SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

February 2018, Vol. 61 022309:1–022309:11
doi: 10.1007/s11432-017-9128-x

A complexity-reduced fast successive cancellation list decoder for polar codes

Qingyun XU, Zhiwen PAN*, Nan LIU & Xiaohu YOU

National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096*, China*

Received 20 February 2017/Accepted 25 April 2017/Published online 20 November 2017

Abstract A multi-bit decision for polar codes based on a simplified successive cancellation (SSC) decoding algorithm can improve the throughput of polar decoding. A list algorithm is used to improve the error-correcting performance. However, list decoders are highly complex compared with decoders without a list algorithm. In this paper, a low-complexity list decoder is proposed, where path-splitting operations for a multi-bit decision can be avoided, if the decoding reliability exceeds a threshold. The threshold is determined based on the reliability of subchannels and positions of decoding nodes. Path splitting rules are designed for multi-bit decision processes, and a complexity-reduced list decoder is proposed based on this. Results show that the number of survival paths can be greatly reduced at the cost of negligible deterioration in block error performance. Thus, the computational complexity can be significantly reduced, especially for a high signal-to-noise ratio (SNR) region.

Keywords polar codes, low-complexity decoder, splitting-reduced, multi-bit decision, list decoder

Citation Xu Q Y, Pan Z W, Liu N, et al. A complexity-reduced fast successive cancellation list decoder for polar codes. Sci China Inf Sci, 2018, 61(2): 022309, doi: 10.1007/s11432-017-9128-x

1 Introduction

Polar coding, first proposed by Arikan [1], is a linear block coding scheme that provably approaches symmetric channel capacity with explicit and deterministic structure for a binary-input discrete memoryless channel (B-DMC). Moreover, it is a promising coding scheme for future wireless communications [2].

Because a low-complexity successive cancellation (SC) decoding algorithm for polar codes can approach capacity when the code length goes to infinite [1], much attention was especially given to efforts toward improving the throughput. The simplified successive cancellation (SSC) algorithm proposed in [3], shows significant improvement in throughput. This complexity reduction is achieved by defining special nodes in the decoding tree. Moreover, constituent codes consisting of all frozen bits (Rate-0 nodes) and all information bits (Rate-1 nodes) can be decoded directly without recursion. The fast simplified successive cancellation (FSSC) algorithm further extends this idea [4]. In the FSSC scheme, a node whose leaves are all information bits, except for the first one, is denoted as a single-parity-check (SPC) node, and only the last leaf of a repetition (REP) node is an information bit. Throughput can be further improved by considering SPC nodes and REP nodes.

However, the error-correcting performance of an SC-based decoder is poor at short and moderate block lengths because subchannels are not completely polarized for finite length polar codes. The successive cancellation list (SCL) introduced in [5] further improves the performance of the SC algorithm. In an

* Corresponding author (email: pzw@seu.edu.cn)

SCL decoder, both 0 and 1 are considered as estimated bits and two decoding paths are generated at each decoding stage. The L most reliable paths are retained in the list. The cyclic redundancy check (CRC) is used in [6] to select the correct decoding path in the SCL algorithm. It is shown that the error-correcting performance of the list decoder with CRC outperforms that of low-density parity-check (LDPC) codes at similar code lengths [5]. However, the complexity of SCL-based algorithms for code length N and list size L is $O(LN\log N)$ [5], which is L times higher than that of SC-based algorithms. Authors in [7] proposed the successive cancellation stack (SCS) scheme, which has a lower computational complexity than that of the SCL decoder. A list decoder is applied in the SSC algorithm in [8] to improve the throughput of the SCL decoder. A fast simplified successive cancellation list (FSSCL) scheme was proposed, applying a list algorithm in the FSSC decoder to further improve the speed of the list decoding algorithm in [9].

The complexity of improved list decoders proposed in [8, 9] is proportional to their list sizes, which are fixed even when the signal-to-noise ratio (SNR) of the channel is high. An adaptive list scheme was proposed in [10], in which the decoder starts with a small list size and then increases it, if there is no survival path passing CRC. A scheme deleting certain paths whose metrics were lower than a threshold was proposed in [11]. A scheme avoiding certain path splitting operations was introduced in [12] to reduce the complexity of the SCL decoder when the reliability of information bit is sufficiently high. A list algorithm with multi-bit decision, based on the SSC algorithm, can reduce computational complexity as well as decoding latency [4, 13]. However, the existing split-reduced or path pruning schemes for the list algorithm are not applicable to multi-bit decision based on the SSC algorithm.

In this paper, we propose a complexity-reduced algorithm for a list decoder based on multi-bit decision. The path splitting of multi-bit decision can be avoided, if the decoding reliability exceeds a threshold. Thus, fewer candidate paths will be generated at each decoding stage due to path splitting reduction. For a high SNR, the number of survival paths is less than the list size throughout the decoding stages. Hence, the computational complexity can be reduced significantly. First, a scheme to determine the thresholds for both leaf nodes and non-leaf nodes in the decoding tree is proposed. Then, path splitting rules for Rate-1, REP, and SPC nodes are proposed, respectively, based on the thresholds.

The remainder of the paper is organized as follows: Section 2 is the necessary background for polar codes. Section 3 is the proposed decoding algorithm. Performance analysis is given in Section 4. Simulation results and analysis are given in Section 5, and Section 6 concludes the paper.

2 Preliminary

2.1 Polar codes

For an (N, K) polar code with block length $N = 2^n$, $n \in \mathbb{N}^*$ and information length K, the generator matrix can be represented as $\boldsymbol{G}_N = \boldsymbol{B}_N \boldsymbol{F}_2^{\otimes n}$ [1], where $\boldsymbol{B}_N$ is an $N \times N$ bit-reversal permutation matrix, $\boldsymbol{F}_2 = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$ is the kernel matrix, and $\otimes$ denotes the Kronecker product. Given the generator matrix, the polar code is encoded as

$$\boldsymbol{x}_1^N = \boldsymbol{u}_1^N \boldsymbol{G}_N, \tag{1}$$

where $\boldsymbol{x}_1^N = (x_1, x_2, \ldots, x_N)$ denotes the encoded bit sequence and $\boldsymbol{u}_1^N = (u_1, u_2, \ldots, u_N)$ denotes the source bit sequence that consists of information bits and frozen bits.

Let $W : \mathcal{X} \to \mathcal{Y}$ denote a B-DMC with channel transition probability $\{W(y|x) : x \in \mathcal{X}, y \in \mathcal{Y}\}$, where $\mathcal{X} = \{0, 1\}$ denotes the input alphabet and $\mathcal{Y} = \{0, 1\}$ denotes the output alphabet. After channel combining and splitting, we have N binary-input subchannels defined by transition probabilities

$$W_N^{(i)}\left(\boldsymbol{y}_1^N, \boldsymbol{u}_1^{i-1} | u_i\right) = \sum_{u_{i+1}^N \in X^{N-i}} \frac{1}{2^{N-1}} W_N\left(\boldsymbol{y}_1^N | \boldsymbol{u}_1^N\right), \tag{2}$$

where

$$W_N\left(\boldsymbol{y}_1^N | \boldsymbol{u}_1^N\right) = \prod_{i=1}^{N} W\left(y_i | x_i\right), \tag{3}$$

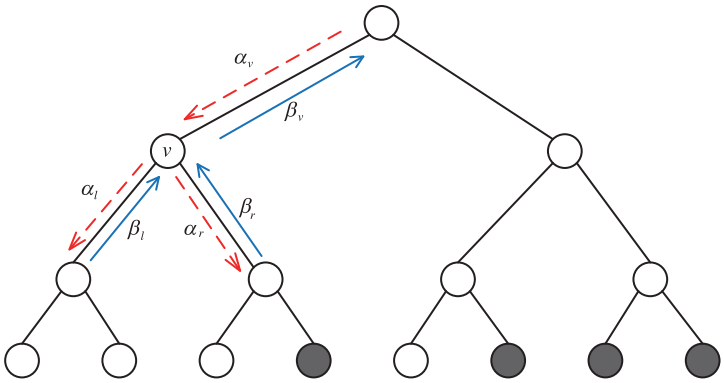

Figure 1 (Color online) Decoding tree of an (8, 4) polar code.

and $\boldsymbol{y}_1^N = (y_1, y_2, \ldots, y_N)$ represents the channel output sequence observed, u_i is the input of $W_N^{(i)}$, $\boldsymbol{u}_1^{i-1} \in \mathcal{X}^{i-1}$ denotes the past channel inputs, and suppose all past input bits can be correctly decoded at earlier stages. The K information bits are transmitted over the K most reliable subchannels, and the remaining N-K subchannels are used for frozen bits [14].

2.2 SC decoding

The SC decoder estimates the information bits sequentially using the received channel output sequence $\boldsymbol{y}_1^N$ and previously estimated $\hat{\boldsymbol{u}}_1^{i-1}$.

Given the block length N, the SC decoding can be represented by a binary tree with depth $n = \log N + 1$ due to the recursive nature of the polar codes. Each node in the decoding tree represents a constituent code of length N_v, where N_v is the number of leaf nodes corresponding to this node. Figure 1 shows a decoding tree of an (8, 4) polar code where hollow leaves denote the frozen bits, and solid leaves denote the information bits.

In general, a node v receives a log-likelihood ratio (LLR) vector $\boldsymbol{\alpha}_v$ from its parent node, and returns a hard decision bit vector $\boldsymbol{\beta}_v$. The node traverses its child nodes recursively in the same manner and the decoding is initiated by feeding the root node with channel LLRs. When a node v is activated, $\boldsymbol{\alpha}_l$ is given by

$$\begin{aligned} a_l[i] &= F\left(\alpha_v[2i-1], \alpha_v[2i]\right) \\ &\approx \operatorname{sgn}\left(\alpha_v[2i-1]\right) \operatorname{sgn}\left(\alpha_v[2i]\right) \min\left(\left|\alpha_v[2i]\right|, \left|\alpha_v[2i-1]\right|\right), \end{aligned} \tag{4}$$

where $\alpha_l[i]$ is the i-th element of $\boldsymbol{\alpha}_l$ and $1 \leqslant i \leqslant N_v/2$.

The decoder will traverse to the left child node (node v_l in Figure 1) by passing $\boldsymbol{\alpha}_l$ to it until it returns a bit sequence $\boldsymbol{\beta}_l$. Then $\boldsymbol{\alpha}_r$ is given by

$$\begin{aligned} \alpha_r[i] &= G\left(\alpha_v[2i-1], \alpha_v[2i], \beta_l[i]\right) \\ &= \alpha_v[2i] - \left(2\beta_l[i] - 1\right) \alpha_v[2i-1], \end{aligned} \tag{5}$$

where $\alpha_r[i]$ is the i-th element of $\boldsymbol{\alpha}_r$ and $1 \leqslant i \leqslant N_v/2$.

The decoder traverses to the right child node (node v_r in Figure 1) by sending $\boldsymbol{\alpha}_r$ to it. After the right child node returns a bit sequence $\boldsymbol{\beta}_r$, bit vector $\boldsymbol{\beta}_v$, to be sent to its parent node (node v_p in Figure 1), is given by

$$\begin{cases} \beta_v[2i-1] = \beta_l[i] \oplus \beta_r[i], \\ \beta_v[2i] = \beta_r[i], \end{cases} \quad \text{for } 1 \leqslant i \leqslant N_v/2, \tag{6}$$

where $\oplus$ denotes an XOR operation.

When a leaf node is activated, $\boldsymbol{\beta}_v$ is set to zero if the leaf node represents a frozen bit, otherwise $\boldsymbol{\beta}_v$ is decoded by

$$\boldsymbol{\beta}_v = h\left(\boldsymbol{\alpha}_v\right), \tag{7}$$

where

$$h(x) = \begin{cases} 0, & x > 0, \\ 1, & \text{else.} \end{cases} \tag{8}$$

When the decoder returns from root node, the estimations of the transmitted bits are

$$\hat{\boldsymbol{u}}_1^N = \boldsymbol{\beta}_v \boldsymbol{G}_N. \tag{9}$$

2.3 SSC and FSSC decoding

The SC decoder traverses the decoding tree recursively until reaching the leaf nodes. The SSC algorithm proposed in [3] obtains the output bits of certain nodes directly using hard decision, and its child nodes do not need to be traversed.

The output of a Rate-0 node is an all-zero vector, and the output of a Rate-1 node is given by

$$\beta_v[i] = h\left(\alpha_v[i]\right) \quad \text{for} \quad 1 \leqslant i \leqslant N_v. \tag{10}$$

The FSSC algorithm further prunes the decoding tree by defining SPC nodes and REP nodes. The output of an SPC node is calculated using (10). The least reliable bit is flipped if the parity-check of $\boldsymbol{\beta}_v$ is not satisfied, where the reliability is defined as the absolute value of $\boldsymbol{\alpha}_v$. The output of an REP node is

$$\beta_v[i] = \begin{cases} 0, & \sum\limits_j \alpha_v[j], \\ 1, & \text{else,} \end{cases} \qquad 1 \leqslant i \leqslant N_v. \tag{11}$$

2.4 List-CRC decoding

The list decoder allows L decoding paths to be retained in the list in order to improve the error-correcting performance. When an information bit is estimated, both 0 and 1 are considered as estimated bits and two decoding paths are generated. Path generation rules for Rate-1, REP, and SPC nodes are discussed in [8, 9], respectively. Each path is decoded by the SC-based algorithm; the L most reliable paths with the largest path metric (PM) are retained at each decoding stage. For a path l, when the i-th bit is estimated, its path metric PM_l^i is updated according to

$$\mathrm{PM}_l^i = \begin{cases} \mathrm{PM}_l^{i-1}, & \text{if} \quad \hat{u}_i = h(\alpha_i), \\ \mathrm{PM}_l^{i-1} - |\alpha_i|, & \text{otherwise,} \end{cases} \tag{12}$$

where α_i is the LLR of the i-th bit.

When list decoding is finished, the survival path that satisfies the CRC constraint is selected as the final decoder output.

3 Complexity-reduced fast SSC list algorithm

In this section, an improved fast SSC list decoding scheme is proposed in which path splitting can be avoided if the decoding reliability exceeds a threshold. First, a scheme is proposed for determining the thresholds according to the reliability of subchannels and locations of nodes. Then, path splitting rules are designed for Rate-1, REP, and SPC nodes, respectively. Finally, a complexity reduced fast SSC list decoding algorithm is proposed based on this. The number of survival paths is much smaller than the list size under the proposed scheme, thus, the computation complexity can be reduced significantly.

3.1 The path splitting threshold for leaf nodes

A new path will not be generated if the decoding reliability (the LLR of a bit) is larger than a threshold that is relevant to the reliability of the subchannel.

Suppose u_i is an information bit that would be observed through a subchannel $W_N^{(i)}\left(\boldsymbol{y}_1^N, \boldsymbol{u}_1^{i-1}\,|u_i\right)$ whose error probability is denoted as $P_e(W_N^{(i)})$, and $1-P_e(W_N^{(i)})$ is regarded as reliability of $W_N^{(i)}$ [12]. The decoding confidence is high enough if $W_N^{(i)}\left(\boldsymbol{y}_1^N, \hat{\boldsymbol{u}}_1^{i-1}\,|u_i\right) > 1-P_e(W_N^{(i)})$. Then, a new path will not be generated if either of the following inequalities are satisfied

$$W_N^{(i)}\left(\boldsymbol{y}_1^N, \hat{\boldsymbol{u}}_1^{i-1}\,|u_i=0\right) > 1-P_e\left(W_N^{(i)}\right), \tag{13}$$

$$W_N^{(i)}\left(\boldsymbol{y}_1^N, \hat{\boldsymbol{u}}_1^{i-1}\,|u_i=1\right) > 1-P_e\left(W_N^{(i)}\right). \tag{14}$$

Therefore, the path splitting threshold of u_i, denoted as TL_N^i, is defined as

$$\mathrm{TL}_N^i = \log\left(\frac{1-P_e\left(W_N^{(i)}\right)}{P_e\left(W_N^{(i)}\right)}\right). \tag{15}$$

Then, the hard decision of (7) becomes

$$\hat{u}_i = \begin{cases} 0, & L(u_i) > \mathrm{TL}_N^i, \\ 1, & L(u_i) < -\mathrm{TL}_N^i, \\ \text{split}, & \text{otherwise}, \end{cases} \tag{16}$$

where $L(u_i)$ denotes the LLR value of the transmitted bit u_i.

In general, there are no analytical expressions of $P_e(W_N^{(i)})$ for a general channel. For a binary erasure channel (BEC), the Bhattacharyya parameter $Z(W_N^{(i)})$ is an upper bound on an error probability of a maximum-likelihood (ML) decision for a subchannel $W_N^{(i)}$, which can be computed explicitly by recursion [1]. Therefore, $P_e(W_N^{(i)})$ can be approximated by $Z(W_N^{(i)})$ for BEC. For the most representative case of an additive white Gaussian noise (AWGN) channel, the LLRs follow an approximate Gaussian distribution. The mean value of LLR can be calculated in a recursive manner [15]. Then, $P_e(W_N^{(i)})$ can be expressed as a Q function. For a general channel, $P_e(W_N^{(i)})$ can be obtained from simulation.

3.2 The path splitting threshold for non-leaf nodes

For the non-leaf node discussed in Subsection 2.3, multi-bits will be decoded simultaneously, and the thresholds mentioned above are not directly applicable for these multi-bit nodes.

A threshold relating to the depth and position of the node can be defined for this case. For clarity, the delivery of LLR messages in the decoding tree for N=8 is illustrated in Figure 2. For a decoding tree, denote V_d^j as the j-th node among the nodes with depth d. The root node V_0^1 has a depth 0, and leaf node has a depth $n=\log_2 N$, where N is code length. Denote $\boldsymbol{\alpha}_{d,j}$ as the LLR vector that node V_d^j receives from its parent node, where $\alpha_{d,j}^{(k)}$ is the k-th element of $\boldsymbol{\alpha}_{d,j}$. In Figure 2, the dotted lines with an arrow denote F function defined in (4), and solid lines with an arrow denote G function defined in (5).

Suppose BPSK modulated code words are transmitted over an AWGN channel with noise variance σ^2. Denote $|\mathrm{E}(\alpha_{d,j}^{(k)})|$ as the absolute value of the expectation of $\alpha_{d,j}^{(k)}$. According to [15], the received LLRs follow a Gaussian distribution with mean $2/\sigma^2$ and variance $4/\sigma^2$. Under a Gaussian approximation, the outputs of the F and G functions follow a Gaussian distribution as well [15] and can be given by

$$\left|\mathrm{E}\left(\alpha_{d+1,2j-1}^{(k)}\right)\right| = \varphi^{-1}\left(1-\left(1-\varphi\left(\left|\mathrm{E}\left(\alpha_{d,j}^{(2k-1)}\right)\right|\right)\right)\left(1-\varphi\left(\left|\mathrm{E}\left(\alpha_{d,j}^{(2k)}\right)\right|\right)\right)\right), \tag{17}$$

$$\left|\mathrm{E}\left(\alpha_{d+1,2j}^{(k)}\right)\right| = \left|\mathrm{E}\left(\alpha_{d,j}^{(2k-1)}\right)\right| + \left|\mathrm{E}\left(\alpha_{d,j}^{(2k)}\right)\right|, \tag{18}$$

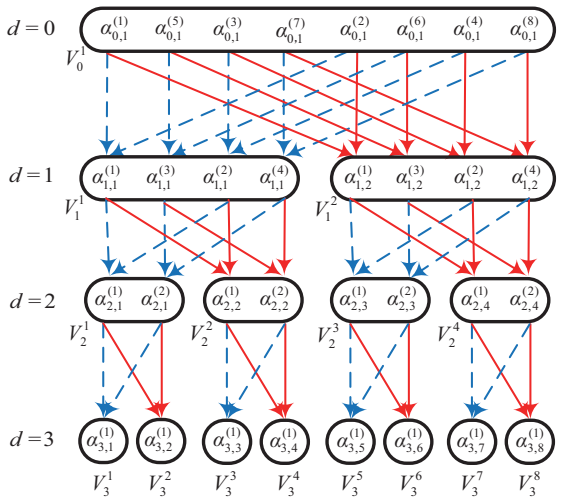

Figure 2 (Color online) Delivery of LLR messages in decoding tree for N=8.

where

$$\varphi(x) = \begin{cases} 1 - \dfrac{1}{\sqrt{4\pi |x|}} \int_{-\infty}^{\infty} \tanh\left(\frac{u}{2}\right) \mathrm{e}^{-\frac{(u-x)^2}{4|x|}} \mathrm{d}u, & x \neq 0, \\ 1, & x = 0. \end{cases}$$

The mean values of the LLRs in the root node V_0^1 have the same absolute value $\mu = 2/\sigma^2$. The LLR values in the left node (V_1^1 in Figure 2) are calculated by F functions. Thus, $|\mathrm{E}(\alpha_{1,1}^{(k)})|$ has the same value for $1 \leqslant k \leqslant N_v$, where N_v is the length of vector $\boldsymbol{\alpha}_{0,1}$. Similarly, all the absolute values of LLRs in the same node have the same mean values. For a given node V_d^j, $|\mathrm{E}(\alpha_{d,j}^{(k)})|$ is fixed, regardless of the full depth of the decoding tree.

In a multi-bit decision, the hard decision function (8) is used to decode LLR values of a non-leaf node. $P_e(\alpha_{d,j}^{(k)})$ is denoted as the error rate of a hard decision for $\alpha_{d,j}^{(k)}$, and $P_e(\alpha_{d,j}^{(k)})$ is given by

$$P_e\left(\alpha_{d,j}^{(k)}\right) = Q\left(\sqrt{\left|\mathrm{E}\left(\alpha_{d,j}^{(k)}\right)\right|} \Big/ 2\right),$$

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} \mathrm{e}^{-\frac{t^2}{2}} \mathrm{d}t.$$

Given σ^2, $P_e(\alpha_{d,j}^{(k)})$ can be calculated in an off-line manner.

It should be noted that the mean value of LLR has a constant absolute value for fixed node V_d^j and σ^2. As illustrated in Figure 2, V_d^j can also be regarded as the j-th leaf node for code length $N' = 2^d$. Therefore, $P_e(\alpha_{d,j}^{(k)})$ is the same as $P_e(W_{N'}^{(j)})$, discussed in Subsection 3.1, and $P_e(W_{N'}^{(j)})$ can be obtained from simulation for a general channel. Thus, hard decisions for LLRs of the same non-leaf node have the same reliability, which can be measured by $1 - P_e(W_{N'}^{(j)})$.

Having determined the metric of decoding reliability, the path splitting threshold for node V_d^j can be defined as

$$\mathrm{TL}_{N'}^{j} = \log\left(\frac{1 - \rho^{n-d} P_e\left(W_{N'}^{(j)}\right)}{\rho^{n-d} P_e\left(W_{N'}^{(j)}\right)}\right), \tag{19}$$

where ρ is a scaling parameter less than 1, $N' = 2^d$, $n = \log_2 N$.

$P_e(W_{N'}^{(j)})$ in (19) is multiplied by a coefficient ρ^{n-d}, compared to (15). For a given i and j, $P_e(W_N^{(i)})$ in (15) will decrease with the increasing of code length N, whereas, $P_e(W_{N'}^{(j)})$ in (19) is constant for a fix d. Because an increased code length will lead to a reduced average error rate for polar codes, and a larger threshold means higher decoding reliability, a larger threshold is needed to avoid the extra error

rate introduced by the new path generating rules. For a given node, the greater the gap between its depth and full depth of the decoding tree, the smaller ρ^{n-d} will be, and the larger the threshold will be, according to (19). Actually, the threshold defined in (15) can be replaced by (19), because for a leaf node, $d = n$ and $N' = N$, Eq. (19) becomes the same as (15).

It should be noted that, ρ affects both the error performance and decoding complexity. A larger ρ leads to lower complexity, but a larger block error rate.

3.3 The path generating rules and decoding algorithm

Having defined the path splitting threshold, the path splitting rules are next designed for Rate-1, REP, and SPC nodes, respectively. Then, a low complexity fast SSC list decoding algorithm is proposed.

Given a node V_d^j in decoding tree, denote $\mathrm{TL}_{N'}^j$ as the path splitting threshold obtained from (19). According to [9], all source paths update their metrics by

$$\mathrm{PM}_s^t = \mathrm{PM}_s^{t-1} - \sum_i |\beta_v[i] - h(\alpha_v[i])|\,|\alpha_v[i]|. \tag{20}$$

No new path will be generated for a Rate-0 node. For a Rate-1 node, at most, three new paths will be generated by flipping a combination of the two least confident bits. $\alpha_v[\min_1]$ and $\alpha_v[\min_2]$ denote the two least elements in vector $|\boldsymbol{\alpha}_v|$. For each source path s, if $|\alpha_v[\min_2]| < \mathrm{TL}_{N'}^j$, three new paths will be generated

$$\begin{aligned}
&\mathrm{P}_1\text{:}\ \ \mathrm{PM}_1^t = \mathrm{PM}_s^{t-1} - |\alpha_v[\min_1]|\,,\\
&\mathrm{P}_2\text{:}\ \ \mathrm{PM}_2^t = \mathrm{PM}_s^{t-1} - |\alpha_v[\min_2]|\,,\\
&\mathrm{P}_3\text{:}\ \ \mathrm{PM}_3^t = \mathrm{PM}_s^{t-1} - |\alpha_v[\min_1]| - |\alpha_v[\min_2]|\,,
\end{aligned}$$

where path P_1 is generated by flipping $\beta_v[\min_1]$, P_2 by flipping $\beta_v[\min_2]$, P_3 by flipping both $\beta_v[\min_1]$ and $\beta_v[\min_2]$. The new path above that contains $|\alpha_v[\min_i]|$ is unnecessary if $|\alpha_v[\min_i]| > \mathrm{TL}_{N'}^j$ for $1 \leqslant i \leqslant 2$.

An SPC node generates, at most, seven new paths by considering the four least reliable bits. $\alpha_v[\min_1]$–$\alpha_v[\min_4]$ denote the four least elements in vector $|\alpha_v|$. Let $q = 1$, if the parity check is satisfied and $q = 0$ otherwise. Seven new paths will be generated if $|\alpha_v[\min_4]| < \mathrm{TL}_{N'}^j$:

$$\begin{aligned}
&\mathrm{P}_1\text{:}\ \ \mathrm{PM}_1^t = \mathrm{PM}_s^{t-1} - q\,|\alpha_v[\min_1]| - |\alpha_v[\min_2]|\,,\\
&\mathrm{P}_2\text{:}\ \ \mathrm{PM}_2^t = \mathrm{PM}_s^{t-1} - q\,|\alpha_v[\min_1]| - |\alpha_v[\min_3]|\,,\\
&\mathrm{P}_3\text{:}\ \ \mathrm{PM}_3^t = \mathrm{PM}_s^{t-1} - q\,|\alpha_v[\min_1]| - |\alpha_v[\min_4]|\,,\\
&\mathrm{P}_4\text{:}\ \ \mathrm{PM}_4^t = \mathrm{PM}_s^{t-1} - |\alpha_v[\min_2]| - |\alpha_v[\min_3]|\,,\\
&\mathrm{P}_5\text{:}\ \ \mathrm{PM}_5^t = \mathrm{PM}_s^{t-1} - |\alpha_v[\min_2]| - |\alpha_v[\min_4]|\,,\\
&\mathrm{P}_6\text{:}\ \ \mathrm{PM}_6^t = \mathrm{PM}_s^{t-1} - |\alpha_v[\min_3]| - |\alpha_v[\min_4]|\,,\\
&\mathrm{P}_7\text{:}\ \ \mathrm{PM}_7^t = \mathrm{PM}_s^{t-1} - q\,|\alpha_v[\min_1]| - |\alpha_v[\min_2]| - |\alpha_v[\min_3]| - |\alpha_v[\min_4]|\,.
\end{aligned}$$

The new paths P_1–P_7 are generated by flipping $\boldsymbol{\beta}_v$ at the corresponding indices in the same way as Rate-1 nodes. The new path above containing $|\alpha_v[\min_i]|$ is unnecessary if $|\alpha_v[\min_i]| > \mathrm{TL}_{N'}^j$ for $1 \leqslant i \leqslant 4$.

The threshold for a REP node is the same as that of the last leaf node that belongs to it, i.e., TL_N^k, where $k = 2^{n-d}j$. If $|\sum_i \alpha_v[i]|$ is less than the threshold, a new path is generated by flipping all bits of $\boldsymbol{\beta}_v$ because REP nodes output all-zero or all-one bits. The path metrics for REP and SPC nodes are updated according to (20) using the new $\boldsymbol{\beta}_v$.

Based on the discussion above, the proposed decoding algorithm can be summarized as Algorithm 1.

Algorithm 1 Complexity reduced fast SSC list decoder

Decoding starts from the root node by setting $\alpha_v[i] = \log\left(\frac{\Pr(y_i|x_i=0)}{\Pr(y_i|x_i=1)}\right)$, for $1 \leqslant i \leqslant N$.
if current node $v \in \{\text{Rate}-0, \text{Rate}-1, \text{REP}, \text{SPC}\}$ **then**
  for each source path l **do**
    Calculate $\boldsymbol{\beta}_v$ according to Subsection 2.3;
    if $|\alpha_v[\min]| > \text{TL}_{N'}^{j}$ **then**
      Do not split new paths;
    else
      Generate new path according to Subsection 3.3;
    end if
  end for
  Prune paths if path number exceeds the specific list size L;
else
  Calculate $\boldsymbol{\alpha}_l$ of each path: $\alpha_l[i] = F\left(\alpha_v[2i-1], \alpha_v[2i]\right)$;
  go to the left child node;
  Calculate $\boldsymbol{\alpha}_r$ of each path: $\alpha_r[i] = G\left(\alpha_v[2i-1], \alpha_v[2i], \beta_l[i]\right)$;
  go to the right child node;
  Calculate $\boldsymbol{\beta}_v$ of each path: $\{\beta_v[2i-1], \beta_v[2i]\} = \{\beta_l[i] \oplus \beta_r[i], \beta_r[i]\}$;
  return to parent node;
end if
When the decoding returns from the root node, select the final output that satisfies the CRC constraint.

4 Analysis of error performance and complexity

4.1 Error performance

Denote $P'_e(\alpha_{d,j}^{(k)})$ as the failure probability that the hard decision of $\alpha_{d,j}^{(k)}$ is incorrect when $|\alpha_{d,j}^{(k)}| > \text{TL}_{N'}^{j}$ (no new path is generated). For the AWGN channel, $\alpha_{d,j}^{(k)}$ follows a Gaussian distribution. Thus,

$$\begin{aligned}
P'_e\left(\alpha_{d,j}^{(k)}\right) &= \Pr\left(\alpha_{d,j}^{(k)} < -\log\left(\frac{1-\rho^{n-d}P_e\left(W_{N'}^{(j)}\right)}{\rho^{n-d}P_e\left(W_{N'}^{(j)}\right)}\right)\right) \\
&< \Pr\left(\alpha_{d,j}^{(k)} < -\log\left(\frac{1-P_e\left(W_{N'}^{(j)}\right)}{P_e\left(W_{N'}^{(j)}\right)}\right)\right) \\
&= Q\left(\frac{\left|\mathrm{E}\left(\alpha_{d,j}^{(k)}\right)\right| + \log\left(1-P_e\left(W_{N'}^{(j)}\right)\right) - \log\left(P_e\left(W_{N'}^{(j)}\right)\right)}{\sqrt{2\left|\mathrm{E}\left(\alpha_{d,j}^{(k)}\right)\right|}}\right) \\
&= Q\left(Q^{-1}\left(P_e\left(W_{N'}^{(j)}\right)\right) + \frac{\log\left(1/P_e\left(W_{N'}^{(j)}\right)-1\right)}{2Q^{-1}\left(P_e\left(W_{N'}^{(j)}\right)\right)}\right). \qquad (21)
\end{aligned}$$

It has been proven in [12] that Eq. (21) is a higher order infinitesimal of $P_e(W_{N'}^{(j)})$ as $P_e(W_{N'}^{(j)})$ goes to infinitesimal. Thus, when $P_e(W_{N'}^{(j)})$ is small, $P'_e(\alpha_{d,j}^{(k)}) < P_e(W_{N'}^{(j)})$. $P_e(W_{N'}^{(j)})$ represents the error probability of the hard decision of $\alpha_{d,j}^{(k)}$, and $P'_e(\alpha_{d,j}^{(k)})$ represents the extra error rate introduced by the proposed path generating rule. Therefore, the proposed algorithm has little degradation on error performance.

4.2 Computational complexity

To simplify the complexity analysis, the complexity is measured by the number of LLR calculation units defined in (4) and (5), as in existing works [11]. Each F or G function is regarded as a calculation unit.

The complexity of the proposed scheme can be given by

$$C_p = \sum_{v \in Z_n} N_v L_v, \tag{22}$$

where L_v denotes the number of survival paths when node v is activated, and Z_n denotes the set of nodes that have to be traversed.

For the list decoder without a path splitting reduction, certain new paths will be generated at each decoding stage, and L_v goes to list size L quickly. Thus, its complexity can be approximated by

$$C_w \approx \sum_{v \in Z_n} N_v L. \tag{23}$$

For the proposed algorithm, there is no new path generated for most decoding stages and L_v increases slowly in the proposed scheme. For a high SNR, $L_v < L$ until the decoding is over. Hence, $C_p < C_w$.

As the SNR goes to infinity, no new path will be generated and L_v=1 for the whole decoding process. Thus, $\lim_{\mathrm{SNR}\to\infty} C_p = C_w/L$. Hence, C_w/L is the lower bound for computational complexity of the proposed algorithm.

5 Simulation results and analysis

Simulation results for an additive white Gaussian noise (AWGN) channel and binary phase-shift keying (BPSK) modulated code words are presented in this section. The 32-bit CRC and code rate R=1/2 are applied for all simulations. The code rate in this work does not include the CRC bits. For example, the (1024, 544) polar code contains 512 valid information bits and 32 CRC bits, thus, the code rate here is 1/2. All the following results are compared with the FSSCL decoding algorithm in [9].

The computational complexity is measured according to (22). The average computational complexity per bit, defined as C_{av}, is used in the simulation results, and

$$C_{av} = \frac{\sum_{v \in Z_n} N_v L_v}{K}.$$

Figure 3 shows the block error rate (BLER) performance of different algorithms for list size L=32, code length N=1024. It can be seen that the BLER decreases with the decrease of ρ for each of the cases. The BLER of the proposed algorithm for $\rho = 0.3$ is very close to that of the FSSCL decoder for L=32. Hence, there is little degradation in error performance for an appropriate ρ.

Figure 4 shows the average computational complexities of different algorithms for different ρ with L=32, N=1024. The reduction of the average complexity is not remarkable at a low level SNR, but complexity decreases rapidly as the SNR increases, because less new paths would be generated as the decoding reliability increases. The complexity is reduced by more than 60% when the SNR is larger than 3 dB, and by more than 90% for 5 dB. The complexity reduction increases with the increase of ρ because the path splitting threshold would be small for a large ρ and more path splitting operations would be reduced. However, a larger ρ may lead to a larger block error rate. Hence, it is important to set a proper ρ for complexity-performance trade-off.

Figure 5 shows the error performance of different algorithms for different node lengths with L=8, ρ=0.3. Figure 6 shows the corresponding average complexities under the same condition. The results show that for short block length N=256 and moderate block length N=1024, the proposed algorithm shows nearly the same BLER performance as the FSSCL and has a slight degradation on BLER for N=4096 when the SNR is very high. The average complexities are reduced considerably for different block lengths, and the reduction is more remarkable with the increase of SNR.

Figure 7 shows the average computational complexity comparison of different decoding algorithms for different list sizes for N=1024, ρ=0.3. Figure 8 shows the corresponding error performance under the same condition. The results show that the proposed algorithm shows nearly the same BLER performance as

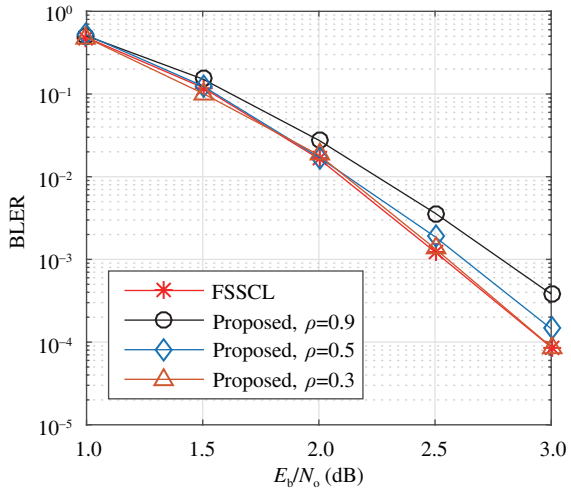

Figure 3 (Color online) BLER performance of different ρ for N=1024, L=32.

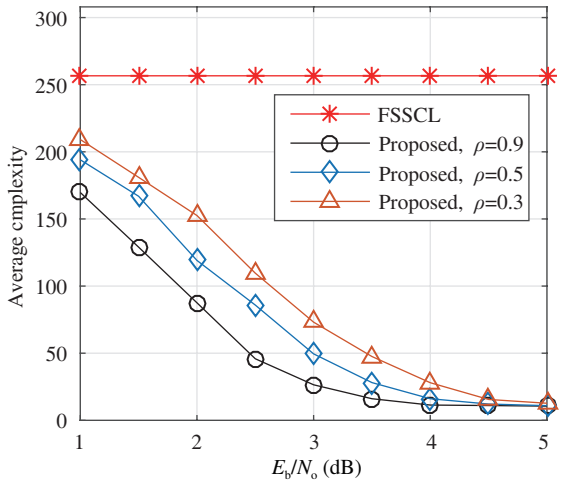

Figure 4 (Color online) Average complexity of different ρ for N=1024, L=32.

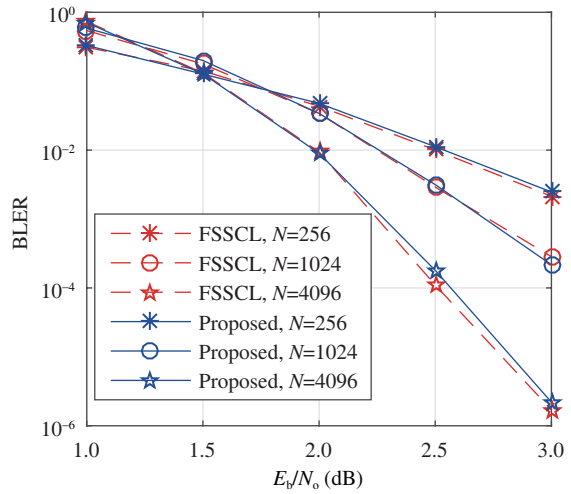

Figure 5 (Color online) BLER performance for L=8, N=256, 1024 and 4096.

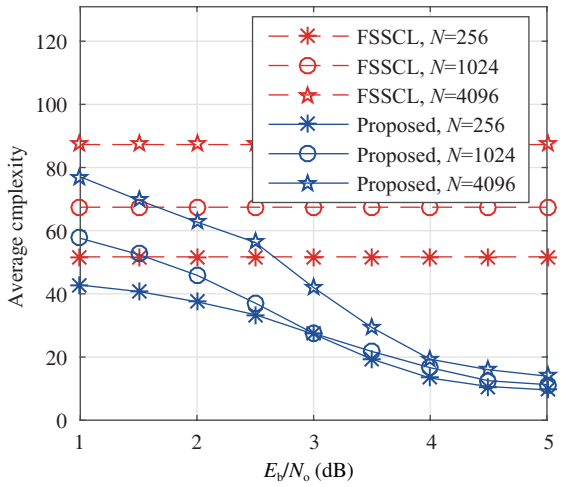

Figure 6 (Color online) Average complexity for L=8, N=256, 1024 and 4096.

the FSSCL for different list sizes. The proposed algorithm can reduce complexity effectively for different list sizes, especially for a large list size. It should be noted that, regardless of list size, the complexity becomes very close to that of the FSSC decoder without a list algorithm at a high SNR region. Hence, when the SNR is sufficiently high, the complexity of the proposed method is $1/L$ of the FSSCL decoder, where L is the list size.

6 Conclusion

In this paper, we proposed an improved list decoding scheme for polar codes that can significantly reduce the computational complexity. Path splitting operations for the multi-bit decision can be reduced if the decoding reliability exceeds a threshold. A scheme is proposed for determining path splitting thresholds for both leaf and non-leaf nodes. Then, path splitting rules are designed for different node types; and a complexity reduced list decoder is proposed based on this. The number of survival paths can be reduced greatly based on the proposed scheme, thus, the computational complexity can be significantly reduced. Results show that the complexity decreases greatly at cost of negligible deterioration in block

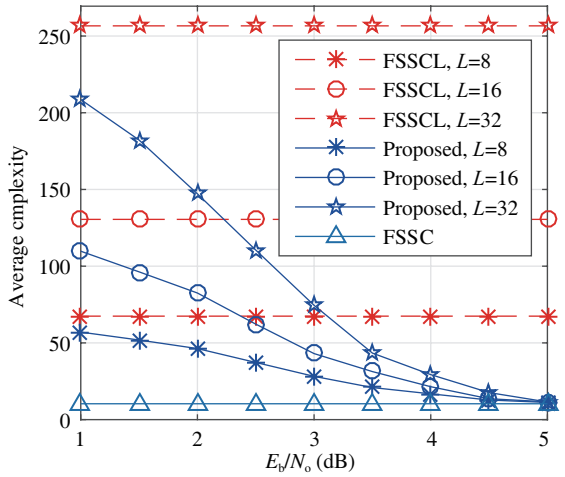

Figure 7 (Color online) Average complexity for N=1024, L=8, 16, 32.

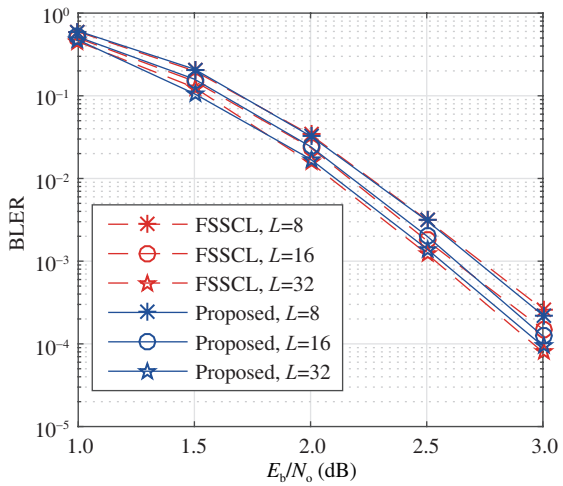

Figure 8 (Color online) BLER performance for N=1024, L=8, 16, 32.

error performance, and becomes close to that of the FSSC decoder in a high SNR condition.

Acknowledgements This work was partially supported by National Major Project (Grant No. 2016ZX030010-11005), National Natural Science Foundation Project (Grant No. 61521061), and Intel Corporation.

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Arikan E. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. IEEE Trans Inf Theory, 2009, 55: 3051–3073
2 Arikan E, Costello D J, Kliewer J, et al. Guest editorial recent advances in capacity approaching codes. IEEE J Sel Areas Commun, 2016, 34: 205–208
3 Alamdar-Yazdi A, Kschischang F R. A simplified successive-cancellation decoder for polar codes. IEEE Commun Lett, 2011, 15: 1378–1380
4 Sarkis G, Giard P, Vardy A, et al. Fast polar decoders: algorithm and implementation. IEEE J Sel Areas Commun, 2014, 32: 946–957
5 Tal I, Vardy A. List decoding of polar codes. IEEE Trans Inf Theory, 2015, 61: 2213–2226
6 Niu K, Chen K. CRC-aided decoding of polar codes. IEEE Commun Lett, 2012, 16: 1668–1671
7 Trifonov P, Miloslavskaya V. Polar codes with dynamic frozen symbols and their decoding by directed search. In: Proceedings of IEEE Information Theory Work, Sevilla, 2013. 1–5
8 Sarkis G, Giard P, Vardy A, et al. Increasing the speed of polar list decoders. In: Proceedings of IEEE Work Signal Process Syst, Belfast, 2014. 1–6
9 Sarkis G, Giard P, Vardy A, et al. Fast list decoders for polar codes. IEEE J Sel Areas Commun, 2016, 34: 318–328
10 Li B, Shen H, Tse D. An adaptive successive cancellation list decoder for polar codes with cyclic redundancy check. IEEE Commun Lett, 2012, 16: 2044–2047
11 Chen K, Li B, Shen H, et al. Reduce the complexity of list decoding of polar codes by tree-pruning. IEEE Commun Lett, 2016, 20: 204–207
12 Zhang Z, Zhang L, Wang X, et al. A split-reduced successive cancellation list decoder for polar codes. IEEE J Sel Areas Commun, 2016, 34: 292–302
13 Yuan B, Parhi K K. LLR-based successive-cancellation list decoder for polar codes with multi-bit decision. IEEE Trans Circuits Syst II Express Briefs, 2016, 64: 21–25
14 Tal I, Vardy A. How to construct polar codes. IEEE Trans Inf Theory, 2013, 59: 6562–6582
15 Wu D, Li Y, Sun Y. Construction and block error rate analysis of polar codes over AWGN channel based on Gaussian ap-proximation. IEEE Commun Lett, 2014, 18: 1099–1102