IET Communications

Research Article

IET Journals
The Institution of Engineering and Technology

# Fast decoding of polar codes using tree structure

ISSN 1751-8628
Received on 8th February 2018
Revised 21st April 2019
Accepted on 13th May 2019
E-First on 24th July 2019
doi: 10.1049/iet-com.2018.5019
www.ietdl.org

*Alia A. Andi*[1] ✉, *Orhan Gazi*[1]

[1]*Department of Electronics and Communication Engineering, Cankaya University, Ankara, Turkey*
✉ *E-mail: alia_eletri@yahoo.com*

**Abstract:** In this study, the authors first propose a tree structure for the successive cancelation (SC) decoding of polar codes. The proposed structure is easy to implement in hardware and suitable for parallel processing operations. Next, using the proposed tree structure, they propose a technique for the fast decoding of polar codes. With the proposed method, it is possible to decode all the information bits simultaneously at the same time, i.e. in parallel. Lastly, they introduce an improved version of the proposed high-speed decoding algorithm. The proposed high-speed decoding approach and its improved version are simulated on the computer environment and their bit error rate performances are compared to the performance of the classical SC method.

## 1 Introduction

Polar codes introduced in [1] are one of the capacity achieving codes on discrete memoryless channels, and their performances are mathematically proven. For a block-length of $N$, they have low encoding and tolerable decoding complexity on the order of $O(N\log N)$. In addition, their construction is very explicit. Polar codes can achieve good performance at even short code lengths. Since their introduction in 2009, polar codes are subjected to many other studies. The sequential bit-by-bit decoding nature of polar codes is one of the main drawbacks for practical implementations. For this reason, a number of studies, which can be classified as latency and computational complexity reduction algorithms, have been performed by the researchers to alleviate the serial decoding drawback of polar codes. The studies on the decoding latency reduction of polar codes are presented in [2–8]. An early stopping criteria for belief propagation polar code decoder is proposed in [9]. In [3], decoding two bits simultaneously, the latency of the polar decoder employing successive cancelation (SC) algorithm is halved. In [4], single instruction multiple data programming model was used to decode $F$ frames in parallel. The proposed approach does not increase the hardware complexity. The parallelism is performed in software. In [8], an improved version of [3] is proposed. However, the computational complexity of the proposed approach increases in an exponential manner and becomes unfeasible for practical systems.

A family of hardware implementation architectures for SC decoders are introduced in [10], where it is shown that decoders employing SC algorithms can be implemented with a number of processing elements on the order of $O(n)$, and a number of memory elements on the order of $O(n)$ with constant throughput. Another study for the practical implementation of polar codes employing belief propagation on field-programmable gate array (FPGA) platform is introduced in [11]. Apart from these studies, approximate weight distribution of polar codes via a polynomial complexity algorithm is inspected in [12].

In [13], the advantage of polar codes over Reed-Muller codes under belief-propagation decoding is inspected. In [14, 15], the improved versions of the SC decoding algorithm are introduced. In [16], the main ideas and techniques of channel polarisation, polar codes and the SC decoding algorithm are presented. In [17], a semi-parallel architecture for the implementation of SC decoding is presented. The problem of efficient construction of polar codes over binary memoryless symmetric channels is considered in [18]. Speeding-up technique has been proposed in [19] where an algorithm is proposed which increases the throughput of the simplified successive-cancelation decoding algorithm three times. The studies in [20–22] focus on the complexity reduction of the encoding, decoding operations and increase the throughput of polar codes.

In this paper, we propose a high-speed polar decoder structure which reduces the decoding latency of polar codes significantly. The proposed technique uses the tree structure of the SC algorithm. In addition, we also introduce an improved version of the proposed high speed polar decoder, which shows better bit error rate (BER) performance. The latency of the proposed approach is the same as the latency of a single bit decoding in classical SC algorithm. In addition, our parallel processing method can be implemented both in hardware and software. That is, without the need of extra hardware, the parallelism can be performed in software as well. For this purpose, digital electronic devices such as FPGAs and hardware programming languages such as VHDL can be utilised. The outline of the paper is as follows. In Section 2, we provide some background information for polar encoding and decoding operations. A tree structure for the SC decoding of polar codes is proposed in Section 3. In Section 4, we introduce our proposed high-speed polar decoding approach along with its improved version. Simulation results are provided in Section 5, and finally, conclusions are drawn in Section 5.

## 2 Background information

In this section, we provide background information about encoding and decoding operation of polar codes.

### 2.1 Polar encoding

Let us denote the $N$-bit information vector by

$$u_1^N = (u_1, u_2, \ldots, u_N).$$

The polar code-word for the data vector $u_1^N$ is obtained using

$$x_1^N = u_1^N \boldsymbol{G}_N$$

where the generator matrix $\boldsymbol{G}_N$ is calculated via

$$\boldsymbol{G}_N = B_N \boldsymbol{F}^{\otimes n}$$

in which $N = 2^n$,

$$\boldsymbol{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

and $B_N$ is found using

$$B_N = \boldsymbol{R}_N (I_2 \otimes B_{N/2})$$

where the initial value of $B_N$, i.e. $B_2$ is $I_2$, and $\boldsymbol{R}_N$ denotes the $N \times N$ reverse shuffle permutation matrix whose operation is explained in

$$(s_1, s_2, s_3, \ldots, s_N)\boldsymbol{R}_N = (s_1, s_3, \ldots, s_{N-1}, s_2, s_4, \ldots, s_N)\,.$$

The Kronecker product of two matrices $\boldsymbol{A} = [\,\cdot\,]_{m\times n}$ and $\boldsymbol{B} = [\,\cdot\,]_{k\times l}$ is obtained as

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} A_{11}B & \ldots & A_{1n}B \\ \vdots & \ddots & \vdots \\ A_{m1}B & \ldots & A_{mn}B \end{bmatrix}$$

and the Kronecker power is defined as

$$\boldsymbol{A}^{\otimes n} = A \otimes A^{\otimes (n-1)}\,.$$

For $N = 4$, the generator matrix can be found as

$$\boldsymbol{G}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

and the structure of the encoder unit can be drawn as in Fig. 1 where $W$ denotes a communication channel.

### 2.2 Decoding of polar codes

In Arikan's original paper, for the decoding of polar codes, the SC decoding algorithm is proposed. In this algorithm, the likelihood ratio for the $i$th bit using the previously decided bits $\hat{u}_1^{i-1}$ is computed as

$$L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) = \frac{W_N^{(i)}(y_1^N, \hat{u}_1^{i-1} | 0)}{W_N^{(i)}(y_1^N, \hat{u}_1^{i-1} | 1)} \tag{1}$$

which can be recursively calculated via

$$L_N^{(2i-1)}(y_1^N, \hat{u}_1^{2i-2}) = \frac{L_{N/2}^{(i)}(y_1^{N/2}, \hat{u}_{1,o}^{2i-2} \oplus \hat{u}_{1,e}^{2i-2}) L_{N/2}^{(i)}(y_{N/2+1}^N, \hat{u}_{1,e}^{2i-2}) + 1}{L_{N/2}^{(i)}(y_1^{N/2}, \hat{u}_{1,o}^{2i-2} \oplus \hat{u}_{1,e}^{2i-2}) + L_{N/2}^{(i)}(y_{N/2+1}^N, \hat{u}_{1,e}^{2i-2})} \tag{2}$$

and

$$L_N^{(2i)}(y_1^N, \hat{u}_1^{2i-1}) = \left[L_{N/2}^{(i)}(y_1^{N/2}, \hat{u}_{1,o}^{2i-2} \hat{u}_{1,e}^{2i-2})\right]^{1-2\hat{u}_{2i-1}} L_{N/2}^{(i)}(y_{N/2+1}^N, \hat{u}_{1,e}^{2i-2})\,. \tag{3}$$

For $N = 1$, the initial likelihood ratio is computed as

$$L_1^1(y_i) = \frac{W(y_i|0)}{W(y_i|1)}$$

and finally, the decision is made according to

$$\hat{u}_i = \begin{cases} 0, & \text{if } L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}) \geq 1 \\ 1, & \text{otherwise}\,. \end{cases} \tag{4}$$

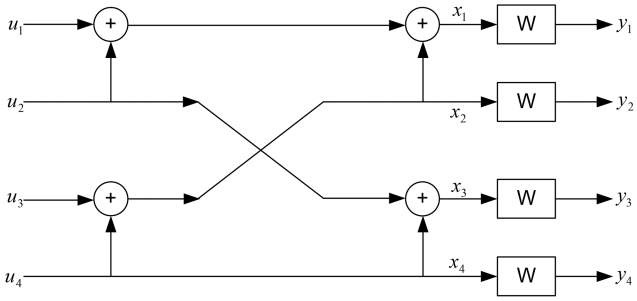


**Fig. 1** *Polar encoder for codeword length $N = 4$*

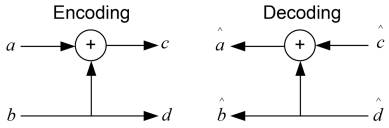


**Fig. 2** *Kernel encoding and decoding units of polar codes*

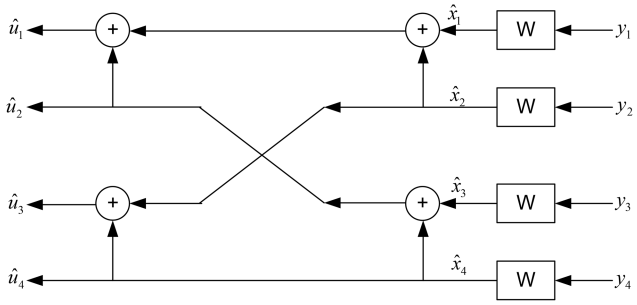


**Fig. 3** *Decoding operation for codeword length $N = 4$*

In [1], a graphical butterfly structure is suggested for the calculation of the likelihood ratios in (2) and (3). However, the structure is complex and not suitable for parallel processing operations. For this reason, we propose a tree structure for the SC decoding of polar codes in the following section.

## 3 Tree structure for the decoding of polar codes

The kernel units that are repeatedly used in encoder and decoder structures of polar codes are depicted in Fig. 2. The kernel unit can also be considered as the smallest polar encoder and decoder units.

Considering the decoder unit in Fig. 2 where *a, b, c, d* and $\hat{a}, \hat{b}, \hat{c}, \hat{d}$ are binary variables, we can calculate the likelihood ratios

$$LR(\hat{a}) = \frac{P(\hat{a}=0)}{P(\hat{a}=1)} \quad LR(\hat{b}) = \frac{P(\hat{b}=0)}{P(\hat{b}=1)} \tag{5}$$

as

$$LR(\hat{a}) = \frac{(1 + LR(\hat{c})LR(\hat{d}))}{(LR(\hat{c}) + LR(\hat{d}))} \tag{6}$$

and

$$LR(\hat{b}) = [LR(\hat{c})]^{1-2\hat{a}} \times LR(\hat{d})\,. \tag{7}$$

It is obvious that the recursive formulas given in (2) and (3) are similar to those given in (6) and (7).

Now, let us explain the proposed tree structure for the decoding of polar codes. To understand the derivation of the proposed algorithm, first, let us give some information about the formation of tree structure used in the decoding operation.

In Fig. 1, the encoding structure of the polar codes for $N = 4$ is illustrated. In the decoding operation, the flow of the signals is reversed. For $N = 4$, the decoding structure with the reversed signal flow is shown in Fig. 3.

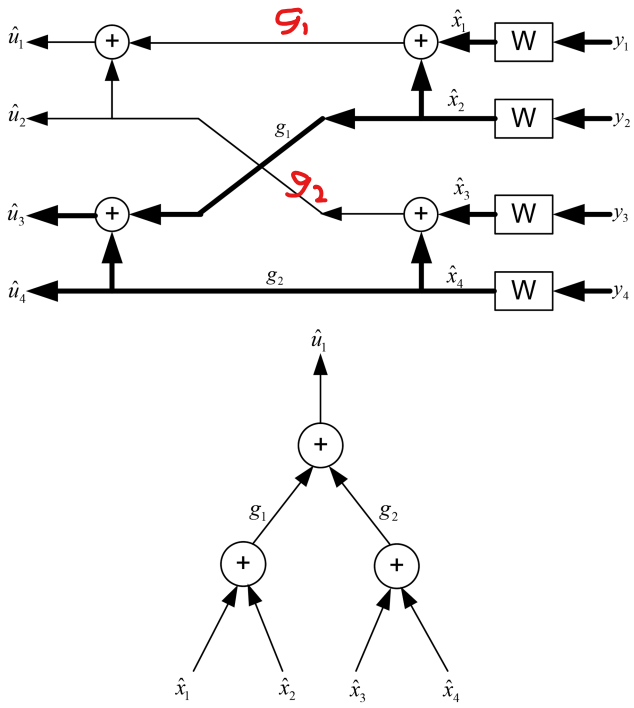


**Fig. 4** *Decoding path for $\hat{u}_1$ and its equivalent tree structure*

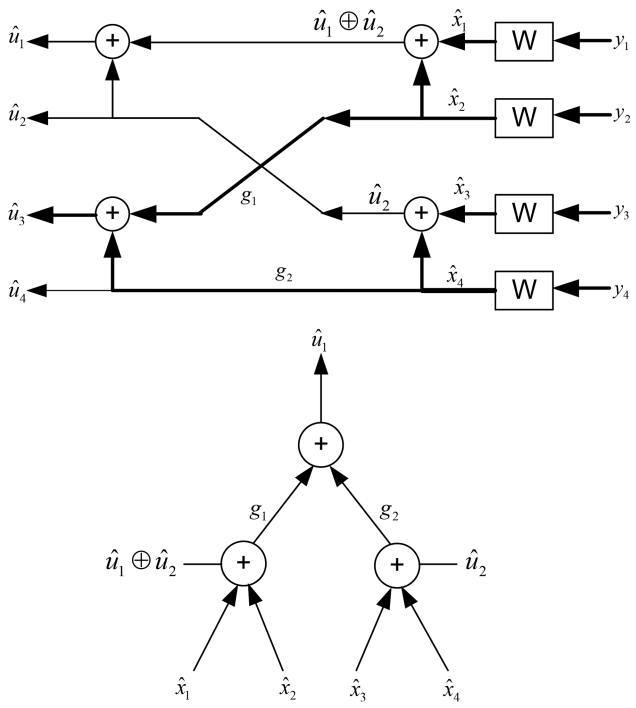


**Fig. 5** *Decoding path for $\hat{u}_3$*

The decoding path for information bit $\hat{u}_1$ and its tree equivalent is shown in Fig. 4 where the likelihoods are calculated as

$$\begin{aligned} LR(g_1) &= \frac{(1 + LR(\hat{x}_1)LR(\hat{x}_2))}{(LR(\hat{x}_1) + LR(\hat{x}_2))} \\ LR(g_2) &= \frac{(1 + LR(\hat{x}_3)LR(\hat{x}_4))}{(LR(\hat{x}_3) + LR(\hat{x}_4))} \\ LR(\hat{u}_1) &= \frac{(1 + LR(g_1)LR(g_2))}{(LR(g_1) + LR(g_2))} \end{aligned} \tag{8}$$

For better comprehension, let us give one more example. The decoding path for $\hat{u}_3$ and its tree equivalent tree structure is depicted in Fig. 5 where the likelihoods are calculated as

$$\begin{aligned} LR(g_1) &= [LR(\hat{x}_1)]^{1-2(\hat{u}_1 \oplus \hat{u}_2)} \times LR(\hat{x}_2) \\ LR(g_2) &= [LR(\hat{x}_3)]^{1-2\hat{u}_2} \times LR(\hat{x}_4) \\ LR(u_3) &= \frac{(1 + LR(g_1)LR(g_2))}{(LR(g_1) + LR(g_2))}. \end{aligned} \tag{9}$$

When the tree structure in Fig. 5 is inspected in detail, we see that some of the nodes have assigned bits, for instance in the tree structure of Fig. 5, $\hat{u}_1 \oplus \hat{u}_2$ is assigned to the lower left node and $\hat{u}_2$ is assigned to the lower right node. Also, if a bit is assigned to a node, we call these nodes as $g$ nodes, on the other hand, if a node does not have any bit assignment, we call such nodes $f$ nodes. In addition, we employ (2) for the likelihood ratio calculation for outputs of the $f$ nodes, and we employ (3) for the likelihood ratio calculation for the outputs of the $g$ nodes.

Now, let us generalie the decoding logic illustrated for $N = 4$ in Figs. 4 and 5. The decoding operation consists of two stages. The first stage is the distribution of previously decoded bits to the nodes, i.e. deciding on the value of node bits. The second stage involves the calculation of the likelihood values. The decoding operation goes in a recursive manner as, distribution, likelihood calculation, distribution, likelihood calculation and so on. In the likelihood calculation stage, node $L$-values are computed for each layer starting from the basement layer and top-most node $L$-value is used for decision. In distribution stage, decided bits are distributed to the nodes and these bits are called node bits. If there is a bit assigned to a node, then (3) is used to compute the node-$L$ value, otherwise, (2) is used. The bit distribution process is performed according to the Algorithm 1 (see Fig. 6).

**Input:** $d_1^R$: is the decoded bit stream, $R$: is length of the decoded bit stream
**if** *$R$ is odd* **then**
  Node-check-bit = $d_R$;
  Left child-node input-bits: $L_b = d_{1,o}^{N-1} \oplus d_{1,e}^{R-1}$;
  Right child-node input-bits: $R_b = d_{1,e}^{R-1}$;
**else**
  Left child node input-bits: $L_b = d_{1,o}^{R} \oplus d_{1,e}^{R}$;
  Right child node input-bits: $R_b = d_{1,e}^{R}$;
**end**
**if** *$R$=1* **then**
  Terminate ;
**else**
  $R = \lfloor R/2 \rfloor$;
  Go to step-1 and repeat $1 - 6$ for the left-child and right-child nodes;
**end**

**Fig. 6** *Algorithm 1: Distribution of decoded bits to the nodes*

A numerical example of the distribution stage for the decoding of sixth bit is depicted in Fig. 7 where it is seen that after the distribution 5 previously decoded bits to the nodes, we see that the node at the top-most level, i.e. level-0, and the nodes at level-2 have bits, i.e. $g$ nodes are available at level-0 and level-2 since we have $6 = 2^0 + 2^2$.

Once the bits are distributed to the nodes, the decoding operation starts which involves the calculation of likelihoods from bottom to the top layer for the decoding of sixth bit.

## 4 Fast decoding of polar codes

In this section, we propose a novel high-speed decoding technique for the decoding of polar codes. The proposed high-speed decoder is based on the introduced tree structure. Our proposed algorithm can decode $N$ successive bits at the same time. Therefore, the latency can be reduced and speed can be increased. The presented $N$-bit-decoding algorithm is able to decode $N$ bits at the same time (in parallel), i.e. it can decode $n$th bit without the need of $(n - 1)$ previously decoded bits.

As mentioned before, using the tree structure, we can divide the decoding operation into two parts, distribution of the previously decoded bits to the nodes, and calculation of node LRs for the

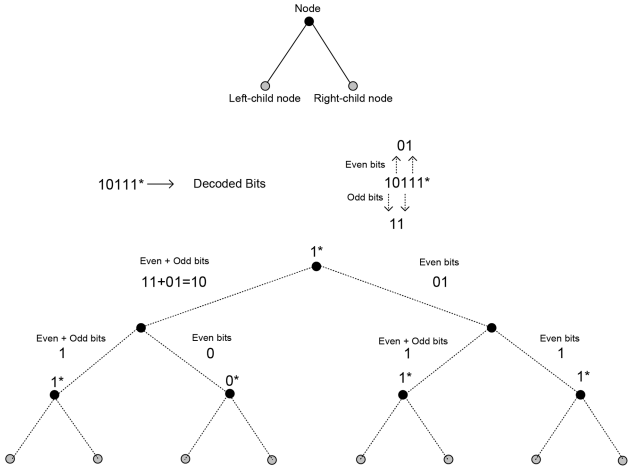


**Fig. 7** *Distribution of the decided bits to the nodes*

**Table 1** Percentage of frozen bits ~~in the second half versus~~ code rate

| Rate | Number of frozen bits | Percentage of frozen bits |
|---|---|---|
| 0.5 | 0 | 0 |
| 0.434 | 67 | 0.13 |
| 0.41 | 91 | 0.177 |
| 0.37 | 130 | 0.2539 |
| 0.359 | 144 | 0.281 |
| 0.33 | 168 | 0.328 |
| 0.32 | 179 | 0.349 |

decoding of the current bit. In the bit distribution stage, the tree is divided into $\log_2(N)+1$ levels, where $N$ is frame length a power of 2. The index of the top level is 0, and the index of the bottom level is $\log_2(N)$. When the previously decoded information bits are distributed to the nodes, we see that nodes at certain levels receive the distributed bits, i.e. $g$-nodes appears at certain levels. Let us call the levels where $g$-nodes appear as active levels. In fact, for the distribution of $l$ previously decoded bits, the active levels can be determined using

$$l = \sum_i 2^i \tag{10}$$

where $i$ denotes the active level index. The $g$-nodes in active levels have labels 0 or 1. As an example, assume that $N = 8$, and the first 5 bits are decoded. To start the decoding of sixth bit, the previously decoded 5 bits are distributed into the nodes, and the number 5 can be written as

$$5 = \sum_i 2^i \rightarrow \; 5 = 2^0 + 2^2 \tag{11}$$

where powers of 2 indicate the locations of $g$ nodes. i.e. they indicate the active levels and the levels with indices 0 and 2 are active levels, and the nodes in these levels are all $g$ nodes, i.e. nodes have bit-labels; the bit-labels either include 0 or 1. All the other nodes in the tree structure are $f$ nodes which do not have bit-labels. After the first stage of the decoding operation, i.e. distribution of previously decoded bits to the nodes, the second stage of the decoding operation starts. In the second stage of the decoding operation, $LR$s of the nodes in each level starting from bottom level to top level are calculated.

### *4.1 High-speed decoding*

For a $N$-bit information sequence, after channel splitting operation, we have $N$ new channels. For these $N$ channels it is seen that most of the low capacity channels occur in the first half, i.e. low capacity channels have indices $1, \ldots, N/2$. And most of the frozen bits are assigned to these low capacity channels. This is the main motivation of our study. Although some of the channels in the second half have low capacities, their number is few considering the total number of channels. For this reason, we dedicate the first group containing $N/2$ channels to the frozen bits, and use the second group containing the rest of the $N/2$ channels for the data bits. This means that the decoding operation starts for the data bits transmitted through the second part of the communication channels. The number of frozen bits and their percentage considering different code rates is tabulated in Table 1 where it is seen that for code rate 0.5, there is no frozen bit in the second half of the information frame. When $N/2$ frozen bits are distributed on the tree, it is seen that $N/2$ nodes get '0' as node-bit. Besides, these $N/2$ nodes exist on the same level which we call as the frozen level. When the bit distribution is performed for the decoding of consecutive data bits, it is seen that these $N/2$ nodes always contain '0's as node-bits, i.e. frozen level stay the same. And some levels depending on the order of the data bit to be decoded become active levels, and the nodes in these active levels either contain '0' or '1', and these nodes are nothing but $g$ nodes. The likelihood ratio at these nodes can be calculated separately for node-bit '0' and node-bit '1' and the larger can be selected to be used for the nodes at upper levels. And this logic can be carried till the top level. In this way, we do not need to know the previously decoded bits, but just need to know the active and frozen levels.

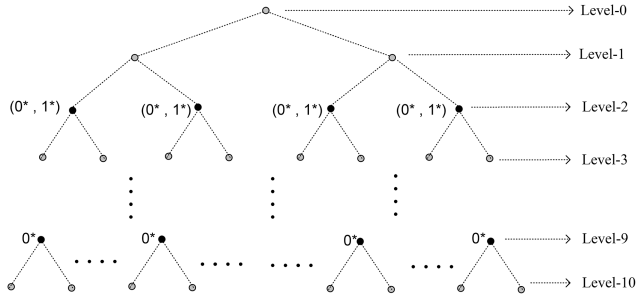


**Fig. 8** *Active and frozen levels for the decoding of data bit* $\hat{u}_{517}$

*Example 1:* Let us illustrate the idea with an example. Assume that $N = 1024$. In this case, the indices of most of the low capacity channels appear in [1 2 3 … $N/2$]. For this reason, we can freeze the first 512 channels and start decoding the bits with indices greater than 512. Assume that we want to decode the bit with index number 517. Then, we distribute the first 516 decoded bits, 512 of them are frozen bits, to the tree nodes. When the first 516 decoded bits are distributed to the nodes, we get a graph like in Fig. 8 where it is seen that level-2 and level-9 active levels, i.e. $g$ nodes appear in these levels. This is expected since $517 = 2^2 + 2^9$, i.e. powers of 2 indicate the active level indices. The nodes in level-9 contain only 0 s. This level can be called as a frozen level since it is filled by zeros via the distribution of frozen bits. On the other hand, the nodes at level-2 may either contain 0 or 1. Since the node bits of level-2 are determined by the distribution of 4 data bits other than the frozen bits, but, we do not need to know the exact values of the node bits at level-2. We can try both 0 and 1 for the calculation of the likelihood for the node output and choose the larger one for the upper nodes.

The advantage of the mentioned method is that we can determine the active levels for the decoding of data bits and data bits can be decoded in a parallel manner without needing the previously decoded bits. Hence, if sufficient place can be found in an electronic device like FPGA, it is possible to decode all the data bits in a concurrent manner. The fast decoding operation can be outlined as in Algorithm 2 (see Fig. 9).

### *4.2 Computational complexity of the proposed approach*

The total number of $f$ nodes and $g$ nodes appearing during the decoding operation is the same and it equals to

1: Input frame length, i.e., $N$, and received symbols.
2: Determine frozen level.
3: Let $r = N/2$.
4: Determine active levels for $r$.
5: Use $g$-nodes for active levels.
6: Use $f$-nodes for inactive levels.
7: For $g$-nodes appearing in the active levels other than the frozen level, use 0 and 1 as bit nodes separately.
8: Calculate $LR$ from the bottom of the tree to the top, use maximum of the two likelihood values for $g$ nodes.
9: Determine the bit $u_{r+1}$.
10: Increment $r$, i.e., $r = r + 1$.
11: Go to step-4.
12 : $When$r=N$stop$.

**Fig. 9** *Algorithm 2: High-speed decoding of polar codes*

| Frozen part | Use classical SC decoding for 50% of the information bits | Use the proposed high speed decoding for the rest of the 50% of the information bits |
|---|---|---|

| 512 Frozen bits | Decode 256 bits using classical SC decoder | Decode 256 bits using the proposed high speed decoding technique |
|---|---|---|

**Fig. 10** *Improved high-speed decoding approach*

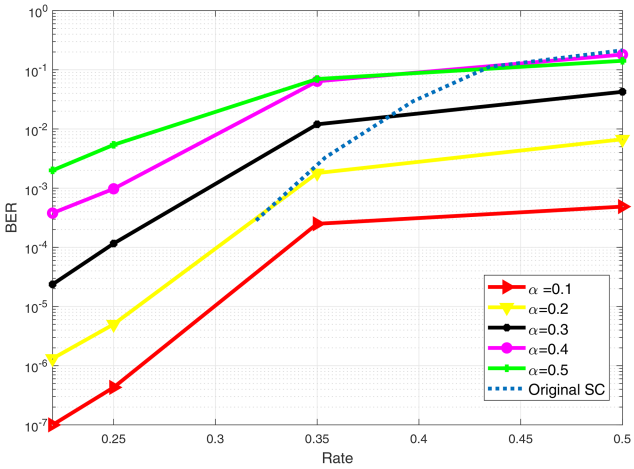


**Fig. 11** *BER versus rate performance comparison for the proposed method and classical SC approach. BEC with $\alpha = 0.5$ is used for SC decoding*

$$K_N = \sum_{i=0}^{n} 2^i$$

where $n = \log_2 N$. For $g$ type nodes, we calculate the probabilities for both $u = 0$ and $u = 1$ separately and choose the one containing more information. For this reason, in our approach, the computation complexity for $g$ type nodes doubles. However, it stays the same for $f$ type nodes. If we indicate the total computational complexity of the *SC* decoding approach by $O(2K_N)$ where $O(\cdot)$ is the *big*-O notation, then the computational complexity of the proposed method can be expressed by $O(3K_N)$. On the other hand, if we indicate the latency of the *SC* decoding algorithm by $O(N)$ where $N$ is the number of data bits to be decoded, then the latency of our proposed technique can be expressed by $O(1)$.

### *4.3 Improved high-speed decoding*

In order to improve the decoding performance of the proposed high-speed decoder, we introduce a new decoding scheme called improved high-speed decoding technique. In this new approach, some percent of the information bits are decoded using the classical SC method and the rest of the bits are decoded at the same time using the proposed parallel decoding approach. With this approach, we aim to increase the BER performance of the communication system. An example of the new approach for $N = 1024$ is depicted in Fig. 10 where we froze the first $N/2$ bits which correspond to low capacity channels, and we employ the SC decoding approach to decode the 50% of the rest of the information bits. Finally, the remaining information bits are decoded in parallel at the same time using the proposed approach.

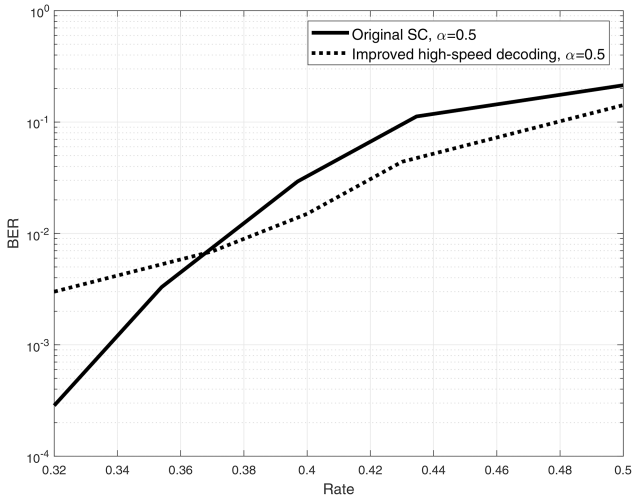


**Fig. 12** *Performance of improved high-speed decoding versus high speed decoding*

With this newly proposed method, the latency of the previously proposed parallel decoding technique increases. However, we obtain better performance considering all parallel decoding approach. In addition, the total latency is still much less than the latency of the SC method. Since the low capacity channels occurs in the first half of the data frame, by freezing the first half of the transmission frame we prevent the error propagation which affects the decoding of the second half, and by successively decoding some of the bits in the second half, we provide more robust bit decisions which will be used for the decoding of the rest of the bits, and for this reason we obtain better results.

## 5 Simulation results

In Figs. 11 and 12, we compare the performance of polar codes using the proposed high-speed decoding approach and SC decoding at block length $2^{10}$ for binary erasure channels, i.e. BECs, with erasure probabilities 0.1, 0.2, 0.3, 0.4 and 0.5 in terms of bit error rate, i.e. BER. In practical systems, usually code rates between 0 and 0.5 are considered for implementations. For this reason, we consider the code rates in the range 0 to 0.5 for the code generation. From simulation results, we can see that the proposed high-speed decoding algorithm gives good performance at low rates, especially for the rates between 0.35 and 0.5. We also performed computer simulations using the improved high decoding method employing the frame structure in Fig. 10. Performance results are depicted in Fig. 12. It is seen from Fig. 12 that for a block-length of 1024 bits, the improved high-speed decoding method gives better performance with reduced latency over the binary erasure channel with erasure probability 0.5. Besides, the proposed method is also simulated in an *AWGN* channel. The performance results are depicted in Fig. 13 where it is seen that the classical *SC* approach and the proposed method have similar performance for the given *SNR* range.

## 6 Conclusions

The contributions in this paper can be listed as follows. (i) First, we proposed a tree structure for the SC decoding of polar codes. (ii) Next using the introduced tree structure, we proposed a high-speed low latency decoding algorithm for polar codes, such that, using the proposed algorithm it is possible to decode all the information bits simultaneously in parallel. BER performance of polar codes using the proposed high-speed decoding method over binary-

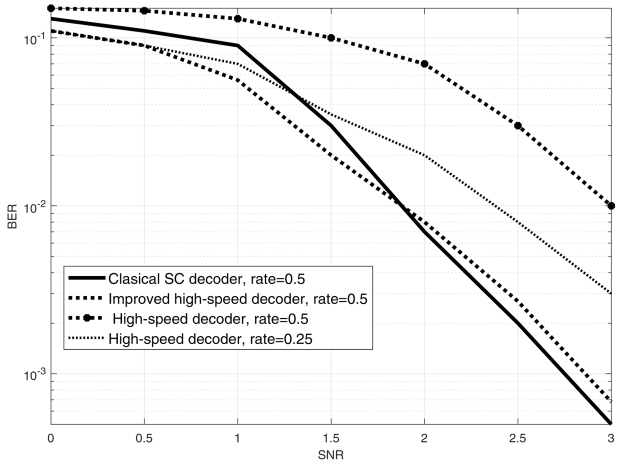


**Fig. 13** *BER versus SNR performance comparison in AWGN channel for different rates*

erasure channels is obtained via computer simulations. From simulation results, we see that proposed high-speed decoding algorithm good performance at high code rates, especially for the rates between 0.35 and 0.5. (iii) The proposed high-speed decoding algorithm gives a great improvement in polar codes decoding speed. For frame length $N = 1024$ and $rate = 0.5$, while original SC decoder can decode only one bit for a decoding stage, the proposed high-speed decoder can decode 512 bits all together at the same decoding stage and since in the proposed high-speed decoding algorithm, we do not need to know the previously decoded bits. (iv) Next, we suggested an improved version of the proposed high-speed decoding technique. The proposed improved high-speed decoding method shows better performance than that of the classical SC decoding method at high rates with much lower decoding latency.

## 7 References


[1] Arikan, E.: 'Channel polarisation: a method for constructing capacity achieving codes for symmetric binary-input memoryless channels', *IEEE Trans. Inf. Theory*, 2009, **55**, pp. 3051–3073

[2] Zhang, C., Parhi, K.K.: 'Low-latency sequential and overlapped architectures for successive cancelation polar decoder', *IEEE Trans. Signal Process.*, 2013, **61**, (10), pp. 2429–2441

[3] Yuan, B., Parhi, K.K.: 'Low-latency successive-cancelation polar decoder architectures using 2-bit decoding', *IEEE Trans. Circuits Syst. I Regul. Pap.*, 2014, **61**, (4), pp. 1241–1254

[4] Le Gal, B., Leroux, C., Jego, C.: 'Multi-gb/s software decoding of polar codes', *IEEE Trans. Signal Process.*, 2015, **63**, (2), pp. 349–359

[5] Fan, Y., Xia, C., Chen, J., *et al.*: 'A low-latency list successive-cancelation decoding implementation for polar codes', *IEEE J Sel. Areas Commun.*, 2016, **34**, (2), pp. 303–317

[6] Li, B., Shen, H., Tse, D., *et al.*: 'Low-latency polar codes via hybrid decoding'. 8th Int. Symp. on Turbo Codes and Iterative Information Processing (ISTC), Bremen, 18–22 August 2014, pp. 223–227

[7] Zhang, C., Yuan, B., Parhi, K.K.: 'Reduced-latency SC polar decoder architectures'. IEEE ICC 2012 – Signal Processing for Communications Symp., Ottawa, ON, 10–15 June 2012

[8] Yuan, B., Parhi, K.K.: 'Low-latency successive-cancelation list decoders for polar codes with multibit decision'. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Singapore, 2014 September1–12

[9] Yuan, B., Parhi, K.K.: 'Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders', *IEEE Trans. Signal Process.*, 2014, **62**, (24), pp. 6496–6506

[10] Leroux, C., Tal, I., Vardy, A., *et al.*: 'Hardware architectures for successive cancelation decoding of polar codes'. Proc. 2011 IEEE Int. Conf. Acoustics Speech and Signal Processing, Prague, 2011, pp. 1665–1668

[11] Pamuk, A.: 'An FPGA implementation architecture for decoding of polar codes'. 8th Int. Symp. on Wireless Communication Systems (ISWCS), Aachen, 6–9 November 2011, pp. 437–441

[12] Valipour, M., Yousefi, S.: 'On probabilistic weight distribution of polar codes', *IEEE Commun. Letters*, 2013, **17**, (11), pp. 2120–2123

[13] Arikan, E.: 'A performance comparison of polar codes and reed-muller codes', *IEEE Commun. Lett.*, 2008, **12**, (6), pp. 447–449

[14] Chen, K.: 'Improved successive cancelation decoding of polar code', *IEEE Trans. Commun.*, 2013, **61**, (8), pp. 3100–3107

[15] Yuan, B.: 'Algorithm and VLSI Architecture for polar codes decoder'. PhD Dissertation, July 2015

[16] Le, D., Wu, X., Niu, X.: 'Decoding schedule generating method for successive-cancelation decoder of polar code', *IET Commun.*, 2016, **10**, (5), pp. 462–467

[17] Leroux, C., Raymond, A.J., Sarkis, G., *et al.*: 'A semi-parallel successive-cancelation decoder for polar codes', *IEEE Trans. Signal Process.*, 2013, **61**, pp. 289–299

[18] Pedarsani, R., Hamed Hassani, S., Tal, I., *et al.*: 'On the construction of polar code'. IEEE Int. Symp., Poland, Gdansk, 2011, pp. 11–15

[19] Sarkis, G., Gross, W.J.: 'Increasing throughput of polar decoder', *IEEE Commun. Lett.*, 2013, **17**, pp. 725–728

[20] Koike-Akino, T., Cao, C., Wang, Y., *et al.*: 'Irregular polar coding for complexity-constrained lightwave systems', *IEEE/OSA J. Lightwave Technol.*, 2018, **36**, (11), pp. 2248–2258

[21] Cao, C., Koike-Akino, T., Wang, Y., *et al.*: 'Irregular polar coding for massive MIMO'. IEEE Global Communication Conf., Singapore, December 2017

[22] Koike-Akino, T., Cao, C., Wang, Y.: 'Turbo product codes with irregular polar coding for high-throughput parallel decoding in wireless OFDM transmission'. 2018 IEEE Int. Conf. on Communications (ICC), Kansas City, MO, 2018, pp. 1–7