IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 9, 2020, accepted November 23, 2020, date of publication November 30, 2020,
date of current version December 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041290

An Improved SC Flip Decoding Algorithm of Polar Codes Based on Genetic Algorithm

XIUMIN WANG[1], QIANGQIANG MA[1], JUN LI[2], HONGCHAO ZHANG[1], AND WENCHAO XU[3]
[1]College of Information Engineering, China Jiliang University, Hangzhou 310018, China
[2]Binjiang College, Nanjing University of Information Science & Technology, Wuxi 214105, China
[3]Department of Computing, The Hong Kong Polytechnic University, Hong Kong

Corresponding author: Jun Li (07a0303105@cjlu.edu.cn)

This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China under Grant Y20F010069, in part by the National Natural Science Foundation of China (NSFC) under Grant 51874264, and in part by the Key Laboratory of Electromagnetic Wave Information Technology and Metrology of Zhejiang Province, College of Information Engineering, China Jiliang University, Hangzhou, China.

ABSTRACT Polar codes have been applied for physical downlink control channel in the 5$^{\text{th}}$ generation wireless communication system. Although successive cancellation flip (SCF) decoding algorithm can improve decoding performance of polar codes, it has also led to the increasing in decoding latency and calculation complexity. Candidate flipping positions set (CFPS) of traditional SCF decoding is consisted of indexes of all information bits. However, some subchannels are reliable enough so that it is almost impossible to cause decoding errors for these subchannels. In order to reduce decoding latency and calculation complexity of SCF decoding algorithm, a new method of constructing the CFPS based on genetic algorithm (GA) is proposed in this paper. What's more, the paper fills a gap of applying GA for decoding of polar codes. In our proposed method, indexes of all information bits are used as individuals of GA. Then through some genetic operations, a vector that can indicate the reliability of all information bits is obtained. Based on the obtained vector, a new CFPS is constructed. Simulation results show that SCF decoding algorithm based on CFPS constructed by GA can achieve competitive decoding performance, while keeping lower calculation complexity and decoding latency. Compared with SCF decoding algorithm based on critical set, the normalized decoding latency of proposed SCF decoding algorithm can be reduced by 39% at 1.5dB when code length and code rate are equal to 1024 and 0.5, respectively.

INDEX TERMS Candidate flipping positions set, genetic algorithm, polar codes, successive cancellation flip.

I. INTRODUCTION

The past decade has seen the rapid development of wireless communication in many aspects. However, compared with wired communication, the channel characteristic of wireless communication is more complex. Because of the channel fading, channel noise and multiple path of wireless channel, decoding performance of wireless communication system is deteriorated. Then channel encoding technology is used in wireless communication to correct decoding error. With the development of wireless communication, channel encoding technology is also improved continuously. To be specific, turbo codes are adopted by the 3$^{\text{th}}$ wireless communication system. Then turbo codes and low density parity check (LDPC) codes are used for channel encoding in the 4$^{\text{th}}$ wireless communication system. Whereas, in the 5$^{\text{th}}$ communication standard, LDPC codes are adopted for data channel while polar codes are used in control channel [1], [2].

Polar codes, proposed by Arıkan in 2009, are the first error correcting codes, provably reaching the Shannon theory limit under successive cancellation (SC) decoding algorithm [3]. With the advantage of low encoding and decoding complexity, low latency and good decoding performance, polar codes have been applied in uplink/downlink control channel of the 5G enhanced mobile broadband (eMBB) scenario. In addition, polar codes have also been used to improve the performance of physical downlink control channel (PDCCH) detection, which is essential for multiple-input multiple-output (MIMO) wireless communication system [4]. Besides, with the characteristic that polarized subchannels tend to

The associate editor coordinating the review of this manuscript and approving it for publication was Donatella Darsena.

become either noiseless or completely noisy as code length approaches infinite, polar codes can perform well in long code length while it has performance loss in short code length. In order to improve the decoding performance of polar codes under finite code length, successive cancellation list (SCL) decoding algorithm was proposed [5]. Unlike original SC decoding algorithm, SCL decoding algorithm preserves multiple SC decoding paths until the end of decoding. However, only one path with the smallest path metric (PM) value can be used as output of decoding. What's more, there is another method in choosing the correct decoding path. By concatenating with cyclic redundancy check (CRC) codes, CRC-aided SCL decoding algorithm was proposed [6]. In this situation, only the path that passes CRC check can be selected as the decoding output. CRC-aided SCL decoding algorithm greatly improves decoding performance of polar codes so that polar codes can provide competitive performance compared with other codes, such as turbo codes and LDPC codes. However, CRC-aided SCL decoding algorithm also causes higher decoding latency and computational complexity due to its decoding characteristic. In order to optimize SCL decoding algorithm of polar codes, some improved SCL decoding algorithms were proposed [7]–[12]. Generally, SC and SCL algorithms both are serial decoding algorithms so that error propagation will occur in their decoding process.

As mentioned above, SC decoding is a kind of serial decoding algorithm. This means that it needs to use previous decoded codewords to decode subsequent codewords. Then if the previous codewords are incorrectly decoded, it is hard to decode successfully because of error propagation. In addition, there are two kinds of error bits during the decoding process [13]. These two types of error bits are as follows:

Type1: type1 refers to the first error bit in every segment of codewords. All bits before these error bits are successfully decoded.

Type2: type2 refers to these error bits caused by type1 and the appearance of type2 error bits is due to the error propagation.

In order to correct type1 error bits, successive cancellation flip (SCF) decoding algorithm was proposed [14]. In contrast to SC decoding, if the decoded codewords can't pass CRC check, the unfrozen bit with the least absolute value of log-likelihood ratio (LLR) will be flipped and subsequent bits of the flipping bit continue to be decoded by using standard SC decoding. The above process will repeat until decoded output passes CRC check or a predetermined maximum number of decoding attempts is reached. In a word, SCF decoding can obtain better decoding performance compared with SC decoding, which has lower complexity compared with SCL decoding. However, the search scope for flipping positions is the entire unfrozen set in SCF decoding [14], causing higher decoding latency and calculation complexity. What's more, bigger search scope for flipping positions can lead to the fact that SCF decoder can't find all type1 errors under limited attempts. In order to reduce decoding complexity and improve decoding performance, critical set (CS) of candidate flipping positions and its corresponding SCF decoding algorithm were proposed in 2017 [15]. During the proposed SCF decoding algorithm, CS is consisted of the first unfrozen bit index of all rate-1 subblocks. Then the search scope for flipping positions is equal to the size of CS, which is smaller than the number of all information bits. The constructed CS contain bit indexes where the first error happens with high probability, and the proposed SCF decoding algorithm based on CS can provide better decoding performance and lower complexity compared with CRC-aided SCL decoding algorithm at medium to high signal to noise ratio (SNR). Similarly, the concept of medium-level bit-channels (MBC) set was also proposed in 2019 [16]. Besides, SCF decoding algorithm based on proposed MBC set has the same decoding performance and calculation complexity with that based on CS. However, the size of LLR value only represents the reliability of its corresponding information bit, but it can't indicate the probability of being type1 errors for every information bit. In order to find type1 errors more efficiently, a dynamic SCF decoding algorithm and its improved version were proposed in 2018 and 2019, respectively [17], [18]. In dynamic SCF decoding algorithm, a new metric to choose flipping positions is defined and the list of candidate flipping bits can be dynamically updated. Dynamic SCF decoding algorithm can reach the lower bound of decoding performance for single bit SCF decoding algorithm, persevering computation complexity closed to SC decoding algorithm. Meanwhile, its computation complexity is further reduced in the improved dynamic SCF decoding algorithm by using an approximate scheme. Besides, there are also some other methods to improve SCF decoding algorithms of polar codes, such as setting thresholds [19], observing channel-induced errors distribution [20] and partitioning [21]. In summary, it is important to find type1 errors with lower latency and higher accuracy for SCF decoding algorithm. Designing a SCF algorithm with better decoding performance, lower complexity and lower latency is of great significance to the development of channel encoding technology.

In recent years, artificial intelligence (AI) technology is fast becoming a key instrument in communication so that more scholars of communication field begin to study how to use AI technology to tackle with communication problems. For instance, neural network has been applied for decoding of polar codes [22]–[24]. Besides, the influence of various network configurations on decoding performance of neural network decoder is investigated [25]. In order to obtain good decoding performance, the train set of neural network decoder of polar codes must include all kinds of codewords. However, the problem of exponential explosion will appear as code length increases. In order to solve the problem of exponential explosion, a kind of partitioned neural network SC decoder was proposed [26]. Whereas the neural network decoder is essentially a classifier, which is limited to code length of polar codes. In SCF decoding algorithm, the problem of finding the candidate flipping positions set (CFPS) can be seen as a problem of finding the optimal solution.

Meanwhile, genetic algorithm (GA) of AI technology is used to search the optimal solution by imitating the natural evolution process. Hence, an improved SCF decoding algorithm of polar codes based on GA is proposed in this paper. It is true that GA has emerging applications in polar codes so far. However, previous studies have reported that GA is only used in the encoding construction of polar codes. The application of GA in decoding of polar codes has been a largely under explored domain. For example, a new frame for constructing polar codes based on GA was proposed in 2019 [27]. Similarly, some researchers also proposed to use GA to construct polar codes without any expert knowledge about channel coding and modulation [28]. Inspired by GA of AI technology, a new method to construct CFPS based on GA is proposed in this paper. Then a SCF decoding algorithm based on the new constructed CFPS is further developed. Results show that compared with other similar state-of-the-art SCF decoding algorithms, the proposed SCF decoding based on GA can achieve competitive decoding performance, while keeping lower decoding latency and computation complexity.

The rest of the paper is organized as follows. In Section II, we briefly review the knowledge of polar codes and some principles of SCF decoding algorithms. The specific construction process of CFPS based on GA is shown in Section III. Then the constructed result of CFPS and its corresponding SCF-GA decoding algorithm are described in detail in Section IV. Simulation results and analyses are presented in Section V. Section VI depicts some conclusions.

II. PRELIMINARIES

A. POLAR CODES

Polar codes are characterized by a three-tuple P(N, K, ξ), where $N = 2^n$ is code length, K is the number of information bits and ξ is set of indices of K information bits. The remaining $N - K$ bits are named as frozen bits, which are known by encoder and decoder. Then code rate R is defined as $R = K/N$. We denote original data vector by $U = u_1^N$, of length N, containing K information bits and $N - K$ frozen bits that are set to zero. Let $X = x_1^N$ represent the encoded codewords and X is obtained by

$$X = UB_N F^{\otimes n}$$

where B_N is a permutation matrix and '$\otimes$' denotes Kronecker product. The kernel $F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. The received vector is $Y = y_1^N = (y_1, y_2, y_3, \cdots, y_N)$.

B. SC DECODING AND SCF DECODING

In SC decoding, each estimated bit $\hat{u}_i$ of bit u_i depends on both Y and the previous estimated $\hat{u}_1^{i-1}$. Each $\hat{u}_i$ can be calculated by (1) [3]

$$\hat{u}_i = \begin{cases} h_i(y_1^N, \hat{u}_1^{i-1}) & i \in \xi \\ u_i & i \notin \xi \end{cases} \tag{1}$$

where h is hard decision function and it is described as

$$h_i(y_1^N, \hat{u}_1^{i-1}) = \begin{cases} 0 & L_i(y_1^N, \hat{u}_1^{i-1}) \geq 0 \\ 1 & L_i(y_1^N, \hat{u}_1^{i-1}) < 0 \end{cases} \tag{2}$$

where $L_i(y_1^N, \hat{u}_1^{i-1})$ denotes LLR value of information bit u_i and it is defined as

$$L_i(\mathrm{y}_1^N, \hat{\mathrm{u}}_1^{i-1}) = \log\left(\frac{\Pr(u_i = 0|y_1^N, \hat{u}_1^{i-1})}{\Pr(u_i = 1|y_1^N, \hat{u}_1^{i-1})}\right) \tag{3}$$

where LLR values of all information bits can be calculated iteratively [1].

In standard SCF decoding, LLR values of all information bits are sorted when original SC decoding fails to pass CRC check, and the CFPS is consisted of indices of all information bits at the moment. Then the bit whose index is in CFPS with the smallest $\left|L_i(y_1^N, \hat{u}_1^{i-1})\right|$ value is flipped and its subsequent bits are decoded continually by standard SC decoding. The process will continue until the maximum value of flipping $T_{\max}$ is reached or the decoding output passes CRC check. The kind of SCF decoding can correct type1 error bits by a certain number of flipping attempts. Then decoding performance of SCF algorithm is better than that of SC algorithm and its calculation complexity is the same with SC algorithm at high SNR.

C. GA AND EXISTING CFPS

GA is a kind of search heuristic algorithm used to solve optimization problem in the field of computer science. It is also a kind of evolutionary algorithm [27]. The frame of GA is shown in Fig. 1. In Fig. 1, the first step of GA is initialization. During the process, every individual of population corresponds to a candidate solution of the optimization problem. Then every individual can be assigned to a fitness evaluated by fitness function. In general, the higher fitness the individual has, the easier it survives in next generation. When the maximum number of iterations is not reached, population can be continuously optimized by selection, crossover, and mutation operations. During the selection operation, these individuals that have higher fitness are selected as parents and can be used to generate off-springs. In crossover operation, new off-springs are generated by exchanging information of their parents. Whereas mutation operation can produce new individuals that don't exist in original population, preserving diversity of population. Through the iterative process in Fig. 1, GA can quickly converge to the optimal solution. Therefore, GA is chosen to construct a new CFPS in this paper.

The CFPS of standard SCF decoding is consisted of indices of all information bits. However, in SCF decoding based on CS [15] and MBC set [16], their CFPSs are CS and MBC set, respectively. These two kinds of SCF decoding algorithms have smaller search scope for candidate flipping positions compared with standard SCF decoding algorithm. The constructed process of CS is described in [15], [29]. Once the encoding construction of polar codes is determined,

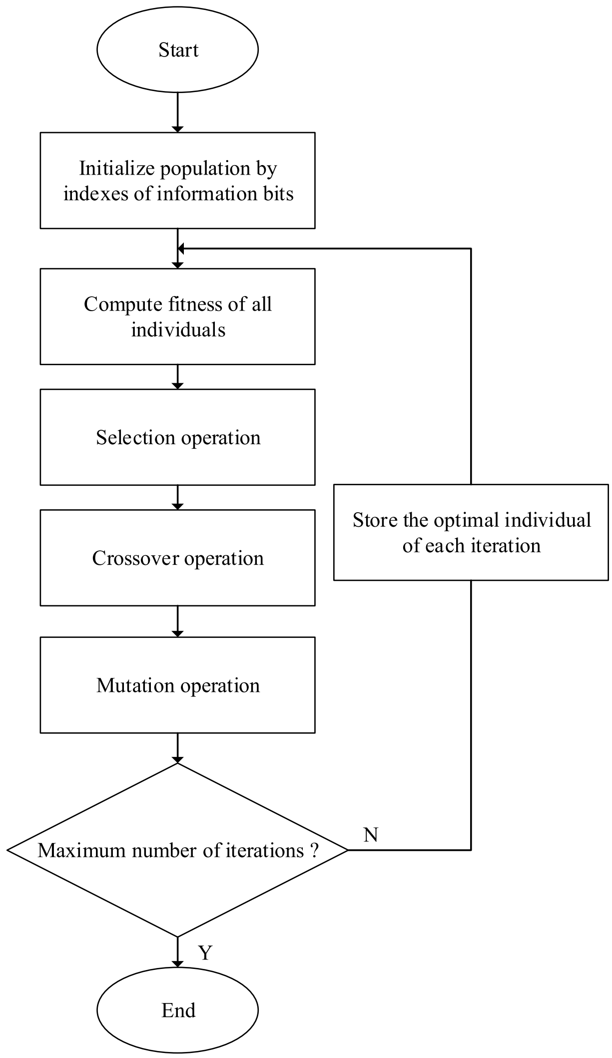


FIGURE 1. Flowchart of GA.

CS is also determined. What's more, the first step of constructing CS is to establish a full binary tree representation of polar codes and find all rate-1 nodes. Then CS can be constructed by the first information bit of each rate-1 node. MBC set is obtained by a search algorithm described explicitly in [16]. By Monte-Carlo simulation over additive white Gaussian noise (AWGN) channel, MBC set can be obtained.

III. CFPS CONSTRUCTION PROCESS BASED ON GA

In this section, constructed CFPS in this paper fills a gap of applying GA for decoding of polar codes. As mentioned above, the quality of CFPS has a huge impact on decoding performance and complexity of SCF decoding algorithm. For example, good CFPS can make SCF decoder find type1 bit errors as soon as possible under limited attempts. In the following section, we discuss the specific constructed scheme of CFPS based on GA.

A. POPULATION AND INITIALIZATION

In this work, each index of information bits is an individual of population. These indices of all information bits form the initial population of GA. Therefore, the size of population is equal to K. Let $P = \{P_i\}, \quad i = 1, 2, \cdots, K$, where P denotes the whole population and P_i represents every individual of population. The remaining $N - K$ frozen bits are known by encoder and decoder, so it doesn't make any sense to flip these bits. Therefore, only these indices of information bits can be used to initialize population. What's more, the encoding method used in this work is binary encoding. Every individual P_i can be represented by a binary vector. For example, if $P_i = 128$, its corresponding binary vector $V_i = \{1, 1, 1, 1, 1, 1, 1\}$.

B. FITNESS

The fitness of every individual in this work is mean value of LLR distribution calculated by Gaussian approximation. In Gaussian approximation over AWGN channel, LLR follows the Gaussian distribution with mean value $m_N^{(i)}$ and variance $2m_N^{(i)}$. Then mean value $m_N^{(i)}$ can be computed iteratively by (4) [30]

$$\begin{aligned} m_{2N}^{(2i-1)} &= \phi^{-1}\left\{1-[1-\phi(\mathrm{m}_N^{(i)})]^2\right\} \\ m_{2N}^{(2i)} &= 2m_N^{(i)} \\ m_1^{(1)} &= \frac{2y}{\sigma^2} \end{aligned} \tag{4}$$

where σ^2 is noise variance of AWGN channel and y is received information from channel. The function $\phi(x)$ is defined as [30]

$$\phi(x) = \begin{cases} \exp(-0.4527x^{0.86} + 0.0218) & 0 < x < 10 \\ \sqrt{\frac{\pi}{x}}\exp(-\frac{x}{4})(1 - \frac{10}{7x}) & x \geq 10 \end{cases} \tag{5}$$

The error probability P_e of every subchannel can be calculated by (6) [30]

$$P_e(i) = \int_{-\infty}^{0} \frac{1}{2\sqrt{\pi m_N^{(i)}}}.\exp(\frac{-(x-m_N^{(i)})^2}{4m_N^{(i)}})dx \tag{6}$$

It is obvious in (6) that there is a negative correlation relation between mean value $m_N^{(i)}$ and error probability P_e. Besides, channel reliability can be reflected by P_e. Generally, the channel with lower P_e is less prone to decoding errors. Therefore, a vector M of length N can be determined by mean value of LLR distribution. The vector M which is consisted of mean values can also reflect reliability of sunchannel. Normally, the channel with higher mean value $m_N^{(i)}$ is more reliable.

The pseudo algorithm of the fitness function is shown in Algorithm 1. In Algorithm 1, a is a logical vector indicating whether the decoding codewords $\hat{u}_1^N$ can pass CRC check. $SC(y_1^N, \xi)$ represents standard SC decoding process. $SCF(y_1^N, \xi, P(i))$ represents the SCF decoding process that the information bit with index $P(i)$ is flipped, and subsequent codewords of flipping bit are decoded by standard SC decoding. After flipping the bit with index $P(i)$, if the decoded codewords pass CRC check, it means that there is a type1 error in the bit with index $P(i)$. Then the fitness of the individual $P(i)$ is set to zero, otherwise its fitness is set to its corresponding mean value. The major function of fitness function is to obtain the fitness vector Fit of the whole population P. In line 1 of Algorithm 1, the received vector y_1^N and logical vector a are initialized. Then the received vector y_1^N is updated by message word that suffers modulation and channel noise in line 3 of Algorithm 1. Standard SC decoding is carried out in line 4 of Algorithm 1. CRC check for decoded codewords is performed in line 5 and the check result is assigned to logical vector a. If the value of a is equal to 1, it means that the decoded codewords can't pass CRC check. Meanwhile, SCF decoding begins to work if and only if the decoded codewords can't pass CRC check. From line 7 to line 14 of Algorithm 1, every individual is assigned to a fitness. When original SC decoding fails, decoded codewords can't pass CRC check. Nevertheless, when the flipping operation $SCF(y_1^N, \xi, P(i))$ is carried out and the decoded codewords under $SCF(y_1^N, \xi, P(i))$ pass CRC check, it means that a type1 error has happened in the individual with index $P(i)$ and we have found the position where the type1 error appears and flipped it. Hence, fitness of the individual with index $P(i)$ is set to zero. In this situation, CRC check is assumed to be correct all the time.

Algorithm 1 Fitness Function

Input: y_1^N, ξ, mean values vector M, population P
Output: fitness value vector Fit
1: **Initialization**: $y_1^N \leftarrow \mathbf{0}$; $a \leftarrow 0$
2: **while** $a == 0$ **do**
3: the received vector y_1^N update
4: $\hat{u}_1^N \leftarrow SC(y_1^N, \xi)$
5: $a = [\text{CRC}(\hat{u}_1^N) == \text{failed}]$
6: **end**
7: **for** $i = 1$ **to** K **do**
8: $\hat{u}_1^N \leftarrow SCF(y_1^N, \xi, P(i))$
9: **if** $\text{CRC}(\hat{u}_1^N) == \text{success}$
10: $Fit(i) = 0$
11: **else**
12: $Fit(i) = M(P(i))$
13: **end**
14: **end**
15: **Return** Fit

C. SELECTION

The major function of selection operation is to select superior individuals from population to generate off-springs and discard inferior individuals of population. There are some selection strategies for GA, such as truncation selection, exponential ranking selection and tournament selection. Among these selection strategies, truncation selection is the most common selection strategy, where the probability of individual being selected is proportional to its fitness. However, an individual whose fitness is zero may never be selected in truncation selection. In contrast to truncation selection, every individual can be assigned a selected probability in exponential ranking selection. Then, tournament selection has the advantage of low complexity and is not easy to fall into local optimum compared with other selection strategies. Therefore, the selection strategy of individuals used in this paper is tournament selection. In tournament selection strategy, a certain number of individuals from the population are selected to compete in each generation, and then the optimal individual is retained in new population. The process will repeat until the size of new population is equal to that of original population.

The pseudo algorithm of selection function is shown in Algorithm 2. In Algorithm 2, P_s is the new population obtained by selection operation. Besides, k is the number of individuals selected to compete during each selection. When value of k is too small, final decoding performance will become worse and complexity will also increase. That is because that population cannot converge to the optimal solution when the number of competitive individuals is too small during every iteration. However, when value of k is too big, the population will quickly converge to a wrong solution so that construction process of CFPS based on GA can't continued to proceed. Therefore, a proper k is important to final decoding performance and complexity. Fit is the fitness of all individuals. $P_{compete}$ is consisted of k individuals selected randomly among the whole population, and F_s represents their fitness vector. In this work, the individual with smaller fitness is retained in next generation. That is because these individuals with lower fitness are more unreliable and type1 errors are easier to happen in these individuals. $P_s(i)$ denotes the individual with the least fitness among k individuals selected in the i-th selection. The selection process of competitive individuals is given in line 2. Then the specific competitive process is presented from line 3 to line 6. Through the selection function, the new selected population P_s can be obtained.

Algorithm 2 Selection Function

Input: fitness vector Fit, population P, k
Output: selected population P_s
1: **for** $i = 1$ **to** K **do**
2: $P_{compete} \leftarrow$ selecting k individuals form P randomly
3: **for** $j = 1$ **to** k **do**
4: $F_s^{(i)}(1, j) = Fit(P_{compete}^{(j)})$
5: **end**
6: $P_s(i) \leftarrow$ the individual with least fitness in $F_s^{(i)}$
7: **end**
8: **Return** P_s

D. CROSSOVER

The crossover operation of GA can help off-springs to inherit good genes from their parents and global search can be achieved by crossover operation. Common crossover methods include one-point crossover, two-point crossover and multi-point crossover. In our work, one-point crossover is used in crossover function.

Algorithm 3 is the pseudo code of crossover function. In Algorithm 3, p_c denotes crossover rate, which is used to determine whether crossover operation needs to be performed in the selected parent individuals. The major function of function rand is to generate a random number whose range is from 0 to 1. Besides, the random number is assigned to flag1. When flag1 is smaller than crossover rate p_c, crossover operation can be carried out. In crossover operation, selected parent individuals can be transformed into binary vector by binary encoding. Then one-point crossover operation is applied for these two binary vectors, such as $V_{parent1}$ and $V_{parent2}$ in Algorithm 3. The position of crossover node denoted by flag2 is also determined randomly. What's more, the major function of function concat is to merge two different binary vectors into one binary vector. When the binary vector P_{child} is obtained, it needs to be converted a decimal real number by function decimal. The validity of new generated individual P_{child} needs to be tested, because every individual for polar codes must be index of information bits. Only these child individuals that pass test can be adopted as effective individuals.

Algorithm 3 Crossover Function

Input: ξ, selected population P_s, crossover rate p_c
Output: crossed population P_c

```
Initialization: i ← 1
while i < K + 1 do
  flag1 ← rand()
  if flag1 ≤ p_c then
    P_parent1, P_parent2 ← selecting parents randomly
      form P_s
    V_parent1, V_parent2 ← Encoding of P_parent1 and
    P_parent2
    L ← length of V_parent1 or V_parent2
    flag2 ← an integer range in [1,L]
    P_child = concat([V_parent1]_1^flag2, [V_parent2]_flag2^L)
   if (decimal(P_child) + 1) ∈ ξ then
     P_c(i) ← decimal(P_child) + 1
     i ← i + 1
   end if
  else
   P_c(i) ← P_s(i)
   i ← i + 1
  end if
end
Return P_c
```

E. MUTATION

Similarly, the pseudo code of the mutation function is shown in Algorithm 4. The mutation operation of GA can make population maintain variety because new individuals that are not existing in original population are generated by mutation operation. The way of simple mutation is chosen for mutation operation in our work. In Algorithm 4, p_m is mutation rate. When the random number flag1 produced by the function

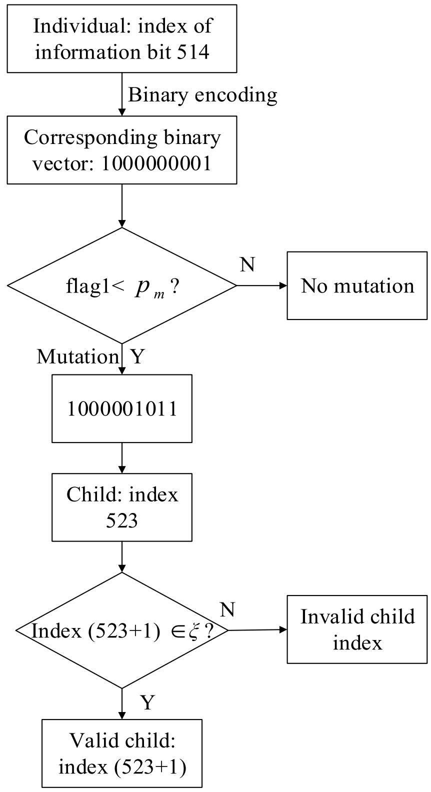


FIGURE 2. **Mutation operation for P(1024,512) polar codes.**

rand is smaller than mutation rate p_m, mutation operation of its corresponding individual $P_c(i)$ is carried out. Besides, the mutation probability for each bit of binary vector $V_m(i)$ is equal. In line 3 of Algorithm 4, a random number flag1 whose range is from 0 to 1 is generated. Then flag1 is used to determine whether to carry mutation operation on the individual $P_c(i)$. If flag1 is smaller than mutation rate p_m, the individual $P_c(i)$ will be transformed into binary vector $V_m(i)$ by binary encoding in line 5 of Algorithm 4. In contrast to flag1, the random number flag2 is used to determine whether to carry mutation operation on each bit of binary vector $V_m(i)$(line 6-line 9). Because of the characteristic that binary vector is consisted of 0 and 1, how to mutate is presented in line 10 of Algorithm 4. With the limit that every individual must be index of information bits, the validity of every new generated individual needs to be tested (line 13-line 16). If flag1 is greater than mutation rate p_m, mutation operation will not be performed (line 17-line 20). Then through the mutation function, new population P_m that has higher population diversity is obtained.

The mutation operation for P(1024,512) polar codes is shown in Fig. 2. Firstly, the selected individual index 514 is transformed into binary vector 1000000001 by binary encoding. The smallest binary number is 0 while the smallest index for polar codes is 1. In order to ensure that every index can correspond to a binary vector, the value of every index is larger than the value of its corresponding binary vector. Secondly, when the value of random number flag1 generated in Algorithm 4 is smaller than mutation rate p_m, mutation operation is carried out on selected individual, otherwise mutation operation is not executed. Finally, with the limit of principles of polar codes, these new child indexes generated by mutation must be indexes of information bits. Then these child indexes that pass test are retained in next population.

Algorithm 4 Mutation Function

Input: ξ, P_c, mutation rate p_m
Output: mutated population P_m
Initialization: $i \leftarrow 1$
while $i < K + 1$ **do**
  flag1 $\leftarrow$ *rand*
  if flag1 $\leq p_m$ **then**
    $V_m^{(i)} \leftarrow$ Binary encoding of $P_c(i)$
    $L \leftarrow$ length of $V_m^{(i)}$
    for $j = 1$ **to** L **do**
      flag2 $\leftarrow$ *rand*()
      if flag2 $\leq p_m$ **then**
        $V_m^{(i)}(1, j) \leftarrow 1 - V_m^{(i)}(1, j)$
      end if
    end for
    if $(decimal(V_m^{(i)}) + 1) \in \xi$ **then**
      $P_m(i) \leftarrow decimal(V_m^{(i)}) + 1$
      $i \leftarrow i + 1$
    end if
  else
    $P_m(i) \leftarrow P_c(i)$
    $i \leftarrow i + 1$
  end if
end
Return P_m

IV. CONSTRUCTED CFPS AND CORRESPONDING SCF-GA DECODING ALGORITHM

A. CONSTRUCTED CFPS

In this section, the constructed CFPS based on GA is presented. $N = 1024$ and $R = 0.5$ of polar codes are used. Binary phase-shift keying (BPSK) modulation and AWGN channel are used for training of GA. Gaussian approximation is used for performing channel estimation and obtaining vector M. What's more, polar codes are concatenated with a CRC-16 with generator polynomial $g(x) = x^{16} + x^{15} + x^2 + 1$. In this regard, the effective code rate of polar codes is $R = (K - 16)/N$. Some hyperparameters of GA are listed in Table 1. Then the specific construction scheme is as follows:

Step 1: Performing channel encoding of polar codes by Gaussian approximation and obtaining vector M. Then initializing the population P by using indexes of all information bits.

Step 2: Selection operation is performed and the selected population P_s is obtained.

Step 3: Carrying out binary encoding on selected population P_s, performing crossover and mutation operation subsequently. After suffering from mutation operation, the individual with the lowest fitness is stored in a vector named *path*.

TABLE 1. Hyperparameters of genetic algorithm.

Hyperparameters	Values
Size of population	512
Crossover rate	1
Mutation rate	0.2
Competitive number of individuals	16
Iterative times	10000
Selection strategy	Tournament selection
Crossover function	One-point crossover
Mutation function	Simple mutation

Step 4: Repeating Step 2-3 until the maximum number of iterations is reached. Then a vector *path*, length of the maximum iteration times, is obtained.

Step 5: Counting the occurrence times of indexes of all information bits in vector *path* and the CFPS is obtained, which is consist of all indexes in vector *path* with non-zero occurrence times.

When SNR is set to 1.5dB, 2dB and 2.5dB, the normalized occurrence times of all indexes in constructed CFPS is shown in Fig. 3. At the same time, these indexes with higher occurrence times are easier to suffer from type1 errors. It can be seen in Fig. 3 that it is almost impossible to cause type1 errors for these greater indexes. The situation can be attributed to the polarization characteristics of polar codes. As code length increases, polarization channels tend to become either noiseless or completely noisy. Besides, the bigger indexes of information bits are, the more reliable its corresponding subchannels are. Therefore, these subchannels with greater indexes are more reliable than that with smaller indexes.

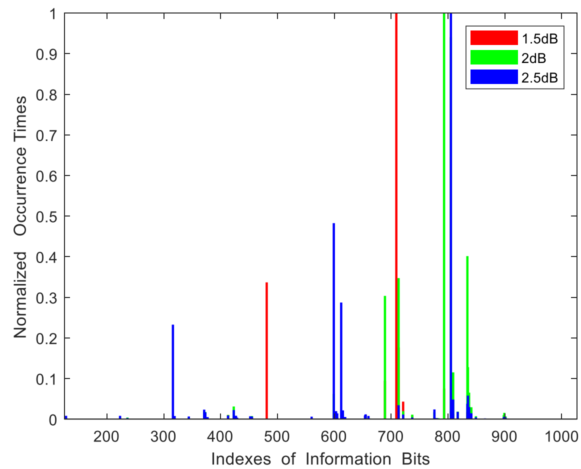


FIGURE 3. Normalized occurrence times of all indexes for information bits in vector *path* for P(1024,512) polar codes.

The size comparison among CFPS based on GA and other state-of-the-art CFPSs for P(1024,512) polar codes is shown in Table 2. In original SCF decoding, the search scope of CFPS is indexes of all information bits. However, for these SCF decoding schemes [14], [15], decoding latency

TABLE 2. Sizes of different CFPSs for p(1024,512) polar codes.

SNR	1dB	1.5dB	2dB	2.5dB
Original SCF [14]	512	512	512	512
MBC Set [16]	-	-	161	143
CS [15]	110	112	117	129
CFPS-GA	87	93	95	103

is determined by the size of search scope [16]. It is obvious that the sizes of MBC, CS and CFPS-GA are smaller than the number of all information bits. For example, when SNR is 2.5dB, compared with original SCF, sizes of MBC, CS and CFPS-GA can be reduced to 7/25, 1/4, 1/5, respectively.

B. SCF-GA DECODING ALGORITHM

After obtaining new CFPS constructed by GA, its corresponding SCF-GA decoding algorithm is shown in Algorithm 5. The decoding process of SCF-GA algorithm is similar to that of original SCF algorithm, and the only difference is the CFPS. As mentioned in Table 2, the size of CFPS constructed by GA is smaller than that of original SCF decoding, which means that type1 errors can be more effectively found by SCF-GA decoding under limited flipping attempts. The maximum number of flipping attempts is denoted by $T_{\max}$ in Algorithm 5.

Algorithm 5 SCF-GA Decoding Algorithm

Input: y_1^N, ξ, constructed CFPS based on GA: CFPS-GA
Output: $\hat{u}_1^N$
Initialization: $y_1^N \leftarrow \mathbf{0}$
  the received vector y_1^N update
  $(\hat{u}_1^N, L_i(y_1^N, \hat{u}_1^{i-1})) \leftarrow SC(y_1^N, \xi)$
  if CRC($\hat{u}_1^N$) == success
    break;
  else
    $\mathbf{I} \leftarrow$ sorting $i \in$ CFPS-GA in an increasing order
      of $|L_i(y_1^N, \hat{u}_1^{i-1})|$
    for $j = 1$ **to** $T_{\max}$ **do**
      $\hat{u}_1^N \leftarrow SCF(y_1^N, \xi, \mathbf{I}(j))$
      if CRC($\hat{u}_1^N$) == success
        break;
      end if
    end for
  end if
Return $\hat{u}_1^N$

V. SIMULATION RESULTS

In this section, the decoding performance and complexity of proposed SCF-GA decoding algorithm are evaluated by simulation. The SNR for CFPS-GA is set to 2.5dB while other hyperparameters about GA are set according to Table 1. The number of individuals in each population is equal to the number of all information bits. What's more, polar codes of length 1024 and 512 are chosen for simulation. Then other hyperparameters (e.g., code rate, channel modulation) are same with that of constructed process of CFPS based on GA. At last, $T_{\max}$ is set to 103 for polar codes of length 1024 and it is 68 for polar codes of length 512.

Fig. 4 shows the frame error rate (FER) of proposed SCF-GA decoding algorithm for P(1024,512) polar codes against other decoding algorithms, where SCF-CS denotes the SCF decoding algorithm based on CS [15]. CRC-SCL2 and CRC-SCL4 represent CRC-aided SCL decoding algorithms with list size of $L = 2$ and $L = 4$, respectively. SCO1 represents ~~the~~ oracle-assisted SC decoding algorithm, whose decoding performance can be regarded as the lower bound of single bit SCF decoding. In SCO1 decoding algorithm, the first type1 error that met by SC decoder can be directly corrected because correct decoded codewords have been known by the decoder. As shown in Fig. 4, the proposed SCF-GA decoding algorithm has almost same FER performance with SCO1 decoding algorithm at all observed SNR. Besides, the FER performance of proposed SCF-GA decoding algorithm is the same as that of original SCF and SCF-CS decoding algorithms when SNR is less than 2dB. However, FER performance of proposed SCF-GA decoding algorithm is better than them when SNR is more than 2dB. Compared with original SCF and SCF-CS decoding algorithms, the proposed SCF-GA decoding algorithm has a performance gain of about 0.1dB when FER is 10^{-3}. It can be observed in Fig. 4 that the proposed SCF-GA algorithm has better FER performance compared with SCF-CS decoding algorithm at high SNR. This is due to the fact that the size of CS gradually increases with SNR. When SNR is set to 2.5dB, its corresponding size of CS is 124 while the number of maximum flipping attempts is set to 103. Therefore, there is a little performance loss for SCF-CS decoding algorithm compared with SCO1 decoding algorithm.

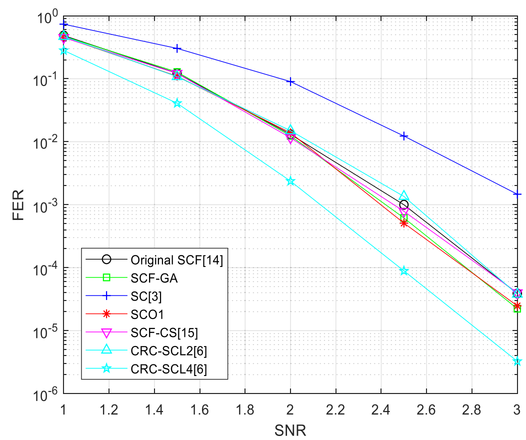


FIGURE 4. FER performance for P(1024,512) polar codes.

It can also be seen in Fig. 4 that FER performance of the proposed SCF-GA decoding algorithm is better than CRC-aided SCL decoding algorithm with $L = 2$ at high SNR region, and exhibits nearly same FER performance as CRC-aided SCL decoding algorithm with $L = 2$ at low SNR region. However, FER performance of CRC-aided

SCL decoding algorithm with $L = 4$ is more excellent than SCF-GA decoding algorithm at all SNR region. Effect of the proposed SCF-GA decoding algorithm on bit error rate (BER) for P(1024,512) polar codes is shown in Fig. 5. As shown in Fig. 5, BER performance of proposed SCF-GA decoding algorithm is better than SC and original SCF decoding algorithms. When SNR is less than 2.2dB, BER performance of SCF-GA decoding algorithm is slightly worse than that of CRC-aided SCL decoding algorithm with $L = 2$, while SCF-GA decoding algorithm outperforms CRC-aided SCL decoding algorithm with $L = 2$ at high SNR region. What's more, BER performance of SCF-GA decoding algorithm is close to that of SCO1 decoding algorithm. Then with the increasing of SNR, the BER gap between SCF-GA decoding algorithm and SCO1 decoding algorithm becomes small gradually.

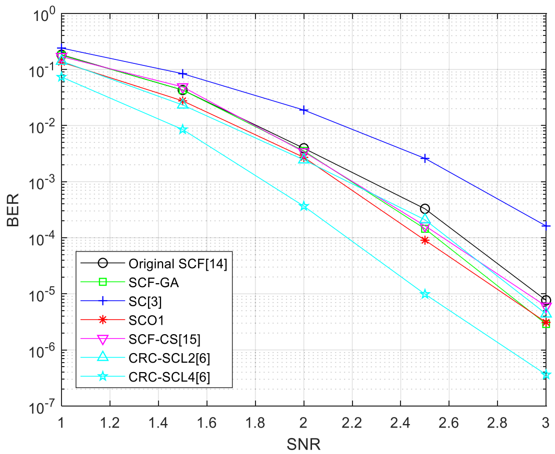


FIGURE 5. BER performance for P(1024,512) polar codes.

Furthermore, in order to verify the universality of the proposed algorithm, Fig. 6 shows FER performance for polar codes with code length $N = 512$ and $R = 0.5$. It can be seen in Fig. 6 that the proposed SCF-GA and SCF-CS decoding algorithms both can provide better decoding performance compared with original SCF decoding when SNR is more than 1.5dB. What's more, these two kinds of decoding algorithms both can reach the decoding performance of SCO1 algorithm when SNR is less than 2.5dB. However, as the increasing of SNR, there is performance loss for proposed SCF-GA decoding algorithm. It is because the size of CFPS for GA is 71 and that is 68 for CS, while the number of maximum flipping times is 68. Hence the proposed SCF-GA decoding algorithm may not find all of the type1 errors under limited flipping attempts. When SNR is less than 1.5dB, SCF-GA decoding algorithm can provide same FER performance with CRC-aided SCL decoding algorithm with $L = 2$. However, SCF-GA decoding algorithm can obtain FER performance gain at higher SNR region. For example, compared with CRC-aided SCL decoding algorithm with $L = 2$, SCF-GA decoding algorithm can have 0.3dB FER performance gain with FER=10^{-3}.

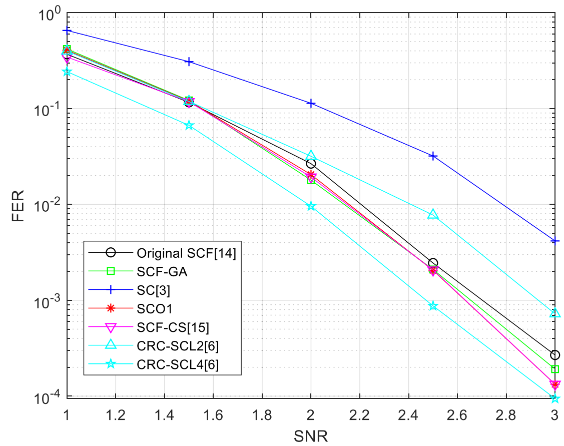


FIGURE 6. FER performance for P(512,256) polar codes.

BER performance of different decoding algorithms with $N = 512$ is shown in Fig. 7. The proposed SCF-GA decoding algorithm has about 0.2dB BER performance gain compared with CRC-aided SCL decoding algorithm with $L = 2$ when BER is 10^{-3}. However, BER performance of CRC-aided SCL decoding algorithm with $L = 4$ is better than SCF-GA decoding algorithm at all SNR region. Then, BER performance of SCF-GA decoding algorithm is better than that of original SCF decoding algorithm. In addition, SCF-GA decoding algorithm demonstrates about 0.1dB performance gain over SCF-CS decoding algorithm when BER is 10^{-3}.

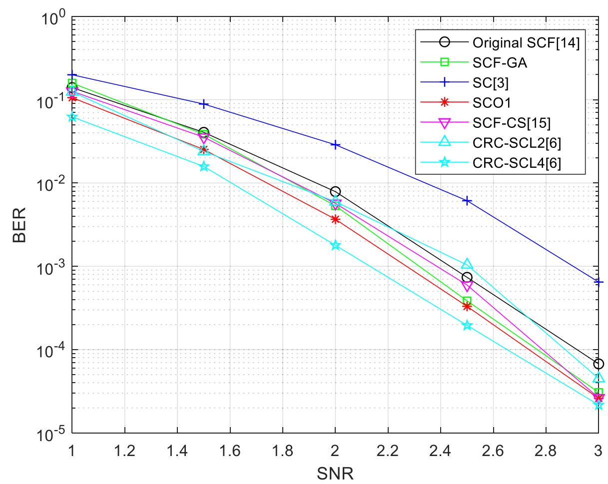


FIGURE 7. BER performance for P(512,256) polar codes.

In Fig. 8 and Fig. 9, we compare the average normalized complexity of proposed SCF-GA decoding algorithm and other decoding algorithms. The average normalized complexity is usually used to describe the computational complexity of SCF decoding algorithm. Besides, the average normalized complexity is normalized by standard SC decoding algorithm. It can be observed that as SNR increases, the average normalized complexity of SCF decoding algorithm quickly decreases to the same level as SC decoding algorithm. This is due to the fact that as SNR increases, the channel tends

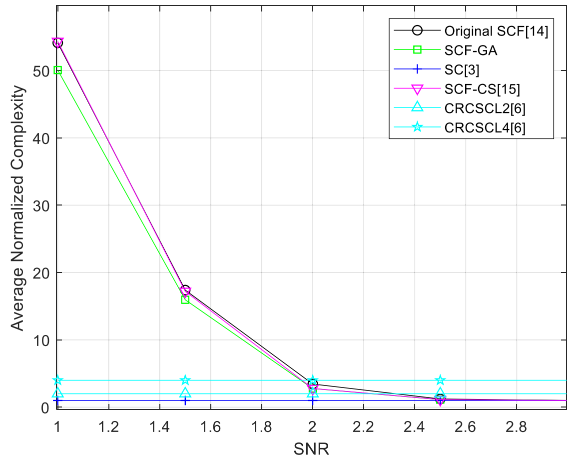


FIGURE 8. The average normalized complexity for P(1024,512) polar codes.

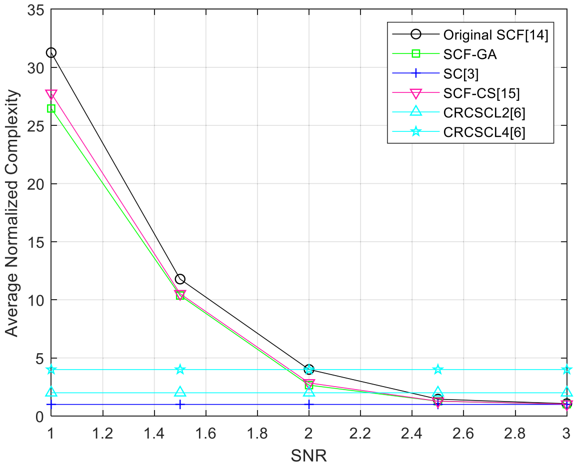


FIGURE 9. The average normalized complexity for P(512,256) polar codes.

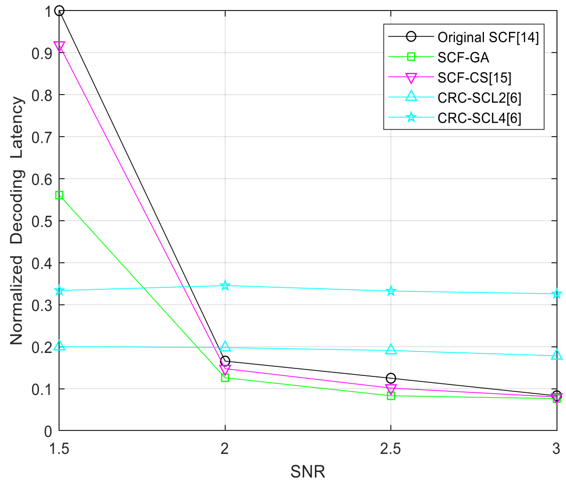


FIGURE 10. The normalized decoding latency for P(1024,512) polar codes.

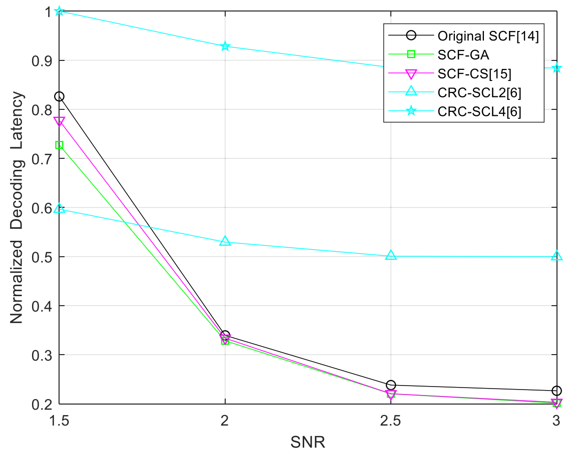


FIGURE 11. The normalized decoding latency for P(512,256) polar codes.

to become more reliable. However, the average normalized complexity of CRC-aided SCL decoding algorithm is always the same as its corresponding value of L, which is independent of SNR. Therefore, in Fig. 8 and Fig. 9, proposed SCF-GA decoding algorithm has higher average normalized complexity compared with CRC-aided SCL decoding algorithm with $L=2$ and $L=4$ at low SNR region for different code lengths of polar codes. Nevertheless, as the increasing of SNR, SCF-GA decoding algorithm is gradually revealing its superiority in average normalized complexity compared with CRC-aided SCL decoding algorithm with $L=2$ and $L=4$. In addition, the average normalized complexity of SCF-GA decoding algorithm is lower than that of SCF-CS and original SCF decoding algorithms at low SNR. It means that SCF-GA decoding algorithm based CFPS constructed by GA needs fewer flipping attempts to pass CRC check, reducing computational complexity of SCF-GA decoding algorithm.

In order to evaluate the decoding latency of proposed SCF-GA decoding algorithm and other decoding algorithms, we reveal the normalized decoding latency curves of different SCF decoding algorithms in Fig. 10 and Fig. 11. The decoding latency at each SNR is obtained by simulating 1×10^5 frames. By measuring time of decoding the same number of codewords, decoding latency of different algorithms can be obtained in our work. In order to ensure the fairness of comparison, simulation conditions are same for all comparative decoding algorithms. Then, decoding latency at each code length is normalized by the maximum value of decoding latency of all comparative algorithms for every code length of polar codes. Finally, normalized decoding latency of every algorithm can be estimated after carrying out the normalization operation. It can be observed in Fig. 10 and Fig. 11 that original SCF decoding algorithm has the highest normalized decoding latency among all comparative SCF decoding algorithms. The reason is that the CFPS of original SCF decoding algorithm is consisted of indexes of all information bits, and all LLR values of information bits need to be sorted to determine the first flipping position. However, it is less prone to causing type1 errors for these greater indexes with higher reliability in Fig. 3. Compared with SCF-CS decoding algorithm, the proposed SCF-GA decoding algorithm has almost same normalized decoding latency at high SNR domain while the normalized decoding latency of proposed SCF-GA is lower than that of SCF-CS decoding algorithm at low SNR.

For example, when code length of polar codes is 1024 and SNR is 1.5dB, the normalized decoding latency of proposed SCF-GA decoding algorithm can be reduced by 39%.

In terms of CRC-aided SCL decoding algorithm, it can be seen in Fig. 10 and Fig. 11 that normalized decoding latency almost remains constant at observed SNR region. In contrast, the normalized decoding latency of SCF algorithm is dependent on SNR. With the increasing of SNR, channel tends to become more reliable. Then SCF decoding algorithm needs fewer flipping attempts to successfully decode, reducing normalized decoding latency. The conclusion can be presented in Fig. 10 and Fig. 11. For P(1024,512) polar codes, the normalized decoding latency of SCF-GA algorithm is higher than CRC-aided SCL with L=4 at low SNR region. However, when the code length is 512, the normalized decoding latency of SCF-GA algorithm is lower than CRC-aided SCL algorithm with L=4 even though it is at low SNR region. This is because the normalized decoding latency of SCF algorithm is mainly determined by the size of CFPS [16]. When code length is equal to 1024, SCF-GA decoding algorithm needs more flipping attempts to accomplish decoding. Nevertheless, when code length is decreased to 512, the number of flipping attempts is greatly reduced so that the normalized decoding latency of CRC-aided SCL algorithm with L=4 is much higher than SCF-GA algorithm. At last, the proposed SCF-GA algorithm demonstrates lower normalized decoding latency compared with CRC-aided SCL algorithm with L=2 at high SNR region for both P(1024,512) polar codes and P(512,256) polar codes, and proposed SCF-GA algorithm also illustrates better decoding performance compared with CRC-aided SCL algorithm with L=2 at high SNR region.

TABLE 3. Normalized decoding time steps for p(1024,512) polar codes.

SNR	1dB	1.5dB	2dB	2.5dB	3dB
SCF [14]	54.11	17.39	3.44	1.22	1.01
SCL [6]	1.50	1.50	1.50	1.50	1.50
SCF-CS [15]	54.32	17.20	2.79	1.13	1.01
SCF-GA	50.07	15.95	2.77	1.12	1.01

TABLE 4. Normalized decoding time steps for p(512,256) polar codes.

SNR	1dB	1.5dB	2dB	2.5dB	3dB
SCF[14]	31.26	11.78	4.00	1.45	1.06
SCL [6]	1.50	1.50	1.50	1.50	1.50
SCF-CS [15]	27.76	10.52	2.85	1.28	1.03
SCF-GA	26.46	10.37	2.66	1.27	1.02

The comparisons of normalized time steps for different code lengths of polar codes are shown in Table 3 and Table 4, respectively. The time steps required for standard SC decoder are $2N$-1 while that required for SCL decoder are $3N$-2 [3], [6]. Then, time steps of different decoders are normalized by that of SC decoder. It can be seen that SCF decoders need more time steps compared with SCL decoder at low SNR. This is because that more SC flipping attempts are required for SCF decoders at low SNR. However, SCF decoders need less time steps compared with SCL decoder with the increasing of SNR. Furthermore, SCF decoder based on CFPS constructed by GA needs the least time steps among all competitive SCF decoders. Besides, decoding latency is determined by the size of search scope for these SCF decoding schemes [14], [15]. It can be seen in Table 2 that the constructed CFPS based on GA has the smallest size of set among all CFPS. Therefore, the SCF decoder based on CFPS constructed by GA needs less time to search for flipping positions.

VI. CONCLUSION

In this work, a new method to construct CFPS by GA has been provided. During the construction process of CFPS based on GA, these indexes of all information bits constitute the initial population of GA. Through some genetic operators, a new CFPS can be obtained. On this basis, the SCF decoding algorithm based on constructed CFPS has been presented. Simulation results show that the SCF decoding algorithm based on new constructed CFPS can achieve competitive decoding performance compared with some state-of-the-art SCF decoding algorithms. Furthermore, the calculation complexity and decoding latency of proposed SCF decoding is lower than that of other SCF decoding algorithms at low SNR. Besides, this work also provides a new insight into the application of AI technology in decoding of polar codes.

REFERENCES

[1] Y. Zhang, K. Peng, J. Song, and Y. Wu, "Quasi-cyclic spatially coupled LDPC code for broadcasting," *IEEE Trans. Broadcast.*, vol. 66, no. 1, pp. 187–194, Mar. 2020.

[2] K. D. Rao and T. A. Babu, "Performance analysis of QC-LDPC and polar codes for eMBB in 5G systems," in *Proc. Int. Conf. Electr., Electron. Comput. Eng. (UPCON)*, Nov. 2019, pp. 1–6.

[3] 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.

[4] A. Jalali and Z. Ding, "Joint detection and decoding of polar coded 5G control channels," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2066–2078, Mar. 2020.

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

[6] K. Niu and K. Chen, "CRC-aided decoding of polar codes," *IEEE Commun. Lett.*, vol. 16, no. 10, pp. 1668–1671, Oct. 2012.

[7] F. Cheng, A. Liu, Y. Zhang, and J. Ren, "Bit-flip algorithm for successive cancellation list decoder of polar codes," *IEEE Access*, vol. 7, pp. 58346–58352, 2019.

[8] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, "Fast list decoders for polar codes," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 318–328, Feb. 2016.

[9] S. A. Hashemi, C. Condo, and W. J. Gross, "Fast and flexible successive-cancellation list decoders for polar codes," *IEEE Trans. Signal Process.*, vol. 65, no. 21, pp. 5756–5769, Nov. 2017.

[10] S. A. Hashemi, M. Mondelli, S. H. Hassani, C. Condo, R. L. Urbanke, and W. J. Gross, "Decoder partitioning: Towards practical list decoding of polar codes," *IEEE Trans. Commun.*, vol. 66, no. 9, pp. 3749–3759, Sep. 2018.

[11] D. Kim and I.-C. Park, "A fast successive cancellation list decoder for polar codes with an early stopping criterion," *IEEE Trans. Signal Process.*, vol. 66, no. 18, pp. 4971–4979, Sep. 2018.

[12] X. Liu, S. Wu, X. Xu, J. Jiao, and Q. Zhang, "Improved polar SCL decoding by exploiting the error correction capability of CRC," *IEEE Access*, vol. 7, pp. 7032–7040, 2019.

[13] X. Wang, H. Zhang, R. Li, L. Huang, S. Dai, Y. Huangfu, and J. Wang, "Learning to flip successive cancellation decoding of polar codes with LSTM networks," in *Proc. IEEE 30th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Istanbul, Turkey, Sep. 2019, pp. 1–5.

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

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

[16] Z. Zhang, K. Niu, and C. Dong, "Multi-Bit-Flipping decoding of polar codes based on medium-level bit-channels sets," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Marrakesh, MO, USA, Apr. 2019, pp. 1–6.

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

[18] N. Doan, S. A. Hashemi, F. Ercan, T. Tonnellier, and W. J. Gross, "Neural dynamic successive cancellation flip decoding of polar codes," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, Nanjing, China, Oct. 2019, pp. 272–277.

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

[20] C. Condo, F. Ercan, and W. J. Gross, "Improved successive cancellation flip decoding of polar codes based on error distribution," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Barcelona, Spain, Apr. 2018, pp. 19–24.

[21] F. Ercan, C. Condo, S. A. Hashemi, and W. J. Gross, "Partitioned successive-cancellation flip decoding of polar codes," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, May 2018, pp. 1–6.

[22] C. Wen, J. Xiong, L. Gui, Z. Shi, and Y. Wang, "A novel decoding scheme for polar code using convolutional neural network," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Jeju, South Korea, Jun. 2019, pp. 1–5.

[23] F. Liang, C. Shen, and F. Wu, "An iterative BP-CNN architecture for channel decoding," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 144–159, Feb. 2018.

[24] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 119–131, Feb. 2018.

[25] J. Seo, J. Lee, and K. Kim, "Decoding of polar code by using deep feed-forward neural networks," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Maui, HI, USA, Mar. 2018, pp. 238–242.

[26] N. Doan, S. Ali Hashemi, and W. J. Gross, "Neural successive cancellation decoding of polar codes," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Kalamata, Greece, Jun. 2018, pp. 1–5.

[27] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Decoder-tailored polar code design using the genetic algorithm," *IEEE Trans. Commun.*, vol. 67, no. 7, pp. 4521–4534, Jul. 2019.

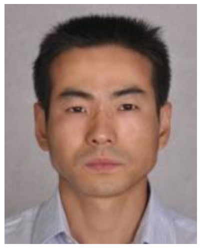
[28] L. Huang, H. Zhang, R. Li, Y. Ge, and J. Wang, "AI coding: Learning to construct error correction codes," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 26–39, Jan. 2020.

[29] X. Wang, T. Wang, J. Li, and Y. Zhang, "Improved multiple bit-flipping fast-SSC decoding of polar codes," *IEEE Access*, vol. 8, pp. 27851–27860, 2020.

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

XIUMIN WANG was born in April 1963. She received the M.E. degree in communication and electronic system from the Dalian University of Technology. She is currently a Professor with the College of Information Engineering, China Jiliang University. Her research interests include channel coding, signal detection, and signal and information processing.

QIANGQIANG MA was born in November 1995. He received the B.E. degree in communication engineering from Shenyang University, in 2018. He is currently pursuing the master's degree with the College of Information Engineering, China Jiliang University. His research interests include channel coding and deep learning.



JUN LI was born in February 1977. He received the Ph.D. degree in communication and information system from the South China University of Technology, in 2005. He worked at ZTE Corporation, in 2005. He has been an Associate Professor with the Binjiang College, Nanjing University of Information Science & Technology. His research interests include channel coding, signal detection, LTE physical layer standard, and wireless resource optimization.

HONGCHAO ZHANG was born in August 1997. She received the B.E. degree in automation from Quzhou University, in 2019. She is currently pursuing the master's degree with the College of Information Engineering, China Jiliang University. Her research interests include channel coding and deep learning.

WENCHAO XU received the B.E. and M.E. degrees from Zhejiang University, Hangzhou, China, in 2008 and 2011, respectively, and the Ph.D. degree from the University of Waterloo, Canada, in 2018. In 2011, he joined Alcatel-Lucent Shanghai Bell Company Ltd., where he was a Software Engineer of Telecom Virtualization. He is currently an Assistant Professor with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. His research interests include wireless communications with emphasis on resource allocation, network modeling, and AI applications.