M) + O(MN) + O(T_1(M-K)K) + O(T_1 M)$. Furthermore, we can get the space and time complexity of the proposed decoding algorithm as $O((T_0 + 1)N\log_2 N)$ and $O(2N) + O((2T_0 + 1)N\log_2 N) + O((T_1 + N + 2)M) + O(T_1 K(N-K))$. From these results, we can find that the complexity of the proposed decoding algorithm mainly depends on T_0 and T_1, values of which depend on the different channel condition, as illustrated in our simulation work.

Input:
The the received vector y_0^{N-1};
Output:
The decoded codeword, $\widehat{u}_0^{N-1}$;
(1) Getting the probability messages $q_0^{N-1}(0)$ and $q_0^{N-1}(1)$ with the received vector y_0^{N-1};
(2) Getting the probability messages of each frozen node $\mathbb{V}_F$;
(3) Getting the error indicator vectors set $\mathbb{C}$ with the Algorithm 1;
(4) **for** each $\vec{c}_k \in \mathbb{C}$ **do**
(5) Correcting the errors indicated by $\vec{c}_k$ with (26);
(6) Recalculating the probability messages of all the nodes of the decoder;
(7) **end for**
(8) Getting the optimal error indicator vector for the decoding;
(9) Getting the decoded codeword $\widehat{u}_0^{N-1}$ by hard decision;
(10) **return** $\widehat{u}_0^{N-1}$;

Algorithm 2: Decoding algorithm based on error checking and correcting.

5. Simulation Results

In this section, Monte Carlo simulation is provided to show the performance and complexity of the proposed decoding algorithm. In the simulation, the BPSK modulation and the additive white Gaussian noise (AWGN) channel are assumed. The code length is $N = 2^3 = 8$, code rate R is 0.5, and the index of the information bits is the same as [1].

5.1. Performance. To compare the performance of SC, SCL, BP, and the proposed decoding algorithms, three optimization targets with 1 bit CRC are used to get the optimal error indicator vector in our simulation, and the results are shown in Figure 4.

As it is shown from Algorithms 1, 2, and 3 in Figure 4, the proposed decoding algorithm almost yields the same performance with the three different optimization targets. Furthermore, we can find that, compared with the SC, SCL, and BP decoding algorithms, the proposed decoding algorithm achieves better performance. Particularly, in the low region of signal to noise ratio (SNR), that is, E_b/N_0, the proposed algorithm provides a higher SNR advantage; for example, when the bit error rate (BER) is 10^{-3}, Algorithm 1 provides SNR advantages of 1.3 dB, 0.6 dB, and 1.4 dB, and when the BER is 10^{-4}, the SNR advantages are 1.1 dB, 0.5 dB, and 1.0 dB, respectively. Hence, we can conclude that performance of short polar codes could be improved with the proposed decoding algorithm.

In addition, it is noted from Theorem 11 that the value of δ depended on the transition probability of the channel and the signal power will affect the performance of the proposed decoding algorithm. Hence, based on Algorithm 1 in Figure 4, the performance of our proposed decoding algorithm with different δ and SNR is also simulated, and the results are shown in Figure 5. It is noticed that the optimal values of δ according to $E_b/N_0 = 1$ dB, $E_b/N_0 = 3$ dB, $E_b/N_0 = 5$ dB, and $E_b/N_0 = 7$ dB are 2.5, 3.0, 5.0, and 5.5, respectively.

5.2. Complexity. To estimate the complexity of the proposed decoding algorithm, the average numbers of parameters T_0 and T_1 indicated in Section 4.5 are counted and shown in Figure 6.

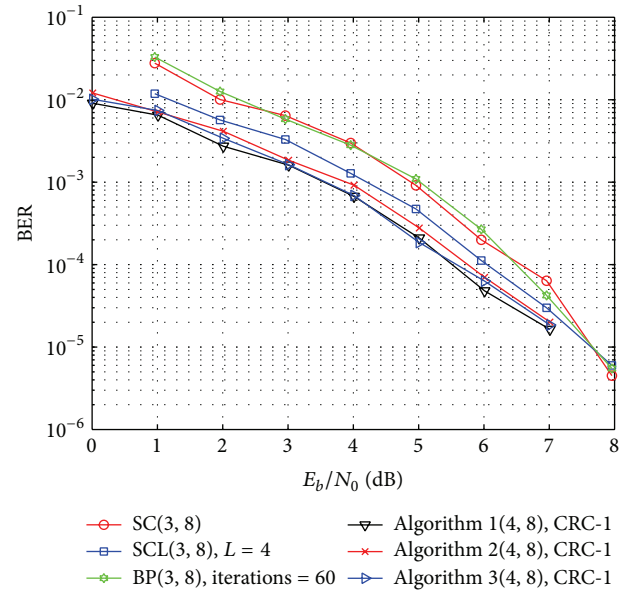

Figure 4: Performance comparison of SC, SCL($L = 4$), BP (iteration number is 60), and the proposed decoding algorithm. Algorithm 1 means that target function to get the optimal error indicator vector is (29), Algorithm 2 means that the target function is (30), and Algorithm 3 means that the target function is (31). δ in Theorem 11 takes the value of 4.

It is noticed from Figure 6 that, with the increasing of the SNR, the average numbers of parameters T_0 and T_1 are sharply decreasing. In particular, we can find that, in the high SNR region, both of the T_0 and T_1 are approaching to a number less than 1. In this case, the space complexity of the algorithm will be $O(N\log_2 N)$, and the time complexity approaches to $O(NM)$. In addition, we further compare the space and time complexity of Algorithm 1 ($\delta = 4$) and SC, SCL ($L = 4$), and BP decoding algorithm, results of which are shown in Figure 7. It is noticed that, in the high

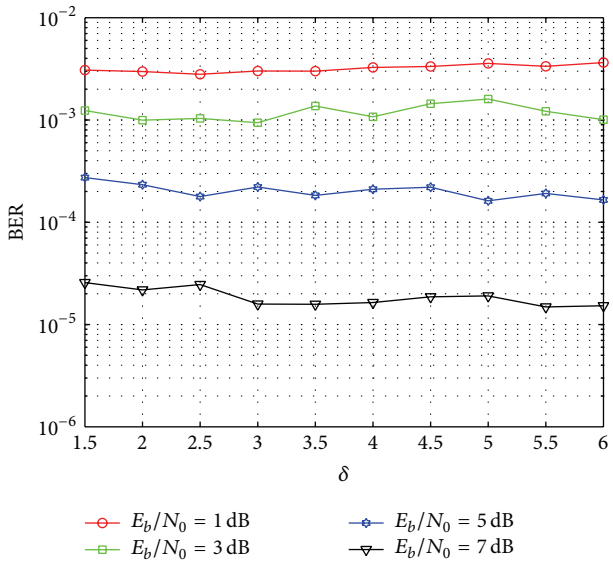

Figure 5: Performance of proposed decoding algorithm with different δ.

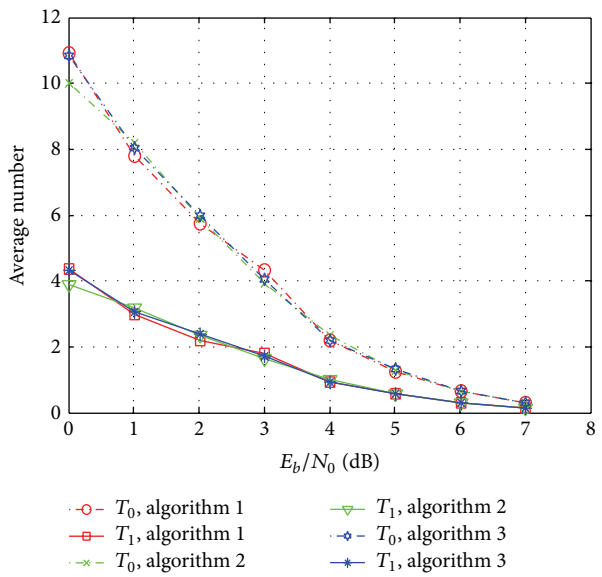

Figure 6: Average number of parameters T_0 and T_1 with $\delta = 4$.

SNR region, the space complexity of the proposed algorithm is almost the same as that of SC, SCL, and BP decoding algorithm, and the space complexity of the proposed algorithm will be close to $O(NM)$. All of the above results have suggested the effectiveness of our proposed decoding algorithm.

6. Conclusion

In this paper, we proposed a parallel decoding algorithm based on error checking and correcting to improve the performance of the short polar codes. To enhance the error-correcting capacity of the decoding algorithm, we derived the error-checking equations generated on the basis of the frozen nodes, and through delving the problem of solving these equations, we introduced the method to check the errors in the input nodes by the solutions of the equations. To further correct those checked errors, we adopted the method of modifying the probability messages of the error nodes with constant values according to the maximization principle. Due to the existence of multiple solutions of the error-checking equations, we formulated a CRC-aided optimization problem of finding the optimal solution with three different target functions, so as to improve the accuracy of error checking. Besides, in order to increase the throughput of decoding, we used a parallel method based on the decoding tree to calculate probability messages of all the nodes in the decoder. Numerical results showed that the proposed decoding algorithm achieves better performance than that of the existing decoding algorithms, where the space and time complexity were approaching to $O(N\log_2 N)$ and $O(NM)$ (M is the number of frozen nodes) in the high signal to noise ratio (SNR) region, which suggested the effectiveness of the proposed decoding algorithm.

It is worth mentioning that we only investigated the error correcting for short polar codes, while for the long-length codes, the method in this paper will yield higher complexity. Hence, in future, we will extend the idea of error correcting in this paper to the research of long code length in order to further improve the performance of polar codes.

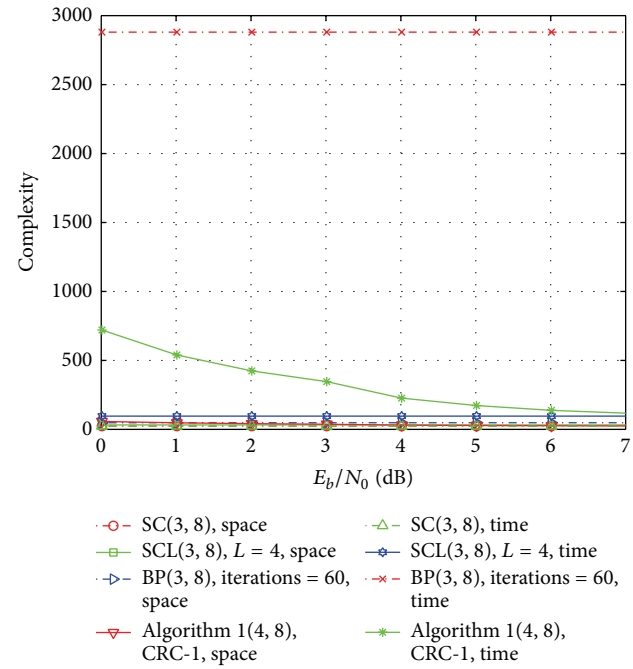

Figure 7: Space and time complexity comparison of SC, SCL($L = 4$), BP (iteration number is 60), and Algorithm 1 ($\delta = 4$).

Appendix

A. Proof of Theorem 1

We can get the inverse of $\mathbf{F}_2$ through the linear transformation of matrix; that is $\mathbf{F}_2^{-1} = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$. Furthermore, we have

$$\begin{aligned}\left(\mathbf{F}_2^{\otimes 2}\right)^{-1} &= \left[\begin{array}{c|c} \mathbf{F}_2 & \mathbf{0}_2 \\ \hline \mathbf{F}_2 & \mathbf{F}_2 \end{array}\right]^{-1} = \left[\begin{array}{c|c} \mathbf{F}_2^{-1} & \mathbf{0}_2 \\ \hline -\mathbf{F}_2^{-1}\mathbf{F}_2\mathbf{F}_2^{-1} & \mathbf{F}_2^{-1} \end{array}\right] \\ &= \left[\begin{array}{c|c} \mathbf{F}_2 & \mathbf{0}_2 \\ \hline \mathbf{F}_2 & \mathbf{F}_2 \end{array}\right] = \mathbf{F}_2^{\otimes 2}.\end{aligned} \tag{A.1}$$

Based on the mathematical induction, we will have

$$\left(\mathbf{F}_2^{\otimes n}\right)^{-1} = \mathbf{F}_2^{\otimes n}. \tag{A.2}$$

The inverse of $\mathbf{G}_N$ can be expressed as

$$\mathbf{G}_N^{-1} = \left(\mathbf{B}_N\mathbf{F}_2^{\otimes n}\right)^{-1} = \left(\mathbf{F}_2^{\otimes n}\right)^{-1}\mathbf{B}_N^{-1} = \mathbf{F}_2^{\otimes n}\mathbf{B}_N^{-1}. \tag{A.3}$$

Since $\mathbf{B}_N$ is a bit-reversal permutation matrix, by elementary transformation of matrix, there is $\mathbf{B}_N^{-1} = \mathbf{B}_N$. Hence, we have

$$\mathbf{G}_N^{-1} = \mathbf{F}_2^{\otimes n}\mathbf{B}_N. \tag{A.4}$$

It is noticed from *Proposition* 16 of [1] that $\mathbf{F}_2^{\otimes n}\mathbf{B}_N = \mathbf{B}_N\mathbf{F}_2^{\otimes n}$; therefore there is

$$\mathbf{G}_N^{-1} = \mathbf{G}_N. \tag{A.5}$$

B. Proof of Theorem 4

We assume the leaf nodes number of the frozen node $v(i, j)$ is Q; that is, $Q = |\mathbb{V}_{v(i,j)}^L|$. If $Q = 2$, based on (25), there is

$$\begin{aligned} p_{v(i,j)}(0) &= p_{v_0}(0)\, p_{v_1}(0) + p_{v_0}(1)\, p_{v_1}(1) \\ p_{v(i,j)}(1) &= p_{v_0}(0)\, p_{v_1}(1) + p_{v_0}(1)\, p_{v_1}(0), \end{aligned} \tag{B.1}$$

where $v_0, v_1 \in \mathbb{V}_{v(i,j)}^L$. Based on the above equations, we have

$$\begin{aligned} &p_{v(i,j)}(0) - p_{v(i,j)}(1) \\ &\quad = \left(p_{v_0}(0) - p_{v_0}(1)\right)\left(p_{v_1}(0) - p_{v_1}(1)\right). \end{aligned} \tag{B.2}$$

Therefore, by the mathematical inducing, when $Q > 2$, we will have

$$p_{v(i,j)}(0) - p_{v(i,j)}(1) = \prod_{k=0}^{Q-1}\left(p_{v_k}(0) - p_{v_k}(1)\right), \tag{B.3}$$

where $v_k \in \mathbb{V}_{v(i,j)}^L$.

To prove Theorem 4, we assume that the values of all the nodes in $\mathbb{V}_{v(i,j)}^L$ are set to 0 without generality. That is to say, when the node $v_k \in \mathbb{V}_{v(i,j)}^L$ is right, there is $p_{v_k}(0) > p_{v_k}(1)$. Hence, based on the above equation, when the probability messages of $v(i, j)$ do not satisfy the reliability condition, that is, $p_{v(i,j)}(0) - p_{v(i,j)}(1) \le 0$, there must exist a subset $\mathbb{V}_{v(i,j)}^{LO} \subseteq \mathbb{V}_{v(i,j)}^L$, and $|\mathbb{V}_{v(i,j)}^{LO}|$ is an odd number, such that

$$\forall v_k \in \mathbb{V}_{v(i,j)}^{LO} \longrightarrow p_{v_k}(0) \le p_{v_k}(1). \tag{B.4}$$

While if $p_{v(i,j)}(0) - p_{v(i,j)}(1) > 0$, there must exist a subset $\mathbb{V}_{v(i,j)}^{LE} \subseteq \mathbb{V}_{v(i,j)}^L$, and $|\mathbb{V}_{v(i,j)}^{LE}|$ is an even number, such that

$$\forall v_k \in \mathbb{V}_{v(i,j)}^{LE} \longrightarrow p_{v_k}(0) \le p_{v_k}(1). \tag{B.5}$$

So the proof of Theorem 4 is completed.

C. Proof of Theorem 9

It is noticed from (1) that the coefficient vector of error-checking equation generated by the frozen nodes in the leftmost column is equal to one column vector of $\mathbf{G}_N^{-1}$, denoted as $\vec{g}_i$, $0 \le i \le N-1$. For example, the coefficient vector of error-checking equation generated by $v(0, 0)$ is equal to $\vec{g}_1 = (1, 1, 1, 1, 1, 1, 1, 1)^T$. Hence, based on the proof of Theorem 1, we have

$$\begin{aligned} \operatorname{rank}\left(\mathbf{E}_{MN}\right) &\ge N - K, \\ \operatorname{rank}\left(\left[\begin{array}{c|c} \mathbf{E}_{MN} & \left(\gamma_0^{M-1}\right)^T \end{array}\right]\right) &\ge N - K. \end{aligned} \tag{C.1}$$

In view of the process of polar encoding, we can find that the frozen nodes in the intermediate column are generated by the linear transformation of the frozen nodes in the leftmost column. That is to say, error-checking equation generated by the frozen nodes in the intermediate column can be linear expressed by the error-checking equation generated by the frozen nodes in the leftmost column. Hence, we further have

$$\begin{aligned} \operatorname{rank}\left(\mathbf{E}_{MN}\right) &\le N - K, \\ \operatorname{rank}\left(\left[\begin{array}{c|c} \mathbf{E}_{MN} & \left(\gamma_0^{M-1}\right)^T \end{array}\right]\right) &\le N - K. \end{aligned} \tag{C.2}$$

Therefore, the proof of (17) is completed.

D. Proof of Theorem 11

To prove Theorem 11, we assume that the real values of all the input nodes are 0 without generality. Given the conditions of transition probability of the channel and constraint of the signal power, it can be easily proved that there exists a positive constant $\beta_0 > 1$, such that

$$\forall v(n,k) \in \mathbb{V}_I \Longrightarrow \frac{1}{\beta_0} \le \frac{p_{v(n,k)}(0)}{p_{v(n,k)}(1)} \le \beta_0, \tag{D.1}$$

where $v(n, k)$ is an input node and $\mathbb{V}_I$ is the input nodes set of the decoder. That is to say, for a frozen node $v(i, j)$ with a leaf nodes set $\mathbb{V}_{v(i,j)}^L$, we have

$$\forall v_k \in \mathbb{V}_{v(i,j)}^L \Longrightarrow \frac{1}{\beta_0} \le \frac{p_{v_k}(0)}{p_{v_k}(1)} \le \beta_0. \tag{D.2}$$

Based on (25) and the decoding tree of $v(i, j)$, we have the probability messages of $v(i, j)$ as

$$\begin{aligned} &p_{v(i,j)}(0) \\ &= \sum_{m=0}^{Q/2-1} \sum_{\substack{\forall\{k_0,\ldots,k_{2m-1}\} \\ \subseteq\{0,\ldots,Q-1\}}} \prod_{l=0}^{2m-1} p_{v_{k_l}}(1) \prod_{\substack{r=0,0\le k_r\le Q-1, \\ k_r\notin\{k_0,\ldots,k_{2m-1}\}}}^{Q-2m-1} p_{v_{k_r}}(0), \\ &p_{v(i,j)}(1) \\ &= \sum_{m=0}^{Q/2-1} \sum_{\substack{\forall\{k_0,\ldots,k_{2m}\} \\ \subseteq\{0,\ldots,Q-1\}}} \prod_{l=0}^{2m} p_{v_{k_l}}(1) \prod_{\substack{r=0,0\le k_r\le Q-1, \\ k_r\notin\{k_0,\ldots,k_{2m}\}}}^{Q-2m-1} p_{v_{k_r}}(0), \end{aligned} \tag{D.3}$$

where $v_{k_l}, v_{k_r} \in \mathbb{V}^L_{v(i,j)}$. Hence, we further have

$$\begin{aligned} &\frac{p_{v(i,j)}(0)}{p_{v(i,j)}(1)} \\ &= \frac{1 + \sum_{m=1}^{Q/2-1} \sum_{\substack{\forall\{k_0,\ldots,k_{2m-1}\} \\ \subseteq\{0,\ldots,Q-1\}}} \prod_{l=0}^{2m-1} \left(p_{v_{k_l}}(0)/p_{v_{k_l}}(1)\right)}{\sum_{m=0}^{Q/2-1} \sum_{\substack{\forall\{k_0,\ldots,k_{2m}\} \\ \subseteq\{0,\ldots,Q-1\}}} \prod_{l=0}^{2m} \left(p_{v_{k_l}}(0)/p_{v_{k_l}}(1)\right)}. \end{aligned} \tag{D.4}$$

With definition of variables $\varphi_0 = p_{v_0}(0)/p_{v_0}(1)$, $\varphi_1 = p_{v_1}(0)/p_{v_1}(1),\ldots$, $\varphi_{Q-1} = p_{v_{Q-1}}(0)/p_{v_{Q-1}}(1)$, $1/\beta_0 \le \varphi_0, \varphi_1, \ldots, \varphi_{Q-1} \le \beta_0$, the above equation will be written as

$$\begin{aligned} &\frac{p_{v(i,j)}(0)}{p_{v(i,j)}(1)} \\ &= f(\varphi_0, \varphi_1, \ldots, \varphi_{Q-1}) \\ &= \big(1 + \cdots + \varphi_{Q-2}\varphi_{Q-1} + \varphi_0\varphi_1\varphi_2\varphi_3 \\ &\qquad + \cdots + \varphi_{Q-4}\varphi_{Q-3}\varphi_{Q-2}\varphi_{Q-1} + \cdots\big) \\ &\quad \times \big(\varphi_0 + \cdots + \varphi_{Q-1} + \varphi_0\varphi_1\varphi_2 \\ &\qquad + \cdots + \varphi_{Q-3}\varphi_{Q-2}\varphi_{Q-1} + \cdots\big)^{-1}. \end{aligned} \tag{D.5}$$

To take the derivative of $f(\varphi_0, \varphi_1, \ldots, \varphi_{Q-1})$, we further define functions as

$$\begin{aligned} &h(\varphi_0, \varphi_1, \ldots, \varphi_{Q-1}) \\ &\quad = 1 + \varphi_0\varphi_1 + \cdots + \varphi_{Q-2}\varphi_{Q-1} \\ &\qquad + \varphi_0\varphi_1\varphi_2\varphi_3 + \cdots + \varphi_{Q-4}\varphi_{Q-3}\varphi_{Q-2}\varphi_{Q-1} + \cdots, \\ &g(\varphi_0, \varphi_1, \ldots, \varphi_{Q-1}) \\ &\quad = \varphi_0 + \cdots + \varphi_{Q-1} + \varphi_0\varphi_1\varphi_2 + \cdots \\ &\qquad + \varphi_{Q-3}\varphi_{Q-2}\varphi_{Q-1} + \cdots. \end{aligned} \tag{D.6}$$

Then, the derivative of $f(\varphi_0, \varphi_1, \ldots, \varphi_{Q-1})$ with respect to φ_k will be

$$\begin{aligned} \frac{\partial f}{\partial \varphi_k} &= \frac{\left(\partial h/\partial\varphi_k\right) g - \left(\partial g/\partial\varphi_k\right) h}{g^2} \\ &= \frac{g g_{\varphi_k=0} - h h_{\varphi_k=0}}{g^2} = \frac{g^2_{\varphi_k=0} - h^2_{\varphi_k=0}}{g^2}, \end{aligned} \tag{D.7}$$

where $g_{\varphi_k=0} = g(\varphi_0, \ldots, \varphi_{k-1}, 0, \varphi_{k+1}, \ldots, \varphi_{Q-1})$ and $h_{\varphi_k=0} = h(\varphi_0, \ldots, \varphi_{k-1}, 0, \varphi_{k+1}, \ldots, \varphi_{Q-1})$. Based on solution of the equations $(\partial f/\partial\varphi_0) = 0$, $(\partial f/\partial\varphi_1) = 0,\ldots$, and $(\partial f/\partial\varphi_{Q-1}) = 0$, we get the extreme value point of $f(\varphi_0, \varphi_1, \ldots, \varphi_{Q-1})$ as $\varphi_0 = \varphi_1 = \cdots = \varphi_{Q-1} = 1$. Based on the analysis of the monotonicity of $f(\varphi_0, \varphi_1, \ldots, \varphi_{Q-1})$, we can get the maximum value as $\delta = f(\underbrace{\beta_0, \beta_0, \ldots, \beta_0}_{Q})$. What is more, we can also get that when $f(\varphi_0, \varphi_1, \ldots, \varphi_{Q-1}) \ge \delta$, there is $\varphi_0 > 1$, $\varphi_1 > 1,\ldots$, and $\varphi_{Q-1} > 1$. That is to say, when $(p_{v(i,j)}(0)/p_{v(i,j)}(1)) \ge \delta$, we will have $p_{v_0}(0) > p_{v_0}(1)$, $p_{v_1}(0) > p_{v_1}(1), \ldots$, and $p_{v_{Q-1}}(0) > p_{v_{Q-1}}(1)$; that is, there is no error in $\mathbb{V}^L_{v(i,j)}$. So the proof of Theorem 11 is completed.

Conflict of Interests

The authors declare that they do not have any commercial or associative interests that represent a conflict of interests in connection with the work submitted.

Acknowledgment

The authors would like to thank all the reviewers for their comments and suggestions.

References

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

[2] E. Arikan and E. Telatar, "On the rate of channel polarization," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT '09)*, pp. 1493–1495, June-July 2009.

[3] S. B. Korada, E. Şaş oğlu, and R. Urbanke, "Polar codes: characterization of exponent, bounds, and constructions," *IEEE Transactions on Information Theory*, vol. 56, no. 12, pp. 6253–6264, 2010.

[4] I. Tal and A. Vardy, "List decoding of polar codes," in *Proceedings of the IEEE International Symposium on Information Theory Proceedings (ISIT '11)*, pp. 1–5, St. Petersburg, Russia, August 2011.

[5] I. Tal and A. Vardy, "List decoding of polar codes," http://arxiv.org/abs/1206.0050.

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

[7] K. Niu and K. Chen, "CRC-aided decoding of polar codes," *IEEE Communications Letters*, vol. 16, no. 10, pp. 1668–1671, 2012.

[8] 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, 2011.

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

[10] 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, 2014.

[11] P. Giard, G. Sarkis, C. Thibeault, and W. J. Gross, "A fast software polar decoder," http://arxiv.org/abs/1306.6311.

[12] E. Arikan, H. Kim, G. Markarian, U. Özür, and E. Poyraz, "Performance of short polar codes under ML decoding," in *Proceedings of the ICT-Mobile Summit Conference*, June 2009.

[13] S. Kahraman and M. E. Çelebi, "Code based efficient maximum-likelihood decoding of short polar codes," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT '12)*, pp. 1967–1971, Cambridge, Mass, USA, July 2012.

[14] N. Goela, S. B. Korada, and M. Gastpar, "On LP decoding of polar codes," in *Proceedings of the IEEE Information Theory Workshop (ITW '10)*, pp. 1–5, Dublin, Ireland, September 2010.

[15] E. Arikan, "A performance comparison of polar codes and reed-muller codes," *IEEE Communications Letters*, vol. 12, no. 6, pp. 447–449, 2008.

[16] N. Hussami, S. B. Korada, and R. Urbanke, "Performance of polar codes for channel and source coding," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT '09)*, pp. 1488–1492, July 2009.

[17] E. Arikan, "Polar codes: a pipelined implementation," in *Proceedings of the 4th International Symposium on Broadband Communication (ISBC '10)*, pp. 11–14, July 2010.

[18] A. Eslami and H. Pishro-Nik, "On bit error rate performance of polar codes in finite regime," in *Proceedings of the 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton '10)*, pp. 188–194, October 2010.

[19] A. Eslami and H. Pishro-Nik, "On finite-length performance of polar codes: stopping sets, error floor, and concatenated design," *IEEE Transactions on Communications*, vol. 61, no. 3, pp. 919–929, 2013.

[20] E. Arikan, "Systematic polar coding," *IEEE Communications Letters*, vol. 15, no. 8, pp. 860–862, 2011.

[21] J. L. Massey, "Catastrophic error-propagation in convolutional codes," in *Proceedings of the 11th Midwest Symposium on Circuit Theory*, pp. 583–587, January 1968.

[22] R. G. Gallager, "Low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 8, pp. 21–28, 1962.

[23] D. Divsalar and C. Jones, "CTH08-4: protograph LDPC codes with node degrees at least 3," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '06)*, pp. 1–5, San Francisco, Calif, USA, December 2006.