An LLR-Based Segmented Flipped SCL Decoding Algorithm for Polar Codes

Anqi Cao, Li Zhang, Jian Qiao, and Yejun He*
Guangdong Engineering Research Center of Base Station Antennas and Propagation
Shenzhen Key Laboratory of Antennas and Propagation
College of Electronics and Information Engineering, Shenzhen University, 518060, China
Email: 1319433421@qq.com, wzhangli@szu.edu.cn, 446941582@qq.com, heyejun@126.com*

***Abstract*—Polar codes have attracted significant attention in the field of communication, for the capacity-achieving under binary-input discrete memoryless channels (BDMCs). On the basis of successive cancellation (SC) decoding algorithm, many excellent methods have been developed to improve the decoding performance on short and medium codes. In this paper, an LLR-based segmented flipped successive cancellation list (LLR-based SF-SCL) decoding algorithm is proposed, where the information bits are divided into segments for early bit-flipping and termination. In order to make the unreliable bits evenly distributed in each sub-block, an LLR-based segmentation scheme is applied. Simulation results show that the proposed segmented flipped algorithm is superior to CRC-aided successive cancellation list (CA-SCL) algorithm in the block error rate (BLER) performance, and LLR-based segmentation scheme has better performance compared to uniform segmentation scheme.**

***Index Terms*—Polar codes, SCL decoding, segmented list, successive cancellation flip (SCF), log-likelihood ratio (LLR).**

I. Introduction

Polar codes were introduced by Arikan with a low complexity decoder (SC decoder) [1]. SC decoding algorithm is the most classical decoding method for polar codes with computational complexity O($Nlog(N)$), where N is the blocklength of the code. Under SC decoding, the Shanon capacity limit can be achieved when the code length N is long enough. However, the decoding performance for short codes is not satisfactory. Therefore, the SCL decoding was presented to improve decoding performance by reserving multiple paths [2]. The CRC (cyclic redundancy check)-aided SCL (CA-SCL) decoding algorithm [3] greatly reduces the BLER of list decoding. However, the computational complexity of list decoding is O($LNlogN$), which is L times that of the SC decoding, where L is the number of reserved paths.

Afisiadis attempted to correct the first error bit with the successive cancellation flip (SCF) algorithm [4], which brings better performance. However, at low signal-to-noise ratios (SNR), the computational complexity of SCF decoding is greatly increased. A new metric to select the candidate bits for flipping was proposed in [5], which reduces the computational complexity and improves the decoding performance. A fixed index selection (FIS) scheme [6] was presented to avoid the computational complexity of sorting the metrics. Nevertheless, the computational complexity brought by flip decoding is still high.

In order to reduce the complexity of list decoding, split-reduced SCL (SR-SCL) algorithm [7] and CRC-aided SR-SCL algorithm [8] reduce the splitting operation in decoding. The segmented CRC-aided SCL (SCA-SCL) algorithm [9] segments the information bits into sub-blocks to delete the wrong decoding paths in time. The segmented scheme reduces the computational complexity without reducing the performance of list decoding.

The contributions of this paper are summarized as follows.

1) The proposed LLR-based segmented flipped SCL decoding algorithm can delete error decoding paths and conduct flip ahead. The algorithm includes the following four steps: a) Segment the information bits, where we segment the information bits according to the LLR distribution which makes the low-reliability channel index evenly distributed in each sub-block. b) Perform SCL decoding for each segment, c) Perform CRC detecting, where we delete the wrong paths in time. d) Perform flip decoding for the failed decoding, where we select the path with the minimum path metric for flipping. If the flip failed, the decoding terminates and the decoding fails. Otherwise, the decoding of the next segment continues.

2) The average decoding list size of LLR-based SF-SCL algorithm at large L values is smaller than L. We compare the LLR-based SF-SCL decoding algorithm with the referenced uniform segmented flipped SCL decoding algorithm in error correction performance and computational complexity. The LLR-based SF-SCL algorithm has better error correction performance and lower computational complexity compared to the referenced uniform SF-SCL algorithm at low SNR.

The remainder of this paper is organized as follows. Section II is a review of polar codes and the related decoding algorithms. Section III describes the SF-SCL decoding based on uniform segmentation. Section IV describes the LLR-based SF-SCL decoding algorithm. Section VI shows the simulation results. Section V draws the conclusions.

II. Preliminaries

A. Polar Codes

Denote the binary memoryless channel by input $x \in \{0,1\}$, output $y \in \mathcal{Y}$, and transition probabilities $W(y|x)$. The information vector is $u_1^N = (u_1, u_2, u_3, ..., u_N)$, the encoded vector is $x_1^N = (x_1, x_2, x_3, ..., x_N)$, and the received vector is $y_1^N = (y_1, y_2, y_3, ..., y_N)$.

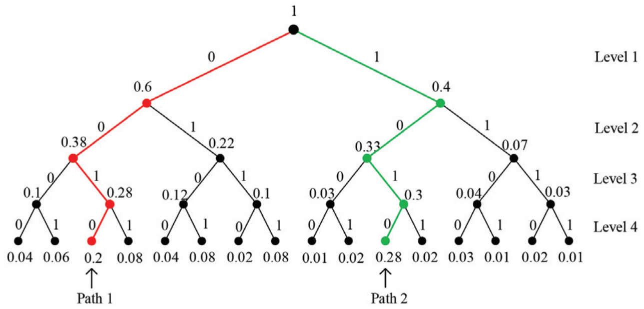

Fig. 1. Decoding tree of SCL algorithm.

The essence of the process of polarization encoding is the process of channel polarization. N BDMCs are converted into $N = 2^n$ polarized subchannels by channel combination and channel separation, where n is a positive integer. The transition probability of the i-th subchannel is $W_N^{(i)}(y_1^N, u_1^{i-1}|u_i), i = 1, 2, ..., N$. According to the reliability of polarization, K channels with higher reliability are selected to transmit information bits. The information indices are denoted by $\mathcal{A}$. Due to incomplete polarization, the channels with the higher error probability are selected to transmit frozen bits, which are set to 0. At this point, the transmission rate is $R = K/N, 0 < K < N$ [10] - [11].

B. SC and SCL Decoding Algorithm

The SC decoding algorithm is a local optimal algorithm. During the decoding process, an optimal decoding result is reserved for each decoding node. At each node, for frozen bits, decoding output is 0; for information bits, the log-likelihood ratio (LLR) of the bit channel [12] is applied:

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

where $\hat{u}_i$ is the estimate of u_i. The decisions are made according to:

$$\hat{u}_i = \begin{cases} 0 & \text{if } L_N^{(i)} \geq 0, \\ 1 & \text{otherwise.} \end{cases} \quad (2)$$

Different from SC decoding algorithm, SCL decoding algorithm does not make decisions immediately. SCL algorithm retains L optimal candidate paths by updating the path metric (PM). After all the bits have been judged, it selects the best path output. As shown in Fig. 1, SC decoding output path 1 and SCL decoding output path 2 for $L = 2$, where both paths may be correct. In order to determine the correct decoding path, the CRC detector is a good decision maker in CA-SCL decoder. If there is no path survival, the decoding path with the smallest metric is retained. The path metrics can be updated by setting:

$$PM_l^{(i)} = \phi(PM_l^{(i-1)}, L_N^{(i)}[l], \hat{u}_i[l]), \quad (3)$$

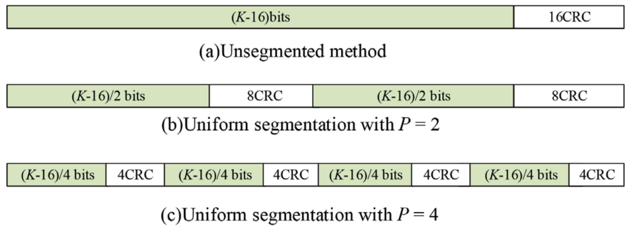

Fig. 2. Segmentation schemes of SCA-SCL decoder.

where

$$\phi(\mu, \lambda, u) = \begin{cases} \mu & \text{if } u = \frac{1}{2}[1 - sign(\lambda)], \\ \mu + |\lambda| & \text{otherwise.} \end{cases} \quad (4)$$

$L_N^{(i)}[l]$ is the LLRs for path l.

C. SCF Decoding Algorithm

SCF decoding algorithm attempts to flip a decision to get the correct decoding when the conventional SC decoding fails. Afisiadis found that SC decoding errors are mostly from error propagation, which is caused by incorrect $\hat{u}_i$ and their partial sums [4]. Since error propagation is caused by an erroneous decision, it is crucial to find the first error bit in SCF decoding.

SCF algorithm first finds the T most likely erroneous bit decisions then tries to find the first error bit. The LLRs of the SC decoding are retained in SCF algorithm. The T least reliable $\hat{u}_i$ with the smallest $|L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}|u_i)|$ values are selected, and their indexes are put into the flip set $\mathcal{U}$. Then it tries to flip the bits in the set in turn. After the F bit flipped, SC decoding is restarted for the following codes. If the path still cannot pass the CRC detector, the flip is restored and the index is deleted from the set. Then the next bit is flipped until the set $\mathcal{U}$ is empty. SCF decoding algorithm corrects a portion of the decoding error and lowers the BLER for polar decoding.

D. SCA-SCL Decoding Algorithm

In order to reduce the computational complexity of the CA-SCL algorithm, SCA-SCL decoding algorithm terminates the error decoding ahead. As shown in Fig. 2, the information bits are uniformly divided into P segments in SCA-SCL decoding algorithm. CRC bits were added after each segment. After each segment is decoded, CRC detectors delete the paths that could not pass. If no path is left, the decoding is declared failed to reduce unnecessary computation.

III. Segmented Flipped SCL Decoding Algorithm

This section presents a simple segmented flipped SCL decoding algorithm for reference with the following proposed algorithm. The uniform segmentation scheme is applied in this referenced algorithm.

According to what Afisiadis described in [4], channel noise and error propagation are the main causes of decoding errors. Although the flipping of the first error bit can effectively avoid error propagation, it is not easy for the decoder to find the first error bit in such long codes, especially when the signal-to-noise ratio is low. In SF-SCL algorithm, the information bits

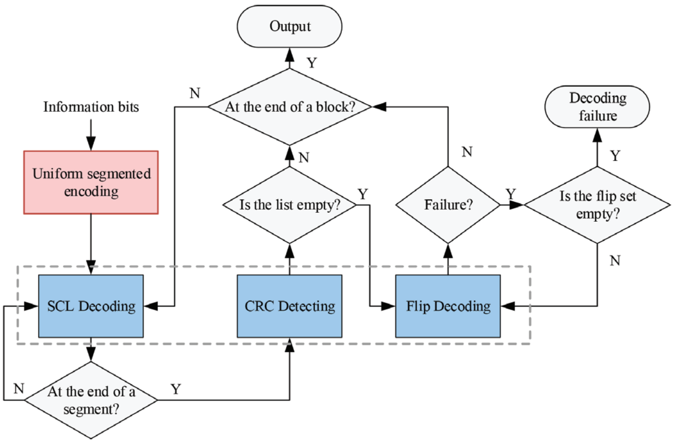

Fig. 3. Flow chart of SF-SCL decoding algorithm.

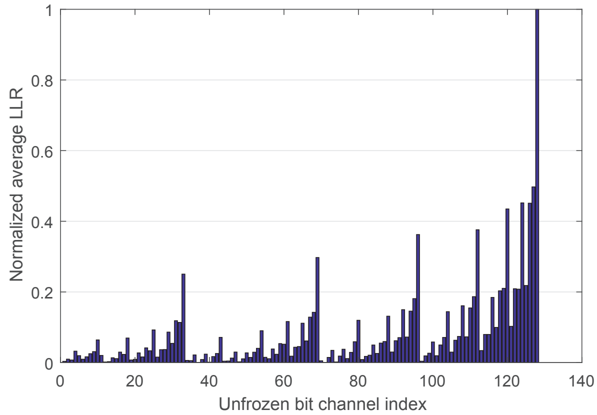

Fig. 4. Normalized average LLR diagram at unfrozen bits with N = 256, R = 0.5, E_b/N_0 = 1.

are divided into P segments, where we named Referenced 1 (SF-SCL, P=2) and Referenced 2 (SF-SCL, P=4). Each segment is checked and flipped separately in the SF-SCL decoding.

In the SF-SCL decoding algorithm, L paths are retained before CRC detecting. If the CRC detecting fails, the algorithm selects a path from the reserved L paths for flipping. The path with the minimum PM_l was chosen in our method. After a decoding path has been selected, SF-SCL decoding algorithm defines a flip set $\mathcal{U}$ consisting of $\hat{u}_i$ with the T_{max} smallest absolute LLRs $|L_N^{(i)}|$ for each sub-block. Because the sub-blocks are flipped separately in the SF-SCL decoding, the chance of flipping is increased and the complexity is decreased. For example, the SCF algorithm takes N bits of decoding computation for flipping once, while SF-SCL algorithm only takes N/P bits of decoding computation.

As shown in Fig. 3, SF-SCL decoding has three operating modes, namely “SCL Decoding”, “CRC Detecting” and “Flip Decoding”. The operating process is as follows.

Step 1: Evenly divide the information bits into P segments and add CRC bits respectively, conduct polarization encoding and send the encoded codes to the decoder.

Step 2: Perform SCL decoding until decoding the last CRC bit for a segment.

Step 3: Perform CRC detecting. If the list is empty, go to Step 4. Otherwise, keep the passed paths and discard other candidate paths; repeat Step 2 and Step 3 for the next segment until all segments are decoded.

Step 4: Perform flip decoding. The path with the minimum PM_l is selected for flipping. The flip set is defined by $\hat{u}_i$ with T_{max} smallest $|L_N^{(i)}(y_1^N, \hat{u}_1^{i-1}|u_i)|$ values of the failed sub-block. If the flipping is successful, reserve the path and repeat Step 2 and Step 3 for next segment until all the codes are decoded. If the flip failed, terminate the decoding and declare the decoding failed. In each flip attempt, a single bit in the set $\mathcal{U}$ is flipped and the following codes are re-decoded. For avoiding too much computational complexity, the SC algorithm is applied after the flipped bit.

TABLE I

THE AVERAGE NUMBER OF FLIP INDEXES DISTRIBUTED IN EACH SEGMENT

E_b/N_0 (dB)	1		2		3	
Segment	S_1	S_2	S_1	S_2	S_1	S_2
Average number	11.31	3.69	10.76	4.23	10.15	4.85

IV. The Proposed LLR-Based SF-SCL Decoding Algorithm

An LLR-based SF-SCL algorithm is proposed in this section. A new segmentation scheme is introduced to improve the probability of successful flipping.

Figure 4 shows the normalized average absolute LLR $|L_N^{(i)}|$ of unfrozen bits. The $|L_N^{(i)}|$ value of the latter channel is relatively high. When $P = 2$, the information bits are evenly divided into two segments (S_1, S_2) in the referenced SF-SCL decoding. Table I is the average number of flip indexes for SCF decoding that distributed in the two segments. Simulation runs 10^4 times with N = 256, R = 0.5, and the size of the flip set T_{max} = 15. Each segment has $N/4$ information bits. Table I shows that the index distributed in S_2 is much less than that in S_1. The absolute value of LLR is used to measure the decoding reliability of the channel. Uniform segmentation makes the low-reliability channel more distributed in the first segment. In order to make the unreliable bits evenly distributed in each sub-block, an LLR-based segmentation scheme is applied in segmented flipped SCL decoding.

In the LLR-based segmentation scheme, there are T_{max} flip opportunities for each segment. When $P = 2$ and the maximum number of flip T_{max} = 15, there are a total of 30 flip opportunities for the 2 sub-blocks. Denote an unreliable set Γ consisting of $\hat{u}_i$ with the $P \times T_{max}$ smallest absolute LLRs. In Fig. 5, it is shown the distribution diagram for unfrozen bit indices in set Γ. The segmentation position for LLR-based SF-SCL decoding with $P = 2$ is also shown in Fig. 5, so that each sub-block contains 15 bits in set Γ. In this way, the unreliable estimation can be more evenly distributed in each segment.

The distribution of low-reliability channel index needs to

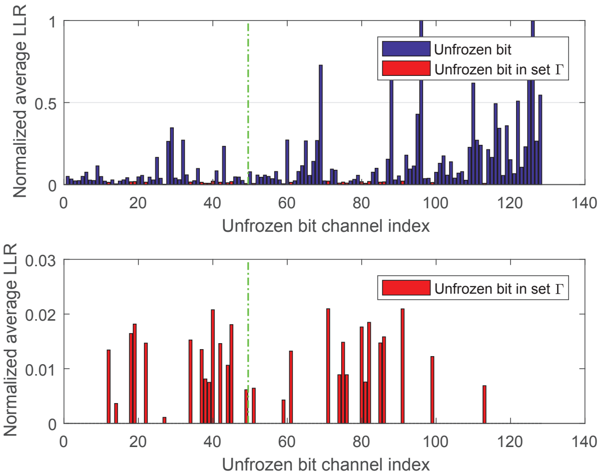

Fig. 5. Normalized average LLR distribution diagram of unfrozen bits. Simulation runs 10^4 times with $N = 256, R = 0.5, E_b/N_0 = 1, P = 2, T_{max} = 15$. Indexes highlight in red correspond to the bits in set Γ.

be counted before LLR-based encoding. Then the appropriate segmentation positions need to be selected to make the low-reliability channel index evenly distributed in each sub-block. The segmented CRC bits are added before the segmentation positions. With the segmentation scheme determined, the following decoding process is similar to the uniform segmentation scheme.

Algorithm 1 explains the LLR-based SF-SCL decoding where $Seg(k)$ is the index of the last bit of S_k. SCL($y_1^N, \mathcal{A}, k$) function performs SCL decoding for the k-th segment. The flip set is also defined by $\hat{u}_i$ with T_{max} smallest $|L_N^{(i)}|$. SC($y_1^N, \mathcal{A}, \mathcal{U}(j)$) function performs SC decoding with the value of $\hat{u}_{\mathcal{U}(j)}$ is flipped.

As we all know, it is very easy for the decoder to find the first error bit in the short sub-block, and multiple error bits could be corrected in the segmented flipped SCL scheme. The LLR-based segmentation avoids distributing error bits densely in the same segment, resulting in the increased probability of successful flipping. Since LLR distribution is not uniform, the segmentation position of LLR-based segmentation is generally in front of the uniform segmentation, which can easily find the first error bit. In terms of computational complexity, flipping in a smaller segment can reduce the computational costs for sorting and re-decoded in each flip attempt. If the flip decoding fails, the decoding terminates earlier to reduce the redundant computation.

V. Experimental Results and Performance Analysis

In this section, we compare the proposed LLR-based segmented flipped decoding algorithm with the referenced SF-SCL algorithm and other related algorithms in error correction performance and computational complexity. The maximum number of flip of SCF algorithm and segmented flipped SCL algorithm is set as $T_{max} = 15$. CRC-8 and CRC-16 generator

Algorithm 1 LLR-based SF-SCL Decoding

Input: $y_1^N, \mathcal{A}, T_{max}, P$;
Output: $\hat{u}_1^N$
$\mathcal{L} \leftarrow \{0\}$;
$PM_0^{(0)} \leftarrow 0$;
$flag \leftarrow 0$;
for each $k \in [1, P]$ **do**
  $(\hat{u}_{Seg(k)+1}^{Seg(k+1)}[l], L(u_i)[l], PM_l^{(k)}) \leftarrow$ SCL($y_1^N, \mathcal{A}, k$) for $\forall l \in \mathcal{L}$;
  for $\forall l \in \mathcal{L}$ **do**
    if CRC($\hat{u}_{Seg(k)+1}^{Seg(k+1)}[l]$)=success **then**
      $flag \leftarrow 1$;
    else
      deletepath(l);
    end if
  end for
  if $flag = 0$ **then**
    $m \leftarrow l$ of smallest $PM_l^{(k)}$;
    $\hat{u}_1^{Seg(k+1)} \leftarrow \hat{u}_1^{Seg(k+1)}[m]$;
    $\mathcal{U} \leftarrow i \in \mathcal{A}$ of T_{max} smallest $|L(y_1^N, \hat{u}_1^{i-1}|u_i)[m]|$;
    for each $j \in [1, T_{max}]$ **do**
      $data_{Seg(k)+1}^{Seg(k+1)} \leftarrow$SC($y_1^N, \mathcal{A}, \mathcal{U}(j)$);
      if CRC($data_{Seg(k)+1}^{Seg(k+1)}$)=success **then**
        $\hat{u}_{Seg(k)+1}^{Seg(k+1)} \leftarrow data_{Seg(k)+1}^{Seg(k+1)}$;
        $flag \leftarrow 1$;
        break;
      end if
    end for
  end if
  if $flag = 0$ **then**
    $\hat{u}_1^N \leftarrow \emptyset$;
    break;
  end if
end for
return $\hat{u}_1^N$;

polynomials are given by [12]

CRC-8:
$$g(x) = x^8 + x^7 + x^6 + x^4 + x^2 + 1 \quad (5)$$

CRC-16:
$$g(x) = x^{16} + x^{15} + x^2 + 1 \quad (6)$$

A. Error Correction Performance

The LLR-based SF-SCL algorithm is simulated and compared with the SCF, the CA-SCL, the SCA-SCL algorithms and the uniform SF-SCL scheme in BLER performance. The CRC-16 is selected for the non-segment schemes, and CRC-8 is selected for schemes with $P = 2$ and $P = 4$. In the case of flip decoding, since each bit-flipping may lead to big changes in the decoding path, it is prone to undetected errors. Therefore, we select CRC-8 instead of CRC-4 for schemes with P = 4. Simulations are under the AWGN channel in the range of $E_b/N_0 = 0.5$ dB to $E_b/N_0 = 3$ dB.

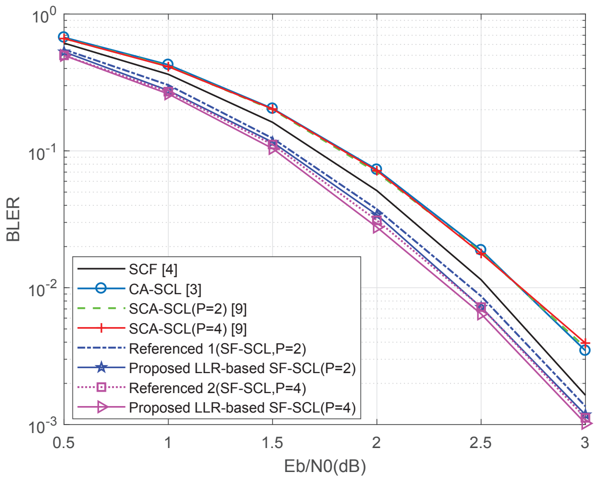

Fig. 6. Comparison of proposed algorithm and SCF algorithm, CA-SCL algorithm, SCA-SCL algorithm and referenced SF-SCL algorithm with N=256, R=0.5, L=2.

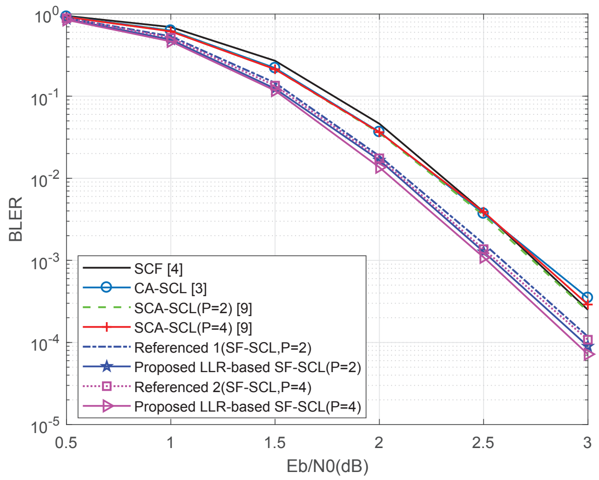

Fig. 7. Comparison of proposed algorithm and SCF algorithm, CA-SCL algorithm, SCA-SCL algorithm and referenced SF-SCL algorithm with N=1024, R=0.5, L=2.

BLER performance of different algorithms with N=256 is shown in Fig. 6. The proposed LLR-based SF-SCL decoding algorithm has corrected some errors in CA-SCL decoding, and the performance of LLR-based SF-SCL algorithm is better than that of SCF algorithm. The decoding performance of SCA-SCL algorithm with P = 4 is almost the same as that of SCA-SCL algorithm with P = 2. The decoding performance of SCA-SCL algorithm is similar to that of CA-SCL algorithm which is consistent with the results discussed in [9]. The performance of SF-SCL decoding algorithms with P = 4 (namely Referenced 2) is better than that of SF-SCL decoding decoding algorithms with P = 2 (namely Referenced 1). The proposed LLR-based SF-SCL decoding algorithm has lower BLER compared to referenced SF-SCL algorithm with the same P. The LLR-based SF-SCL algorithm with P = 4 demonstrates a 0.34 dB performance gain over the CA-SCL algorithm at BLER of 10^{-2}.

Figure 7 compares the BLER performance of different algorithms with N=1024. LLR-based SF-SCL decoding algorithm with P = 4 has the lowest BLER. The BLER of LLR-based SF-SCL algorithm is lower than that of referenced uniform SF-SCL algorithm with the same P. The proposed LLR-based SF-SCL algorithm with P = 4 demonstrates a 0.27 dB performance gain over the CA-SCL algorithm at BLER of 10^{-3}.

Figure 8 compares the BLER performance of the proposed LLR-based SF-SCL algorithm and CA-SCL algorithm with different list size L. The BLER performance of LLR-based SF-SCL algorithm is better than that of CA-SCL algorithm at the same L. The bigger the L is, the better the BLER performance is.

B. Computational Complexity

Since the computational complexity of SC list decoding is proportional to the average list size, the average list size is usually used to describe the computational complexity [9]. The list size is represented as

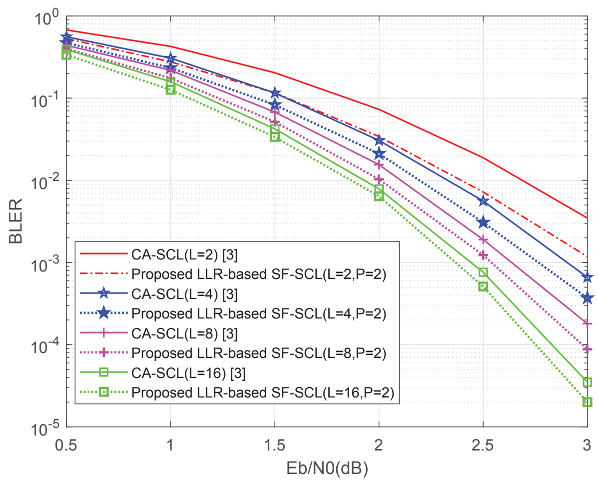

Fig. 8. Comparison of the proposed LLR-based SF-SCL algorithm and SCL algorithm with N=256, R=0.5.

$$\mathcal{L} = L\frac{\mathcal{K}}{P} + \sum_{j=1}^{\mathcal{T}} L_{flip}(j) \tag{7}$$

where $\mathcal{K}$ is the number of decoded segment, $\mathcal{T}$ is the number of flip decoding, and $L_{flip}(j)$ is the list size of j-th flip decoding which is defined as b/N, b is the number of re-decoded bits. If the codes decoded successfully, $\mathcal{K}$ is equal to P, then the average list size without the flipping part is L. The computational complexity with $\mathcal{L}$ =1 is $O(N\log N)$.

The relative average list size $\mathcal{L}_a$ is defined as

$$\mathcal{L}_a = \mathcal{L}_{avg} - L \tag{8}$$

where $\mathcal{L}_{avg}$ is the average list size.

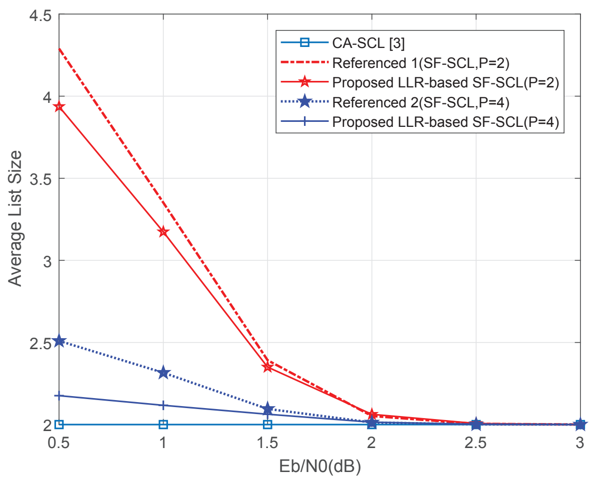

Fig. 9. Average list size of different segmentation schemes with N=1024, R=0.5, L=2.

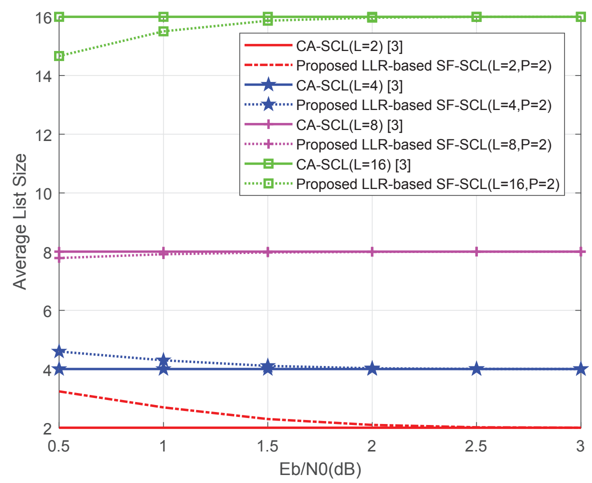

Fig. 10. Average list size of LLR-based SF-SCL decoding with N=256, R=0.5.

Figure 9 shows the average list size of different segmentation schemes with $N = 1024$, $L = 2$. The average list size of LLR-based SF-SCL algorithm is smaller than that of referenced SF-SCL algorithm. The average list sizes of both LLR-based SF-SCL algorithm and referenced SF-SCL algorithm drop below 2.5 at E_b/N_0 = 1.5, and the average list sizes are very close to 2 at E_b/N_0 = 2. When $P = 4$, the average list size of LLR-based segmented scheme is 2.17 at E_b/N_0 = 0.5. It is obvious that more segments result in the lower computational complexity.

Figure 10 shows the average list size of LLR-based SF-SCL decoding with different L. When $L < 8$, the average list size of LLR-based SF-SCL decoding is slightly greater than that of CA-SCL decoding. When $L \geq 8$, the average list size of LLR-based SF-SCL decoding is smaller than that of CA-SCL decoding at low E_b/N_0. Simulation results indicate that the large L will reduce the relative average list size of the proposed LLR-based SF-SCL algorithm, even make the average list size smaller than L.

VI. Conclusion

In this paper, a segmented flipped SCL decoding algorithm including the LLR-based segmentation method is proposed to correct the decoding error bits. Simulation results demonstrate that the proposed algorithm obtains better decoding performance compared to the CA-SCL algorithm, and the LLR-based SF-SCL decoding algorithm has lower BLER and smaller average list size compared to uniform SF-SCL decoding algorithm. Due to the early termination of the error decoding, the average list size of LLR-based SF-SCL decoding is smaller than that of CA-SCL decoding algorithm with the large L.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grants 61372077, 61801299, and 61871433, in part by the Shenzhen Science and Technology Programs under Grants ZDSYS201507031550105, JSGG20180507183215520, JCYJ20170302150411789, JCYJ20170302142515949, and GCZX 2017040715180580 and GJHZ 20180418190529516, and in part by the Guangzhou Science and Technology Program under Grant 201707010490.

References

[1] E. Arikan, “Channel Polarization: A Method for Costructing Capacity-Achieving Codes for Symmetic Binary-Input Memoryless Channels,” *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051-3073, Jul.2009.
[2] I. Tal and A. Vardy, “List decoding of polar codes,” *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213-2226, May. 2015.
[3] K. Niu and K. Chen, “CRC-Aided Decoding of Polar codes,” *IEEE Communications Letters*, vol. 16, no. 10, pp. 1668-1671, Oct. 2012.
[4] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, “A low-complexity improved successive cancellation decoder for polar codes,” in *Proc. IEEE 48th Asilomar Conference on Signals, Systems and Computers*, pp. 2116-2120, Nov. 2014.
[5] L. Chandesris, V. Savin, and D. Declercq, “An improved SCFlip decoder for polar codes,” in *Proc. IEEE Global Communications Conference (GLOBECOM)*, pp. 1-6, Dec. 2016.
[6] Carlo Condo, Furkan Ercan, and Warren J. Gross, “Improved successive cancellation flip decoding of polar codes based on error distribution,” in *Proc. IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 19-24, Apr. 2018.
[7] Z. Zhang, L. Zhang, X. Wang, C. Zhong and H. V. Poor, “A Split-Reduced Successive Cancellation List Decoder for Polar Codes,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 2, pp. 292-302, Feb. 2016.
[8] Li Zhang, Anqi Cao, Jian Qiao and Yejun He, “A CRC-Aided SR-SCL Decoding Algorithm,” in *Proc. IEEE MTT-S International Wireless Symposium (IWS)*, May 2019, in press.
[9] H. Zhou, C. Zhang, W. Song, S. Xu, and X. You, “Segmented CRC-aided SC list polar decoding,” in *Proc. IEEE Vehicular Technology Conference*, pp. 1-5, May 2016.
[10] K. Chen, B. Li, H. Shen, J. Jin and D. Tse, “Reduce the Complexity of List Decoding of Polar Codes by Tree-Pruning,” *IEEE Communications Letters*, vol. 20, no. 2, pp. 204-207, Feb. 2016.
[11] Z. Huang, C. Diao, J. Dai, C. Duanmu, X. Wu and M. Chen, “An Improvement of Modified Successive-Cancellation Decoder for Polar Codes,” *IEEE Communications Letters*, vol. 17, no. 12, pp. 2360-2363, December 2013.
[12] A. Balatsoukas-Stimming, M. B. Parizi, and A. Burg, “LLR-based successive cancellation list decoding of polar codes,” *IEEE Trans. Signal Processing*, vol. 63, no. 19, pp. 5165-5179, Oct. 2015.