Syndrome Check aided Fast-SSCANL Decoding Algorithm for Polar Codes

Choangyang Liu[1], Wenjie Dai[1], and Rui Guo[1*]
[1] Communication College, Hangzhou Dianzi University
Hangzhou, Zhejiang 310018 China
[e-mail : 1697792272@qq.com, dai1213@163.com, guorui@hdu.edu.cn]
[*]Corresponding author: Rui Guo

Received November 6, 2023; revised March 21, 2024; accepted April 20, 2024; published May 31, 2024

Abstract

The soft cancellation list (SCANL) decoding algorithm for polar codes runs L soft cancellation (SCAN) decoders with different decoding factor graphs. Although it can achieve better decoding performance than SCAN algorithm, it has high latency. In this paper, a fast simplified SCANL (Fast-SSCANL) algorithm that runs L independent Fast-SSCAN decoders is proposed. In Fast-SSCANL decoder, special nodes in each factor graph is identified, and corresponding low-latency decoding approaches for each special node is propose first. Then, syndrome check aided Fast-SSCANL (SC-Fast-SSCANL) algorithm is further put forward. The ordinary nodes satisfied the syndrome check will execute hard decision directly without traversing the factor graph, thereby reducing the decoding latency further. Simulation results show that Fast-SSCANL and SC-Fast-SSCANL algorithms can achieve the same BER performance as the SCANL algorithm with lower latency. Fast-SSCANL algorithm can reduce latency by more than 83% compared with SCANL, and SC-Fast-SSCANL algorithm can reduce more than 85% latency compared with SCANL regardless of code length and code rate.

Keywords: Polar codes, SCAN decoding algorithm, SCANL decoding algorithm, Special nodes, Syndrome check.

1. Introduction

Arikan first proposed polar codes and proved that these codes can reach the channel using successive cancellation (SC) decoding in a binary memoryless channel as the code length tends to infinity [1]. However, SC algorithm gets poor decoding performance for finite-length polar codes, and has high decoding latency due to its serial characteristic.

Many improved algorithms have been proposed to reduce the decoding latency of the SC, such as [2][3][4][5]. A kind of fast SC decoding algorithm was proposed in [2] for the first time. It presented three special nodes: single parity check (SPC) node, repetition (REP) node, and RS (REP-SPC) node. Different fast decoding methods were used at these special nodes to reduce the decoding latency. In [3], the author extended the fast SC decoding algorithm, and proposed five special type nodes corresponding to different frozen bit sequences. Based on the five special nodes proposed in [3], fast parallel list decoders were proposed in [4], which reduced the decoding latency significantly. Unlike the algorithms in [2][3][4], a pruning method for SC decoding algorithm was proposed in [5]. The proposed method can achieve lower decoding latency without affecting the bit-error-rate (BER) performance by pruning the unnecessary subtrees if the syndrome check is satisfied.

As SC decoding is hard-decision decoding algorithm, a soft cancellation (SCAN) decoding algorithm based on SC decoding was proposed in [6] for the first time. The decoding performance of SCAN decoding algorithm is better than that of the SC decoding algorithm. However, the serial characteristic of SCAN decoding algorithm leads to high latency. To reduce the latency, a fast and low-latency decoding algorithm based on SCAN was proposed in [7]. The algorithm adopted different low latency methods for Rate-0 and Rate-1 nodes. In [8], the author proposed an improved SCAN decoding to reduce decoding latency further. It used the REP and SPC nodes to simplify the traversal of the decoding tree. Later, a fast SCAN decoding algorithm was proposed in [9] to further reduce latency. The fast SCAN algorithm identified five special nodes in the decoding tree and proposed low-latency decoding methods for each type. Based on the SCAN algorithm, some SCAN bit flip algorithms were proposed. Wang proposed (SCAN-BF) decoding algorithm to reduce errors caused by error propagation in [11]. Zhang propose Fast-SCANF decoder to accelerate the decoding speed using fast processing mechanism in [12]. Meanwhile, soft cancellation list (SCANL) decoding algorithm was proposed in [10] to improve the decoding performance of SCAN decoding. It ran multiple SCAN decoding processes with different factor graphs. Although it gets better performance, it has higher latency. The author proposed a kind of generalized SCAN Bit-Flipping decoding algorithm in [13], it improved the SCAN-BF decoding algorithm by correcting prior information of polar codes. The author proposed a soft-output list (SOL) decoder, which considers both hypotheses of 0 and 1 for unreliable bits and keeps the a-priori likelihoods for reliable bits, and can provide better performance than the SCANL algorithm with same list size [14].

In this paper, fast simplified SCANL (Fast-SSCANL) decoding algorithm based on the SCANL decoder is proposed first. Several special nodes are used in Fast-SSCANL, and the corresponding low latency decoding methods are designed. Then, a syndrome check aided Fast-SSCANL (SC-Fast-SSCANL) decoding algorithm is proposed by extending the Fast-SSCANL algorithm further. The proposed SC-Fast-SSCANL algorithm can stop traversing the factor graph when the ordinary nodes satisfy the syndrome check.

The rest of this paper is organized as follows. Section 2 provides the general descriptions of SCAN and SCANL decoding algorithms for polar codes. Fast-SSCANL and SC-Fast-SSCANL decoding algorithms are proposed in Section 3. Section 4 analyzes the decoding

performance and latency of the proposed algorithms. Finally, Section 5 concludes the paper.

2. Preliminaries of Polar Encoding and Decoding

Polar code is a kind of linear block code based on channel polarization [1]. For (N, K) polar code, we generally select K subchannels with the highest channel reliability from N subchannels to transmit information. The bits transmitted on these subchannels are called information bits, and those transmitted on the remaining subchannels are called frozen bits, which are often set to zero. F and F_c represent the frozen and information bit index sets respectively. In practice, source message, denoted as $\boldsymbol{u}=\{u_0, u_1, ..., u_{N-1}\}$, is a vector with its elements index coming from F and F_c. Then, the encoded message x can be calculated as $x=\boldsymbol{G}_N \times \boldsymbol{u}$, where $\boldsymbol{G}_N$ is the generator matrix, which is calculated by $\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}^{\otimes \log_2 N}$. At the receiver, the decoder generates an estimate of the source message $\boldsymbol{u}$ based on the received message.

2.1 SCAN Decoding Algorithm

The SCAN decoding algorithm ran soft information iterative transfer decoding based on the factor graph. For polar code with code length N, its factor graph consists of $n(n=\log_2 N)$ stages as shown in **Fig. 1**. Soft information is divided into left information λ_i and right information β_i. The left information λ_i^0 on the right side of the factor graph is initialized as the received information $y=\{LLR_0, ..., LLR_{N-1}\}$. Meanwhile, on the left side of the factor graph, the right information β_i^n is initialized to ∞ if $i \in F$, and the other right information is initialized to 0.

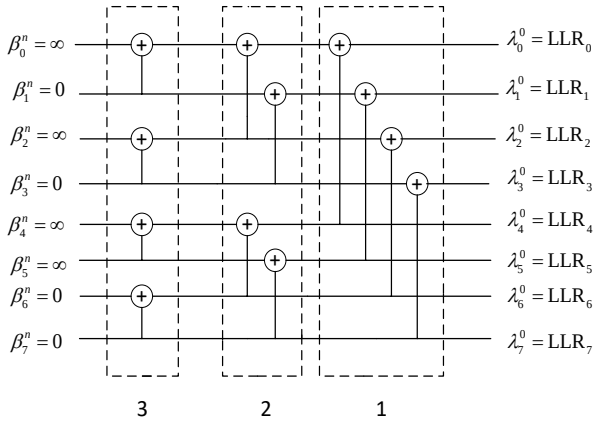


Fig. 1. SCAN decoding factor graph with code length $N=8$.

Soft information is transferred in both directions through the factor graph. Hence, the update rule can be described from **Fig. 2** as:

$$\begin{aligned} \lambda_a &= f\left(\lambda_b, \lambda_d + \beta_c\right) \quad \lambda_c = f\left(\lambda_b, \beta_a\right) + \lambda_d \\ \beta_b &= f\left(\beta_a, \lambda_d + \beta_c\right) \quad \beta_d = f\left(\lambda_b, \beta_a\right) + \beta_c \end{aligned} \tag{1}$$

Where $f(x, y) = \text{sign}(x) \times \text{sign}(y) \times \min(|x|, |y|)$.

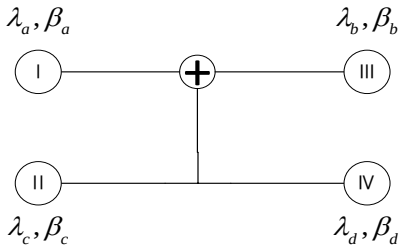


Fig. 2. Unit factor graph of SCAN algorithm.

2.2 SCANL Decoding Algorithm

The SCANL decoding algorithm is an improvement of the SCAN algorithm. By permuting the n stages of the factor graph, $n!$ different equivalent factor graphs can be obtained, where each factor graph is defined by the permutation π_i. Hence, the factor graph shown in **Fig. 1** is defined by the permutation $\pi_0 = \{3, 2, 1\}$. For example, the permutation $\pi_1 = \{1, 3, 2\}$ can be obtained by permuting π_0. Then, the equivalent factor graph of **Fig. 1** can be generated according to the permutation π_1, as shown in **Fig. 3**.

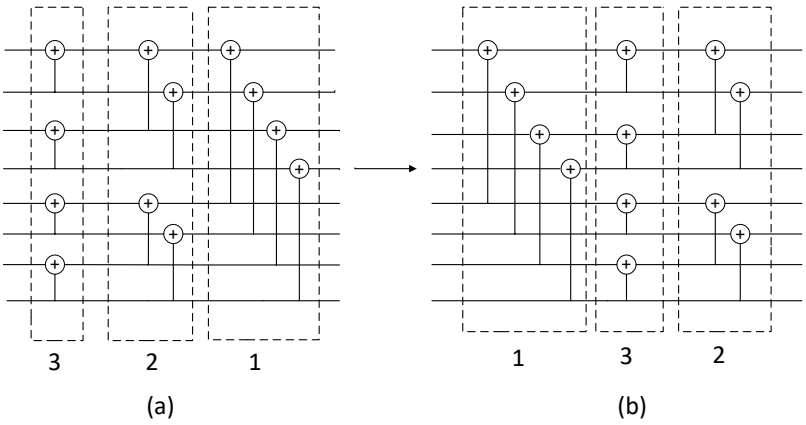


Fig. 3. Equivalent factor graph permutation process from π_0 to π_1.

According to different permutations, the Hamming distance between permutations is calculated by:

$$HD\left(\pi_a, \pi_b\right) = n + 1 - \sum_{s=0}^{n} \delta_{\pi_a(s), \pi_b(s)} \tag{2}$$

where δ is the Kronecker delta function.

Based on the Hamming distance between permutations L permutations $\prod(\pi)=\{\pi_0,\pi_1,...,\pi_{L-1}\}$ with larger Hamming distance are selected, and the corresponding L equivalent factor graphs are generated. After receiving the information, the SCANL decoding algorithm runs L independent SCAN decoders. Each decoder relies on a different equivalent factor graph, and returns path metric PM. The path metric PM update rule is described as follows:

$$PM = PM - \lambda_i, i \in F \tag{3}$$

where λ_i represents the left information returned from the frozen bit node. Then, the decoding bit sequence with the lowest path metric PM_{min} is selected as the output of SCANL decoder.

3. Proposed Decoding Algorithm

In this section, we first propose the Fast-SSCANL decoding algorithm. Then, as an improvement of Fast-SSCANL algorithm, the SC-Fast-SSCANL decoding algorithm is proposed.

3.1 Fast-SSCANL Decoding

Since factor graph permutation is difficult to implement in practice, we hope to find an implementation-friendly permutation method that is equivalent to factor graph permutation. First, according to the codeword position in the factor graph in **Fig. 3** (a), the codeword position in the factor graph (b) is permuted as shown in **Fig. 4**. Note that the i-th component of the bit sequence $\hat{u}_i$ is connected to the i-th component of the information sequence LLRs by associating a "$\oplus$" to 0 and a "$\bullet$" to 1. Then, the binary expansion of the integer i is used to represent the corresponding order of those "$\oplus$" and "$\bullet$" operations. By permuting the stages of the factor graph, we could simply permute the order of those "$\oplus$" and "$\bullet$". Therefore, applying permutation to the binary expansion of i can be equivalent to applying the same permutation to the stages of the factor graph.

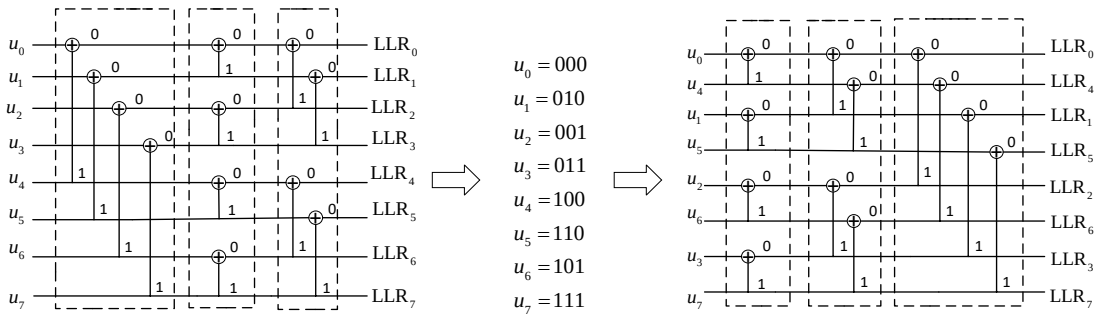


Fig. 4. Mapping process from factor graph permutation to the codeword position permutation.

According to **Fig. 4**, there is a one-to-one mapping relationship between the permutation on the factor graph stage and the permutation on the codeword position. The relationship allows us to use the same factor graph by permuting the received information, thus being more implementation-friendly.

So, the received information $y=\{LLR_0,...,LLR_{N-1}\}$ is permuted according to the permutation set $\pi=\{\pi_0,...,\pi_{L-1}\}$ at the initialization stage before decoding. Then, L different permutations of received information $Y=\{y_0,...,y_{L-1}\}$ are obtained. Fast-SSCANL algorithm runs L independent Fast-SSCAN decoders, while Y is used to initialize all the decoders.

Fast-SSCAN decoder can provide the same soft output as SCAN, while reducing latency by adopting several special nodes. The simplified thought in Fast-SSCAN is done by rate-0 and rate-1 special nodes. A rate-0 node always returns $\beta_i^t=\{\infty,...,\infty\}$, conversely, a rate-1 node returns $\beta_i^t=\{0,...,0\}$. Here we extend the special node to RI node and FS node.

A. REP node

In **Fig. 5**, the factor graph can be represented as a decoding tree. The REP node occurs when all leaf nodes of a node are frozen bit nodes except the rightmost one.

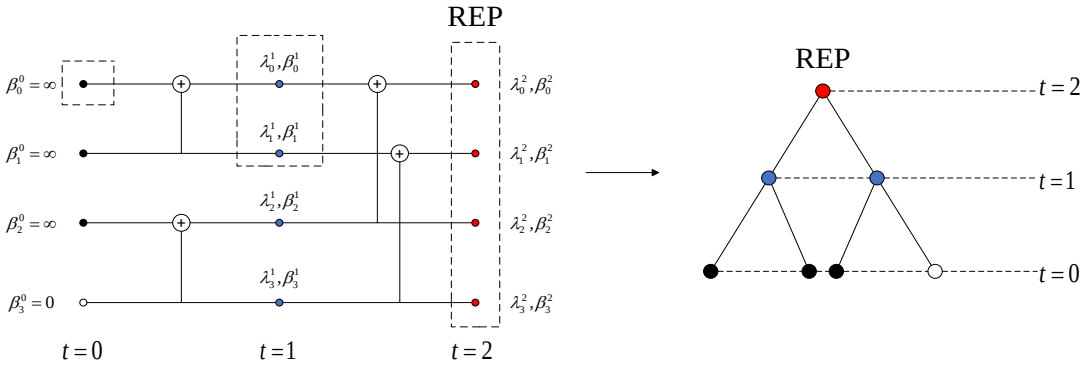


Fig. 5. Decoding factor graph for the REP node.

The right information is passed from stage $t=0$ to stage $t=log_2 N$ iteratively, and the left information is passed from stage $t=log_2 N$ to stage $t=0$ iteratively. By this process, the right information β_i^t returned from the REP node can be calculated directly from the left information λ_j^t without traversing the factor graph. It can be described as follows:

$$\beta_i^t=\sum_{j=0}^{2^t-1}\lambda_j^t-\lambda_i^t,0\le i\le 2^t-1 \tag{4}$$

Where $2\le t\le log_2 N$.

B. SPC node

In an SPC node, the first leaf node is frozen bit node while all the other leaf nodes represent information bit nodes, as shown in **Fig. 6**.

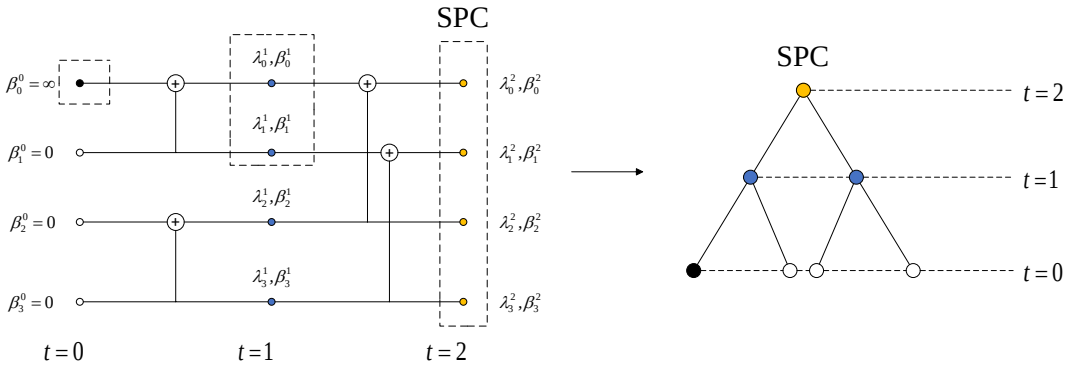


Fig. 6. Decoding factor graph for the SPC node.

Similar to the REP node, the right information β_i^t returned from the SPC node can also be obtained without traversing the factor graph. The calculation process can be described as follows:

$$\beta_i^t = \min_{\substack{j\neq i \\ 0\le j\le 2^t-1}} \left(|\lambda_j^t|\right) \prod_{j=0,j\neq i}^{2^t-1} \operatorname{sign}\left(\lambda_j^t\right), 0\le i\le 2^t-1 \tag{5}$$

Where $2\le t\le log_2 N$. It should be noted that in (5), $\prod_{j=0,j\neq i}^{2^t-1} \operatorname{sign}\left(\lambda_j^t\right)$ still needs a large number of multiplication operations, so in order to further reduce the amount of computation, (5) can be rewritten as:

$$\beta_i^t = (-1)^{P\oplus h_i} * \min_{\substack{j\neq i \\ 0\le j\le 2^t-1}} \left(|\lambda_j^t|\right) \tag{6}$$

where h_i is the hard decision taken on λ_i^t, and P is the overall parity:

$$P = \bigoplus_{i=0}^{2^t-1} h_i \tag{7}$$

In order to further reduce the latency, we expand the above four special nodes (rate-0, rate-1, REP, SPC) to get the following new special nodes, namely RI node and FS node, and design the corresponding low-latency decoding method.

C. RI node

The RI node merges an REP node and rate-1 node to reduce the latency, as shown in **Fig. 7**, where rate-1 node is represented by white node.

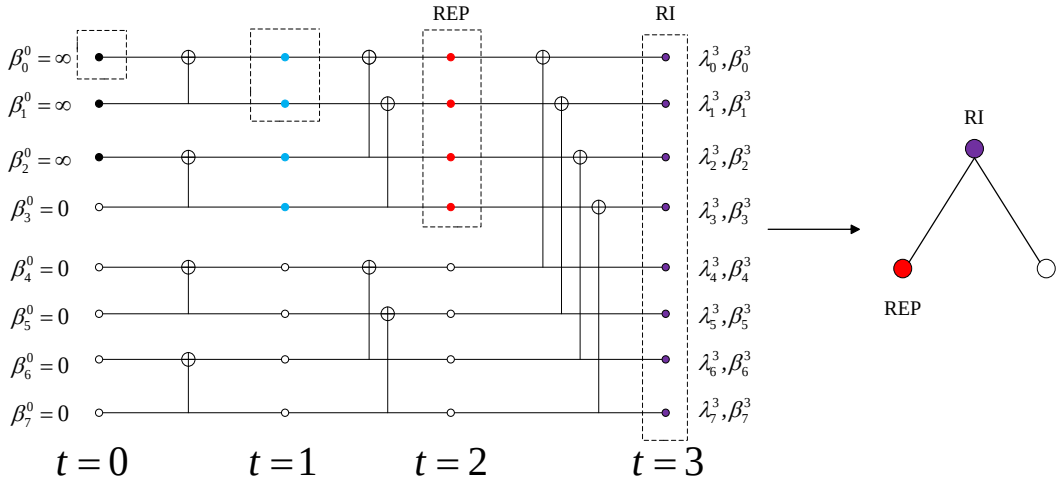


Fig. 7. Decoding factor graph for the RI node.

The right information returned from the RI node can still be calculated from the left information without traversing the factor graph. By combining (4) and the computations related to the right information in (1), the following equation can be obtained:

$$\beta_i^t = \begin{cases} f\left(\sum_{j=0,j\neq i}^{2^{t-1}-1} f\left(\lambda_j^t, \lambda_{j+2^{t-1}}^t\right), \lambda_{i+2^{t-1}}^t\right), 0 \le i \le 2^{t-1}-1 \\ f\left(\sum_{k=0,k\neq i-2^{t-1}}^{2^{t-1}-1} f\left(\lambda_k^t, \lambda_{k+2^{t-1}}^t\right), \lambda_{i-2^{t-1}}^t\right), 2^{t-1} \le i \le 2^t-1 \end{cases} \tag{8}$$

Where $3 \le t \le log_2 N$.

D. FS node

Similar to the RI node, the FS node merges an SPC node and rate-0 node, as shown in **Fig. 8**, where rate-0 node is represented by black node.

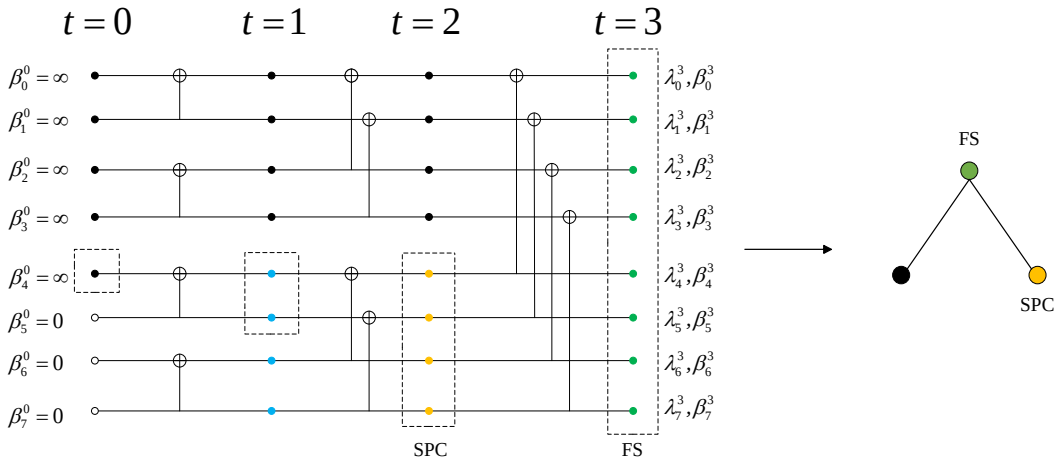


Fig. 8. Decoding factor graph for the FS node.

By combining (6) and the computations related to the right information in (1), the following equation can be obtained:

$$\beta_i^t = \begin{cases} (-1)^{P \oplus h_{i+2^{t-1}}} * \min\limits_{\substack{j \neq i \\ 0 \le j \le 2^{t-1}-1}} \left(| \lambda_j^t + \lambda_{j+2^{t-1}}^t | \right) + \lambda_{i+2^{t-1}}^t, 0 \le i \le 2^{t-1}-1 \\ (-1)^{P \oplus h_i} * \min\limits_{\substack{k \neq i \\ 2^{t-1} \le k \le 2^t-1}} \left(| \lambda_k^t + \lambda_{k-2^{t-1}}^t | \right) + \lambda_{i-2^{t-1}}^t, 2^{t-1} \le i \le 2^t-1 \end{cases} \tag{9}$$

where $h_i = \begin{cases} 0, \left(\lambda_i^t + \lambda_{i-2^{t-1}}^t \right) \ge 0 \\ 1, else \end{cases}$, $p = \bigoplus\limits_{i=2^{t-1}}^{2^t-1} h_i$, $3 \le t \le log_2 N$.

3.2 SC-Fast-SSCANL Decoding

The latency reduction of the proposed Fast-SSCANL algorithm is based on the above six kinds of special nodes, but not all factor graphs are composed of special nodes. For non-special nodes named ordinary nodes, a simple way is also deeded to calculate the right information instead of updating all soft information by using formula (1). SC-Fast-SSCANL algorithm based on syndrome check is proposed, SC-Fast-SSCANL runs L independent SC-Fast-SSCAN decoders. The syndrome check is proposed for ordinary nodes in SC-Fast-SSCAN decoder. According to the check message, the ordinary nodes that satisfying the syndrome check will be decoded by a low latency decoding method, thereby reducing the decoding latency further. The left information returned from the ordinary node can be easily converted to the estimated right information. A check matrix can be used to check if these estimated right information values can represent a potential transmitted codeword or not. The syndrome check is descripted in Algorithm 1 in detail.

Algorithm 1 Syndrome check

input: current factor graph stage t of the ordinary node, left information λ^t, frozen bit index set F

output: check message *flag*

function syndrome check $\left(t, \lambda^t, F\right)$:

flag $\leftarrow 0$
FOR $i = 0 : 2^t - 1$
    $\beta^t[i] \leftarrow h\left(\lambda^t[i]\right)$
END FOR
$\hat{u}_i = \left\{u_0, \ldots, u_{2^t-1}\right\} \leftarrow \beta^t * \boldsymbol{G}_t$
IF $\hat{u}_{i \in F} = \hat{u}_i * \boldsymbol{G}_t * \boldsymbol{H}^T$
    flag $\leftarrow 1$
    FOR $i = 0 : 2^t - 1$
        IF $i \notin F$

```
10              IF  û_i = 1
11                  Continue;
12              ELSE
13                  flag ← 0
14                  Break;
15              END IF
16          ELSE
17              Continue;
18          END IF
19      END FOR
20  ELSE
21      flag ← 0
22  END IF
23  Return flag
```

Where the h function in line 4 of Algorithm 1 can be described as $h(x)=\begin{cases}0, x\geq 0\\ 1, else\end{cases}$, and there are also three important parameters in lines 5 and 6 such as $\boldsymbol{G}_t$ and $\boldsymbol{H}$. $\boldsymbol{G}_t$ can be calculated as $\boldsymbol{G}_t=\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}^{\otimes t}$. $\boldsymbol{H}$ is the syndrome check matrix in which the i-th row consists of the i-th column of $\boldsymbol{G}_t$, where $i\in F$.

The detailed description of the SC-Fast-SSCANL decoding algorithm is shown in Algorithm 2.

Algorithm 2 SC-Fast-SSCANL decoding

input: channel received information y, permutation set π, frozen bit index set F, number of iterations T

output: decoding estimated bit vector $\hat{u}$

function SC-Fast-SSCANL (y,π,F,T):

```
1   PM ← ∞
2   FOR i = 0 : L − 1
3       PM_i, û[i] ← SC-Fast-SSCAN(y, π, F, T)
4       IF PM_i < PM
5           PM ← PM_i
6           û ← π^{-1}(û[i])
7       END IF
8   END FOR
```

9 **Return** $\hat{u}$

After receiving the check message, if $flag$ is equal to 1, the syndrome check is successful. The right information returned from the ordinary nodes is directly obtained by the hard decision of the left information, without traversing the factor graph. The process can be described as follows:

$$\beta_i^t = \begin{cases} +\infty, \lambda_i^t \geq 0 \\ 0, else \end{cases} \tag{10}$$

Where $0 \leq i \leq 2^t - 1$. If $flag$ is equal to 0, the syndrome check fails. The ordinary nodes still run SCAN decoding. The sub-function SC-Fast-SSCAN is a part of SC-Fast-SSCANL decoding algorithm. And it is shown in Algorithm 3.

Algorithm 3 SC-Fast-SSCAN

input: channel received information y, permutation set π, frozen bit index set F, number of iterations T

output: estimated bit vector $\hat{u}$, path metric PM

function SC-Fast-SSCAN (y, π, F, T):

1 $PM \leftarrow 0$
2 $\lambda_i^0 \leftarrow \pi(y)$
3 **FOR** $i = 0 : N-1$
4 **IF** $\pi_l[i] \in F$
5 $\beta_i^n \leftarrow \infty$
6 **ELSE**
7 $\beta_i^n \leftarrow 0$
8 **END IF**
9 **END FOR**
10 **FOR** $t = 0 : T-1$
11 **FOR** $i = 0 : N-1$
12 right to left propagation until λ_i^n, and identify node types
13 **IF** special nodes
14 stop traversing the factor graph, use (4)-(9)
15 **ELSE**
16 $flag \leftarrow$ syndrome check(t, λ^t, F)
17 **IF** $flag = 1$
18 stop traversing the factor graph, and use (10)
19 **END IF**

END IF

IF $\pi_l[i] \in F$

$\hat{u}_i \leftarrow 0$

$PM_l \leftarrow PM_l - \lambda_i$

ELSE

$\hat{u}_i \leftarrow h\left(\lambda_i^n + \beta_i^n\right)$

END IF

left to right propagation with β_i^n

END FOR

$\hat{u} \leftarrow \pi^{-1}(\hat{u})$

END FOR

Return $\hat{u}$, PM

In summary, different type nodes are proposed in SC-Fast-SSCAN decoder. For several special nodes, the subsequent factor graph will not be traversed, thus reducing the latency. For the ordinary nodes that satisfy the syndrome check, the nodes will execute a hard decision directly without traversing the subsequent factor graph to reduce latency. The process of SC-Fast-SSCAN decoder is shown in **Fig. 9**.

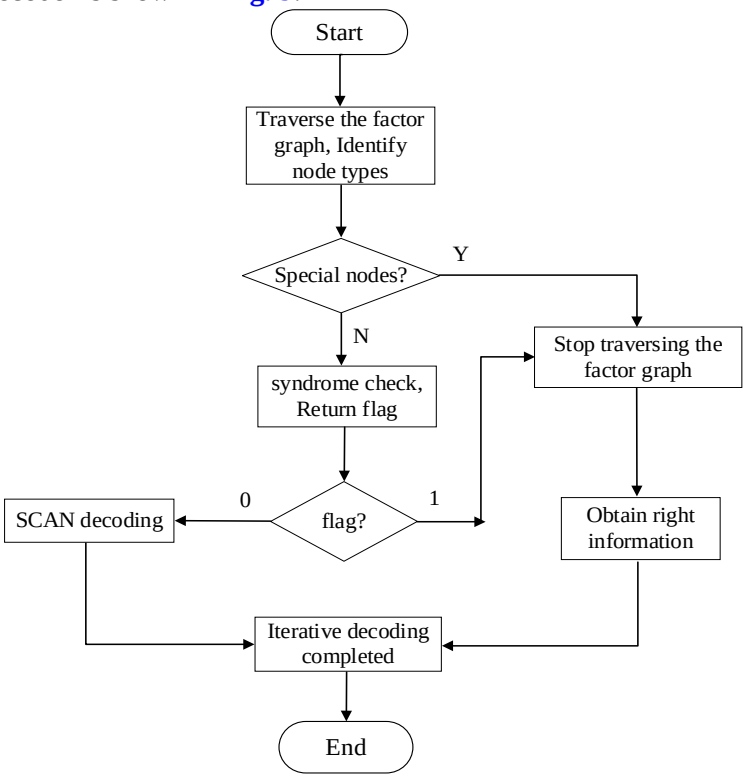
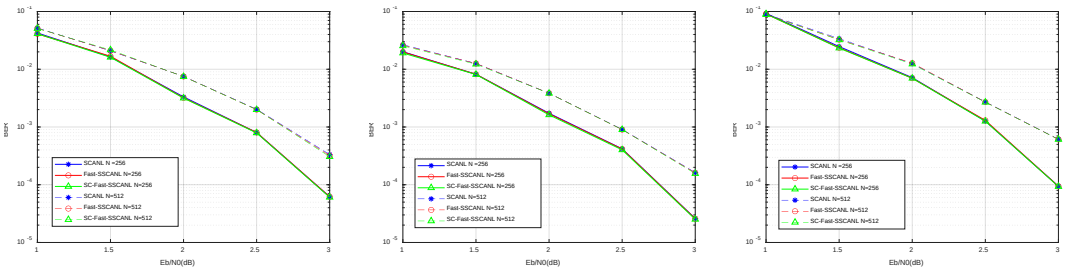


Fig. 9. Flow chart of SC-Fast-SSCAN decoder.

4. Simulation Results and Performance Analysis

4.1 Decoding Performance Analysis

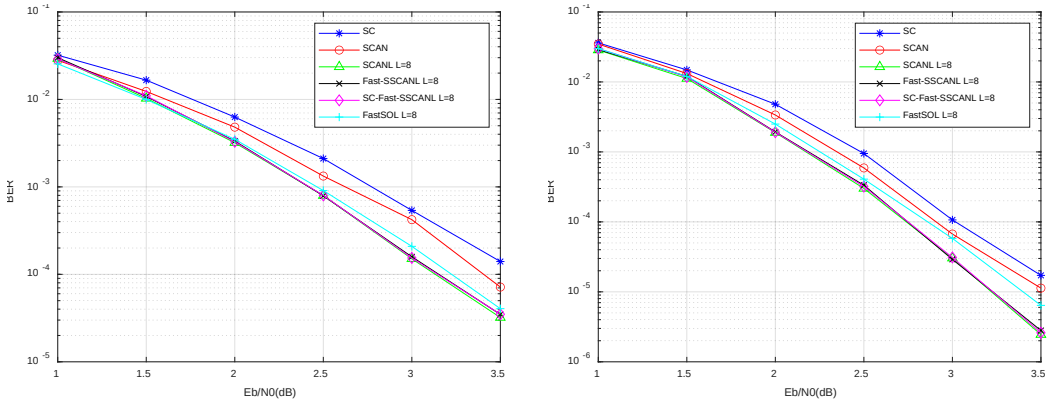
In **Fig. 10**, we compared the BER performance of the proposed Fast-SSCANL and SC-Fast-SSCANL algorithms with SCANL algorithm under different code length N and code rate R.The related parameter L is set to 8. From **Fig. 10**, it can be seen that the BER performance of the SC-Fast-SSCANL decoding algorithm is similar to the performance of the other decoding algorithms under different code length and code rate.



(a) $R=1/4$ (b) $R=1/2$ (c) $R=3/4$

Fig. 10. BER performance of various decoding algorithms.

Fig. 11 illustrated the BER performance comparison between SC, SCAN, FastSOL (A Soft-Input Soft-Output Polar Decoding Algorithm in [14]), SCANL, and the proposed Fast-SSCANL, SC-Fast-SSCANL decoding algorithm with code rate $R=1/2$. Under different code length N , the BER performance of the proposed SC-Fast-SSCANL algorithm outperforms SC, SCAN and FastSOL algorithms, and is similar to that of SCANL and Fast-SSCANL algorithms.



(a) $N=256$ (b) $N=512$

Fig. 11. The BER performance of different decoding algorithms.

4.2 Decoding Latency Analysis

In this section, we evaluate the decoding latency of the proposed Fast-SSCANL and SC-Fast-SSCANL algorithms, and comparison with the latency of SCANL algorithm. Here, we suppose that hard decisions on LLRs and bit operations are executed instantaneously, and operations involving real numbers (addition, comparison) require one clock cycle. With this assumption, one SCAN update rule such as $\lambda_a = f\left(\lambda_b, \lambda_d + \beta_c\right)$ in (1) consumes 2 clock cycles, since it is composed of one $f\left(x, y\right)$ operation followed by an addition operation. If we use SCAN algorithm to decode the factor graph shown in **Fig. 5**, the SCAN decoding has a latency of $2\times 4\times 4 = 32$ clock cycles. So, the SCAN latency for a node at stage t is $L_{latency} = 2\times 4\times t\times 2^{t-1} = t\cdot 2^{t+2}$, the overall SCANL latency is then $L_{latency} = t\cdot 2^{t+2}\cdot L$.

The proposed Fast-SSCANL decoding runs L independent Fast-SSCAN decoders. In Fast-SSCAN, we consider the decoding of rate-0 and rate-1 as instantaneous decoding, since the returned values of β_i^t are constant. The latency of Fast-SSCAN to decode an REP node of size 2^t is $L_{latency} = 2$, since the sum of LLRs in (4) takes 1 clock cycle, while removing one LLR value takes 1 clock cycle. Fast-SSCAN decoder use (6) to decode an SPC node. According to (6), hard decisions and (7) are executed instantaneously, hence "$*$" operation and the search of the least reliable LLRs are taken into consideration. we assume that the minimum search operation and "$*$" operation both take 1 clock cycle, so the latency of Fast-SSCAN to decode an SPC node is $L_{latency} = 2$.

When Fast-SSCAN decoder decodes an RI node, it can be seen from (8) that the decoding of an RI node of size 2^t can be divided into two parts. We assume that the first 2^{t-1} indexed and the last 2^{t-1} indexed values are interpreted as two independent parts that can be decoded in parallel, taking 1 cycle. As seen in **Fig. 7**, use (8) to decode RI node at stage $t=3$, we can get both β_0^3 (as in (11)) and β_4^3 (as in (12)), β_1^3 and β_5^3, and so on. Thus, we can find that $\sum_{j=0,j\neq i}^{2^{t-1}-1} f\left(\lambda_j^t, \lambda_{j+2^{t-1}}^t\right)$ is same as $\sum_{k=0,k\neq i-2^{t-1}}^{2^{t-1}-1} f\left(\lambda_k^t, \lambda_{k+2^{t-1}}^t\right)$. $\sum_{j=0,j\neq i}^{2^{t-1}-1} f\left(\lambda_j^t, \lambda_{j+2^{t-1}}^t\right)$ requires 2^{t-1} $f\left(x, y\right)$ operations and $4^{t-1}-2^t$ addition operations. So, the decoding latency of an RI node at stage t requires $L_{latency} = 2^{t-1} + 4^{t-1} - 2^t + 1 = 4^{t-1} - 2^{t-1} + 1$.

$$\beta_0^3 = f\left(f\left(\lambda_1^3, \lambda_5^3\right) + f\left(\lambda_2^3, \lambda_6^3\right) + f\left(\lambda_3^3, \lambda_7^3\right), \lambda_4^3\right) \tag{11}$$

$$\beta_4^3 = f\left(f\left(\lambda_1^3, \lambda_5^3\right) + f\left(\lambda_2^3, \lambda_6^3\right) + f\left(\lambda_3^3, \lambda_7^3\right), \lambda_0^3\right) \tag{12}$$

Same as RI node, according to (9), the decoding of an FS node also is interpreted as two independent parts that can be decoded in parallel, then taking 2 clock cycles. We also find that $\min_{\substack{j\neq i \\ 0\le j\le 2^{t-1}-1}} \left(\left|\lambda_j^t + \lambda_{j+2^{t-1}}^t\right|\right)$ is same as $\min_{\substack{k\neq i \\ 2^{t-1}\le k\le 2^t-1}} \left(\left|\lambda_k^t + \lambda_{k-2^{t-1}}^t\right|\right)$. $\min_{\substack{j\neq i \\ 0\le j\le 2^{t-1}-1}} \left(\left|\lambda_j^t + \lambda_{j+2^{t-1}}^t\right|\right)$ requires one minimum search operation and 2^{t-1} addition operations. Therefore, for Fast-SSCAN decoder, an FS node at stage t needs $L_{latency} = 2^{t-1} + 3$ clock cycles.

The final proposed SC-Fast-SSCANL decoding algorithm relies on L independent SC-Fast-SSCAN decoders. Compared with Fast-SSCAN decoder, SC-Fast-SSCAN decoder adds syndrome check operation, thus the latency of syndrome check should be taken into account.

According to algorithm 1, hard decisions and bit operations are executed instantaneously, hence "*" operations and the operation to get $\boldsymbol{H}$ are considered. The operation to get $\boldsymbol{H}$ costs 1 clock cycle. The latency for SCAN in case of an ordinary node at stage t is $L_{latency} = t \cdot 2^{t+2}$, however the latency of an ordinary node at stage t with successful syndrome check can be reduced to $L_{latency} = 4$.

Once the latency of special nodes has been analyzed, we analyze the number of special nodes. Since Fast-SSCANL and SC-Fast-SSCANL have the same codeword structure, the frequency of occurrence of special nodes is same in Fast-SSCANL and SC-Fast-SSCANL. Take a polar code of $N = 1024$ and $R = 0.25, 0.5, 0.75$ for example, the number of special nodes is shown in **Table 1**. From **Table 1**, Low-rate polar codes are more likely to have rate-0, REP and FS nodes, and high-rate polar codes are more likely to have rate-1, SPC and RI nodes.

Table 1. Number of special nodes for various code rates

Special node	$R = 0.25$	$R = 0.5$	$R = 0.75$
rate-0	80	48	16
rate-1	8	56	128
REP	120	160	72
SPC	24	144	160
RI	16	64	96
FS	64	88	32

Based on the above descriptions, we simulate and analyze the latency of SCANL, Fast-SSCANL and SC-Fast-SSCANL algorithms under various code lengths N and rates R. The latency data is shown in **Table 2**.

Table 2. Latency of SCANL, Fast-SSCANL and SC-Fast-SSCANL algorithms

Algorithm	$N = 256$			$N = 512$			$N = 1024$		
	$R = \frac{1}{4}$	$R = \frac{1}{2}$	$R = \frac{3}{4}$	$R = \frac{1}{4}$	$R = \frac{1}{2}$	$R = \frac{3}{4}$	$R = \frac{1}{4}$	$R = \frac{1}{2}$	$R = \frac{3}{4}$
SCANL	65536			147456			327680		
Fast-SSCANL	8385	10321	8454	16579	24183	17105	36474	42271	38666
SC-Fast-SSCANL	7512	9789	7472	14515	19907	15483	32525	37028	34124

From **Table 2**, it can be seen that both Fast-SSCANL and SC-Fast-SSCANL have less latency than SCANL, where SC-Fast-SSCANL is even better. Fast-SSCANL can reduce the latency

of SCANL of more than 83% regardless of code length and code rate, however SC-Fast-SSCANL can reduce SCANL latency by more than 85%.

4.3 Decoding Complexity Analysis

In this paper, the complexity is analyzed according to the number of add operations and the number of multiplication operations. Compared to SCANL decoding algorithm and SSCANL decoding algorithm, SC-Fast-SSCANL decoding algorithm involves the computation of special nodes, the number of add and multiplication operations generated by different types of nodes are different. The complexity of special nodes is analyzed as shown in **Table 3**.

Table 3. Computational complexity for special nodes

Special node	Number of additive operations	Number of multiplication operations
REP	$2^{2n-t}-2^{n+1}$	0
SPC	0	2^{n-t}
RI	$3\times 2^{n-t}$	2^{n+1-t}
FS	$2^{n-t}+2^{n-1}$	2^{n-t}

Where $n=\log_2 N$ and the parameter t denotes the stage of the decoding factor graph where the special node is located.

Taking $N=1024$, $R=0.25,0.5,0.75$ for example, the occurrence frequency of the four special nodes is analyzed below. From **Table 4**, it can be seen that low rate codes rates are more probable to have REP nodes and FS nodes, while high rate codes are more probable to have SPC nodes and RI nodes.

Table 4. Number of various special nodes

Special node	$R=0.25$	$R=0.5$	$R=0.75$
REP	120	160	72
SPC	24	144	160
RI	16	64	96
FS	64	88	32

Table 5 and **Table 6** respectively show the reduction in addition and multiplication compared to the SCANL algorithm for the proposed SSCANL, Fast-SSCANL, and SC-Fast-SSCANL decoding algorithms respectively.

Table 5. Add operation reduction results compared to SCANL algorithm

Algorithm	$N=256$			$N=512$			$N=1024$		
	$R=\frac{1}{4}$	$R=\frac{1}{2}$	$R=\frac{3}{4}$	$R=\frac{1}{4}$	$R=\frac{1}{2}$	$R=\frac{3}{4}$	$R=\frac{1}{4}$	$R=\frac{1}{2}$	$R=\frac{3}{4}$
SSCANL	59.4%	48.2%	15.5%	61.6%	48.9%	14.3%	60.3%	49.4%	16.4%
Fast-SSCANL	81.2%	70.7%	76.6%	82.6%	72.1%	74.9%	82.2%	78.7%	81.5%
SC-Fast-SSCANL	86.2%	83.1%	85.3%	87.8%	85.5%	88.6%	89.5%	88.1%	89%

Table 6. Multiplication operation reduction results compared to SCANL algorithm

Algorithm	$N=256$			$N=512$			$N=1024$		
	$R=\frac{1}{4}$	$R=\frac{1}{2}$	$R=\frac{3}{4}$	$R=\frac{1}{4}$	$R=\frac{1}{2}$	$R=\frac{3}{4}$	$R=\frac{1}{4}$	$R=\frac{1}{2}$	$R=\frac{3}{4}$
SSCANL	74.6%	49.1%	25.3%	75.9%	50.2%	24.8%	75.3%	49.2%	25.6%
Fast-SSCANL	83.8%	74.3%	80.1%	86.4%	75.8%	82.6%	87.7%	76.5%	85.2%
SC-Fast-SSCANL	88.9%	86.5%	87.3%	90.4%	88.6%	88.2%	93.2%	89.7%	91.3%

As can be seen from **Table 5** and **Table 6**, SC-Fast-SSCANL decoding algorithm has lowest computational complexity. And the lower the bit rate, the more obvious the reduction effect. Compared with SCANL decoding algorithm, SC-Fast-SSCANL decoding algorithm can reduce the number of addition operations by at least 83.1% and the number of multiplication operations by 86.5%.

5. Conclusion

Fast-SSCANL decoding algorithm and its' improved algorithm (SC-Fast-SSCANL) are proposed as an improvement of the SCANL decoding algorithm in this paper. The proposed Fast-SSCANL and SC-Fast-SSCANL algorithms can achieve the same performance as that of the SCANL algorithm with lower latency and lower computational complexity. Compared with the SCANL decoding algorithm, Fast-SSCANL can reduce latency by more than 83%, however SC-Fast-SSCANL can reduce SCANL latency by more than 85%. Compared with the SCANL decoding algorithm, SC-Fast-SSCANL can reduce the number of addition operations by at least 83.1% and the number of multiplication operations by 86.5%.

References

[1] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on information Theory*, vol. 55, no. 7, pp. 3051-3073, July. 2009. Article (CrossRef Link)

[2] G. Sarkis, P. Giard, A. Vardy, C. Thibeault and W. J. Gross, "Fast polar decoders: Algorithm and implementation," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 946-957, May. 2014. Article (CrossRef Link)

[3] M. Hanif and M. Ardakani, "Fast successive-cancellation decoding of polar codes: Identification and decoding of new nodes," *IEEE Communications Letters*, vol. 21, no. 11, pp. 2360-2363, Nov. 2017. Article (CrossRef Link)

[4] M. H. Ardakani, M. Hanif, M. Ardakani and C. Tellambura, "Fast successive-cancellation-based decoders of polar codes," *IEEE Transactions on Communications*, vol. 67, no. 7, pp. 4562-4574, July. 2019. Article (CrossRef Link)

[5] H. Yoo and I. C. Park, "Efficient pruning for successive-cancellation decoding of polar codes," *IEEE Communications Letters*, vol. 20, no. 12, pp. 2362-2365, Dec. 2016. Article (CrossRef Link)

[6] U. U. Fayyaz and J. R. Barry, "Low-complexity soft-output decoding of polar codes," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 958-966, May. 2014. Article (CrossRef Link)

[7] J. Lin, C. Xiong and Z. Yan, "Reduced complexity belief propagation decoders for polar codes," in *Proc. of 2015 IEEE Workshop on Signal Processing Systems (SiPS)*, Hangzhou, China, pp. 1-6, 2015. Article (CrossRef Link)

[8] M. Cai, S. Li and Z. Liu, "An Improved Simplified Soft Cancellation Decoding Algorithm for Polar Codes Based on Frozen Bit Check," in *Proc. of 2021 IEEE 21st International Conference on Communication Technology (ICCT)*, Tianjin, China, pp. 127-131, 2021. Article (CrossRef Link)

[9] C. Pillet, C. Condo and V. Bioglio, "Fast-SCAN decoding of polar codes," in *Proc. of 2021 11th International Symposium on Topics in Coding (ISTC)*, Montreal, QC, Canada, pp. 1-5, 2021. Article (CrossRef Link)

[10] C. Pillet, C. Condo and V. Bioglio, "SCAN list decoding of polar codes," in *Proc. of ICC 2020-2020 IEEE International Conference on Communications (ICC)*, Dublin, Ireland, pp. 1-6, 2020. Article (CrossRef Link)

[11] X. Wang, H. Liu, J. Li, Z. Zheng and J. He, "Scan-bf decoding of polar codes," *IEEE Communications Letters*, vol. 25, no. 8, pp. 2507-2511, 2021. Article (CrossRef Link)

[12] L. Zhang, Y. Sun, Y. Shen, W. Song, X. You and C. Zhang, "Efficient fast-SCAN flip decoder for polar codes," in *Proc. of 2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, Daegu, Korea, pp. 1-5, 2021. Article (CrossRef Link)

[13] L. Chen and G. Rui, "Generalized SCAN Bit-Flipping Decoding Algorithm for Polar Code," *KSII Trans. Internet Inf. Syst.*, vol. 17, no. 4, pp. 1296-1309, 2023. Article (CrossRef Link)

[14] Y. Shen, W. Zhou, Y. Huang, Z. Zhang, X. You and C. Zhang, "Fast iterative soft-output list decoding of polar codes," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1361-1376, 2022. Article (CrossRef Link)

Liu Chongyang is a Master student in Hangzhou Dianzi University. His research interests include channel coding and wireless communication.

Dai Wenjie is a Master student in Hangzhou Dianzi University. His research interests include channel coding and wireless communication.

Guo Rui received the Ph.D. degree from the Zhejiang University, Hangzhou, China, in 2007. He is currently an associate professor with the School of Communication Engineering, Hangzhou Dianzi University, Hangzhou, China. He is a visiting scholar at Oregon State University from August 2018 to August 2019. His research interests include wireless communication and channel coding.