2023 32nd Wireless and Optical Communications Conference (WOCC) | 979-8-3503-3715-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/WOCC58016.2023.10139571

Improved Stack Decoding for PAC Codes

Li Zhang, Haina Liu, and Yejun He⋆
Guangdong Engineering Research Center of Base Station Antennas
Shenzhen Key Laboratory of Antennas and Propagation
College of Electronics and Information Engineering, Shenzhen University, 518060, China
Email: wzhangli@szu.edu.cn, qq970033251@163.com, heyejun@126.com⋆

***Abstract*—Several classical decoding algorithms, such as Fano decoding, list decoding and list Viterbi decoding algorithm have been proposed for polarization-adjusted convolutional (PAC) codes by some scholars. Inspired by the decoding algorithms mentioned above, two algorithms called stack Viterbi decoding (SVD) and critical set-aided stack decoding are presented in this paper. Simulation results show that performance of the proposed algorithm with stack depth of 100 slightly outperforms that of the conventional stack decoding algorithm with the same stack depth at the cost of higher complexity, and the sorting complexity of conventional stack decoding (CSD) can be further reduced by means of the proposed critical set-aided stack decoding algorithm.**

***Index Terms*—Polarization-adjusted codes, stack decoding, Viterbi decoding**

I. Introduction

Polarization-adjusted convolutional (PAC) codes proposed by Arikan are concatenated codes of convolution codes and polar codes [1]. The entire coding scheme of PAC codes illustrates that convolutional codes serve as an outer code and polar codes serve as an inner code. According to Arikan's perspective, polar codes has no error correction ability due to rate 1 of the inner code. Therefore, PAC codes are often considered as an irregular convolutional codes transmitted through the polarized channels which are created by polar transforms, instead of a type of concatenated codes. With the help of polarized channels, the performance of convolutional codes can be further improved.

Although tree codes of convolutional codes in PAC codes are irregular, it can also be decoded in the same way as regular tree codes. It can be shown that with the help of the sequential decoder, the curve of performance for PAC codes can get close to the binary-input additive white Gaussian noise (BIAWGN) dispersion bound approximation curve. Besides, Arikan also revealed that PAC codes can outperform polar codes with cyclic redundancy check (CRC) width size of 8 and list size of 32 [1]. Sequential decoding such as Fano algorithm was used to decode PAC codes, which had low space complexity at the cost of unbearable time complexity at low Signal to Noise Ratio (SNR) [1]. Later, as a typical decoding algorithm in polar codes, list decoding was employed to PAC codes in [2], which showed an outstanding performance near to the Fano decoding with the moderately large list size (e.g., list size=128). Its constant decoding complexity is an advantage at low SNR regime but become a disadvantage at high SNR regime compared to sequential decoding algorithms. Enlightened by the existing algorithms in convolutional codes, Rowshan applied Viterbi algorithm (VA) and list Viterbi algorithm (LVA) to PAC codes, and showed that compared with the global sorting strategy used in [2], LVA has a significantly lower sorting complexity [3].

The main contributions of this paper are as follows.

- Motivated by the LVA in [3] and the stack decoding in [4], an algorithm referred as stack Viterbi decoding algorithm (SVD) is proposed in this work.
- With the help of the critical set (CS) proposed in [5], we present a critical set-aided stack algorithm to further reduce the sorting complexity.

Similar to the conventional stack decoding algorithm in convolutional codes, SVD maintains a fixed stack depth. That is to say, SVD works in the same way as conventional convolutional codes until the maximum depth of the stack is reached. When the maximum depth is reached, conventional stack algorithm deletes the path located at the bottom of the stack after sorting while SVD would regard each of the states in trellis as a stack, and then operate each state as a conventional stack, namely, sorting the paths in each stack and then deleting the one at the bottom of the stack. Numerical results show that the proposed SVD with stack depth of 100 slightly outperforms the conventional stack decoding algorithm with the same stack depth at the cost of higher complexity, and the proposed critical set-aided stack decoding algorithm forms a split-reduced behavior so that the sorting complexity can be further reduced.

II. PRELIMINARIES

A. PAC Codes

PAC code is usually characterized by four parameters $(N, K, \mathcal{A}, \mathbf{c})$, where $N = 2^n$ $(n \geq 1)$ represents the block length of PAC codes, K represents the number of information bits, $\mathcal{A}$ is the index set of information bits with the cardinality $|\mathcal{A}| = K$, and c is an impulse response (a.k.a generator polynomial). Fig. 1 shows an overall coding scheme of PAC codes. As shown in Fig. 1, a source word d is firstly fed into the rate-profiling block. In accordance with the rule of rate profiling block, d is inserted into a data carrier word v so that $\mathrm{v}_{\mathcal{A}} = \mathrm{d}$ and $\mathrm{v}_{\mathcal{A}^c} = 0$. Then, convolution output is obtained by

$$\mathrm{u}_1^N = \mathrm{v}_1^N \mathrm{G}_\mathrm{c}, \tag{1}$$

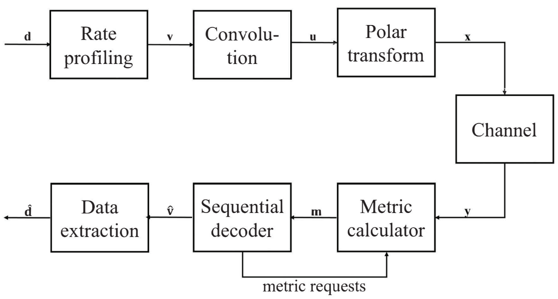

Fig. 1. PAC coding scheme.

where $\mathrm{G_c}$ is the generator matrix of convolutional codes with the form of upper-triangular Toeplitz. Alternatively, convolution operation is characterized by an impulse response $\mathrm{c} = (c_0, ..., c_m)$, where the elements in vector c corresponds to the entries in matrix $\mathbf{T}$. By convention, we often assume that $c_0 \neq 0$ and $c_m \neq 0$. The parameter $m+1$ is known as the constraint length of the convolutional codes. The convolution output can also be written as

$$u_i = \sum_{j=0}^{m} c_j v_{i-j}, \tag{2}$$

where $v_{i-j} = 0$ for $j \geq i$. Immediately, codeword u_1^N is transformed into the codeword x_1^N by the polar transform mentioned above and then transmitted over a noisy channel.

At the side of receiver, polar demapper and convolutional demapper collaborate to decode the estimated data carrier $\hat{\mathrm{v}}$. Polar demapper (often a successive cancellation demapper) calculates soft message (often the channel log-likelihood ratios) according to the received signal y and the feedback bit $\hat{u}_i$. Utilizing the soft message, convolutional demapper gives an estimated data carrier $\hat{\mathrm{v}}$ bitwise and the feedback bit $\hat{u}_i$. The decoding process continues in this way until $\hat{\mathrm{v}}$ is determined or a predefined stopping rule is triggered. Finally, the message extraction block recover $\hat{\mathrm{d}}$ from $\hat{\mathrm{v}}$ in accordance with the rule set by rate-profiling block.

B. Stack Decoding

Niu applied stack decoding algorithm to polar codes, and named it as successive cancellation stack (SCS) decoding algorithm in [4]. The results showed that SCS with stack depth 100 had performance comparable with the successive cancellation list (SCL) with list size 20 under the (256, 128) polar code and the binary input additive white Gaussian noise (BI-AWGN) channel. However, in [4], the path metric was $M(d_1^i) = \log W_N^{(i)}(y_1^N, d_1^{i-1}|d_i)\ i \in \mathcal{A}$, which meant that paths of different lengths were compared in an unfair way. SCS must have a moderately large stack depth to compensate the unfairness. In [6], a more reasonable path metric for comparing paths of different lengths is named as Fano metric given by

$$\log P\left(\mathbf{a}^{(\ell)}|\tilde{\mathbf{r}}\right) = \sum_{i=0}^{nn_\ell - 1} \underbrace{(\log P\left(r_i|a_i^{(\ell)}\right)}_{\text{ML-metric}} \underbrace{- \log P\left(r_i\right) - R}_{\text{path-length bias}}). \tag{3}$$

Note that the channel seen by convolutional codes is memoryless, whereas the channel for polar codes is a polarized channel with memory. The formula (3) needs to be modified slightly to adapt for stack decoding of PAC codes. Moradi investigated the sequential decoding metric function of PAC codes in [7]. The branch metric for ith position is given by

$$\gamma\left(u_i; \mathbf{y}_1^N, \mathbf{u}_1^{i-1}\right) = \begin{cases} 1 - \log_2\left(1 + \frac{1}{z_i}\right) - b_i, & \text{if } u_i = 0; \\ 1 - \log_2\left(1 + z_i\right) - b_i, & \text{if } u_i = 1. \end{cases} \tag{4}$$

where $z_i := \frac{P(y_1^N, u_1^{i-1}|u_i=0)}{P(y_1^N, u_1^{i-1}|u_i=1)}$ is the likelihood ratio and its logarithmic form is known as log-likelihood ratio (LLR), and b_i is a bias term. In this paper, we take $b_i = 1.35$ for $i \in \mathcal{A}$ and $b_i = 0$ for $i \notin \mathcal{A}$.

III. Improved Stack Decoding Algorithm

Viterbi algorithm is an optimal decoding algorithm for convolutional codes while its complexity is 2^m which grows exponentially with the number of states. It is suggested in [8] that one can obtain better performance with longer constraint length. But it is unfeasible due to the prohibitive memory consumption. To overcome the conflict above, a list-type Viterbi algorithm was introduced in [9], where instead of one path, L paths with smallest path metric were preserved and extended at each decoding step. Even if algorithms mentioned above were designed for convolutional codes, both of them can be applicable for PAC codes due to the similarity of code structure. In [3], Rowshan noted the similarity between list Viterbi decoding and list decoding of PAC codes, namely, list Viterbi decoding sorting paths locally at each state (paths often dispersed at different state) while list decoding sorting paths globally. Actually, Rowshan regarded each state as a list with size $L = L_G/2^m$. In this paper, we consider each state as a stack as shown in Fig. 2.

A. Stack Viterbi Decoding

Algorithm 1 illustrates the stack Viterbi algorithm. In the beginning of the algorithm, there is only one empty path in the stack. At each decoding iteration, one path is popped from the stack. When the current position of the popped path is not in the set $\mathcal{A}$, the decoder knows its value and takes the value of $\mathcal{A}[i]$ (usually $\mathcal{A}[i] = 0$). Depending on the current state S and the generator polynomial $\mathbf{g}$ of the popped path, $\hat{v}_i$ is encoded into $\hat{u}_i$. Then, by the use of the LLR computed in line 31, one can calculate the path metric. Note that the LLR used in this paper is in the form of λ_n^i, where the subscript n and the superscript i denote the nth stage and the ith bit channel,

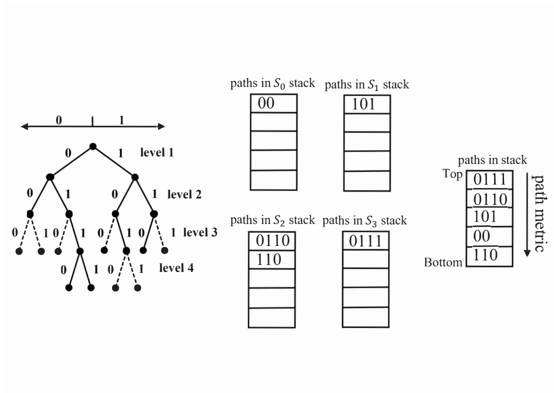

Fig. 2. Example of SVD for $m = 2$.

respectively, and the γ is the branch metric introduced in (4). After updating the partial sums and renewing the indicator T for stack depth and the current position i, the path will be pushed into the stack and the paths in the stack will be sorted in an ascending order. On the contrary, if the current position of the popped path is in the set $\mathcal{A}$, the algorithm will make two different decisions. If $T \leq D - 2$ (D is the maximum depth of the stack), the algorithm will behave same as the conventional stack decoding. Otherwise, the path popped previously will be pushed into the stack, and then arrange the paths in stack according to the state of the paths. At each of states, the most promising path will be extended and duplicated depending on whether the current position of the most promising path is in $\mathcal{A}$ or not, and the variable pathToBePruned is used to count the number of the paths to be pruned. Whenever a new path is spawned, it will be pushed into the stack. Finally, the paths in the stack will be sorted in an ascending order, and the stack Π retains last $|\Pi|$ – pathToBePruned paths. Note that the subroutines *updateLLRs updatePartialSums* are identical to the ones used in SC decoding, and *conv1bEnc* is identical to the one used in [3].

B. Critical Set-Aided Stack Decoding

A novelty notion named as critical set was introduced in [5]. Briefly speaking, critical set assures that when the first incorrectly estimated information bit comes out, it will be included in the critical set with a high probability, even for low SNR. Also, with the oracle-assisted decoder defined in [10], one incorrectly estimated bit accounts for the majority when a frame error occurs, and it is meaningful to correct the error bits due to the channel noise rather than the error bits inflicted by the error propagation. Consequently, we propose a decoding algorithm called critical set-aided stack decoding herein. Without loss of generality and for the purpose of the simplicity, the conventional stack decoding is used in the algorithm.

Algorithm 2 details the process. The proposed algorithm does the same thing as the conventional stack decoding when

Algorithm 1: Stack Viterbi Decoding of PAC Codes.

Input: $\mathcal{A}$, D, $\mathbf{g}$, $\lambda_n^{1,N}$
Output: estimated vector $\hat{\mathbf{d}}$

$\Pi \leftarrow \{\phi\}$
$m \leftarrow |\mathbf{g}| - 1$
$i \leftarrow 1$, $T \leftarrow 1$
while $i \neq N + 1$ **do**
  $\pi \leftarrow$ pop(Π), $i \leftarrow$ getCurrentPosition(π);
  $T \leftarrow T - 1$;
  if $i \in \mathcal{A}$ **then**
    if $T \leq D - 2$ **then**
      $\pi' \leftarrow$ copy(π);
      $\lambda_0^i[\pi] \leftarrow$ updateLLRs(π, i, $\lambda[\pi]$, $\beta[\pi]$);
      $[\hat{v}_i[\pi], \hat{v}_t[\pi']] \leftarrow (0, 1)$;
      $[\hat{u}_i[\pi]$, S$[\pi]] \leftarrow$ conv1bEnc($\hat{v}_i[\pi]$, S$[\pi]$, $\mathbf{g}$);
      $[\hat{u}_i[\pi']$, S$[\pi']] \leftarrow$ conv1bEnc($\hat{v}_t[\pi']$, S$[\pi]$, $\mathbf{g}$);
      $M_i(\pi) \leftarrow M_{i-1}(\pi) + \gamma(\hat{u}_i[\pi])$;
      $M_i(\pi') \leftarrow M_{i-1}(\pi) + \gamma(\hat{u}_i[\pi'])$;
      $\beta[\pi] \leftarrow$ updatePartialSums($\hat{u}_i[\pi]$, $\beta[\pi]$);
      $\beta[\pi'] \leftarrow$ updatePartialSums($\hat{u}_i[\pi']$, $\beta[\pi]$);
      $\Pi \leftarrow$ push(π, π');
      $T \leftarrow T + 2$, $(i, i') \leftarrow (i + 1, i' + 1)$;
      $\Pi \leftarrow$ sort(Π)
    else
      $\Pi \leftarrow$ push(π);
      $T \leftarrow T + 1$;
      pathToBePruned $\leftarrow 0$;
      for $s \leftarrow 1$ ***to*** 2^m **do**
        extend the most promising path at each state and increase the pathToBePruned according to the $\mathcal{A}$;
        $\Pi \leftarrow$ push(newPath);
      $\Pi \leftarrow$ sort(Π);
      $\Pi \leftarrow \Pi$[pathToBePruned : $|\Pi|$]
  else
    $\lambda_0^i[\pi] \leftarrow$ updateLLRs(π, i, $\lambda[\pi]$, $\beta[\pi]$);
    $\hat{v}_i[\pi] = \mathcal{A}[i]$;
    $[\hat{u}_i[\pi]$, S$[\pi]] \leftarrow$ conv1bEnc($\hat{v}_i[\pi]$, S$[\pi]$, $\mathbf{g}$);
    $M_i(\pi) \leftarrow M_{i-1}(\pi) + \gamma(\hat{u}_i[\pi])$;
    $\beta[\pi] \leftarrow$ updatePartialSums($\hat{u}_i[\pi]$, $\beta[\pi]$);
    $T \leftarrow T + 1$, $i \leftarrow i + 1$;
    $\Pi \leftarrow$ push(π);
    $\Pi \leftarrow$ sort(Π)
$\hat{\mathbf{d}} \leftarrow$ extractData($\hat{v}_1^N[-1]$, $\mathcal{A}$)
return $\hat{\mathbf{d}}$

the index of the popped path is not in the set $\mathcal{A}$. On the other hand, the proposed algorithm will judge further whether the index is in the critical set $\mathcal{CS}$ or not. If $i \in \mathcal{CS}$, the function *stackDecoding* performs the extension and the duplication of the path π, just like the conventional stack decoding algorithm. Otherwise, the proposed algorithm just extends the current path π to the path corresponding to the larger path metric, and records $\hat{v}_i[\pi]$, $\hat{u}_i[\pi]$ and the next state S$[\pi]$.

IV. Numerical Results

In this section, the error correction performance and the decoding complexity of the algorithm 1 are illustrated and analyzed. The effect of the algorithm 2 is also shown in this section. The rate profiling rule used in this paper is RM-Poalr [11]. The modulation applied to the codewords is binary phase shift keying (BPSK) and the codewords are transmitted over

Algorithm 2: Critical Set-Aided Stack decoding of PAC Codes.

Input: $\mathcal{A}$, D, $\mathbf{g}$, $\lambda_n^{1,N}$, $\mathcal{CS}$
Output: estimated vector $\hat{\mathbf{d}}$
$\Pi \leftarrow \{\phi\}$
$m \leftarrow |\mathbf{g}| - 1$
$i \leftarrow 1$, $T \leftarrow 1$
while $i \neq N+1$ **do**
  $\pi \leftarrow$ pop(Π), $i \leftarrow$ getCurrentPosition(π);
  $T \leftarrow T - 1$;
  if $i \in \mathcal{A}$ **then**
    if $i \in \mathcal{CS}$ **then**
      stackDecoding(π, i, $\mathbf{g}$)
    else
      $\lambda_0^i[\pi] \leftarrow$ updateLLRs(π, i, $\lambda[\pi]$, $\beta[\pi]$);
      $[\hat{v}_0[\pi], \hat{v}_1[\pi]]$ = (0, 1);
      $[\hat{u}_0[\pi]$, S$_0[\pi]] \leftarrow$ conv1bEnc($\hat{v}_0[\pi]$, S$[\pi]$, $\mathbf{g}$);
      $[\hat{u}_1[\pi]$, S$_1[\pi]] \leftarrow$ conv1bEnc($\hat{v}_1[\pi]$, S$[\pi]$, $\mathbf{g}$);
      $M_0(\pi) \leftarrow M_{i-1}(\pi) + \gamma(\hat{u}_0[\pi])$;
      $M_1(\pi) \leftarrow M_{i-1}(\pi) + \gamma(\hat{u}_1[\pi])$;
      $M_i(\pi) \leftarrow$ max($M_0(\pi)$, $M_1(\pi)$);
      $t \leftarrow$ argmax($M_0(\pi)$, $M_1(\pi)$);
      $[\hat{v}_i[\pi], \hat{u}_i[\pi]$, S$[\pi]] \leftarrow [\hat{v}_t[\pi], \hat{u}_t[\pi]$, S$_t[\pi]]$;
      $\beta[\pi] \leftarrow$ updatePartialSums($\hat{u}_i[\pi]$, $\beta[\pi]$);
      $T \leftarrow T+1$, $i \leftarrow i+1$;
      $\Pi \leftarrow$ push(π);
      $\Pi \leftarrow$ sort(Π)
  else
    $\lambda_0^i[\pi] \leftarrow$ updateLLRs(π, i, $\lambda[\pi]$, $\beta[\pi]$);
    $\hat{v}_i[\pi] = \mathcal{A}[i]$;
    $[\hat{u}_i[\pi]$, S$[\pi]] \leftarrow$ conv1bEnc($\hat{v}_i[\pi]$, S$[\pi]$, $\mathbf{g}$) ;
    $M_i(\pi) \leftarrow M_{i-1}(\pi) + \gamma(\hat{u}_i[\pi])$;
    $\beta[\pi] \leftarrow$ updatePartialSums($\hat{u}_i[\pi]$, $\beta[\pi]$);
    $T \leftarrow T+1$, $i \leftarrow i+1$;
    $\Pi \leftarrow$ push(π);
    $\Pi \leftarrow$ sort(Π)
$\hat{\mathbf{d}} \leftarrow$ extractData($\hat{v}_1^N[-1]$, $\mathcal{A}$)
return $\hat{\mathbf{d}}$;

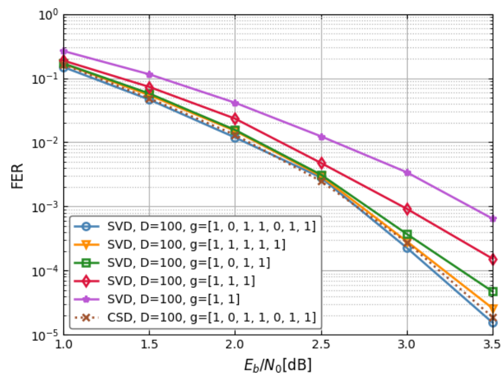

Fig. 3. FER comparison under SVD with various generator polynomials.

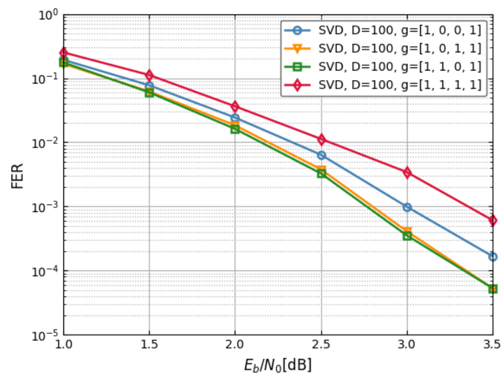

Fig. 4. FER comparison under SVD with various fixed-length generator polynomials.

the BI-AWGN. List decoding of PAC codes is denoted by L_G in this paper. Frame error rate (FER) is used to measure the error correction performance. The decoding complexity and the sorting complexity are measured by counting the average node visitings in the code tree (which is known as ANV) and the average running time, respectively. Fig. 3 illustrates the performance of the stack Viterbi decoding (denoted by SVD) under different generator polynomials $\mathbf{g}$. Note that the changes in generator polynomials $\mathbf{g}$ both have impact on encoding procedure and decoding procedure. Usually, more previous bits participate in convolution with the increase of the $\mathbf{g}$. It can be seen from the Fig. 3 that our proposed algorithm slightly outperforms the conventional stack decoding algorithm (denoted by CSD). Furthermore, Fig. 4 compares the performance of the stack Viterbi decoding under four kinds of generator polynomial. It can be seen that the performance of various generator polynomials varies from each other. Although generator polynomial $\mathbf{g}$ has an impact on the performance of PAC codes, there is no a systematic method to identify a good generator polynomial so far. Fig. 5 shows the comparison of performance of SVD with LVA. It can be seen from the Fig. 5 that the stack Viterbi decoding has a comparable performance than the list Viterbi decoding under certain generator polynomial $\mathbf{g}$ (e.g., $\mathbf{g} = [1,1,1]$), while with a worse performance under certain generator polynomial $\mathbf{g}$ (e.g., $\mathbf{g} = [1,1]$). Besides, we can see from the Fig. 3 and Fig. 5 that the stack Viterbi decoding outperforms the list decoding (also outperforms the list Viterbi decoding with $\mathbf{g} = [1,1,1]$ and $L = 32$) under certain generator polynomial $\mathbf{g}$ (e.g., $\mathbf{g} = [1,0,1,1,0,1,1]$).

Fig. 6 and Fig. 7 reveal complexity under different algorithms, where generator polynomials used in SVD, CSD, CS-Aided and list decoding are $\mathbf{g} = [1,0,1,1,0,1,1]$ and $\mathbf{g} = [1,1,1]$ for LVA. Note that as the list decoding of PAC codes, LVA has a constant decoding complexity regardless of the SNR, while SVD and conventional stack decoding have a varied decoding complexity due to the property of the sequential decoding. And our proposed algorithm slightly outperforms the conventional stack decoding at the cost of a larger complexity. Besides, although the proposed complexity-reduced algorithm almost has no difference with the conventional stack algorithm in

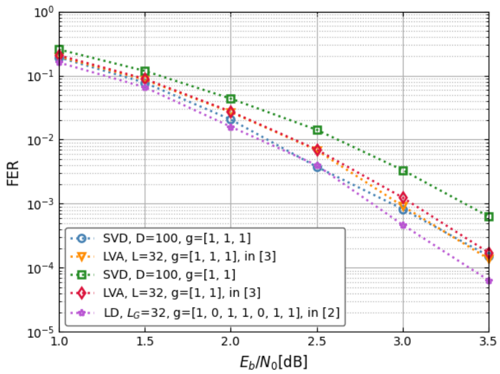

Fig. 5. FER comparison under SVD, LVA and list decoding of PAC codes (denoted by L_G).

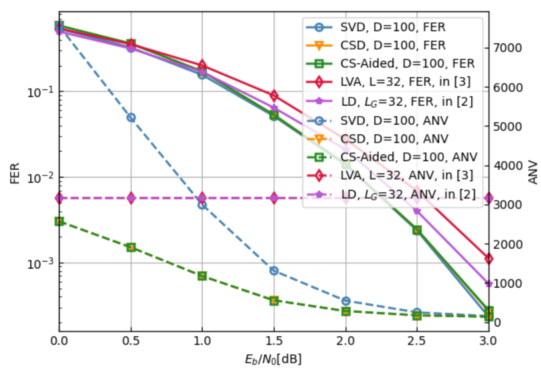

Fig. 6. FER (solid line) and ANV (dotted line) comparison under various algorithms.

terms of ANV, it can still reach our purpose and a following qualitative analysis is made to clarify it. Recall that except for the decoding complexity, the sorting complexity is also included in the overall complexity. In our proposed algorithm, a path only forks when the index of current bit is in the critical set, which means that the number of paths in the stack is usually less than the conventional stack algorithm. Hence, less number of paths is equivalent to the smaller sorting complexity. With respect to the average running time of the proposed algorithm, the simulation result confirms our assumption.

V. Conclusion

In this paper, we proposed a special variation of conventional stack decoding algorithm named stack Viterbi decoding and the critical set-aided stack decoding algorithm. We studied the performance and the complexity of the SVD, and compared it with other decoding algorithms (e.g., LVA). Simulation results showed that SVD with $D = 100$ slightly outperformed the conventional stack decoding algorithm with the same stack depth at the cost of higher complexity. Besides, to verify

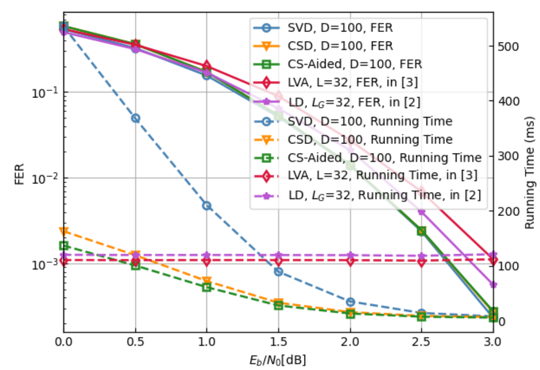

Fig. 7. FER (solid line) and running time (dotted line) comparison under various algorithms.

the effect of the proposed complexity-reduced algorithm, we investigated the complexity behavior of the conventional stack decoding algorithm under the proposed algorithm. The result showed that the overall complexity can be further reduced by decreasing the sorting complexity while exhibiting almost no degradation in the performance.

Acknowledgment

This work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant No. 62071306, and in part by Shenzhen Science and Technology Program under Grants JCYJ20200109113601723, JSGG20210420091805014, JSGG20210802154203011.

References

[1] E. Arıkan, "From sequential decoding to channel polarization and back again." *arXiv preprint arXiv*:1908.09594 (2019).
[2] H. Yao, A. Fazeli, and A. Vardy, "List decoding of Arikans PAC codes." in *Proc. IEEE International Symposium on Information Theory (ISIT)*, 2020, pp. 443-448.
[3] M. Rowshan and E. Viterbo, "List Viterbi decoding of PAC codes." *IEEE Transactions on Vehicular Technology*, vol. 70, no. 3, pp. 2428-2435, Mar. 2021.
[4] K. Niu and K. Chen, "Stack decoding of polar codes." *Electronics letters* vol. 16, no. 10, Oct. 2012, pp. 1668-71.
[5] Z. Zhang, et al, "Progressive bit-flipping decoding of polar codes over layered critical sets." in *Proc. GLOBECOM 2017-2017 IEEE Global Communications Conference*, Singapore, 2017, pp. 1-6.
[6] R. Fano, "A heuristic discussion of probabilistic decoding." *IEEE Transactions on Information Theory*, vol. 9, no. 2, pp. 64-74, Apr. 1963.
[7] M. Moradi, "On sequential decoding metric function of polarization-adjusted convolutional (PAC) codes." *IEEE Transactions on Communications*, vol. 69, no. 12, pp. 7913-7922, Dec. 2021.
[8] T.K.Moon, *Error correction coding: mathematical methods and algorithms*. New Jersey, NJ, USA: John Wiley & Sons, 2005, pp 451-525.
[9] T. Hashimoto, "A list-type reduced-constraint generalization of the Viterbi algorithm." *IEEE Transactions on Information Theory*, vol. 33, no. 6, pp. 866-876, Nov. 1987.
[10] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, "A low-complexity improved successive cancellation decoder for polar codes." in *Proc. 48th Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, USA, 2014, pp. 2116-2120.
[11] B. Li, H. Shen, and D. Tse, "A RM-polar codes." *arXiv preprint arXiv*:1407.5483 (2014).