SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

November 2025, Vol. 68, Iss. 11, 212301:1–212301:15
https://doi.org/10.1007/s11432-024-4320-3

Tanner-graph-assisted belief propagation decoding for large kernel polar codes: low-complexity design and enhancement method

Yangyang LIU, Yu ZHANG, Guanghua LIU, Jiaxi ZHOU, Lixia XIAO & Tao JIANG*

Research Center of 6G Mobile Communications, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China

Received 5 November 2024/Revised 20 January 2025/Accepted 20 February 2025/Published online 16 September 2025

Abstract Large kernel polar codes can provide outstanding error-correction performance for finite-length coding, which is expected to support ultra-high reliability and ultra-low latency communication (URLLC) for 6G. However, the complexity of decoding for large kernel polar codes grows exponentially with the kernel size. In this paper, we study Tanner-graph-assisted (TGA) decoding to further reduce complexity while maintaining satisfactory error-correction performance. We first construct a low-complexity parallel TGA belief propagation (TGA-BP) decoder. The decoder takes the kernel matrix that achieves the optimal exponent as the Tanner graph and selects key nodes for iterative decoding. In particular, the Tanner graph and iterative decoding equations for arbitrary dimensional linear binary kernels are derived. Then, a two-step Tanner graph optimization strategy is further proposed to enhance the TGA-BP. It includes a kernel matrix generation method that obtains different kernels with the optimal exponent and a kernel matrix selection method that maximizes the mean log-likelihood ratios (mLLR). The simulation results demonstrate that our scheme achieves significant complexity reduction and error-correction improvement compared to large kernel polar codes under successive cancellation (SC) decoding over the additive white Gaussian noise (AWGN) channel.

Keywords polar codes, large kernels, belief propagation decoding, Tanner graph, URLLC

Citation Liu Y Y, Zhang Y, Liu G H, et al. Tanner-graph-assisted belief propagation decoding for large kernel polar codes: low-complexity design and enhancement method. Sci China Inf Sci, 2025, 68(11): 212301, https://doi.org/10.1007/s11432-024-4320-3

1 Introduction

Polar codes, capable of theoretically achieving Shannon's capacity, have already made their way into the specification for the reliable channel control in 5G new radio [1, 2]. The original design of polar codes was based on the polarization phenomenon caused by the Kronecker product of the two-dimensional kernel $\boldsymbol{F}_2 = \left[\begin{smallmatrix}1 & 0\\ 1 & 1\end{smallmatrix}\right]$. When it comes to the ultra-reliability and ultra-low latency communication (URLLC) scenarios of 6G, 5G polar codes can hardly meet the stringent requirements. The reason is that shorter code length will be employed for error-correction codes to reduce latency in 6G URLLC. In this case, the channel polarization is greatly weakened, thus dramatically eroding the performance of polar codes [3]. Consequently, improving the polarization capability of polar codes under short block lengths has become an urgent need.

Large kernel polar codes, which can improve polarization capabilities by constructing large polarization kernel matrices [4, 5], have attracted increased attention. Ref. [6] validated that better polarization effect can be achieved for a large kernel size, where the necessary and sufficient conditions for a large kernel matrix with polarization effect are also provided. In [7], decompositions were used to design good binary kernels. In [8], non-binary kernels with a larger exponent based on Reed-Solomon codes are provided. In [9], linear and non-linear binary kernels with maximal polarization exponents up to dimension 16 were investigated. With these technologies, polar codes have been shown to improve the asymptotic error probability and are poised to enhance their competitiveness compared to other state-of-the-art channel coding schemes.

* Corresponding author (email: tao.jiang@ieee.org)

Despite the promising theoretical results, large kernels pose challenges for implementing the polar decoders. Generally, the complexity of straightforward successive cancellation (SC) decoder for kernel $\boldsymbol{F}_l$-based polar codes with length N behaved like $O(2^l N\log_l N)$ [6], which grows exponentially with the kernel size l. Taking the kernel size $l = 16$ and the code length $N = 256$ as an example, the classic polar SC decoding requires 2048 XOR units for computational complexity [10]. When it comes to large kernel polar codes, the direct computation cost may increase by 2^{16} times. Therefore, it is not suitable to straightforwardly apply conventional decoding algorithms for large kernel polar codes, and low-complexity schemes should be put forward.

There exist several studies that address the decoding complexity challenge of large kernel polar codes. An l-formula method for simplification of the log-likelihood ratio (LLR) expressions was suggested in [11]. However, only l-formula for $\boldsymbol{F}_3$ and $\boldsymbol{F}_4$ were given. Ref. [12] tried to generalize this formulation, but provided l-formula of kernels up to dimension 11. An approximate kernel processing method based on window decoding was suggested in [13], and some kernels of dimensions 16 and 32 were published together. The corresponding polar codes were shown to have lower decoding complexity compared to $\boldsymbol{F}_2$-based polar codes with the same performance. Notably, not all kernels support this approach in an efficient way. To this end, Ref. [14] proposed a unified scheme for kernel processing based on the recursive trellis, which uses the max operator to approximate the summation operator in the computation of kernel processing. Unfortunately, the window decoding or recursive trellis, both have a tradeoff between complexity and error correction performance. Recently, a W-expression method has been proposed by considering the bit channel transition probability [15], which reduces the complexity of the straightforward SC decoder to $O(l^2 N\log_l N)$ while maintaining the error correction performance advantage of the large kernels. However, the maximum kernel processing dimension is 16, and the computational complexity is still higher than that of the classic polar codes.

To summarize, the above-mentioned studies mainly improved serial SC decoding to reduce complexity. On the one hand, the computational complexity grows at least quadratically with the kernel size, which is still higher than that of the classic polar codes. On the other hand, the serial decoding structure will incur additional delay, which deviates from the ultra-low latency vision of 6G URLLC. Consequently, finding large kernels with good polarization properties and low decoding complexity to meet the stringent constraints of 6G URLLC remains, in general, an open problem.

It is a fact that classical $\boldsymbol{F}_2$-based polar codes exploit the Tanner graph tool, which enables their serial or parallel decoding with low complexity [16,17]. Motivated by this, Tanner-graph-assisted parallel decoding for large kernel polar codes is studied in this paper. The challenges of applying the Tanner graph for large kernel polar decoding lie in two aspects. First, the Tanner graph of classic polar and low-density parity-check (LDPC) codes [18] is no longer applicable, and one must redesign the Tanner graph and decoding algorithm for large kernels. Second, given the kernel size, there are a large number of kernel matrices with optimal polarization effect, which leads to a huge potential candidate space for the Tanner graphs. Therefore, selecting the most suitable one for decoding is very tricky. We address these two challenges by designing a unified low-complexity decoding framework and then investigating the Tanner graph optimizing problem. The main contributions can be summarized as follows.

(1) We propose a parallel Tanner-graph-assisted belief propagation (TGA-BP) decoder for large kernel polar codes. The decoder takes the kernel matrix that achieves the optimal polarization exponent as the Tanner graph and selects key nodes for iterative decoding. In particular, the Tanner graph and iterative decoding equations for arbitrary dimensional linear binary kernels are derived. To the best of our knowledge, this is the first time that the Tanner graph is employed to assist large kernel polar codes decoding.

(2) Since different kernel matrices with the same polarization exponent result in differentiated Tanner graphs, which will cause fluctuations in TGA-BP decoding performance. We construct a large polarization kernel matrix generation method based on a genetic algorithm, which has the ability to obtain different kernels with the optimal polarization exponent. Furthermore, a Tanner graph selection strategy that maximizes the mean log-likelihood ratios (mLLR) is designed. This enables the TGA-BP decoder to select the appropriate Tanner graph for decoding.

(3) Extensive numerical results validate our proposed TGA-BP decoder is capable of providing significant performance gain over existing SC decoders for large kernel polar codes. Particularly, under the additive white Gaussian noise (AWGN) channel, when the kernel dimension $l = 7$ and the code length $N = 343$, a frame error rate (FER) performance gain of 0.3 dB is obtained. Moreover, TGA-BP decoding has a lower computational cost, reducing the complexity from $O(l^2 N\log_2 N)$ to $O(\frac{2EN}{l}\log_l N)$, where the

maximum of E is $\frac{l^2-2l+2}{2}$. In addition, we simulate TGA-BP list (TGA-BPL) decoding, which also has great performance advantages compared with classic polar successive cancellation list (SCL) decoding.

The remainder contents are organized as follows. In Section 2, we introduce the basics of polar codes and large kernel polar codes. A detailed Tanner graph design and iterative decoding formula derivation are given in Section 3. In Section 4, we elaborate on the designed kernel matrix generation scheme and Tanner graph selection strategy of the enhanced TGA-BP. Section 5 presents extensive simulation results. Finally, we conclude our paper in Section 6.

2 Preliminaries

2.1 Polar codes

Owing to the channel polarization phenomenon, the capacity of N binary-input coordinate channels is extremely distributed in [19]. Symbol (N, K) is defined as a polar code with code length N and information length K. $\mathcal{I}$ indicates the index set that belongs to information bits, which can be effectively constructed by [20]. Assuming u_1^N as the input vector, codeword x_1^N is generated by Kronecker power $\boldsymbol{F}_2{}^{\otimes n}$ as follows:

$$x_1^N = u_1^N \boldsymbol{F}_2^{\otimes n}, \boldsymbol{F}_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}. \tag{1}$$

Decoding of polar codes is performed by SC decoding on the Tanner graph of the code. We assume that x_1^N is binary phase shift keying (BPSK) modulated and transmitted over an AWGN channel. The received sequence y_1^N is converted to log-likelihood ratio (LLR) and then input into the Tanner graph for decoding. Specifically, the sequence u_1^N is decoded bit by bit in the order from u_1 to u_N. Taking the code length $N = 4$ as an example, the Tanner graph is illustrated in Figure 1. Nodes receive channel LLRs and are expressed as $\mathrm{LLR}_i = \frac{2y_i}{\sigma^2}$. According to the LLR_5 and LLR_7 of nodes 5 and 7, the LLR value of node a is calculated as

$$\mathrm{LLR}_a = f(\mathrm{LLR}_5, \mathrm{LLR}_7). \tag{2}$$

Similarly, the LLR value of node b is computed as

$$\mathrm{LLR}_b = f(\mathrm{LLR}_6, \mathrm{LLR}_8). \tag{3}$$

Furthermore, the LLR of node 1 is obtained by

$$\mathrm{LLR}_1 = f(\mathrm{LLR}_a, \mathrm{LLR}_b), \tag{4}$$

where the function f is defined as $f(x, y) = \ln(\frac{1+\mathrm{e}^{x+y}}{\mathrm{e}^x+\mathrm{e}^y})$. Decoding of u_1 can be implemented using LLR_1; i.e., one can make decisions:

$$\hat{u}_i = \begin{cases} 0, & i \notin \mathcal{I}, \\ 1, & \mathrm{LLR}_i < 0, i \in \mathcal{I}, \\ 0, & \mathrm{LLR}_i \geqslant 0, i \in \mathcal{I}. \end{cases} \tag{5}$$

According to the decoding result $\hat{u}_1$ of node 1, the LLR of node 2 is calculated as

$$\mathrm{LLR}_2 = g(\mathrm{LLR}_a, \mathrm{LLR}_b, u_1), \tag{6}$$

where the function g is expressed as $g(x, y, u_i) = (-1)^{u_i}x + y$. By analogy, one can deduce the decoding result of each u_i based on the Tanner graph.

2.2 Large kernel polar codes

Similar to polar encoding, the codeword x of large kernel polar codes can also be obtained from the generator matrix $\boldsymbol{G}_N$, which is expressed as $x_1^N = u_1^N \boldsymbol{G}_N$. The difference is that the generator matrix of the large kernel polar codes consists of a high-dimensional polarization kernel matrix. Specifically, the generator matrix of a (N, K) large kernel polar code is specified as $\boldsymbol{G}_N = \boldsymbol{F}_l^{\otimes n}$, where $N = l^n$, and $\boldsymbol{F}_l$ denotes polarization kernel with a size of $l \times l$.

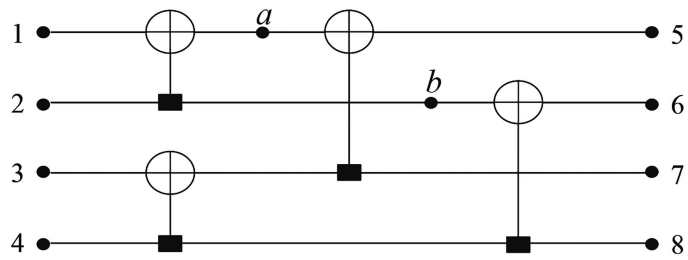

Figure 1 Tanner graph of the $\boldsymbol{F}_2$-based polar code with $N = 4$.

The code structure of large kernel polar codes has the same function as the generator matrix, which represents the internal relationship between the input vector and the codeword. An example with a kernel size of 3 is shown in Figure 2. It is divided into p stages from right to left, and each stage contains N/l kernels of the same size. The key to code structure design for large kernel polar codes is to obtain the connection vector between any two adjacent stages. The detailed acquisition process of the connection vectors is summarized as follows.

Step 1. Calculate the connection vectors between stage 1 and stage $p-1$, which can be calculated as

$$w_i = (\alpha_i, \alpha_i + \beta_{i+1}, \alpha_i + 2 \times \beta_{i+1}, \ldots, \alpha_i + (N/\beta_{i+1} - 1)\beta_{i+1}), \tag{7}$$

where $i \in [2, p-1]$, $\beta_1 = 1$, $\beta_i = \prod_{j=1}^{i-1} lj$, $\alpha_i = (1, li+1, 2 \times li+1, \ldots, (\beta_i - 1) \times li + 1, \ldots, li, li+li, 2 \times li + li, \ldots, (\beta_i - 1) \times li + li)$.

Step 2. Compute the connection vector w_p between the $(p-1)$-th stage and the p-th stage, which is defined as

$$w_p = (1, lp+1, 2 \times lp+1, \ldots, (\beta_p - 1) \times lp + 1, \ldots, lp, lp+lp, 2 \times lp + lp, \ldots, (\beta_p - 1) \times lp + lp). \tag{8}$$

Step 3. Obtain w_1, which represents the connection vector between the codeword x and the first stage. In particular, the existence of vector w_1 makes the codeword generated according to the generator matrix strictly consistent with that obtained from the Tanner graph.

With a code structure, SC decoding can be performed on it. However, due to the lack of graph analysis tools, such as the Tanner graph of polar codes, the kernels in the code structure are considered black boxes. Therefore, SC decoding of large kernel polar codes relies on the basic iterative formula of a general kernel matrix $\boldsymbol{F}_l$:

$$W_l^{(i)}(y_1^l, u_1^{i-1}|u_i) = \frac{1}{2^{l-1}} \sum_{u_{i+1}^l} W_l(y_1^l|u_1^l) = \frac{1}{2^{l-1}} \sum_{u_{i+1}^l} W(y_1|(u_1^l F_l)_1) \cdots W(y_l|(u_1^l F_l)_l), \tag{9}$$

where $u_{i+1}^l \in \{0,1\}^{l-i}$. The LLR of the i-th bit channel is defined as

$$\mathrm{LLR}_l^{(i)} = \ln \frac{W_l^{(i)}(y_1^l, u_1^{i-1}|u_i = 0)}{W_l^{(i)}(y_1^l, u_1^{i-1}|u_i = 1)} = \ln \frac{\sum_{u_{i+1}^l} W(y_1|(u_{1,u_i=0}^l F_l)_1) \cdots W(y_l|(u_{1,u_i=0}^l F_l)_l)}{\sum_{u_{i+1}^l} W(y_1|(u_{1,u_i=1}^l F_l)_1) \cdots W(y_l|(u_{1,u_i=1}^l F_l)_l)}, \tag{10}$$

where $u_{1,u_i=1}^l$ represents $(u_1^{i-1}, u_i = 1, u_{i+1}^l \in \{0,1\}^{l-i})$. As the kernel size increases, the complexity of (10) will increase exponentially. Obviously, it is not practical to directly perform SC decoding.

Several studies have focused on reducing the complexity of (10), such as l-formula [12] and W-expressions [15], but these measures can only handle kernels up to 16. To make matters worse, compared with the classic polar codes SC decoding, the computational complexity is still high.

2.3 Motivation

By comparing the decoding process of polar codes and large kernel polar codes, we can observe that although both employ SC decoding, polar codes can greatly reduce the computational complexity using the Tanner graph tool. As for large kernel polar codes, although decoding can be performed according to the code structure shown in Figure 2, the intrinsic connection between the input and output of each kernel, referred to as the kernel Tanner graph (KTG) in this paper, is a black box. Therefore, the LLR of

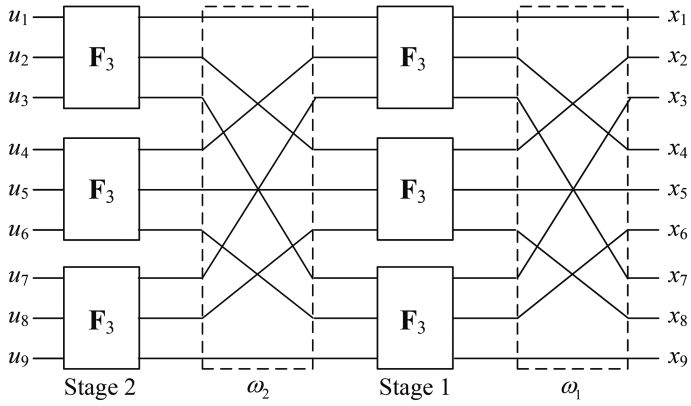

Figure 2 Code structure of the $\boldsymbol{F}_3$-based polar code with $N = 9$.

each kernel needs to be calculated according to (10), which brings about huge computational complexity. If the Tanner graph for a given kernel is available, then the computation of (10) can be replaced. This motivates us to find suitable kernel Tanner graphs for large kernel polar codes, which can efficiently assist in their decoding.

3 TGA-BP decoding for large kernel polar

In this section, we design a highly parallel belief propagation decoding by employing kernel Tanner graph for polar codes with arbitrary dimension linear binary kernel. Moreover, we give the complexity representation of the conceived TGA-BP decoding.

3.1 Kernel Tanner graph design

The Tanner graph of the classic polar codes is essentially another representation of its generator matrix $\boldsymbol{G}_N$, which represents the internal relationship between the input vector u_1^N and the codeword x_1^N. Inspired by this, the polarization kernel matrix is regarded as a generator matrix, and the intrinsic connection between its input and output is explored. In [9], the linear binary kernel with the maximum exponent in different dimensions has been given, take the kernel of size 5 as an example:

$$\boldsymbol{F}_5 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{bmatrix}. \tag{11}$$

The codewords generated by $\boldsymbol{F}_5$ can be expressed as $x_1^5 = u_1^5 \boldsymbol{F}_5$, and each codeword x_i is determined by the element 1 in the i-th column of $\boldsymbol{F}_5$. For example, the elements in the third and the fifth row of the third column are 1, i.e., $F_{3,3} = 1$ and $F_{5,3} = 1$; then x_3 is obtained by the XOR operation of the input bits u_3 and u_5, which is called a check function. More specifically, the parity functions of x_1^5 are represented as

$$\begin{cases} x_1 = u_1 \oplus u_2 \oplus u_3 \oplus u_4 \oplus u_5, \\ x_2 = u_2 \oplus u_5, \\ x_3 = u_3 \oplus u_5, \\ x_4 = u_4, \\ x_5 = u_5. \end{cases} \tag{12}$$

According to (12), codeword x_5 equals input bit u_5. Therefore, u_5 and x_5 are directly connected in the kernel Tanner graph corresponding to $\boldsymbol{F}_5$. Likewise, u_4 and x_4 are also directly connected by a straight line. As for x_3, it is obtained by exclusive or (XOR) u_3 and u_5, forming an XOR circuit structure between

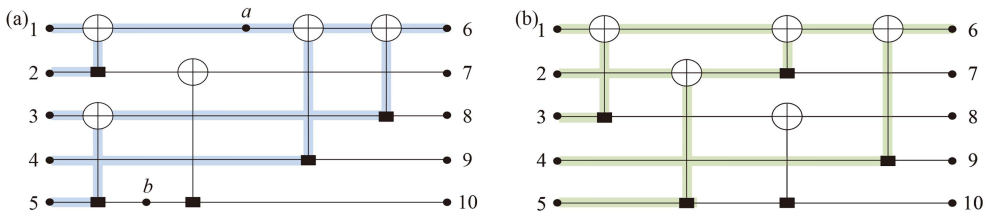

Figure 3 (Color online) Different kernel Tanner graphs corresponding to $\boldsymbol{F}_5$. (a) Kernel Tanner graphs with $x_1 = u_1 \oplus u_2 \oplus u_4 \oplus x_3$; (b) kernel Tanner graphs with $x_1 = u_1 \oplus u_3 \oplus x_2 \oplus u_4$.

u_3, u_5, x_3, and x_5. When comes to x_2, an XOR structure is generated by u_2, u_5, x_2, and x_5. Unlike x_2^5, each codeword can be uniquely determined. The check function of x_1 has multiple representations, such as $x_1 = u_1 \oplus u_2 \oplus u_4 \oplus x_3$ or $x_1 = u_1 \oplus u_3 \oplus x_2 \oplus u_4$. It is certain that these two different check functions for x_1 lead to different kernel Tanner graphs, which are plotted in Figures 3(a) and (b), respectively. Notably, the leftmost nodes 1–5 in the kernel Tanner graph represent input u_1^5, and the rightmost nodes 6–10 represent codewords x_1^5. Selecting a suitable one from the differentiated kernel Tanner graphs will be discussed in Subsection 4.2.

Given any polarization kernel matrix and the same process, one can easily obtain its corresponding kernel Tanner graph based on the check functions. Equipped with the kernel Tanner graph, the code structure is no longer a black box, and a Tanner graph is formed for the large kernel polar code. Note that in this work, the kernel Tanner graph and the Tanner graph are strictly corresponding, that is, given a kernel Tanner graph, by filling it with the black boxes in the code structure, a unique Tanner graph is obtained. Then, large kernel polar codes can be decoded like polar codes by tracking the evolution of LLR values in the Tanner graph. Next, the belief propagation decoding based on the Tanner graph is given for large kernel polar codes.

3.2 Belief propagation decoding

In general, the TGA-BP decoding of large kernel polar codes relies on the iteration of left information L and right information R between adjacent stages in the Tanner graph. Consequently, the crucial to TGA-BP decoding is to calculate the L and R information of the kernel Tanner graph. Specifically, the kernel Tanner graph is divided into s layers from left to right, each layer contains T nodes, where T is the dimension of the kernel matrix. There are a total of $T(s-2)$ nodes in the kernel Tanner graph except for its own input and output. It is worth noting that not all of these $T(s-2)$ nodes are meaningful, only some contain valuable L and R information, which are called key nodes (KN). All key nodes are gathered into a set $\mathcal{Q} = (Q_1, Q_2, \ldots, Q_z)$, where z is the number of KN. When the kernel matrix is obtained, its corresponding kernel Tanner graph and $\mathcal{Q}$ are also determined. What pays more attention is that $\mathcal{Q}$ needs to satisfy the flowing constraints: (i) the L and R information of any Q_i can be expressed by Q_j $(j \neq i)$, as well as the input and output nodes of the kernel Tanner graph; (ii) the number of KN should be as small as possible. Relying on the calculated L and R of $\mathcal{Q}$, one can deduce the left and right information of the kernel Tanner graph based on the kernel $\boldsymbol{F}_2$ processing.

For further explanation, we utilize the $\boldsymbol{F}_5$ kernel Tanner graph in Figure 3(a) to illustrate. In particular, the kernel Tanner graph is divided into 5 layers, with a total of 15 nodes excluding input and output. According to the constraint principles of $\mathcal{Q}$, it requires at least two KN, depicted as a and b in Figure 3(a), to satisfy condition (i). Precisely, the L and R information of node a can be calculated through b and the input and output nodes of kernel Tanner graph according to the left and right information processing of kernel $\boldsymbol{F}_2$, which are expressed as

$$L_a = g(g(L_6, g(R_3, R_5 + L_b) + L_8), R_4 + L_9), \tag{13}$$

$$R_a = g(R_1, R_2 + g(L_7, R_b + L_{10})), \tag{14}$$

where the function g is defined as

$$g(x, y) = \frac{1 + x \times y}{x + y}. \tag{15}$$

In the same manner, the L and R information of node b are computed as

$$L_b = L_{10} + g(R_2 + g(R_1, L_a), L_7), \tag{16}$$

Algorithm 1 TGA-BP for large kernel polar codes.

Require: N, y_1^N, p, N_{iter}, w, $\mathcal{I}$, $\mathcal{Q}$;
Ensure: $\hat{u}_1^N$;
1: **Initial**: $L_1^{(i)} = \text{LLR}(y_i)$, $R_p^{(i)} = 0 (i \in \mathcal{I})$, $R_p^{(i)} = +\infty (i \in \mathcal{I}^{\mathcal{C}})$;
2: **for** $t = 1$ to N_{iter} **do**
3: **for** $s = 1$ to p **do**
4: **if** $\boldsymbol{F}_{ls} = \boldsymbol{F}_2$ **then**
5: Calculate L of all kernels in the s-th stage according to traditional polar;
6: **else**
7: Calculate L of all KTG in the s-th stage based on the obtained L and R of $\mathcal{Q}$;
8: **end if**
9: $L_s^{(i)} = L_s^{w_s^{-1}(i)}, i = 1, 2, \ldots, N$;
10: **end for**
11: **for** $s = p$ to 1 **do**
12: **if** $\boldsymbol{F}_{ls} = \boldsymbol{F}_2$ **then**
13: Compute R of all kernels in the s-th stage according to traditional polar;
14: **else**
15: Compute R of all KTG in the s-th stage based on the obtained L and R of $\mathcal{Q}$;
16: **end if**
17: $R_s^{(i)} = R_s^{w_s(i)}, i = 1, 2, \ldots, N$;
18: **end for**
19: Add $L_p^{(i)}$ and $R_p^{(i)}$ to obtain the LLR value LLR_1^N;
20: **end for**
21: Make a hard decision on the LLR_1^N to obtain the decoding result $\hat{u}_1^N$;

$$R_b = R_5 + g(R_3, L_8 + g(g(R_a, R_4 + L_9), L_6)). \tag{17}$$

Furthermore, after possessing the L and R of nodes a and b obtained by (13)–(17), the L information of the input side nodes of the kernel Tanner graph can be derived as

$$L_1 = g(L_a, R_2 + g(L_7, R_b + L_{10})), \tag{18}$$

$$L_2 = g(L_7, R_b + L_{10}) + g(R_1, L_a), \tag{19}$$

$$L_3 = g(L_8 + g(g(R_a, R_4 + L_9), L_6), R_5 + L_b), \tag{20}$$

$$L_4 = L_9 + g(R_a, g(L_6, g(R_3, R_5 + L_b) + L_8)), \tag{21}$$

$$L_5 = L_b + g(R_3, L_8 + g(g(R_a, R_4 + L_9), L_6)). \tag{22}$$

Similarly, the R information of the nodes on the output side of the kernel Tanner graph is calculated as

$$R_6 = g(g(R_a, R_4 + L_9), g(R_3, R_5 + L_b) + L_8), \tag{23}$$

$$R_7 = g(R_2 + g(R_1, L_a), R_b + L_{10}), \tag{24}$$

$$R_8 = g(R_3, R_5 + L_b) + g(g(R_a, R_4 + L_9), L_6), \tag{25}$$

$$R_9 = R_4 + g(R_a, g(L_6, g(R_3, R_5 + L_b) + L_8)), \tag{26}$$

$$R_{10} = R_b + g(R_2 + g(R_1, L_a), L_7). \tag{27}$$

TGA-BP decoding for large kernel polar codes is described in detail in Algorithm 1. In the Tanner graph, the L and R information of all KN are initialized to 0, the left information $L_1^{(i)}$ in the first stage and the right information $R_p^{(i)}$ in the p-th stage are initialized, where

$$L_1^{(i)} = \text{LLR}(y_i), \tag{28}$$

$$R_p^{(i)} = \begin{cases} 0, \text{ if } i \in \mathcal{I}, \\ +\infty, \text{ if } i \in \mathcal{I}^{\mathcal{C}}. \end{cases} \tag{29}$$

As described in lines 1–20 of Algorithm 1, TGA-BP decoding starts from the first stage of the Tanner graph. According to the L and R information calculation formula of KN, the L information corresponding to the kernel Tanner graph of each stage is calculated sequentially to the left. After reaching the p-th stage, the R information of the kernel Tanner graph corresponding to each stage in the Tanner graph is calculated from left to right. When the number of iterations T is arrived, TGA-BP decoding is accomplished with the estimated decoding sequence $\hat{u}_1^N$.

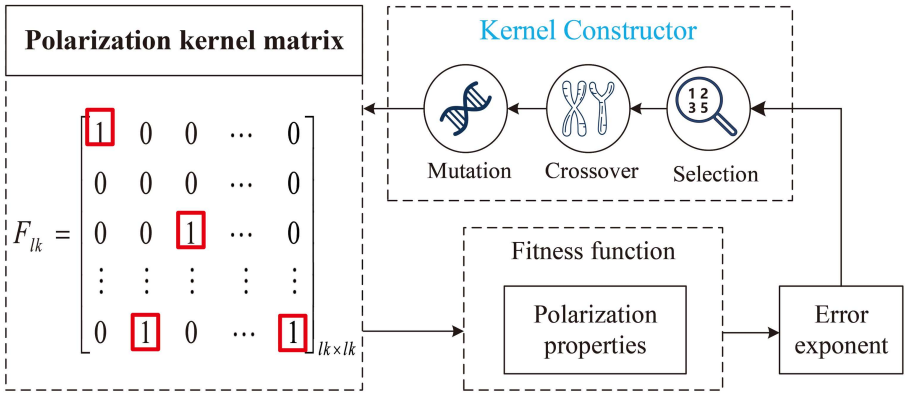

Figure 4 (Color online) Construction architecture of polarization kernel matrix based on genetic algorithm.

Remark 1. For dimension $l = 2$, the kernel matrix with the optimal polarization effect is $\boldsymbol{F}_2 = \left[\begin{smallmatrix}1 & 0\\ 1 & 1\end{smallmatrix}\right]$. Applying the kernel Tanner graph design method in Subsection 3.1 to $\boldsymbol{F}_2$, we can find that there is only one, and its corresponding check functions are $x_1 = u_1 \oplus u_2$ and $x_2 = u_2$. This means that when the kernel size is 2, TGA-BP decoding degenerates into the classic polar belief propagation decoding.

It can be seen from Algorithm 1 that the decoding complexity of TGA-BP is negligible compared to that of polar belief propagation decoding. When the standard Landau notation $O(\cdot)$ is employed for denoting the complexity, a belief propagation (BP) decoder for classic polar can be implemented with complexity $O(TN\log_2 N)$. Algorithm 1 requires $\frac{2EN}{l}$ left and right information calculations at each stage of the Tanner graph, where E is the number of XORs in the kernel Tanner graph. Therefore, assuming that the code structure is composed of the same kernel Tanner graph, the complexity of Algorithm 1 is $O(T\frac{2EN}{l}\log_l N)$.

4 Enhanced TGA-BP decoding

In this section, we propose a two-step scheme to improve TGA-BP decoding for large kernel polar codes. First, we generate a polarization kernel matrix construction architecture, which provides as many candidates of kernel Tanner graph as possible. Subsequently, the most suitable kernel Tanner graph is chosen to assist belief propagation decoding.

4.1 Kernel construction

Differentiated Tanner graphs lead to fluctuations in TGA-BP decoding performance. The factors that affect the Tanner graph under the same kernel size mainly include different kernel matrices and different parity functions designed under the same kernel. Therefore, it is crucial to obtain as many Tanner graph candidates as possible and select the appropriate ones for decoding.

We first designed a kernel matrix construction strategy to obtain as many kernel Tanner graphs as possible. The kernel matrix is expected to have excellent polarization effects, such as smaller scaling exponents or larger error exponents (EE). More importantly, considering the complexity of decoding, under the same kernel size and polarization performance, it demands to have minimum number of 1 in the kernel matrix. Based on the above premise, an offline kernel matrix construction scheme based on genetic algorithm (GA) is illustrated in Figure 4. Specifically, we take the position indices of element 1 in $\boldsymbol{F}_l$ as the target to be optimized by the GA. The selection, crossover, and mutation of the GA are regarded as a kernel constructor, which continuously generates better positions for elements 1 according to the fitness value, as shown by the red mark in Figure 4. In particular, the fitness value is defined as an indicator representing the performance of the kernel, which can be calculated as

$$\mathrm{EE}(\boldsymbol{F}_l) = \frac{1}{l}\sum_{i=0}^{l-1}\log_l(D_i), \tag{30}$$

where D is referred to as the partial distances profile. To sum up, given the kernel size l and the required EE, the construction steps of GA are outlined as follows.

(1) Initialize the kernel population. GA usually starts with a randomly generated population of candidate individuals, where each candidate individual competes with each other and only a few of the fittest survive. Surviving individuals will then undergo evolutionary transitions (i.e., mutation and crossover) to produce offspring representative of the new population. In this work, the initialized population contains S kernel matrix individuals, and the number of elements 1 in all kernels is consistent, assumed to be k. In addition, it is worth noting that the kernel individual in any step is in the standard form [9], which is a lower triangular matrix with all diagonal elements set to 1 and needs to satisfy the polarization conditions.

(2) Generate new kernel population. Calculate the fitness value of the population, and employ the obtained $\mathrm{EE}_k^{(j)}$ $(j = 1, \ldots, S)$ of each individual as the input of the kernel constructor. Then, perform selection, crossover, and mutation operations of the GA to generate a new population. The purpose of this process is to create kernel matrices with better polarization effects under the constraint k.

As an example, we show the process of generating a new matrix by the kernel constructor when the kernel size is 4. For space considerations, the size of the kernel matrix is converted from 4×4 to 1×16. For the kernel individuals $\boldsymbol{F}_4^{(1)} = [1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1]$ with $k = 9$, suppose the individual crossed with it is $\boldsymbol{F}_4^{(2)} = [1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 1]$ and the simplest single midpoint crossover is adopted. Since all kernel matrices are in standard form, only the elements at positions $\{5, 9, 10, 13, 14, 15\}$ in $\boldsymbol{F}_4^{(1)}$ and $\boldsymbol{F}_4^{(2)}$ participate in the crossover operation. Then, the elements located at 13, 14, and 15 in $\boldsymbol{F}_4^{(1)}$ are replaced by the corresponding elements in $\boldsymbol{F}_4^{(2)}$, resulting in the formation of a new kernel $\boldsymbol{F}_4^{(1)} = [1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 1]$. It is worth noting that the number of 1 in $\boldsymbol{F}_3^{(1)}$ is 10 now, and one of the positions $\{5, 9, 10, 13, 14, 15\}$ must be selected to be reversed to 0. Assuming it is 10, thereby completing the crossover operation, the final kernel is $\boldsymbol{F}_4^{(1)} = [1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1]$.

(3) Obtain required kernel matrices. Loop step (2) continuously. Meanwhile, the polarization kernel matrix with the most advanced polarization performance $\mathrm{EE}_k^{(o)}$ is recorded. If $\mathrm{EE}_k^{(o)} < \mathrm{EE}$, go to step (4), otherwise, keep the current k unchanged and record all subsequent kernels with required EE until the maximum number of iterations is reached.

(4) Let $k = k + 1$ and repeat the above process.

$$\boldsymbol{F}_5^{(2)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{bmatrix}. \tag{31}$$

For further explanation, we compared one of the GA-based polarization kernel $\boldsymbol{F}_5^{(1)}$ in (11) with $\boldsymbol{F}_5^{(2)}$ in (31). It can be seen that while they have the same kernel size and error exponent, the kernel Tanner graph obtained by $\boldsymbol{F}_5^{(1)}$ has lower complexity, as shown in Figure 3, where there are 5 XOR structures. In contrast, $\boldsymbol{F}_5^{(2)}$ has at least 6 XOR structures. The consistent phenomenon also exists when coming to other larger kernels, such as kernel $\boldsymbol{F}_{10}^{(1)}$ with the maximum exponent constructed in [9], and

$$\boldsymbol{F}_{10}^{(2)} = [800; C00; A00; 500; 180; D40; 5A0; 350; F28; 5FC] \tag{32}$$

with the same EE is designed by GA. In particular, in (32), in order to save space we divide the rows of the original kernel into 4-bit segments and complement the last segment to 4-bit by 0. Then, convert these segments into corresponding hexadecimal numbers.

Note that after obtaining the minimum k of the kernel matrix corresponding to the required EE, the kernel constructor is still working continuously, with the goal of obtaining multiple different kernel matrices under the k constraint, in other words, enough candidate kernel Tanner graphs.

4.2 Selection of efficient Tanner graph

Of course, the kernel Tanner graph of the same kernel may be different due to different designs of the parity function. All kernel Tanner graphs corresponding to the kernels obtained by GA constitute the candidate set for TGA-BP decoding, and the most effective one needs to be selected. The decoding

performance of a Tanner graph is almost determined by the reliability of each input bit u_i, and the upper bound of the error probability of decoding is defined as

$$P \leqslant \sum_{i \in \mathcal{I}} P_e(u_i), \tag{33}$$

where $\mathcal{I}$ denotes the information bits set, and $P_e(u_i)$ represents the probability of making an incorrect decision on bit u_i. Therefore, one can aim to minimize this upper bound by choosing the information set $\mathcal{I}^R$ to contain the K locations with the most reliable metrics. In other words, $\mathcal{I}^R$ can be regarded as the solution of the optimization problem

$$\mathcal{I}^R = \underset{\mathcal{I} \subset [N]}{\operatorname{argmin}} \sum_{i \in \mathcal{I}} \max(P_e(u_i)). \tag{34}$$

Given the Tanner graph of large kernel polar codes, the reliability value of each bit u_i can be computed by tracking the evolution of the mLLR in the graph. In particular, under the Gaussian assumption, the analytical relationship between the error probability $P_e(u_i)$ and the mLLR m_i of the i-th bit on the input side of the Tanner graph is $P_e(u_i) = Q(\sqrt{m_i/2})$, where Q is defined as

$$Q(x) = \frac{1}{\sqrt{(2\pi)}} \int_x^{+\infty} \mathrm{e}^{-\frac{u^2}{2}} \, \mathrm{d}z. \tag{35}$$

Furthermore, the decoding error probability of the Tanner graph can hence be lower bound by

$$P \geqslant \max_{i \in \mathcal{I}} Q(\sqrt{m_i/2}). \tag{36}$$

The derivative of Q is calculated as

$$Q'(x) = -\frac{1}{\sqrt{(2\pi)}} \mathrm{e}^{-\frac{x^2}{2}}. \tag{37}$$

It can be seen that when $x \in (0, +\infty)$, $Q'(x) < 0$. In other words, when $x > 0$, the Q function is monotonically decreasing. According to (36), in order to obtain a lower bound on the decoding error probability, for $i \in \mathcal{I}$, one needs to maximize the value of $Q(\sqrt{m_i/2})$. Since the value of m_i is always greater than 0, the maximum $Q(\sqrt{m_i/2})$ can be obtained by a small m_i. Therefore, the optimization problem in (34) can be transformed into

$$\mathcal{I}^R = \underset{\mathcal{I} \subset [N]}{\operatorname{argmax}} \sum_{i \in \mathcal{I}} \min(m_i). \tag{38}$$

According to (38), the performance of the Tanner graph is closely related to the mLLR of the input side bits. To this end, the scheme with the maximization of the sum of mLLR of the information bit channels is employed as the selection strategy of the Tanner graph, which is defined as

$$m_j = \underset{j \in M}{\operatorname{argmax}} \sum_{i \in \mathcal{I}(j)} \mathrm{mLLR}_i(j), \tag{39}$$

where $M = \{m_1, m_2, \ldots, m_z\}$ is a set of z candidate Tanner graphs, generated by z kernel Tanner graphs, $\mathcal{I}(j)$ is the information bits positions of the j-th Tanner graph, and $\mathrm{mLLR}_i(j)$ represents the mLLR value of the i-th bit channel of Tannrt graph m_j.

As for the transfer of mLLR in the Tanner graph, the process is similar to the $\boldsymbol{F}_2$-based polar codes. By taking a single XOR operation as a unit, from the output side of the Tanner graph, the mLLR of all nodes is calculated sequentially to the left. Let m_i denote the mLLR on the input side of the Tanner graph, and μ_i denote the output side. As an example, the mLLR evolution equations for $\boldsymbol{F}_5$ in Figure 3(a) are calculated as

$$m_1 = \phi(\phi(\phi(\mu_6, \mu_8), \mu_9), \phi(\mu_7, \mu_{10})), \tag{40}$$

$$m_2 = \phi(\phi(\mu_6, \mu_8), \mu_9) + \phi(\mu_7, \mu_{10}), \tag{41}$$

$$m_3 = \phi(\mu_6 + \mu_8, \mu_7 + \mu_{10}), \tag{42}$$

Algorithm 2 Enhanced TGA-BP decoding.

Require: l, S, k, N_{iter}, EE;
Ensure: $\hat{u}_1^N$;
1: Initialize the kernel population $\boldsymbol{P}$ with S individuals under the restriction of k;
2: **while** EE(o) $\leqslant$ EE **do**
3: **for** $t = 1$ to N_{iter} **do**
4: According to (30), calculate the fitness value EE(j) of each individual in $\boldsymbol{P}$;
5: $\boldsymbol{P}_{\text{new}} \leftarrow$ Selection, crossover, and mutation;
6: **end for**
7: EE(o) $\leftarrow$ The smallest polarization exponent in $\boldsymbol{P}$;
8: $k = k + 1$;
9: Initialize population $\boldsymbol{P}$ under the restriction of k;
10: **end while**
11: Initialize the kernel population $\boldsymbol{P}$ with S individuals under the restriction of k;
12: **for** $t = 1$ to N_{iter} **do**
13: According to (30), calculate the fitness value EE(j) of each individual in $\boldsymbol{P}$;
14: $\boldsymbol{P}_r(i) \leftarrow$ All kernels in $\boldsymbol{P}$ where EE(o) = EE;
15: $\boldsymbol{P}_{\text{new}} \leftarrow$ Selection, crossover, and mutation;
16: **end for**
17: Design all Tanner graphs for each kernel in $\boldsymbol{P}_r(i)$, and the number is $|\boldsymbol{P}_r(i)|$;
18: $M = \sum_{i=1}^{N_{\text{iter}}} |\boldsymbol{P}_r(i)|$;
19: **for** $z = 1$ to M **do**
20: Calculate mLLR(g) ($g = 1, 2, \ldots, N$) of the i-th Tanner graph;
21: $m(i) = \sum_{g \in \mathcal{I}}^{|\mathcal{I}|} \text{mLLR}(g)$;
22: **end for**
23: Take the largest $m(i)$ as the Tannrt graph;
24: $\hat{u}_1^N \leftarrow$ Calling Algorithm 1;

$$m_4 = \phi(\mu_6, \mu_8) + \mu_9, \tag{43}$$

$$m_5 = \mu_6 + \mu_7 + \mu_8 + \mu_{10}, \tag{44}$$

where operation ϕ is marked as

$$\phi(\mu_1, \mu_2, \ldots, \mu_j) = \psi^{-1}\left(1 - \prod_{s=1}^{j}(1 - \psi(\mu_s))\right), \tag{45}$$

$$\psi(\mu) = 1 - \frac{1}{\sqrt{(4\mu\pi)}} \int_{-\infty}^{+\infty} \tanh\frac{z}{2} \mathrm{e}^{-\frac{(z-\mu)^2}{4\mu}} \,\mathrm{d}z. \tag{46}$$

Based on the above discussion, the Tanner graph selection strategy of large kernel polar codes is recapitulated as follows.

(1) The rightmost stages of the z independent Tanner graphs are initialized to the mLLR received from the channel. Note that for BPSK over an AWGN channel, the initial mLLR is given by $\frac{2}{\sigma^2}$, where σ^2 denotes the noise variance.

(2) According to the evolution formula of mLLR on different kernel Tanner graphs, the mLLR of each stage in the m_i ($i = 1, 2, \ldots, z$) Tanner graph is calculated. When the N nodes on the input side of the first stage are completed, we define it as $m_1^N(i)$ ($i = 1, 2, \ldots, z$).

(3) The indexes corresponding to the largest K values in $m_1^N(i)$ form the information set $\mathcal{I}(i)$, where $\mathcal{I}(i) \in [N]$. According to (39), the Tanner graph with the largest sum of bit channel reliabilities is selected and used to assist BP decoding.

Enhanced TGA-BP decoding for large kernel polar codes is described in Algorithm 2. It is worth noting that in practical applications, the determination of the polarization kernel matrix and the final Tanner graph in lines 1–21 of Algorithm 2 can be done offline. For enhanced TGA-BP decoding, compared with the original algorithm, only the kernel Tanner graph is changed. Therefore, after obtaining the optimal kernel Tanner graph, the enhanced TGA-BP can still employ Algorithm 1 for decoding. It should be also noted that the proposed TGA-BP decoder can be easily extended to a more advanced BP list (BPL) by permutation of the Tanner graph, which further enhances the error-correction capability of the proposed decoding scheme.

5 Simulation results

In this section, numerical results are implemented to evaluate the performance of the devised TGA-BP decoder. We assume BPSK transmission with $\{+1, -1\}$ over AWGN channel. All code rates $R = N/K$ are

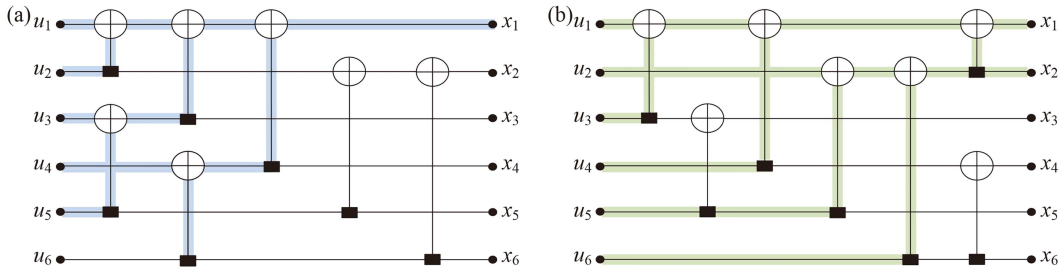

Figure 5 (Color online) Different kernel Tanner graphs corresponding to $\boldsymbol{F}_6$. (a) Kernel Tanner graph 1 with $x_1 = u_1 \oplus u_2 \oplus x_3 \oplus x_4$; (b) kernel Tanner graph 2 with $x_1 = u_1 \oplus u_3 \oplus u_4 \oplus x_2$.

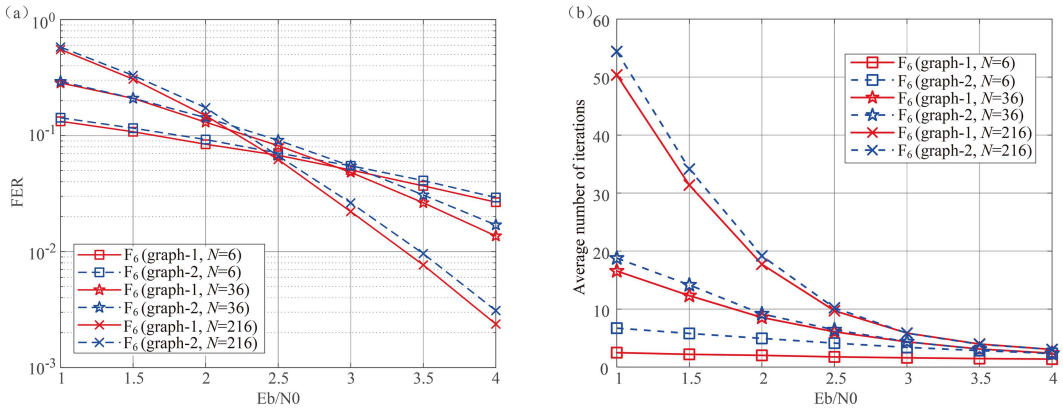

Figure 6 (Color online) FER performance and average decoding iterations of TGA-BP with $\boldsymbol{F}_6$ under different kernel Tanner graphs. (a) FER performance; (b) average decoding iterations.

1/2 and the FER is used as a criterion to measure decoding performance. Specifically, we demonstrate the following aspects: (1) the impact of kernel Tanner graph; (2) comparison of error correction performance with polar codes, large kernel polar codes, and hybrid kernel codes; (3) computational complexity analysis.

5.1 Impact of Tanner graph

We examined the impact of the kernel Tanner graph of the polarization kernel matrix on the FER performance of the TGA-BP decoder. Note that the TGA-BP decoder employs an early termination criterion [21]. Specifically, Figure 5 shows different kernel Tanner graphs corresponding to $\boldsymbol{F}_6$ with error exponent 0.4512, which is defined as

$$\boldsymbol{F}_6 = [80; C0; A0; 90; E8; D4].$$

The check functions of kernel Tanner graph 1 of $\boldsymbol{F}_6$ as plotted in Figure 5(a) are designed as $x_1 = u_1 \oplus u_2 \oplus x_3 \oplus x_4$, $x_2 = u_2 \oplus u_5 \oplus u_6$, $x_3 = u_3 \oplus u_5$, $x_4 = u_4 \oplus u_6$, $x_5 = u_5$, and $x_6 = u_6$. Different from the codeword x_1 of kernel Tanner graph 1, the check equation of kernel Tanner graph 2 as plotted in Figure 5(b) is $x_1 = u_1 \oplus u_3 \oplus u_4 \oplus x_2$. Using the Tanner graph selection strategy of (39), when code length $N = 36$ with $\sigma^2 = 0.5$, the sum of mLLR reliability of Tanner graph 1 is 470.38, while Tanner graph 2 is 469.59. This implies that the performance of graph 1 is better than that of graph 2, and the same for other code lengths. This is also verified in Figure 6(a). We can observe that under different code lengths, the FER of kernel Tanner graph 1 performs better than graph 2.

In addition, the effect of the kernel Tanner graph on the number of iterations of the TGA-BP decoder is shown in Figure 6(b), where the maximum number of iterations is set to 200. The results show that a prominent kernel Tanner graph can not only have outstanding error correction performance, but also significantly reduce the number of decoding iterations, in other words, it has low computational complexity. A consistent phenomenon also exists when coming to other kernels. Therefore, we can draw a conclusion that the enhanced TGA-BP for large kernel polar codes is necessary in Section 4.

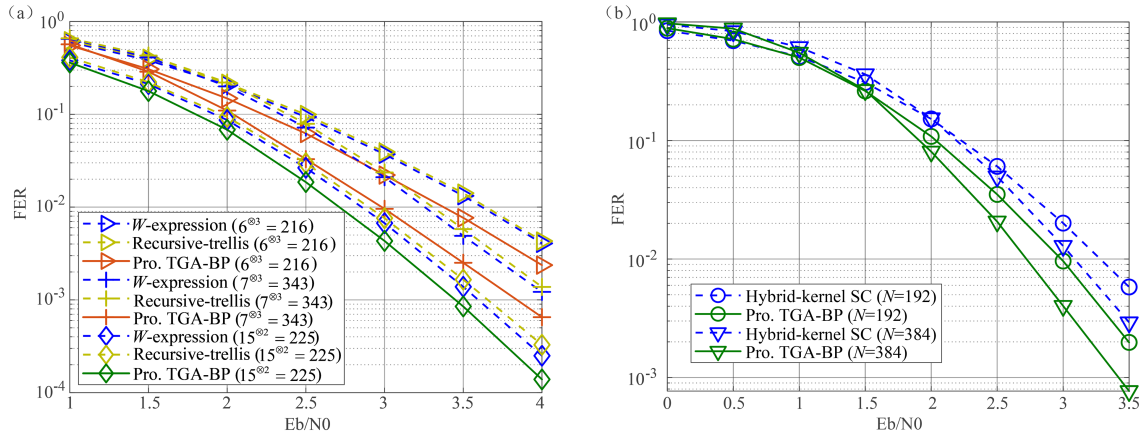

Figure 7 (Color online) Enhanced TGA-BP decoding performance for large kernels. (a) Compared with W-expression and recursive-trellis; (b) compared with hybrid-kernel SC decoding.

5.2 Error-correction performance

Figure 7(a) portrays the FER performance of TGA-BP decoding for large kernel polar codes under different kernel sizes; the recursive-trellis [14] and W-expression [15] methods are employed to serve as a benchmark. For kernel $\boldsymbol{F}_6$, its Tanner graph is given in Figure 5(a). $\boldsymbol{F}_7$ with error exponent 0.4579 obtained by GA is expressed as

$$\boldsymbol{F}_7 = [80; C0; A0; 30; B8; E4; 2E].$$

The check functions of kernel Tanner graph corresponding to $\boldsymbol{F}_7$ are designed as $x_1 = u_1 \oplus u_3 \oplus u_5 \oplus x_2$, $x_2 = u_2 \oplus u_6$, $x_3 = u_3 \oplus x_4 \oplus x_6$, $x_4 = u_4 \oplus u_5$, $x_5 = u_5 \oplus u_7$, $x_6 = u_6 \oplus u_7$, and $x_7 = u_7$.

As can be seen, whether TGA-BP or SC decoding, the error-correction performance of both is almost determined by the exponent of the kernel matrix. Simulations show that the error-correction capability of the TGA-BP exceeds both recursive-trellis and W-expression for large kernel polar codes with different kernel sizes. More specifically, when the kernel size $l = 15$, code length $N = 225$, and FER $= 10^{-3}$, the devised TGA-BP decoding obtains a gain of about 0.3 dB compared with W-expression method. Moreover, since the TGA-BP engages in parallel decoding, it is endowed with high throughput and low latency characteristics to meet 6G URLLC requirements.

Hybrid-kernel codes are a more general form of large kernel polar codes, whose generator matrix can be composed of a mixture of differentiated polarization kernels [22]. Figure 7(b) demonstrates the performance of the TGA-BP decoder with different code lengths for hybrid-kernel polar codes, and the SC decoding is also employed to serve as a benchmark. For the hybrid-kernel code with length $N = 192$, its generator matrix is expressed as $\boldsymbol{G}_N = \boldsymbol{F}_2^{\otimes 6} \otimes \boldsymbol{F}_3$. For the length $N = 384$, the generator matrix $\boldsymbol{G}_N = \boldsymbol{F}_2^{\otimes 7} \otimes \boldsymbol{F}_3$, where $\boldsymbol{F}_3 = [1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1]$. As for the kernel Tanner graph of $\boldsymbol{F}_3$, its check functions are designed as $x_1 = u_1 \oplus u_2$, $x_2 = u_2 \oplus u_3$, and $x_3 = u_3$. Overall, we observe that the FER of TGA-BP outperforms SC decoding for hybrid-kernel polar codes under different code lengths. For example, when the code length $N = 384$ and FER $= 10^{-2}$, the devised TGA-BP decoding obtains a gain of about 0.3 dB compared with SC.

To present further insights, Figure 8(a) plots the error correction performance of the enhanced TGA-BP at different kernel dimensions. As can be seen, the error-correction performance of enhanced TGA-BP for larger kernel polar codes is almost determined by the EE. Specifically, the EE of $\boldsymbol{F}_3$, $\boldsymbol{F}_5$, $\boldsymbol{F}_6$, and $\boldsymbol{F}_7$ is smaller than that of $\boldsymbol{F}_4$, so their performance is worse than that of $\boldsymbol{F}_4$-based polar. For $\boldsymbol{F}_{15}$ and $\boldsymbol{F}_{16}$, although their EE values are slightly greater than $\boldsymbol{F}_4$, they achieve a significant improvement in error correction performance.

Figure 8(b) compares the FER performance of TGA-BPL decoding for large kernel polar codes with classic $\boldsymbol{F}_2$-based polar SC and SCL decoding. In particular, after the kernel matrix of the enhanced TGA-BP is determined, the L best-performing ones are selected as the kernel Tanner graph for TGA-BPL. The simulations confirm that large kernel polar codes with proposed decoding methods achieve significant error-correction performance gains than $\boldsymbol{F}_2$-based polar under both SC and SCL decoding. For example, when the code length $N = 256$, $L = 32$, and FER$= 10^{-3}$, the devised TGA-BPL decoding obtains a gain

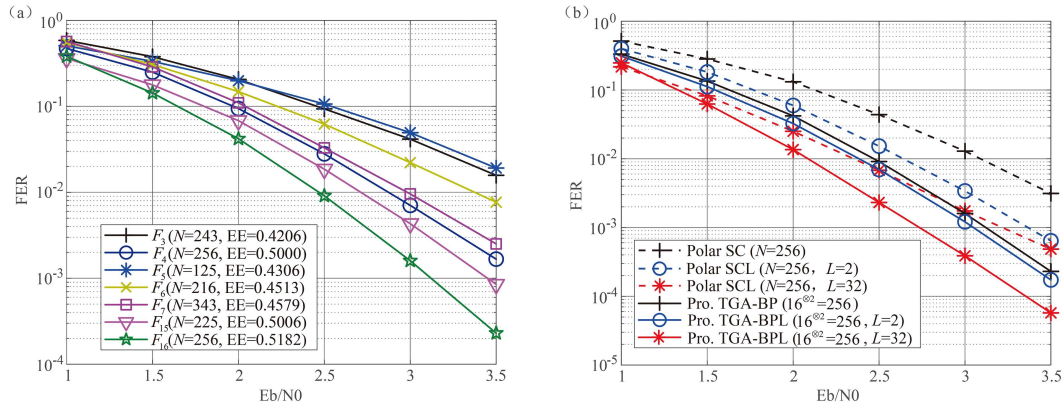

Figure 8 (Color online) Enhanced TGA-BP decoding compared with $\boldsymbol{F}_2$-based polar. (a) FER of enhanced TGA-BP under different kernels; (b) compared with polar SCL decoding.

Table 1 Comparison of the complexity for different decoding schemes.

Classic polar (SC)	Large kernel (straightforward SC)	Large kernel (W-expression)	Proposed (TGA-BP)
$O(N\log_2 N)$	$O(2^l N\log_l N)$	$O(l^2 N\log_l N)$	$O(T\frac{2EN}{l}\log_l N)$

Table 2 Average number of operations for kernel $\mathrm{LLR}_l^{(i)}$.

m	SC	l-formula [12]	W-formula [15]	TGA-BP
2	3.0	2.0	4.0	6.0
3	9.3	3.7	8.7	8.0
4	22.5	2.0	4.0	12.0
5	49.6	7.8	19.2	12.0
6	105.0	11.3	25.3	14.0
7	217.7	21.9	33.7	15.4
8	446.3	42.8	44.8	18.0
9	908.4	52.1	56.7	20.0
10	1841.4	83.8	67.4	20.4
11	3721.8	246.0	118.5	21.8
12	7505.5	673.8	171.0	23.0
13	15121.8	1271.5	240.2	24.9
14	30425.6	3790.6	372.0	25.7
15	61165.1	10736.6	481.3	27.2
16	122878.1	34145.0	630.1	28.5

of about 0.5 dB compared with SCL.

5.3 Complexity analysis

In the TGA-BP decoding complexity analysis in Subsection 3.2, we obtained the expression of the TGA-BP decoding complexity as $O(T\frac{2EN}{l}\log_l N)$. Table 1 shows the complexity comparison of different decoding strategies for classical polar and large kernel polar. It can be found that the proposed TGA-BP can achieve a complexity similar to that of classical polar codes. This is a rough statistic because there are a lot of differentiated mathematical calculations in the calculation of left and right information. Therefore, in this subsection, we explain the decoding complexity of TGA-BP in a more detailed way.

The code structure of large kernel polar codes shows that the complexity of both TGA-BP and SC decoding is proportional to the kernel processing. Therefore, only the decoding computational cost of a single kernel is considered. The number of mathematical operations required to calculate the $\mathrm{LLR}_l^{(i)}$ dominates the computation cost of the kernel processing. For the l-formula [12], it mainly contains two operations, "$\cdot$" and $\Diamond$. A "$\cdot$" includes one multiplication $\times$, and a $\Diamond$ includes one $\times$, one division $\div$, and two additions $+$. Differently, in the W-formula [15], a "$\cdot$" includes two $\times$ and one $+$, and a $\Diamond$ includes four $\times$ and two $+$. As for TGA-BP, it is mainly g function calculation, which contains one $\times$, one $\div$, and two $+$. Because the execution speed of division is lower than multiplication, and both are much

lower than addition, the additions in our complexity analysis are neglected in this work. In addition, we assume one $\div$ equates two $\times$ to simplify the comparison.

The average number of multiplications operations of the kernel $\mathrm{LLR}_l^{(i)}$ by computation straightforward SC, l-formula, W-formula, and TG-BP are recapitulated in Table 2. It can be observed that the SC decoding based on the W-formula achieves a considerable reduction in the number of operations compared with the l-formula, for $l \geqslant 10$. However, both l-formula and W-formula have higher complexity than TGA-BP when $l > 6$. What is more noteworthy is that the larger the kernel dimension, the more significant the advantage of TGA-BP.

6 Conclusion

In this paper, we proposed a Tanner-graph-assisted belief propagation decoding for large kernel polar codes to meet the stringent reliability and latency constraints in 6G URLLC. The Tanner graph and iterative decoding equations of arbitrary-dimensional linear binary polarization kernel matrices are designed. Next, we presented an offline machine learning-based kernel matrix generation architecture, which has the ability to obtain different kernels while ensuring a consistent polarization effect. Benefiting from the above-mentioned strategies, our scheme achieves significant complexity reduction and error-correction enhancement compared to large kernel polar codes under SC decoding.

References

1 Wang T, Qu D, Jiang T. Parity-check-concatenated polar codes. IEEE Commun Lett, 2016, 20: 2342–2345
2 Wu Y Z, Li L, Fan P Z. A fast parallel SC-Fano decoding algorithm for PAC codes. Sci China Inf Sci, 2023, 66: 152301
3 Jiang T, Liu Y, Xiao L, et al. PCC polar codes for future wireless communications: potential applications and design guidelines. IEEE Wireless Commun, 2024, 31: 414–420
4 Arikan E. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. IEEE Trans Inform Theor, 2009, 55: 3051–3073
5 Ashikhmin A, Trifonov P. Fast successive cancellation decoding of polar codes with large kernels. IEEE Trans Commun, 2025, 73: 3–11
6 Korada S B, Sasoglu E, Urbanke R. Polar codes: characterization of exponent, bounds, and constructions. IEEE Trans Inform Theor, 2010, 56: 6253–6264
7 Presman N, Shapira O, Litsyn S, et al. Binary polarization kernels from code decompositions. IEEE Trans Inform Theor, 2015, 61: 2227–2239
8 Mori R, Tanaka T. Source and channel polarization over finite fields and Reed-Solomon matrices. IEEE Trans Inform Theor, 2014, 60: 2720–2736
9 Lin H P, Lin S, Abdel-Ghaffar K A S. Linear and nonlinear binary kernels of polar codes of small dimensions with maximum exponents. IEEE Trans Inform Theor, 2015, 61: 5253–5270
10 Feng Z, Niu C, Zhang Z, et al. List-serial pipelined hardware architecture for SCL decoding of polar codes. China Commun, 2023, 20: 175–184
11 Bonik G, Goreinov S, Zamarashkin N. Construction and analysis of polar and concatenated polar codes: practical approach. 2012. ArXiv:1207.4343
12 Huang Z, Zhang S, Zhang F, et al. On the successive cancellation decoding of polar codes with arbitrary linear binary kernels. 2017. ArXiv:1701.03264
13 Trofimiuk G, Trifonov P. Window processing of binary polarization kernels. IEEE Trans Commun, 2021, 69: 4294–4305
14 Trifonov P, Karakchieva L. Recursive processing algorithm for low complexity decoding of polar codes with large kernels. IEEE Trans Commun, 2023, 71: 5039–5050
15 Huang Z, Jiang Z, Zhou S, et al. On the non-approximate successive cancellation decoding of binary polar codes with medium kernels. IEEE Access, 2023, 11: 87505–87519
16 Liu Y, Xiao L, Liu W, et al. Channel coding for satellite-terrestrial integrated communication: classic applications, key technologies, and challenges. IEEE Wireless Commun, 2024, 31: 348–354
17 Hu Y Y, Pan Z W, Guan Y L. Asynchronous multilevel bit-interleaved polar-coded modulation. Sci China Inf Sci, 2023, 66: 132304
18 Pang Q, Ma Z, Tang X. Unreliability normalization weighted bit-flipping algorithms of LDPC decoding for ReRAM systems. Sci China Inf Sci, 2024, 67: 1–2
19 Yu Y R, Pan Z W, Tan X S, et al. A latency-reduced successive cancellation list decoder for polar codes. Sci China Inf Sci, 2019, 62: 029302
20 Tal I, Vardy A. How to construct polar codes. IEEE Trans Inform Theor, 2013, 59: 6562–6582
21 Yuan B, Parhi K K. Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders. IEEE Trans Signal Process, 2014, 62: 6496–6506
22 Zhao Y, Yin Z, Yang Z, et al. Generalized constituent nodes for simplified successive-cancellation decoding of multi-kernel polar codes. IEEE Commun Lett, 2023, 27: 1272–1276