Journal of Systems Engineering and Electronics
Vol. 33, No. 4, August 2022, pp.856 – 866

Improved encoding structure and decoding algorithms for spinal codes

HUANG Wensha[1] and WANG Lina[1,2,*]

1. School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China;
2. Shunde Graduate School, University of Science and Technology Beijing, Beijing 100083, China

Abstract: To improve the error correction performance, an innovative encoding structure with tail-biting for spinal codes is designed. Furthermore, an adaptive forward stack decoding (A-FSD) algorithm with lower complexity for spinal codes is proposed. In the A-FSD algorithm, a flexible threshold parameter is set by a variable channel state to narrow the scale of nodes accessed. On this basis, a new decoding method of AFSD with early termination (AFSD-ET) is further proposed. The AFSD-ET decoder not only has the ability of dynamically modifying the number of stored nodes, but also adopts the early termination criterion to curtail complexity. The complexity and related parameters are verified through a series of simulations. The simulation results show that the proposed spinal codes with tail-biting and the AFSD-ET decoding algorithms can reduce the complexity and improve the decoding rate without sacrificing correct decoding performance.

Keywords: spinal code, tail-biting encoding, lower decoding complexity, early termination strategy.

DOI: 10.23919/JSEE.2022.000084

1. Introduction

Spinal codes, as a new kind of rateless codes, were proposed by Perry et al. in 2012 [1]. Different from the traditional channel encoding algorithms, the key idea of spinal codes is that a similar structure of convolutional code [2,3] and an irreversible hash function to generate pseudorandom sequences are adopted. Compared with the fixed-rate codes such as turbo codes [4,5] and low-density parity check (LDPC) codes [6,7], rateless codes can adapt to the dynamic channel and generate infinite modulation symbols to send continuously without feedback [6]. As shown in [8,9], spinal codes have been proved that they can achieve the capacities of both additive white Gaussian noise (AWGN) channel and the binary symmetric channel (BSC) with short message length and pseudorandom like codewords.

Perry mentioned that the error bits of spinal codes are mainly concentrated in several symbols at the end of information [1]. Yu et al. showed that spinal codes have the potential unequal error protection (UEP) property [10]. To solve the high decoding error, a novel encoding structure was proposed by adding tail bits to the head of source message. The protection of tail bits was strengthened, thereby improving the bit-error-rate (BER). The effect of tail-biting on the original spinal codes is similar to adding tail symbols of original sequences, which can improve the overall error-control performance. The difference is that the former is added to the head, but the latter is added to the tail. The conclusion showed that the addition of extra tail symbols gains the higher reliability.

Based on the maximum likelihood (ML) decoding, the bubble decoding algorithm effectively reduces the decoding complexity [11,12]. Yang et al. [13] proposed a lower complexity algorithm named forward stack decoding (FSD) algorithm by dividing the decoding tree into several layers. When the channel state fluctuates greatly, the FSD algorithm searches more invalid nodes, which is not ideal for reducing the complexity. Therefore, an adaptive FSD (A-FSD) algorithm is proposed in this paper. According to different channel states, a specific threshold which is inversely proportional to the signal to noise ratio (SNR) is designed to control the number of storage nodes. The A-FSD algorithm can dynamically adjust the number of node search through the real-time channel state, so as to adaptively optimize the decoding complexity.

According to different search criteria, tree search strategies can be divided into depth-first tree search and breadth-first tree search [14,15]. Different from the breadth-first tree search, the depth-first tree search starts from a certain node and accesses its first subnode. If the subnode meets certain conditions, it goes to the next layer and searches the first node of this subnode, and then it

Manuscript received August 25, 2020.

*Corresponding author.

This work was supported by the National Natural Science Foundation of China (61701020) and the Scientific and Technological Innovation Foundation of Shunde Graduate School, USTB (BK19BF009).

searches down to the leaf node according to the search criteria. The depth-first search algorithm has the feature of implicit early termination (ET). The ET criterion [16,17] can be described in a decoding tree as searching for a node along a path, terminating the search for a node whose path metric is greater than a specified metric value, returning to the beginning and continuing the search along the next path. For spinal codes, there are many improved algorithms in encoding and decoding. Hu et al. designed a block dynamic decoding (BDD) algorithm to facilitate reliable data transmission [18]. Xu et al. adapted a sliding feedback decoding (SFD) algorithm, which uses a sliding window to locate a layer for local decoding [19]. Chen et al. proposed a multiplicative repetition-based scheme instead of the hash function and designed both frozen-aided and cyclic redundancy check-aided decoding algorithms to improve the spectral efficiency [20]. However, the speed of reaching the end of the decoding tree is slow and the complexity is still high. It can be noted that most of the algorithms are carried out from the breadth-first tree search. Inspired by depth-first search and ET strategy, an AFSD-ET algorithm is designed to further optimize the decoding computation based on the A-FSD scheme. The new algorithm divides the tree into two groups. Meanwhile, in the process of node detection, the number of storage nodes is determined by setting a threshold level, and the ET criterion is adopted to stop the decoding process immediately. It can flexibly adjust the path of decoding search to reduce the number of nodes to be calculated. Through theoretical analysis and simulations, the AFSD-ET algorithm is capable of decreasing the complexity and accelerating the efficiency of searching the correct path.

The rest of this paper is organized as follows. Section 2 describes the tail-biting encoding structure. In Section 3, based on the FSD decoding method, the lower decoding complexity schemes including A-FSD and AFSD-ET are presented. And then the complexity of four decoding algorithms is discussed. The performance of the proposed algorithms is evaluated through a series of simulations in Section 4. And the conclusions are drawn in Section 5.

2. Tail-biting spinal codes

2.1 Tail-biting encoding structure

The encoding process is improved on the original structure as follows:

(i) Coding block segmentation

Divide the *n*-bits message M into $d=n/k$ segments which are consisted of k non-overlapping bits, i.e., $M = m_1, m_2, m_3, \cdots, m_d$.

(ii) Setting tail-biting parameters Δ

The last Δ segments of the block M are added to the head of original sequences, and the order of the blocks added in the head is the same as tail blocks, so it is called tail-biting encoding structure. Suppose that two information block to the header are added, setting up $\Delta = 2$, and the new sequence is $M' = m_{d'-1}, m_{d'}, m_1, \cdots, m_{d-1}, m_d$ with the same head and tail obtained, among which $m_{d'-1} = m_{d-1}$, $m_{d'} = m_d$.

(iii) Coded transmission

The initial state value s_0 and the first information segment of the new sequence M' are input into the hash function $h(\cdot)$ to generate the next state value. Then the state value and the next segment are sent to the hash function, and the process is carried out successively until all information is processed by $h(\cdot)$. Eventually, a total of $d+\Delta$ state values are generated. The hash function formula is given as follows:

$$\begin{cases} s_i = h(s_{i-1}, m_i) \\ s_0 = 0^v \end{cases}. \tag{1}$$

The initial hash value s_0 is known by both the encoder and the decoder, and it is assumed that s_0 is the zero set of v-dimension. Then, these states are used as seeds for the random number generator (RNG) [21,22] and multi-batch output generates a sequence of c-bit pseudo-random numbers. The RNG formula is given as

$$\text{RNG}: \{0,1\}^v \times L \to \{0,1\}^c$$

where L is the number of passes.

Finally, the sequence of c-bit is mapped into the encoded symbol $x_{i,j}$ suitable for channel transmission, where i is the ith state value and j is the batch sequence of symbols.

The tail-biting, also known as cyclic encoding, ensures that the initial and termination states of each component encoder are the same [23,24]. In previous research, tail-biting convolutional codes could overcome the rate loss caused by initializing the coder with known bits [25], and the tail-biting encoding structure could solve the existing error-level phenomenon in turbo codes [26].

Fig. 1 illustrates the encoding structure of tail-biting for spinal codes with $\Delta = 2$. The decoding process of tail-biting structure is roughly the same as the original process, only the optimal path is determined when decoding output is different. Assume that the decoder uses the bubble algorithm, and the original spinal codes select B candidate paths reserved in the decoding tree extending to the last level. After tail-biting treatment, when selecting the optimal decoding output from B candidate paths, the reliability of the paths is sorted first, and the path information corresponding to the paths with the same decoding results of the top Δ segments and the last Δ segments are selected as the decoding results. If the B paths do not meet the condition requirement, the highest reliability is selected as the optimal path, and the top Δ segments replace the last Δ information segments as the final decoding result (Δ is a positive integer).

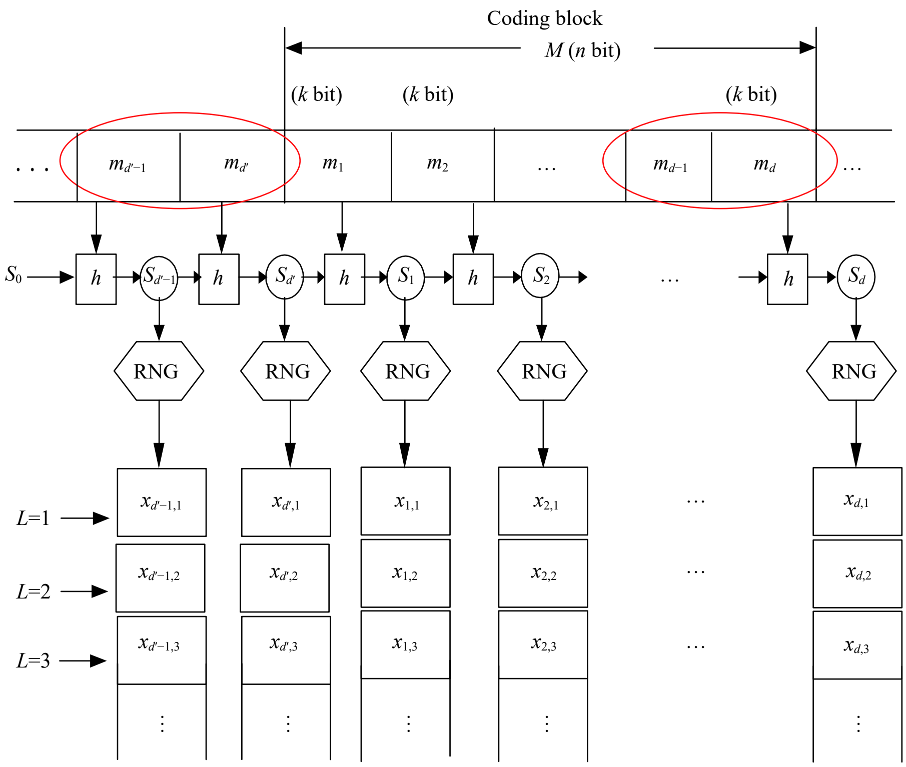


Fig. 1 Encoding structure of tail-biting for spinal codes

2.2 Error probability analysis and comparison

The main purpose of tail-biting encoding structure is to improve the transmission reliability, so the performance is compared through calculating the symbol error rate of the proposed and the original structure. According to [10], the block error rate of the original spinal codes is calculated, where $\hat{M} = M$ represents the probability that the input information sequence M is the same as the decoded output sequence $\hat{M}$.

$$\begin{aligned} P_e = 1 - P\left(\hat{M} = M\right) = \\ 1 - P(\hat{m}_1 = m_1, \hat{m}_2 = m_2, \cdots, \hat{m}_d = m_d) = \\ 1 - P(\hat{m}_1 = m_1) \cdot P(\hat{m}_2 = m_2 | \hat{m}_1 = m_1) \cdot \\ \cdots P(\hat{m}_d = m_d | \hat{m}_1 = m_1, \cdots, \hat{m}_{d-1} = m_{d-1}). \end{aligned} \tag{2}$$

Let p_i be the probability that the first i−1 information is successfully decoded and the ith segment fails. $\hat{m}_i$ denotes the estimation of m_i, so (2) is simplified as

$$P_e = 1 - \prod_{i=1}^{d} (1 - p_i) \tag{3}$$

where $p_i = P(\hat{m}_i \neq m_i | \hat{m}_1 = m_1, \cdots, \hat{m}_{i-1} = m_{i-1})$.

At the transmitter, the mutual information between the block m_d and the transmission code symbols $x_1, x_2, \cdots, x_{d-1}$ is zero and independent of each other. The information of m_d is only carried by the coding symbol x_d, therefore, the average error probability of the last information block can be taken as the lower bound of the average error probability of the whole spinal codes.

$$P_e \geqslant p_d \tag{4}$$

where p_d indicates that the decoding of the first d−1 information segments is correct. While the decoding failure probability of the dth information block can be regarded as the error probability of a single information block, it can be regarded as the error probability of a single information block.

For the tail-biting spinal codes with parameter Δ, the total error probability is as follows:

$$\begin{aligned} P_e = 1 - P(\hat{m}_{d'-\Delta+1} = m_{d'-\Delta+1}, \hat{m}_{d'-\Delta+2} = m_{d'-\Delta+2}, \\ \cdots, \hat{m}_1 = m_1, \cdots, \hat{m}_d = m_d) = \\ 1 - \prod_{j=1}^{\Delta} \left(1 - p'_j\right) \cdot \prod_{i=1}^{d} (1 - p_i) \end{aligned} \tag{5}$$

where p'_j is the error probability of the added tail-biting block, when $\Delta = 1$,

$$p'_1 = P(\hat{m}_{d'} \neq m_{d'}), \tag{6}$$

$$p_i = P(\hat{m}_i \neq m_i | \hat{m}_{d'} = m_{d'}, \hat{m}_1 = m_1, \cdots, \hat{m}_{i-1} = m_{i-1}). \tag{7}$$

For $i = d$, the decoding algorithm corresponding to the encoding structure of the tail-biting spinal code requires that the first information block of the block sequences is the same as the last one. When $\hat{m}_{d'} = m_{d'}$ is true, $\hat{m}_d = m_d$ must be established. Therefore, $p_d = P(\hat{m}_d \neq m_d | \hat{m}_{d'} = m_{d'}, \hat{m}_1 = m_1, \cdots, \hat{m}_{d-1} = m_{d-1})$ is zero, and the error probability can be written as

$$P_e = 1 - (1 - p'_1) \cdot \prod_{i=1}^{d-1} (1 - p_i). \tag{8}$$

It can be found that the segment leading to the error of the tail-biting spinal code is m_{d-1}. That is, the lower bound of the overall BER is given as

$$P_e \geqslant p_{d-1}. \tag{9}$$

Set up $\Delta = 2$, and

$$p'_1 = P(\hat{m}_{d'-1} \neq m_{d'-1}), \tag{10}$$

$$p'_2 = P(\hat{m}_{d'} \neq m_{d'} | \hat{m}_{d'-1} = m_{d'-1}), \tag{11}$$

$$p_i = P(\hat{m}_i = m_i | \hat{m}_{d'-1} = m_{d'-1}, \cdots, \hat{m}_{i-1} = m_{i-1}). \tag{12}$$

For $i = d{-}1$ or $i = d$, $\hat{m}_{d'-1} = m_{d'-1}$, $\hat{m}_{d'} = m_{d'}$, the error probability of p_{d-1} and p_d are 0. Thus we have

$$P_e = 1 - (1 - p'_1)(1 - p'_2) \cdot \prod_{i=1}^{d-2} (1 - p_i). \tag{13}$$

It is obvious that the error correction capability of the spinal codes with $\Delta = 2$ is determined by the information block m_{d-2}.

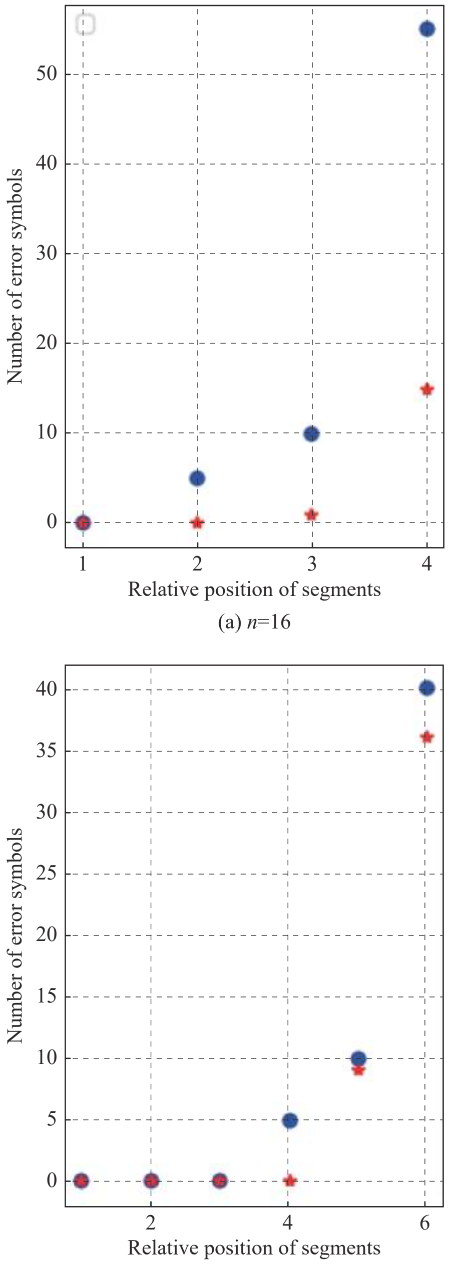
$$P_e \geqslant p_{d-2}. \tag{14}$$

By analogy, the error formula of spinal codes under different tail-biting parameters can be obtained, owing to $p_{i-1} \leqslant p_i (1 < i \leqslant d)$. Compared with the original structure, the lower bound of error probability and the error correction performance of the tail-biting spinal codes are effectively improved.

2.3 Performance analysis of tail-biting spinal codes

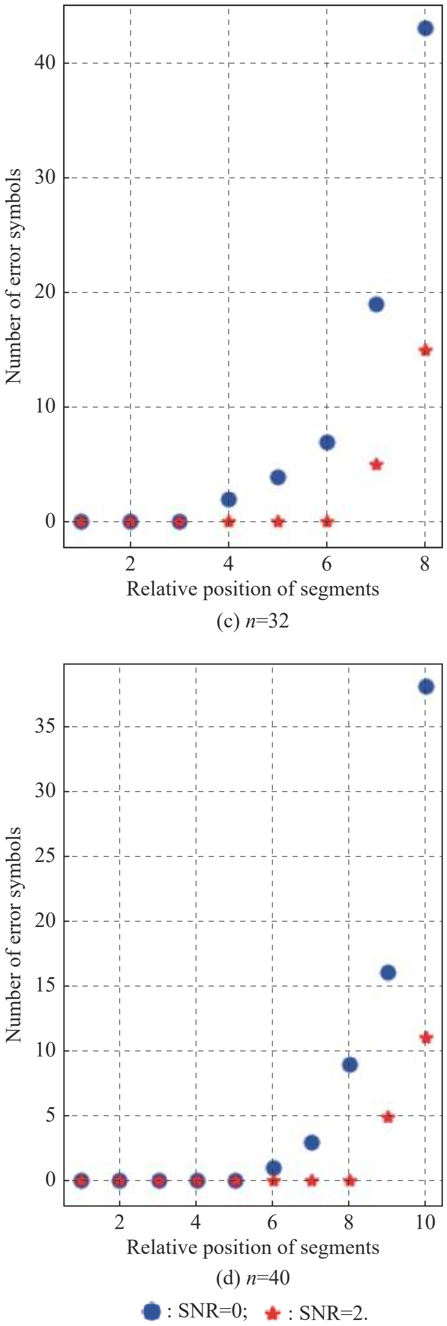
By observing the location distribution with error prone symbols, the appropriate Δ is selected. The simulation conditions are as follows: assume that the channel model is AWGN, the information is n, the segment length $k = 4$, the number of passes $L = 8$, the pruning parameter B of each layer is 32 by using the bubble algorithm, and the tail-biting parameter is Δ.

When 1 000 frames of message are transmitted, the relationship between the relative position of information block and the number of error symbols is shown in Fig. 2. The error location of spinal codes is mainly concentrated in the tail, and its error correction performance becomes worse with the decline of SNRs. The information length is different, the error probability of the symbol in the front 3/4 position is lower, while the error probability of the information block in the last 1/4 position is higher.



(a) n=16

(b) n=24



(c) n=32

(d) n=40

Fig. 2 Error location and numbers of messages for original spinal codes

Assume that 32 bits are transmitted per frame, the comparison between the original and the tail-biting spinal codes with parameter $\Delta = 1$ and $\Delta = 2$ is conducted.

The symbol error rate (SER) of spinal codes is shown in Fig. 3. For different encoding structures, with the increase of SNRs, the SER decreases rapidly. In the same channel state, the SER of the designed structure is better than that of the original structure. And the error correction ability of the structure with two tail symbols in the head is more outstanding than that of the structure with one tail symbol. Therefore, the tail-biting spinal codes have the better performance.

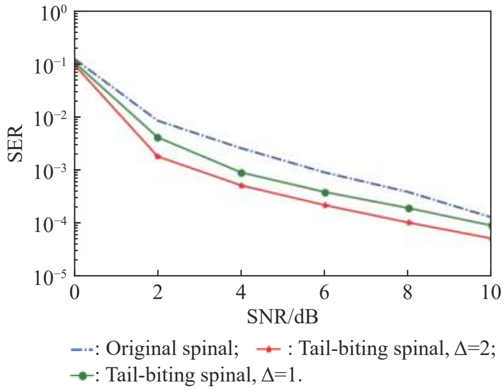


Fig. 3 SER comparison of spinal codes with different encoding structures

Because the number of transmission passes is reduced, the number of channel coding symbols required for correct decoding is reduced, and the transmission overhead is also reduced accordingly.

3. Improved decoding algorithms

3.1 Forward stack decoding algorithm

Based on the principle of the stack algorithm, the researcher proposes the FSD algorithm with lower decoding complexity. The FSD method divides the entire tree into several units. By dividing the n/k-layer decoding tree into n/kD units, the parameter D ($D<n$) is the integer that decides the layer of each unit.

The search process of the FSD algorithm can be depicted in Fig. 4. The FSD algorithm starts with the root node, searches the stack in the first basic unit, obtains the optimal node in this unit, and saves it. Then, according to the results of the first unit, the stack is carried out in the second unit to obtain the optimal node. With the same method, the last unit is searched and the optimal node path is output. Compared with the stack decoding algorithm, the FSD algorithm controls all units independently, restrains the large jump between nodes in the stack decoding process, and effectively reduces the search amount of invalid nodes. At the same time, the maximum number of storage nodes B is set in the path reservation stack B and the maximum number of searching nodes C is controlled in the search expansion of each unit. The storage requirement for the stack capacity is significantly reduced in the whole decoding process of spinal codes, which is conducive to the simplification of the decoding equipment in the specific engineering practice.

It can be seen from Fig. 4 that obviously, the number of nodes in the reserved stack B remains unchanged after the search expansion of each unit is completed. By analyzing the complexity of the FSD algorithm, it can be seen that the number of reserved nodes per unit directly affects the computational complexity and BER performance of the spinal codes. When the channel conditions are relatively stable, the fixed nodes will cause the reservation of an invalid path and increase the overall decoding complexity of the decoding tree. When the channel states are bad, the optimal node expansion will lead to more error paths searched, which is not ideal for further reducing the decoding complexity of the spinal codes. Therefore, new decoding algorithms are needed.

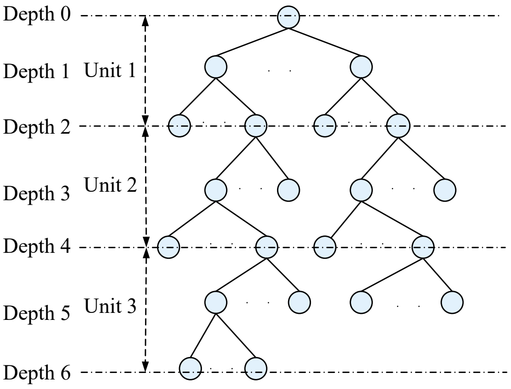


Fig. 4 Search process of FSD algorithm (*D*=2)

3.2 Adaptive forward stack decoding algorithm

In this subsection, an A-FSD algorithm is proposed. This algorithm can dynamically adjust the number of storage nodes by combining channel state information and reduce the decoding complexity of spinal codes.

The A-FSD algorithm designs a threshold to control the number of storage nodes per unit based on the FSD algorithm. The threshold is mainly determined by the minimum path metric of the current unit and the variance of intra-channel noise. Define a threshold of every unit *i*. It can be expressed as $\varGamma_i = \varepsilon_{i,\min} + 2^{\tau} \cdot f(\mathrm{SNR})$, where $\varepsilon_{i,\min}$ is the minimum metric in the *i*th unit and τ represents a specific threshold control index. The $f(\mathrm{SNR})$ denotes the currently estimated channel SNRs function whose value is inversely proportional to SNR. That is, it is positively increased with channel interference and noise power. Therefore, in the *i*th unit of the decoding tree, the cumulative path metrics of the nodes with a depth of *iD* in search stack *C* are compared with the size of $\varGamma_i$. When the value is greater than $\varGamma_i$, the node is discarded. Judge whether the number of surviving nodes is less than *B*. If it is, then all nodes are saved in stack *B*, and recode the number of actual nodes as B_i. If not, only the first *B* nodes with the greatest reliability are saved.

In the process of node selection using an adaptive method, the determination of threshold $\varGamma_i$ is related to the decoding effect of spinal codes. The positive and negative effects vary greatly and cannot be ignored. If the threshold is too small, the bad effect is that the correct decoding path is easily discarded, and the information transmission becomes worse. On the contrary, the total decoding complexity will increase while the quality of information transmission is stable. In a word, for different channel conditions, different thresholds need to be set up, the size of thresholds should also be taken into account to minimize the loss of performance and reduces the complexity as much as possible.

The adaptive adjustment of the A-FSD decoder needs the cooperation of the channel estimation algorithm. Typical SNR estimation methods for AWGN channels include M2M4 estimation [27,28], singular value decomposition [29] and data fitting algorithm [30]. In this paper, we adopt the data fitting algorithm with simple calculation and accurate estimation.

The adaptive property of the A-FSD algorithm is mainly reflected in blind channel estimation using received signals to control the number of nodes. Apparently, the complexity of the improved A-FSD algorithm is better than the FSD algorithm, because the number of storage nodes for the A-FSD algorithm is no more than the capacity of stack *B*.

3.3 Adaptive forward stack decoding with ET

Based on the principle of the A-FSD algorithm, a new algorithm named AFSD-ET is proposed. It can achieve a better tradeoff between performance and decoding complexity.

The main ideas of the improved AFSD-ET algorithm are as follows. Firstly, based on the FSD decoding structure, the search unit size is set to *D*. Therefore, the whole decoding tree is divided into n/kD units, and then the divided decoding units are divided into two groups, set the grouping parameter to $T, T \in \{1,2,\cdots,n/kD\}$. The first group is defined as the unit number less than or equal to the block parameter *T*, the decoding algorithm is based on the A-FSD decoder principle, and the remaining units are automatically divided into the second group. Secondly, the nodes stored in the reserved path stack *B* after decoding in the first group are taken out and arranged in the order of path metrics. They are called pseudo root nodes and are used as the initial decoding nodes in the second group. The first pseudo root node (the minimum cumulative path metric) is regarded as the root node whose metric value is not zero, and then it is extended to the leaf node according to the A-FSD decoding principle. At this time, the path metric value of the optimal path is recorded as the threshold value $\varLambda_{\min}$. Eventually, the A-FSD algorithm is used to search for the second pseudo root node. $\Lambda_{\min}$ is used as the threshold in the process of searching. For nodes whose path metric value is greater than $\Lambda_{\min}$, the search is stopped by using the ET criterion. Only the nodes less than the threshold are searched until the leaf node is found. If the metric value of the optimal leaf node is less than $\Lambda_{\min}$, then the threshold $\Lambda_{\min}$ is updated by this metric value, and the remaining pseudo root nodes are detected in turn according to the above method. The path corresponding to $\Lambda_{\min}$ is taken as the decoding output. Fig. 5 shows a decoding tree of the AFSD-ET algorithm, where the values of T and D are set to be 2.

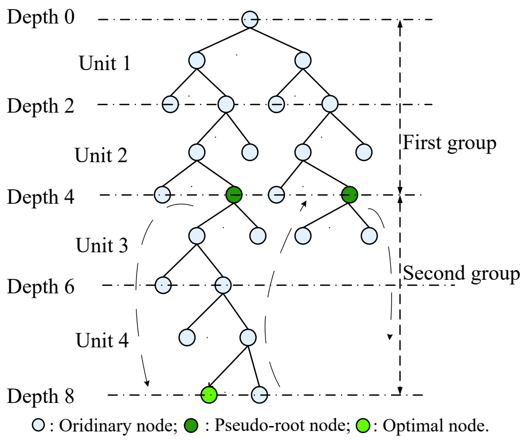


Fig. 5 The AFSD-ET algorithm (Example of an 8-depth decoding tree with T=2 and D=2)

The proposed algorithm takes each pseudo root node as a whole and adopts the depth first search strategy to speed up the speed of decoding to the leaf node in the process of detecting the second group of search points in the decoding tree. At the same time, combined with the ET criterion, it can flexibly adjust the decoding search path, significantly reducing the number of search nodes.

The specific steps of the AFSD-ET decoding algorithm can be described as follows.

Step 1 Initialize parameters. The message M with transmission length of n bits is divided into n/k depths in k bits. If the size of search unit is D, the whole decoding tree contains n/kD units. The front T units (including the Tth unit) are divided into the first group, and the remaining units are divided into the second group.

Step 2 In the first group, the A-FSD algorithm is used to search from the root node to the DT th layer. In the DT th layer, all nodes stored in the reserved path stack B are taken out, and the number of nodes is counted as B_1 who are called pseudo root nodes according to the path metric value from small to large. It is marked as $\boldsymbol{X}_{DT}=[X_{DT}^1, X_{DT}^2,\cdots,X_{DT}^j,\cdots,X_{DT}^{B_1}]$ and its corresponding cumulative path measure is marked as $\Lambda_{DT}^j\,(\Lambda_{DT}^j\leqslant\Lambda_{DT}^{j+1})$.

Step 3 Start the search of the second group of decoding tree, initialize j = 1, start with pseudo root node X_{DT}^1, use the A-FSD algorithm to search until the leaf nodes are found, select the minimum path metric value in stack B of reserved path stack as $\Lambda_{\min}$, which is called ET search threshold, and record its corresponding decoding path $\tilde{x}=[\tilde{x}_1,\tilde{x}_2,\cdots,\tilde{x}_{n/k}]$.

Step 4 j=j+1, judge $\Lambda_{\min}$ and Λ_{DT}^j. If $\Lambda_{DT}^j<\Lambda_{\min}$, execute Step 5; if $\Lambda_{DT}^j\geqslant\Lambda_{\min}$, terminate the search according to the ET strategy, and directly execute Step 7.

Step 5 Set $\Lambda_{\min}$ as the threshold in each unit. In the search process, if the path metric of a node is greater than the threshold, it will not be extended by using the ET criterion. If it is less than the threshold, the subnode will be extended until the leaf nodes are found. If the minimum path metric $\Lambda'_{\min}<\Lambda_{\min}$ in the path stack B is retained, $\Lambda_{\min}$ is updated with $\Lambda'_{\min}$ and the corresponding decoding path $\tilde{x}$ is updated.

Step 6 Judge the size of j and B_1. If $j<B_1$, perform Step 4; otherwise, go to Step 7.

Step 7 Select the path corresponding to $\Lambda_{\min}$ at this time as the final decoding output, and the decoding ends.

The detailed AFSD-ET algorithm is illustrated in Algorithm 1.

Algorithm 1 The AFSD-ET

```
Initialize set parameter $T, D,$ Stack $B$ and Stack $C$
decoding from the root node
for $t \leftarrow 1$ to $DT$-depth do
    expand nodes with the A-FSD algorithm
end for
save $B_1$ nodes in $DT$-depth and it is named as pseudo-root node
for $j \leftarrow 1$ to $B_1$ do
    put the $j$th pseudo-root node into Stack $B$
    for $i \leftarrow 1$ to $\dfrac{(n/k)-DT}{D}$ do
        store the nodes to the Stack $C$
        if $j$=1 then
            get the optimal node in $DT$-depth to design the threshold for per unit
        else
            if path metric of the $j$th pseudo-root node is greater than $\varepsilon_{\min}$ then
                stop the extension of this pseudo-root node in advance by using the ET criterion
                break
            else
                continue
        end if
    set the threshold as $\varepsilon_{\min}$ to control the number of reserved nodes in $DT$-depth
end for
```

```
    save the current optimal node, recode the cost metric as $\varepsilon_{\min}$ and the corresponding decoding is $\tilde{\boldsymbol{x}}$.
end for
return $\tilde{\boldsymbol{x}}$
```

Before the start of the second group of decoding, the pseudo root nodes are sorted in advance, which increases the probability that the output path along the first candidate node becomes the maximum likelihood path, and by updating $\varLambda_{\min}$, the path metric value of the maximum likelihood path can be approached with greater probability.

3.4 Theoretical analysis of decoding complexity

Assume an n-bit message M, which is divided into n/k layers by the segment length k. Only a pass is considered for the decoding process. The decoding complexity of the bubble algorithm is $O(nB\cdot 2^k(v+k+\lg B))$, where v is the length of the hash value and B denotes the pruning width [13]. The FSD algorithm divides the n/k-depth decoding tree into n/kD layers. Assume that the number of selected nodes is B_F in each unit, the upper bound and lower bound of the FSD algorithm for spinal decoding are $O_{\max}(n/k\cdot B_F\cdot 2^k(v+k+\lg B_F))$ and $O_{\min}(n/kD\cdot B_F\cdot 2^k(v+k+\lg B_F))$ respectively [20]. Due to the parameter B_F always less than or equal to B, the complexity of the FSD algorithm must be better than the bubble algorithm.

With the same idea as the FSD algorithm, the A-FSD algorithm adds one step in each unit. Set the threshold value to adjust the reserved nodes. We define B_R as the actual reserved nodes for each unit and the maximum reserved nodes set to B_A. The complexity of the A-FSD algorithm is $O\left(\sum\limits_{R=1}^{n/(kD)} B_R\cdot 2^k(v+k+\lg B_R)\right)$ and the upper bound is $O_{\max}(n/k\cdot B_A\cdot 2^k(v+k+\lg B_A))$. Compared with the FSD algorithm, taking the same maximum quantity of storage nodes for the two algorithms, the decoding complexity of the A-FSD is the same. For the decoding of the AFSD-ET algorithm, the A-FSD algorithm at the front DT-depth of decoding tree and the complexity is $O\left(\sum\limits_{R=1}^{T} B_R\cdot 2^k\cdot (v+k+\lg B_R)\right)$. For the remaining $n/kD-T$ units, set the actual storage nodes as B_R. Keeping B_T pseudo-root nodes for DT-depth and decoding for the first one, we can compute the complexity as $O\left(\sum\limits_{R=T+1}^{n/(kD)} B_R\cdot 2^k(v+k+\lg B_R)\right)$. Considering that only the first pseudo root node is extended, the rest B_T-1 stops ahead of time, we can obtain the minimum complexity of the AFSD-ET algorithm for spinal codes as $O_{\min}\left(\sum\limits_{R=1}^{n/(kD)} B_R\cdot 2^k(v+k+\lg B_R)\right)$. Because of the uncertainties of related parameters, it is impossible to determine the specific complexity relationship, but it can be clearly seen from the decoding formula that the AFSD-ET algorithm can achieve a smaller extreme decoding complexity under certain conditions.

4. Performance evaluation

To further evaluate the performance of the improved spinal codes, a serial of simulations are carried out. In this paper, the feedback delay from the decoder to the sender is neglected. All simulations use the AWGN channels, and the parameters of the spinal codes are n = 32, k = 4, L = 8, D = 2. The storage parameter B is 32. The maximum size of stack C is set to 4 096.

4.1 Parameters selection of the A-FSD algorithm

It can be seen from Fig. 6 that the BER performance among the bubble, the FSD and the A-FSD algorithms are very close at $\tau=5$. Thus it can be considered that they are consistent in error control ability of spinal codes. The performance of the A-FSD decoding algorithm with different values of τ is analyzed. Under the same channel environment, with the increase of τ value, the BER performance is closer to the other two algorithms. When a certain value is taken, the performance of A-FSD decoding is very different from that of the traditional spinal decoding algorithm. The main reason for the performance difference is that when τ is smaller, the threshold of the number of surviving control nodes is also smaller, which increases the probability of deleting the correct path, resulting in the degradation of the decoding performance of spinal codes. With the increase of τ value, the threshold is enlarged, and the node stored in the reserved stack of the A-FSD algorithm is close to that of the FSD algorithm, so its error correction performance is also improved.

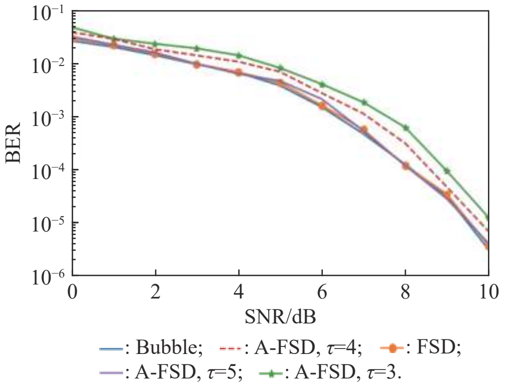


Fig. 6 BER for spinal codes with A-FSD algorithm

The complexity of different decoding methods refers to the average number of nodes accessed about a frame of information in the decoding process. Fig. 7 simulates the calculation costs of different algorithms. Through comparison, it can be found that under the same SNRs, compared with the bubble decoding algorithm, the average number of access nodes of the FSD algorithm and the A-FSD algorithm is greatly reduced, but the calculation amount of the FSD algorithm and the A-FSD algorithm under lower SNRs is still high. With different parameter τ, the number of access nodes decreases in the whole SNR range, especially when the control parameters are small, the calculation amount of nodes is significantly reduced. To sum up, the proposed novel algorithm not only guarantees the transmission performance of spinal codes, but also optimizes the decoding complexity compared with the traditional decoding algorithm.

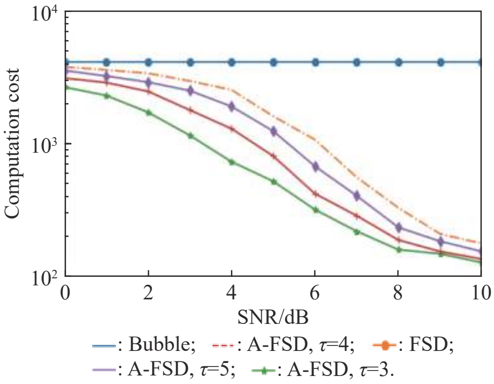


Fig. 7 Computation cost for spinal codes

4.2 Parameters comparison of AFSD-ET algorithm

In the AFSD-ET algorithm, the value of T directly affects where the second group begins to use depth first search and ET strategy to further optimize decoding. The simulation results with different T are shown in Fig. 8. At low SNRs, the performance of AFSD-ET is similar with different packet parameters. With the increase of SNRs, the performance difference gradually increases, and the maximum performance gap can reach 2.4 dB. Under the same threshold control parameters, when $T = 4$, the BER performance of the decoding algorithm is poor. With the decrease of T, the error control ability of spinal codes is gradually improved. When $T = 1$, the decoding performance is optimal. When $\tau = 5$, $T = 2$ and $\tau = 5$, $T = 3$, the performance gap of the AFSD-ET algorithm is very small. It can be seen that for the improved algorithm, when the first group of decoding tree contains similar units, its performance is closer.

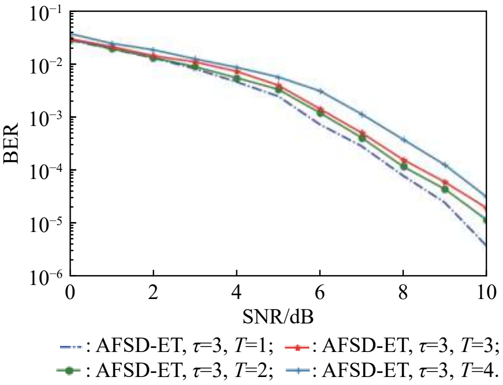


Fig. 8 BER of AFSD-ET decoding algorithms

It can be seen from Fig. 9 that when $\tau = 5, T = 1$, the AFSD-ET algorithm has the largest number of access nodes. The main reason is that the number of layers requiring depth first search is large, resulting in a large difference in path metrics. Starting from the second pseudo root node, it needs to be extended to deeper nodes to effectively use the ET strategy. Therefore, the number of search nodes is increased with smaller T. When $\tau = 5$, $T = 4$, because A-FSD decoding structure is used in the whole decoding process, the depth first search and ET criteria are not used, so the decoding complexity is also large. When SNR $<$ 5 dB, the complexity of $T = 3$ is slightly better than $T = 2$, but the difference between the two cases is small; when SNR $>$ 5 dB, the complexity of $T = 2$ is lower than $T = 3$, and the number of nodes visited is different. It can be seen that the more similar the number of units contained in the two groups of the whole decoding tree, the better the advantages of depth first search and ET criteria can be achieved by setting the grouping parameters. When the performance is similar, the decoding complexity of spinal codes decreases faster.

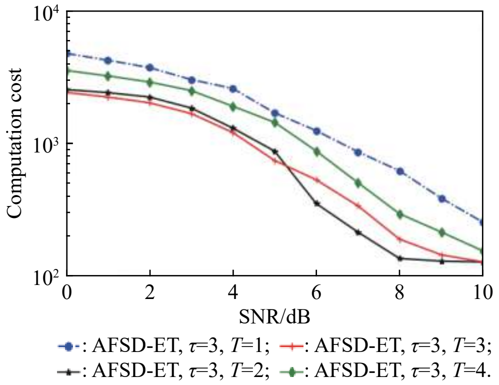


Fig. 9 Computation cost of AFSD-ET decoding algorithm

4.3 Comparison of four decoding algorithm

Fig. 10 provides the BER comparison of two traditional decoding algorithms and two improved decoding algorithms. It can be found that with the increase of SNRs, the error correction performance of several decoding algorithms is improved. At the same time, it can be observed that the BER performance of the AFSD-ET algorithm proposed in the previous section is slightly improved in the whole SNR range. However, on the whole, the BER performance of the AFSD-ET algorithm is similar to that of other algorithms.

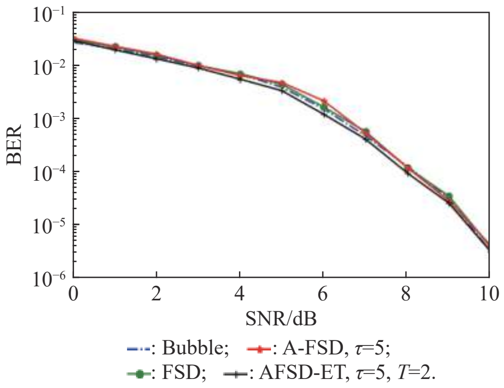


Fig. 10 BER comparison of four decoding algorithms

The number of access nodes of four decoding algorithms with the same code rate are compared in Fig. 11. The result shows that the complexity of the two improved algorithms proposed in this paper is lower than that of the traditional spinal codes, and the complexity of the AFSD-ET algorithm decreases more obviously. Therefore, it is verified that the use of depth first search and ET strategy can greatly reduce the decoding tree search.

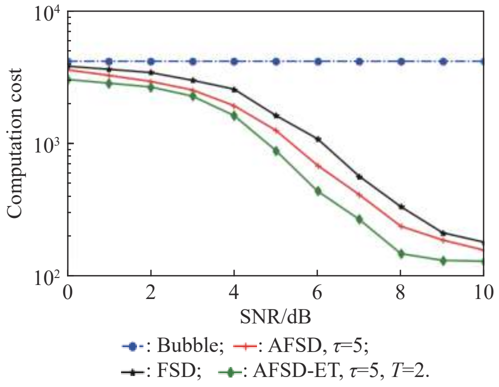


Fig. 11 Computation cost of four decoding algorithms

4.4 Performances of improved spinal codes

The BER performances of tail-biting spinal codes and comparable traditional spinal codes with bubble decoder or AFSD-ET decoder under the rate being 1 are shown in Fig. 12. According to the simulation results, considering two different decoding algorithms separately, the reliability of the tail-biting encoding structure is better than that of the one without adding the tail symbol. Compared with the conventional decoding algorithm, the improved decoder provides better error control performance for the tail-biting scheme. We can see from the curves that the spinal codes perform the best by adding tail-biting symbols and using the novel AFSD-ET decoding method over the AWGN channel. Thus, the transmission of the messages by applying tail-biting spinal codes with the proposed decoding method exhibits better performance.

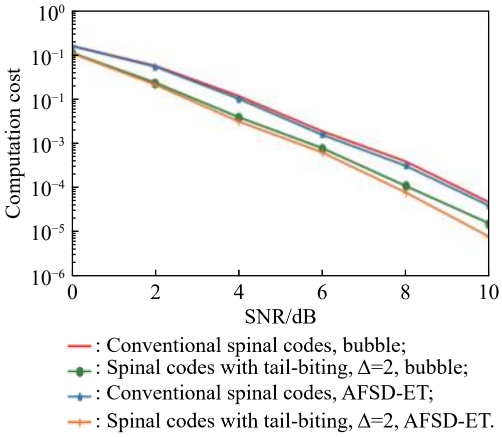


Fig. 12 BER comparison for spinal codes

5. Conclusions

In this paper, firstly, to solve the problem that the error bits of spinal codes are always concentrated in the last several segments, a novel structure of spinal codes with the tail-biting is proposed. The proposed coding structure can effectively enhance the reliability in transmission. Secondly, in the light of the problem of high computation of the decoding algorithm, an adaptive algorithm is designed to control the number of stored nodes according to the threshold value, which needs to be set by the blind channel estimation algorithm to predict the SNR of the AWGN channel. The A-FSD reduces the computation to a certain extent. In order to further optimize calculations, the AFSD-ET decoder is proposed, which combines the depth-first search and ET criteria, and effectively controls the tradeoff between the error performance and calculation amount of Spinal codes. Moreover, the spinal codes perform well by using the tail-biting structures and adopting the novel AFSD-ET decoding method over the AWGN channel.

References

[1] JONATHAN P, HARI B, DEVAVRAT S. Rateless spinal codes. Proc. of the 10th ACM Workshop on Hot Topics in Networks, 2011. DOI:10.1145/2070562.2070568.

[2] ELIAS P. Coding for noisy channels. IRE Convention Record, 1995: 37–46.

[3] MAHRAN A, SAMY R. Distance threshold Viterbi deco-

ding of convolutional codes. Proc. of the 7th International Japan-Africa Conference on Electronics, Communications, and Computations, 2019. DOI: 10.1109/JAC-ECC48896.2019.9051105.

[4] WILDE M M, HSIEH M H, BABAR Z. Entanglement-assisted quantum turbo codes. IEEE Trans. on Information Theory, 2014, 60(2): 1203–1222.

[5] ATTARKASHANI A, HAMOUDA M. Performance analysis of turbo codes and distributed turbo codes in buffer-aidedrelay systems. IEEE Trans. on Communications, 2019, 67(7): 4620–4633.

[6] ALEXIOS B S, BURG A. Density evolution for min-sum decoding of LDPC codes under unreliable message storage. IEEE Communications Letters, 2014, 18(5): 849–852.

[7] ZHANG J, BAI B M. Tail-biting globally-coupled LDPC codes. IEEE Trans. on Communications, 2019, 67(12): 8206–8219.

[8] BALAKRISHNAN H, IANNUCCI P, PERRY J, et al. De-randomizing Shannon: the design and analysis of a capacity-achieving rateless code. ArXiv preprint arXiv: 1206.0418, 2012.

[9] YANG W Q, LI Y, YU X P, et al. Two-way spinal codes. Proc. of the IEEE International Symposium on Information Theory, 2016: 1919–1923.

[10] YU X P, LI Y, YANG W Q, et al. Design and analysis of unequal error protection rateless spinal codes. IEEE Trans. on Communications, 2016, 64(11): 4461–4473.

[11] PERRY J, IANNUCCI P A, FLEMING K E, et al. Spinal codes. Proc. of the ACM International Conference on the Applications, Technologies, Architectures, and Protocols for Computer Communication, 2012, 42(4): 49–60.

[12] XU X, WU S H, DONG D, et al. High performance short polar codes: a concatenation scheme using spinal codes as the outer code. IEEE Access, 2018, 6: 70644–70654.

[13] YANG W Q, LI Y, YU X P, et al. A low complexity sequential decoding algorithm for rateless spinal codes. IEEE Communications Letters, 2015, 19(7): 1105–1108.

[14] YEH C C, LU H L, YEH J J, et al. Path exploration based on Monte Carlo tree search for symbolic execution. Proc. of the Conference on Technologies and Applications of Artificial Intelligence, 2017: 33–37.

[15] CHEN J N, CHEN S Y, QI Y L, et al. Intelligent massive MIMO antenna selection using Monte Carlo tree search. IEEE Trans. on Signal Processing, 2019, 67(20): 5380–5390.

[16] LI F M, WU A Y. A new stopping criterion for efficient early termination in turbo decoding design. Proc. of the International Symposium on Intelligent Signal Processing and Communication Systems, 2005: 585–588.

[17] HSU Y C, LIN J R, CHEN M J, et al. Acceleration of depth intra coding for 3D-HEVC by efficient early termination algorithm. Proc. of the IEEE Asia Pacific Conference on Circuits and Systems, 2018: 127–130.

[18] HU Y M, LIU R K, BIAN H X, et al. Design and analysis of a low complexity decoding algorithm for spinal codes. IEEE Trans. on Vehicular Technology, 2019, 68(5): 4667–4679.

[19] XU S Y, WU S H, LUO J X, et al. Low complexity decoding for spinal codes: sliding feedback decoding. Proc. of the IEEE 86th Vehicular Technology Conference, 2017. DOI: 10.1109/VTCFall.2017.8287919.

[20] CHEN P Y, BAI B M, LI Q, et al. Multiplicative repetition-based spinal codes with low computational complexity. IET Communications, 2017, 11(17): 2660–2666.

[21] ALLARD L J, DOBELL A, HULL T. Mixed congruential random number generators for decimal machines. Journal of the ACM, 1963, 10(2): 131–141.

[22] AL-MHADAWI M, ALBAHRANI A. Hybrid method as pseudo-random bits generator. Proc. of the International Conference of Computer and Applied Sciences, 2019: 250–255.

[23] WANG X T, QIAN H, XU J, et al. An efficient CAV-based decoding algorithm for tail-biting codes. Proc. of the IEEE Global Telecommunications Conference, 2011. DOI: 10.1109/GLOCOM.2011.6133569.

[24] MA X, LIN W C, CAI S H, et al. Statistical learning aided decoding of BMST tail-biting convolutional code. Proc. of the IEEE International Symposium on Information Theory, 2019: 1352–1356.

[25] WANG X T, QIAN H, XU J, et al. Trap detection based decoding algorithm for tail-biting convolutional codes. Journal of Electronics and Information Technology, 2011, 33(10): 2300–2305.

[26] WANG Y, DU X J, LI H, et al. Tail-biting theory for turbo codes. Proc. of the IEEE Annual Wireless and Microwave Technology Conference, 2006. DOI: 10.1109/WAMICON.2006.351949.

[27] BAKKAIL M, STEPHEN A, AFFES S. Generalized moment-based method for SNR estimation. Proc. of the IEEE Wireless Communications and Networking Conference, 2007: 2228–2232.

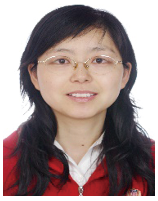
[28] SEKOKOTOANA L E, TAKAWIRA F, OYERINDE O O. Least mean squares channel estimation for downlink non-orthogonal multiple access. Proc. of the IEEE AFRICON, 2019. DOI: 10.1109/AFRICON46755.2019.9133868.

[29] EDFORS O, SANDELL M, WILSON J J, et al. OFDM channel estimation by singular value decomposition. IEEE Trans. on Communications, 1998, 46(7): 931–939.

[30] XU Q, ZHOU J. Improved SNR estimation algorithm. Proc. of the International Conference on Computer Systems, Electronics and Control, 2017: 1458–1461.

Biographies

HUANG Wensha was born in 1995. She is pursuing her M.S. degree at the Department of Communication Engineering, School of Computer and Communication Engineering, University of Science and Technology Beijing. Her research interest is channel coding in deep space.
E-mail: 18801475622@163.com



WANG Lina was born in 1977. She received her M.S. and Ph.D. degrees in communication and information systems from Harbin Institute of Technology in 2001 and 2004, respectively. From 2010 to 2012, she was engaged in postdoctoral research at the University of Science and Technology Beijing. She is currently a professor in the Department of Communication Engineering, School of Computer and Communication Engineering, University of Science and Technology Beijing. Her research interests include space communications, satellite positioning algorithms, and rateless codes.
E-mail: wln_ustb@126.com