TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 03/20 pp656–670
DOI: 10.26599/TST.2023.9010038
Volume 29, Number 3, June 2024

Spatially Coupled Codes via Bidirectional Block Markov Superposition Transmission

Gaoyan Li, Shancheng Zhao*, Haiqiang Chen, and Jinming Wen

Abstract: In this paper, we present a new class of spatially coupled codes obtained by using both non-recursive and recursive block-oriented superposition. The resulting codes are termed as bidirectional block Markov superposition transmission (BiBMST) codes. Firstly, we perform an iterative decoding threshold analysis according to protograph-based extrinsic information transfer (PEXIT) charts for the BiBMST codes over the binary erasure channels (BECs). Secondly, we derive the generator and parity-check matrices of the BiBMST codes. Thirdly, extensive numerical results are presented to show the advantages of the proposed BiBMST codes. Particularly, our numerical results show that, under the constraint of an equal decoding latency, the BiBMST codes perform better than the recursive BMST (rBMST) codes. However, the simulation results show that, in finite-length regime, negligible performance gain is obtained by increasing the encoding memory. We solve this limitation by introducing partial superposition, and the resulting codes are termed as partially-connected BiBMST (PC-BiBMST) code. Analytical results have confirmed the advantages of the PC-BiBMST codes over the original BiBMST codes. We also present extensive simulation results to show the performance advantages of the PC-BiBMST codes over the spatially coupled low-density parity-check (SC-LDPC) codes, spatially coupled generalized LDPC (SC-GLDPC) codes, and the original BiBMST codes in the finite-length regime.

Key words: block Markov superposition transmission (BMST); protograph-based extrinsic information transfer (PEXIT); capacity-approaching codes; spatial coupling

1 Introduction

Error correction codes are indispensable for communication systems, including wireless communication, maritime information networks[1], vehicular communication networks[2], and helicopter-satellite communication system[3]. In the fifth-generation (5G) mobile communications, polar codes[4, 5] and low-density parity-check (LDPC) codes[6–8] are employed for error correction. However, polar codes and LDPC block codes are not suitable for streaming applications. In 1999, Felstrom and Zigangirov[9] proposed the convolutional LDPC codes, which may find applications in streaming. The convolutional LDPC codes were constructed by combining convolutional codes with LDPC block codes. The idea of convolutional LDPC codes was then generalized as spatially coupled LDPC (SC-LDPC) codes[10]. The SC-LDPC codes can be generated from a

• Gaoyan Li, Shancheng Zhao, and Jinming Wen are with College of Information Science and Technology, Jinan University, Guangzhou 510632, China. E-mail: ligaoyan@stu2021.jnu.edu.cn; shanchengzhao@jnu.edu.cn; jinming.wen@mail.mcgill.ca.
• Shancheng Zhao and Haiqiang Chen are with Guangxi Key Laboratory of Multimedia Communications and Network Technology, Guangxi University, Nanning 530004, China. E-mail: haiqiang@gxu.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2022-11-17; revised: 2023-03-27; accepted: 2023-05-05

series of LDPC block codes by reconnecting the edges of their Tanner graphs[11]. The SC-LDPC codes were proved to have the threshold saturation phenomenon[10–14], which means that the iterative decoding threshold of an SC-LDPC ensemble can achieve the maximum a posteriori (MAP)[15, 16] threshold of the underlying uncoupled code ensemble over binary-input memoryless output-symmetric channels. Spatially coupling can also be applied to the low-density generator matrix (LDGM) codes[17–19] and turbo-like codes[20–24].

Recently, a class of big convolutional codes called block Markov superposition transmission (BMST) which is constructed by spatial coupling of the generator matrices of the short codes was proposed in Refs. [25, 26]. A BMST code can be viewed as a serially concatenated code that consists of an inner code and an outer code. The outer code is referred to as the basic code. For BMST construction, any short code (linear or non-linear) with efficient encoding and decoding algorithms can be employed as the basic code[25]. For example, BMST codes based on short Hadamard transform (HT) codes[27], Bose-Chaudhuri-Hocquenghem (BCH) codes[28], repetition codes (RC), and single-parity-check (SPC) codes[29] had been investigated in the literature. The inner code is a rate-one block-oriented feedforward (non-recursive) convolutional code. To achieve enhanced flexibility, the systematic BMST of repetition (BMST-R) codes were proposed in Ref. [30]. Existing numerical results showed that BMST-R code performs well in a wide range of coding rates. The technique of superposition transmission is used in LDPC coded links with feedback to enhance the spectral efficiency[31].

When compared with SC-LDPC codes, the BMST code performs well in the waterfall region but has a higher error floor and a large decoding complexity[26]. To reduce the decoding complexity, the recursive block Markov superposition transmission (rBMST) was proposed in Ref. [32]. The rBMST codes were obtained by replacing the block-oriented feedforward convolutional encoder in BMST with a block-oriented recursive convolutional encoder. Numerical results showed that rBMST codes perform better than BMST codes even with a much smaller coupling memory. In addition, under the constraint of an equal decoding latency, rBMST codes admit better performance in the waterfall region and lower implementation costs when compared with the SC-LDPC codes.

Decoding complexity and decoding latency are key considerations when selecting channel codes[33]. For BMST and rBMST codes, the decoding latency is determined by the product of the length of the basic code and the decoding window size. In practice, both BMST codes and rBMST codes require a large encoding memory and hence a large decoding window size to achieve near-capacity performance. As a result, for a fixed decoding latency, basic codes of short length should be used, which may weaken the performance gain of BMST and rBMST codes. The study in Ref. [33] confirmed the huge impact of the length of the basic code on the decoding performance. In addition, the decoding complexity of a BMST code or an rBMST code grows quadratically with the encoding memory. Based on the above observations, in this paper, we are intended to construct BMST-like codes with enhanced performance and reduced decoding complexity.

In this paper, we extend the BMST and the rBMST constructions to present the bidirectional block Markov superposition transmission (BiBMST) codes, in which a rate-one block-oriented convolutional encoder with both non-recursive and recursive connections is taken as the inner encoder. The main contributions of this paper are summarized as follows:

(1) We have presented the encoder and decoder of the BiBMST codes and derived their generator and parity-check matrices.

(2) We have derived the protograph-based extrinsic information transfer (PEXIT) charts for BiBMST codes, which can be used to predict their iterative decoding threshold.

(3) We have investigated the impact of various parameters on the performance of the BiBMST codes. We have also carried out performance comparisons to show the advantages of the BiBMST codes over the rBMST codes in terms of decoding performance and computational complexity.

(4) To further enhance the performance of BiBMST codes, we introduced the partially connected BiBMST (PC-BiBMST) codes, in which a portion of the connection coefficients of the inner block-oriented convolutional encoder is set to be zero. We use the PEXIT to optimize the connection pattern of the PC-BiBMST codes under a given maximum allowable encoding memory. Our numerical results show that the optimized PC-BiBMST codes perform better than the

original BiBMST codes, SC-LDPC codes, and spatially coupling generalized LDPC (SC-GLDPC) codes.

The rest of the paper is structured as follows. In Section 2, we introduce the encoding and decoding algorithms for the BiBMST codes. The derivation of the generator and parity-check matrices is given in Section 3. In Section 4, we derive the PEXIT charts. The impacts of various parameters on the performance of BiBMST codes are investigated in Section 5. Performance and complexity comparisons are also presented in Section 5. In Section 6, we introduce the PC-BiBMST codes and their optimization. Section 7 concludes the paper.

2 Bidirectional Block Markov Superposition Transmission

The encoding and decoding algorithms for BiBMST codes are presented in this section.

2.1 Encoding of BiBMST

An information sequence $\boldsymbol{u}$ of length Lk is divided into L data blocks, each of which is of length k. That is, we have $\boldsymbol{u}=(\boldsymbol{u}^{(0)},\boldsymbol{u}^{(1)},\ldots,\boldsymbol{u}^{(L-1)})$. The parameter L is referred to as the coupling length. The encoding diagram of a BiBMST code is shown in Fig. 1, in which C denotes the basic code. Assume that a linear block code of length n and dimension k is used as the basic code. For convenience, the basic code is denoted as $C[n,k]$. The encoding algorithm of a BiBMST code with basic code $C[n,k]$ and encoding memory m is described in Algorithm 1, where, for $i=1,2,\ldots,2m$, $\boldsymbol{\Pi}_i$ is a binary $n\times n$ permutation matrix.

Let $C[n,k]$ be the basic code. The code rate of a BiBMST code is

$$R_{\text{BiBMST}}=\frac{Lk}{(L+T)n}=\frac{L}{L+T}R \tag{1}$$

where T is the termination length. Similar to SC-LDPC codes, the BiBMST code has a rate loss due to termination. This rate loss is negligible for large L. When all L pieces of length k are input to the encoder, we simply input T $(T\geqslant m)$ all-zero sequences of length k into the encoder during the termination process to ensure the transmission performance of the last few data blocks.

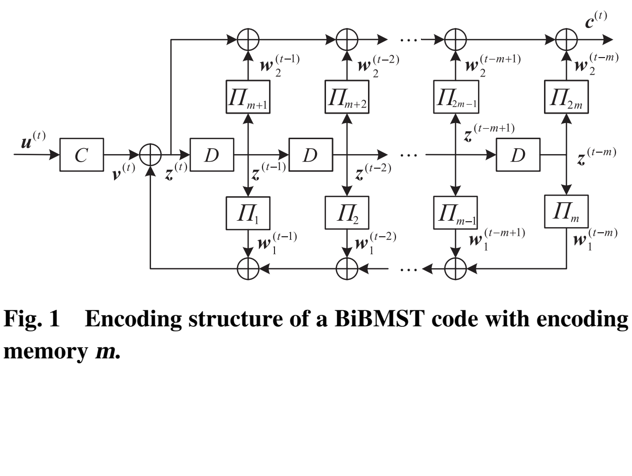

Fig. 1 Encoding structure of a BiBMST code with encoding memory *m*.

Algorithm 1 Encoding of a BiBMST code

```
(1) Initialization: For $t<0$, set $\boldsymbol{c}^{(t)}=\boldsymbol{0}\in F_2^n$.
(2) Loop: For $t=0,1,\ldots,L-1$,
    • Encoding: Encoding the $t$-th data block $\boldsymbol{u}^{(t)}$ by the encoder of the basic code, resulting in $\boldsymbol{v}^{(t)}\in F_2^n$.
    • Interleaving: For $1\leqslant i\leqslant m$, interleave $\boldsymbol{z}^{(t-i)}$ by the interleaver $\boldsymbol{\Pi}_i$ and $\boldsymbol{\Pi}_{m+i}$, resulting in the sequences $\boldsymbol{w}_1^{(t-i)}=(\boldsymbol{z}^{(t-i)})\boldsymbol{\Pi}_i$ and $\boldsymbol{w}_2^{(t-i)}=(\boldsymbol{z}^{(t-i)})\boldsymbol{\Pi}_{m+i}$.
    • Superposition: Compute $\boldsymbol{z}^{(t)}=\boldsymbol{v}^{(t)}+\Sigma_{i=1}^m\boldsymbol{w}_1^{(t-i)}$ and $\boldsymbol{c}^{(t)}=\boldsymbol{v}^{(t)}+\Sigma_{i=1}^m(\boldsymbol{w}_1^{(t-i)}+\boldsymbol{w}_2^{(t-i)})$; take $\boldsymbol{c}^{(t)}$ as the $t$-th transmitted block.
(3) Termination: For $t=L,L+1,\ldots,L+T-1$, set $\boldsymbol{u}_t^{(t)}=0\in F_2^n$ and compute $\boldsymbol{c}_t$ following Step (2).
```

2.2 Decoding algorithm

For the t-th transmitted block $\boldsymbol{c}^{(t)}$, the correspondingly received signal is denoted as $\boldsymbol{y}^{(t)}$. We use the iterative sliding-window decoder with a decoding delay of d to recover $\boldsymbol{u}^{(t)}$. The iterative sliding-window decoder operates on a subgraph of the normal graph of the BiBMST code.

For completeness, we present the normal graph of a BiBMST code. The normal graph can be divided into layers, where each layer consists of a $\boxed{\text{C}}$ node, a $\boxed{=}$ node, two $\boxplus$ nodes, and $2m$ $\boxed{\Pi_i}$ nodes. Please refer to Fig. 2 for reference. Node $\boxed{\text{C}}$ represents the constraint that the sequence must be a codeword of the basic code $C[n,k]$. Node $\boxplus$ represents the constraint that the sum of all connecting variables is zero over F_2. Node $\boxed{=}$ represents the constraint that all connecting variables take the same value. Node $\boxed{\Pi_i}$ represents the constraint of the i-th interleaver. For an encoding memory of m, the degrees of the node $\boxplus$ and $\boxed{=}$ are $m+2$ and $2m+2$, respectively.

Similar to SC-LDPC and rBMST codes[32], an iterative sliding-window decoder can be used for decoding. An example of the sliding-window decoder with a decoding delay of $d=3$ is shown in Fig. 2. The decoder operates on a subgraph of the normal graph consisting of $W=d+1$ layers. The first layer in the decoding window is called the target layer. The

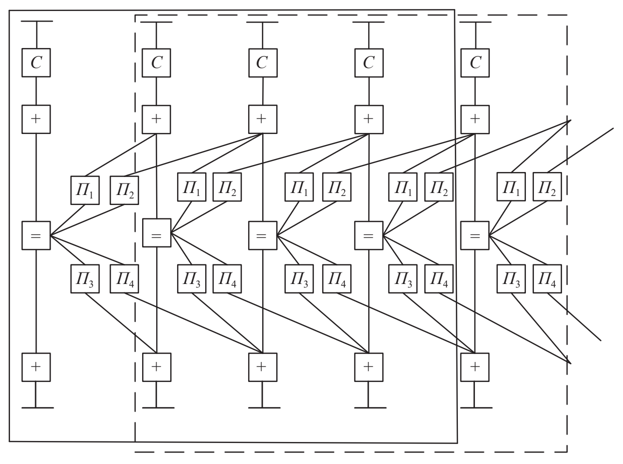

Fig. 2 Normal graphical representation: A BiBMST code with encoding memory $m = 2$ and decoding delay $d = 3$.

message updating rule in a decoding layer is $\boxed{+}\rightarrow\boxed{=}\rightarrow\boxed{+}\rightarrow\boxed{C}\rightarrow\boxed{+}\rightarrow\boxed{=}\rightarrow\boxed{+}$. In our simulations, we use the entropy-based stopping criterion to reduce unnecessary iterations. For a decoding window, the decoding process of the target layer stops and the decoding window slides down when the stopping criterion is satisfied or when the maximum number of iterations is reached.

3 Matrix Representation of BiBMST Codes

In this section, we first derive the parity-check matrices of BiBMST codes and then derive their generator matrices. We use $\boldsymbol{G}$ to denote the generator matrix and $\boldsymbol{H}$ to denote the parity-check matrix of $C[n,k]$. Let $\boldsymbol{\Pi}_i$ represent i-th permutation matrix of size $n\times n$. Based on the Step superposition in Algorithm 1, we have

$$\begin{aligned}\boldsymbol{c}^{(t)} =&\boldsymbol{v}^{(t)}+\sum_{i=1}^{m}\boldsymbol{w}_1^{(t-i)}+\sum_{i=1}^{m}\boldsymbol{w}_2^{(t-i)}=\\ &\boldsymbol{v}^{(t)}+\sum_{i=1}^{m}\boldsymbol{z}^{(t-i)}\boldsymbol{\Pi}_i+\sum_{i=1}^{m}\boldsymbol{z}^{(t-i)}\boldsymbol{\Pi}_{m+i}\end{aligned}\tag{2}$$

$$\boldsymbol{z}^{(t)}=\boldsymbol{v}^{(t)}+\sum_{i=1}^{m}\boldsymbol{z}^{(t-i)}\boldsymbol{\Pi}_i\tag{3}$$

From Eq. (3), we have

$$\boldsymbol{v}^{(t)}=\boldsymbol{z}^{(t)}+\sum_{i=1}^{m}\boldsymbol{z}^{(t-i)}\boldsymbol{\Pi}_i\tag{4}$$

Then we have

$$\left(\boldsymbol{v}^{(0)},\boldsymbol{v}^{(1)},\ldots,\boldsymbol{v}^{(L+T-1)}\right)=\left(\boldsymbol{z}^{(0)},\boldsymbol{z}^{(1)},\ldots,\boldsymbol{z}^{(L+T-1)}\right)\boldsymbol{\Pi}^{(1)}\tag{5}$$

where $\boldsymbol{\Pi}^{(1)}$ is an upper bounded block matrix with $(L+T)$ row blocks and $(L+T)$ column blocks, and is given as

$$\boldsymbol{\Pi}^{(1)}=\begin{bmatrix} \boldsymbol{I} & \boldsymbol{\Pi}_1 & \ldots & \boldsymbol{\Pi}_m & & & & \\ & \boldsymbol{I} & \boldsymbol{\Pi}_1 & \ldots & \boldsymbol{\Pi}_m & & & \\ & & \ddots & \ddots & \ddots & \ddots & & \\ & & & \boldsymbol{I} & \boldsymbol{\Pi}_1 & \ldots & \boldsymbol{\Pi}_m \\ & & & & \ddots & \ddots & \vdots \\ & & & & & \boldsymbol{I} & \boldsymbol{\Pi}_1 \\ & & & & & & \boldsymbol{I}\end{bmatrix}.$$

where I is the identity matrix of order L+T.

Similarly, from Eqs. (2) and (3), we have

$$\boldsymbol{c}^{(t)}=\boldsymbol{z}^{(t)}+\sum_{i=1}^{m}\boldsymbol{z}^{(t-i)}\boldsymbol{\Pi}_{m+i}.$$

To derive the relationship between $\boldsymbol{z}^{(t)}$ and $\boldsymbol{c}^{(t)}$, we first define the matrices $\boldsymbol{P}_t$ and $\boldsymbol{P}$. Firstly, let $\boldsymbol{P}_0=\boldsymbol{I}$. For $t>0$, define $\boldsymbol{P}_t$ as $\boldsymbol{P}_t=\sum_{1\leqslant i\leqslant m}\boldsymbol{P}_{t-i}\boldsymbol{\Pi}_{m+i}$, where, for $t<0$, $\boldsymbol{P}_t$ is the all-zero matrix of size $n\times n$. Based on the above definitions, the relationship between $\boldsymbol{z}^{(t)}$ and $\boldsymbol{c}^{(t)}$ is given as

$$\left(\boldsymbol{z}^{(0)},\boldsymbol{z}^{(1)},\ldots,\boldsymbol{z}^{(L+T-1)}\right)=\left(\boldsymbol{c}^{(0)},\boldsymbol{c}^{(1)},\ldots,\boldsymbol{c}^{(L+T-1)}\right)\boldsymbol{P}\tag{6}$$

where $\boldsymbol{P}$ is an upper bounded block matrix with $(L+T)$ row blocks and $(L+T)$ column blocks and is given as

$$\boldsymbol{P}=\begin{bmatrix} \boldsymbol{I} & \boldsymbol{P}_1 & \boldsymbol{P}_2 & \cdots & \boldsymbol{P}_{L+T-1} \\ & \boldsymbol{I} & \boldsymbol{P}_1 & \cdots & \boldsymbol{P}_{L+T-2} \\ & & \ddots & \ddots & \vdots \\ & & & \boldsymbol{I} & \boldsymbol{P}_1 \\ & & & & \boldsymbol{I}\end{bmatrix}.$$

From Eqs. (5) and (6), we have

$$(\boldsymbol{v}^{(0)},\boldsymbol{v}^{(1)},\ldots,\boldsymbol{v}^{(L+T-1)})=(\boldsymbol{c}^{(0)},\boldsymbol{c}^{(1)},\ldots,\boldsymbol{c}^{(L+T-1)})\boldsymbol{P}\boldsymbol{\Pi}^{(1)}\tag{7}$$

Since $\boldsymbol{v}^{(t)}$ is a codeword of the basic code and $\boldsymbol{v}^{(t)}=\boldsymbol{0}\in F_2^n$ for $t\geqslant L$, we have

$$\begin{aligned}&(\boldsymbol{c}^{(0)},\boldsymbol{c}^{(1)},\ldots,\boldsymbol{c}^{(L+T-1)})\boldsymbol{P}\boldsymbol{\Pi}^{(1)}\cdot\\ &\mathrm{diag}(\underbrace{\boldsymbol{H}^{\mathrm{T}},\ldots,\boldsymbol{H}^{\mathrm{T}}}_{L},\underbrace{\boldsymbol{I},\ldots,\boldsymbol{I}}_{T})=0\end{aligned}\tag{8}$$

where the superscript T denotes matrix transposition, and $\mathrm{diag}(\boldsymbol{H}^{\mathrm{T}},\ldots,\boldsymbol{H}^{\mathrm{T}},\boldsymbol{I},\ldots,\boldsymbol{I})$ is the block diagonal matrix with $\boldsymbol{H}^{\mathrm{T}}$ and $\boldsymbol{I}$ on the diagonal. Therefore, the parity-check matrix of the BiBMST code is shown as

$$\boldsymbol{H}_{\mathrm{BiBMST}}=\mathrm{diag}(\underbrace{\boldsymbol{H}^{\mathrm{T}},\ldots,\boldsymbol{H}^{\mathrm{T}}}_{L},\underbrace{\boldsymbol{I},\ldots,\boldsymbol{I}}_{T})(\boldsymbol{\Pi}^{(1)})^{\mathrm{T}}\boldsymbol{P}^{\mathrm{T}}\tag{9}$$

A protograph-based SC-GLDPC code with coupling width (syndrome former memory) w and coupling length L_{SC} can be described as

$$\boldsymbol{B}_{\text{SC-GLDPC}} = \begin{bmatrix} \boldsymbol{B}_0 & & & \\ \boldsymbol{B}_1 & \boldsymbol{B}_0 & & \\ \vdots & \boldsymbol{B}_1 & \ddots & \\ \boldsymbol{B}_w & \vdots & \ddots & \boldsymbol{B}_0 \\ & \boldsymbol{B}_w & \ddots & \boldsymbol{B}_1 \\ & & \ddots & \vdots \\ & & & \boldsymbol{B}_w \end{bmatrix},$$

where component base matrices $\boldsymbol{B}_i\,(0 \leqslant i \leqslant w)$ of size $b_c \times b_v$ represent the edge connections from variable nodes to generalized constraint (GC) nodes corresponding to an (n,k) linear block code. This protograph is then subjected to a graph lifting operation, which is done by replacing all nonzero entry B_{ij} in $\boldsymbol{B}_{\text{SC-GLDPC}}$ with a sum of B_{ij} nonoverlapping randomly selected $M \times M$ permutation matrices and replacing all-zero entry B_{ij} with the all-zero matrix of size $M \times M$, thereby creating the parity-check matrix of size $(L+w)Mb_c \times LMb_v$.

Similarly, the parity-check matrix of BiBMST code is

$$\boldsymbol{H}_{\text{BiBMST}} = \begin{bmatrix} \boldsymbol{HS}_0 & & & & & \\ \vdots & \boldsymbol{HS}_0 & & & & \\ \boldsymbol{HS}_{L-1} & \ddots & \ddots & & & \\ \boldsymbol{S}_L & \boldsymbol{S}_{L-1} & \ddots & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \boldsymbol{S}_0 & \\ \boldsymbol{S}_{L+T-1} & \boldsymbol{S}_{L+T-2} & \cdots & \cdots & \boldsymbol{S}_1 & \boldsymbol{S}_0 \end{bmatrix},$$

where $\boldsymbol{S}_j = \Sigma_{i=0}^{m} \boldsymbol{\Pi}_i^{\mathrm{T}} \boldsymbol{P}_{j-i}^{\mathrm{T}}$ and default $\boldsymbol{\Pi}_0$ is the all-zero matrix. Noting that both BiBMST codes and SC-GLDPC codes are constructed with basic codes. They are different in the following aspects.

(1) Firstly, a BiBMST code is defined by its encoding process. Instead, the GLDPC codes and SC-GLDPC codes are defined by Tanner graphs.

(2) Secondly, the parity-check matrices of BiBMST codes are always dense. On the other hand, the parity-check matrices of GLDPC and SC-GLDPC codes are typically sparse. Furthermore, the parity-check matrices of BiBMST codes are not used for decoding.

For completeness, we derive the generator matrices of BiBMST codes. We complete the derivation in two steps. We first define the matrices $\boldsymbol{Q}_t$ as follows:

$$\boldsymbol{Q}_t = \begin{cases} \boldsymbol{0}, & \text{if } t < 0; \\ \boldsymbol{I}, & \text{if } t = 0; \\ \sum_{1 \leqslant i \leqslant m} \boldsymbol{Q}_{t-i} \boldsymbol{\Pi}_i, & \text{if } t > 0; \end{cases}$$

where $\boldsymbol{0}$ is the $n \times n$ all-zero matrix and $\boldsymbol{I}$ is the $n \times n$ identity matrix. From Eqs. (2) and (3), we have

$$\boldsymbol{c}^{(t)} = \boldsymbol{z}^{(t)} + \sum_{i=1}^{m} \boldsymbol{z}^{(t-i)} \boldsymbol{\Pi}_{m+i} \tag{10}$$

Therefore, we obtain the generator matrix as

$$\boldsymbol{G}_{\text{BiBMST}} = \text{diag}(\underbrace{\boldsymbol{G}, \ldots, \boldsymbol{G}}_{L}) \boldsymbol{Q} \boldsymbol{\Pi}^{(2)} \tag{11}$$

where $\boldsymbol{Q}$ is a block matrix with L row blocks and $L+T$ column blocks, and $\boldsymbol{\Pi}^{(2)}$ is an upper bounded block matrix with $L+T$ row blocks and $L+T$ column blocks. The matrix $\boldsymbol{Q}$ is given as

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{Q}_1 & \boldsymbol{Q}_2 & \cdots & \cdots & \cdots & \boldsymbol{Q}_{L+T-1} \\ & \boldsymbol{I} & \boldsymbol{Q}_1 & \cdots & \cdots & \cdots & \boldsymbol{Q}_{L+T-2} \\ & & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & & \boldsymbol{I} & \boldsymbol{Q}_1 & \cdots & \boldsymbol{Q}_T \end{bmatrix}.$$

Apparently, $\text{Rank}(\boldsymbol{G}_{\text{BiBMST}}) = kL$ since the rank of $\boldsymbol{G}$ is k, and $\boldsymbol{Q}\boldsymbol{\Pi}^{(2)}$ is of full rank.

Remarks: When $m > 1$, the parity-check matrix $\boldsymbol{H}_{\text{BiBMST}}$ and the generator matrix $\boldsymbol{G}_{\text{BiBMST}}$ of a BiBMST code are both dense matrices irrespective of the densities of $\boldsymbol{H}$ and $\boldsymbol{G}$. This is different from SC-LDPC codes, SC-LDGM codes, BMST codes, and rBMST codes. The parity-check matrices of SC-LDPC codes are sparse, and the generator matrices of SC-LDGM codes are sparse. Furthermore, the BMST codes admit sparse generator matrices and dense parity-check matrices, while the rBMST codes admit sparse parity-check matrices and dense generator matrices.

Example 1: Consider the BiBMST with the basic code $\mathcal{C}_{\text{rc}}[4,2]$. The generator matrix and parity-check matrix of $\mathcal{C}_{\text{rc}}[4,2]$ are

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

Let $m = 2$ and $T = 2$. Similarly, We assume that the four interleavers used for encoding are defined by the permutation matrices $\boldsymbol{\Pi}_1$, $\boldsymbol{\Pi}_2$, $\boldsymbol{\Pi}_3$, and $\boldsymbol{\Pi}_4$ as follows:

$$\boldsymbol{\Pi}_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \text{and } \boldsymbol{\Pi}_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

$$\boldsymbol{\Pi}_3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \text{and } \boldsymbol{\Pi}_4 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

Based on our derivation, when $L = 1$, the generator matrix $\boldsymbol{G}_1$ is

$$\boldsymbol{G}_1=\boldsymbol{G}\cdot\begin{bmatrix}\boldsymbol{I} & \boldsymbol{Q}_1 & \boldsymbol{Q}_2\end{bmatrix}\cdot\begin{bmatrix}\boldsymbol{I} & \boldsymbol{\varPi}_3 & \boldsymbol{\varPi}_4\\ 0 & \boldsymbol{I} & \boldsymbol{\varPi}_3\\ 0 & 0 & \boldsymbol{I}\end{bmatrix}=$$

$$\begin{bmatrix}1&0&1&0&0&1&1&0&1&1&1&1\\0&1&0&1&0&1&1&0&1&1&1&1\end{bmatrix},$$

and the parity-check matrix $\boldsymbol{H}_1$ is

$$\boldsymbol{H}_1=\begin{bmatrix}1&0&1&0&0&0&0&0&0&0&0&0\\0&1&0&1&0&0&0&0&0&0&0&0\\0&0&0&0&1&0&0&0&0&0&0&0\\1&0&0&1&0&1&0&0&0&0&0&0\\1&0&0&1&0&0&1&0&0&0&0&0\\0&0&0&0&0&0&0&1&0&0&0&0\\0&0&1&1&0&0&0&0&1&0&0&0\\0&1&1&0&1&0&0&1&0&1&0&0\\0&1&1&0&1&0&0&1&0&0&1&0\\0&0&1&1&0&0&0&0&0&0&0&1\end{bmatrix}.$$

It can be checked that $\boldsymbol{G}_1\boldsymbol{H}_1^{\mathrm{T}}$ is a zero matrix.

When $L=2$, the generator matrix $\boldsymbol{G}_2$ is given as

$$\boldsymbol{G}_2=\begin{bmatrix}\boldsymbol{G} & \boldsymbol{F}_1 & \boldsymbol{F}_2 & \boldsymbol{F}_3\\ \boldsymbol{0} & \boldsymbol{G} & \boldsymbol{F}_1 & \boldsymbol{F}_2\end{bmatrix},$$

where $\boldsymbol{F}_1=\boldsymbol{G}(\boldsymbol{\varPi}_3+\boldsymbol{Q}_1)$, $\boldsymbol{F}_2=\boldsymbol{G}(\boldsymbol{\varPi}_4+\boldsymbol{\varPi}_3\boldsymbol{Q}_1+\boldsymbol{Q}_2)$, and $\boldsymbol{F}_3=\boldsymbol{G}(\boldsymbol{\varPi}_4\boldsymbol{Q}_1+\boldsymbol{\varPi}_3\boldsymbol{Q}_2+\boldsymbol{Q}_3)$, and the parity-check matrix $\boldsymbol{H}_2$ is

$$\boldsymbol{H}_2=\begin{bmatrix}\boldsymbol{H} & 0 & 0 & 0\\ \boldsymbol{H}\boldsymbol{S}_1 & \boldsymbol{H} & 0 & 0\\ \boldsymbol{S}_2 & \boldsymbol{S}_1 & \boldsymbol{I} & 0\\ \boldsymbol{S}_3 & \boldsymbol{S}_2 & \boldsymbol{S}_1 & \boldsymbol{I}\end{bmatrix},$$

where $\boldsymbol{S}_1=\boldsymbol{\varPi}_1^{\mathrm{T}}+\boldsymbol{P}_1^{\mathrm{T}}$, $\boldsymbol{S}_2=\boldsymbol{\varPi}_2^{\mathrm{T}}+\boldsymbol{\varPi}_1^{\mathrm{T}}\boldsymbol{P}_1^{\mathrm{T}}+\boldsymbol{P}_2^{\mathrm{T}}$, and $\boldsymbol{S}_3=\boldsymbol{\varPi}_2^{\mathrm{T}}\boldsymbol{P}_1^{\mathrm{T}}+\boldsymbol{\varPi}_1^{\mathrm{T}}\boldsymbol{P}_2^{\mathrm{T}}+\boldsymbol{P}_3^{\mathrm{T}}$.

4 Iterative Decoding Thresholds Analysis of BiBMST

In this section, we drive the PEXIT[34] functions to predict the iterative decoding thresholds of the BiBMST codes over the binary erasure channels (BECs).

4.1 Transfer function of the component decoder

A BiBMST code can be constructed based on protograph. Figure 3a illustrates the protograph of an uncoupled BiBMST code with $m=2$. In this protograph, the information nodes and the signal parity-check nodes $\boxplus$ are connected by the generator matrix $\boldsymbol{G}$ of basic code. The protograph of an uncoupled BiBMST code is replicated L times, with each protograph labeled by its corresponding time t. The edges of the nodes $\boxed{=}$ at time t are then reconnected to the nodes $\boxed{=}$ at time $t+k$, for $0\leqslant t\leqslant L-1$ and

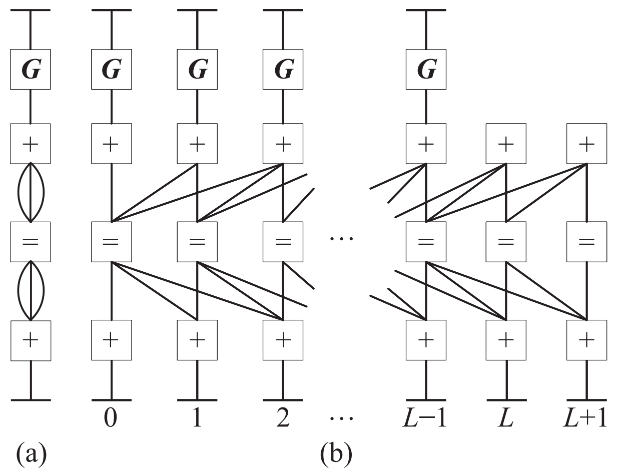

Fig. 3 Protograph corresponding to (a) an uncoupled BiBMST code with $m=2$ and (b) a BiBMST code with $m=2$ and $T=2$.

$1\leqslant k\leqslant m$, resulting in a couple of chains that correspond to a BiBMST code. An example of a BiBMST code with encoding memory $m=2$ is shown in Fig. 3b. Similar to the protograph-based LDPC codes, PEXIT charts can be used to predict the iterative decoding thresholds for BiBMST codes when the sizes of the interleavers are infinite. To derive the PEXIT functions for BiBMST codes, we need the transfer functions for all nodes in the protograph, including the node $\boxed{=}$, the node $\boxplus$, and the basic decoder node.

When the input message number is j, both node $\boxplus$ and node $\boxed{=}$ have a degree of $j+1$. In BEC, the extrinsic mutual information (MI) between a node's message transmission and its corresponding codeword bit is equal to its erasure probability. Let x_i be the input erasure probability to a node of degree $j+1$, where $1\leqslant i\leqslant j$. We use P_{plu} and P_{eq} to denote the extrinsic output erasure probability of a node $\boxplus$ and a node $\boxed{=}$, respectively. Then we have

$$P_{\mathrm{plu}}=f_{\mathrm{plu}}(x_1,x_2,\ldots,x_j)=1-(1-x_1)(1-x_2)\cdots(1-x_j) \tag{12}$$

and

$$P_{\mathrm{eq}}=f_{\mathrm{eq}}(x_1,x_2,\ldots,x_j)=x_1x_2\cdots x_j \tag{13}$$

where f_{plu} denotes the transfer function of the node $\boxplus$, and f_{eq} denotes the transfer function of the node $\boxed{=}$.

4.2 PEXIT charts

Let ϵ denote the channel erasure probability of the BEC. For a BiBMST code with basic code C, in the i-th iteration, we use $p_{\mathrm{eq},i}^{(t,t+j)}$ to denote the extrinsic erasure probability from node $\boxminus$ of the t-th layer to the lower node $\boxplus$ of the $(t+j)$-th layer. Similarly, the extrinsic erasure probability from node $\boxminus$ of the t-th layer to the upper node $\boxplus$ of the $(t+j)$-th layer is denoted as $q_{\mathrm{eq},i}^{(t,t+j)}$. In addition, we let $p_{\mathrm{plu},i}^{(t,t+j)}$ and $q_{\mathrm{plu},i}^{(t,t+j)}$ denote the extrinsic erasure probabilities from the lower node $\boxplus$ of the t-th layer and the upper node $\boxplus$ of the t-th layer to the node $\boxminus$ of the $(t+j)$-th layer, respectively. Meanwhile, in the i-th iteration, we use $C_{\mathrm{in},i}^{(t)}$ and $C_{\mathrm{out},i}^{(t)}$ to denote the extrinsic input erasure probability and the extrinsic output erasure probability of the basic decoder of the t-th layer, respectively.

For the initialization of the extrinsic information erasure probabilities of the above nodes, if $i<0$ or $t_1<0$, we set $p_{\mathrm{plu},i}^{(t_1,t_2)}=q_{\mathrm{plu},i}^{(t_1,t_2)}=1$ and $p_{\mathrm{eq},i}^{(t_1,t_2)}=q_{\mathrm{eq},i}^{(t_1,t_2)}=0$. We first derive the PEXIT functions for the forward decoding process. The PEXIT functions within a layer are written as

$$
\begin{aligned}
p_{\mathrm{plu},i}^{(t,t)} &= f_{\mathrm{plu}}\left(\epsilon, 1-\prod_{k=1}^{m}\left(1-p_{\mathrm{eq},i}^{(t-k,t)}\right)\right),\\
q_{\mathrm{eq},i}^{(t,t)} &= f_{\mathrm{eq}}\left(p_{\mathrm{plu},i}^{(t,t)}, \prod_{k=1}^{m} p_{\mathrm{plu},i-1}^{(t+k,t)}, \prod_{k=1}^{m} q_{\mathrm{plu},i-1}^{(t+k,t)}\right),\\
C_{\mathrm{in},i}^{(t)} &= f_{\mathrm{plu}}\left(q_{\mathrm{eq},i}^{(t,t)}, 1-\prod_{k=1}^{m}\left(1-q_{\mathrm{eq},i}^{(t-k,t)}\right)\right)
\end{aligned}
\tag{14}
$$

The PEXIT functions for information updating from the t-th layer to the $(t+l)$-th layer are written as

$$
\begin{aligned}
q_{\mathrm{plu},i}^{(t,t)} &= f_{\mathrm{plu}}\left(C_{\mathrm{out},i}^{(t)}, 1-\prod_{k=0}^{m}\left(1-q_{\mathrm{eq},i}^{(t-k,t)}\right)\right),\\
q_{\mathrm{eq},i}^{(t,t+l)} &= q_{\mathrm{plu},i}^{(t,t)} p_{\mathrm{plu},i}^{(t,t)} \prod_{k=1,k\neq l}^{m} q_{\mathrm{plu},i-1}^{(t+k,t)} \prod_{k=1}^{m} p_{\mathrm{plu},i-1}^{(t+k,t)},\\
p_{\mathrm{eq},i}^{(t,t+l)} &= q_{\mathrm{plu},i}^{(t,t)} p_{\mathrm{plu},i}^{(t,t)} \prod_{k=1,k\neq l}^{m} p_{\mathrm{plu},i-1}^{(t+k,t)} \prod_{k=1}^{m} q_{\mathrm{plu},i-1}^{(t+k,t)}
\end{aligned}
\tag{15}
$$

where $1\leqslant l\leqslant m$.

We then derive the PEXIT functions for the backward decoding process. The PEXIT functions within the t-th layer are written as

$$
\begin{aligned}
p_{\mathrm{plu},i}^{(t,t)} &= f_{\mathrm{plu}}\left(\epsilon, 1-\prod_{k=1}^{m}\left(1-p_{\mathrm{eq},i}^{(t-k,t)}\right)\right),\\
q_{\mathrm{eq},i}^{(t,t)} &= f_{\mathrm{eq}}\left(\prod_{k=0}^{m} p_{\mathrm{plu},i}^{(t+k,t)}, \prod_{k=1}^{m} q_{\mathrm{plu},i}^{(t+k,t)}\right),\\
C_{\mathrm{in},i}^{(t)} &= f_{\mathrm{plu}}\left(q_{\mathrm{eq},i}^{(t,t)}, 1-\prod_{k=1}^{m}\left(1-q_{\mathrm{eq},i}^{(t-k,t)}\right)\right)
\end{aligned}
\tag{16}
$$

The updating functions from the t-th layer to the $(t-l)$-th layer are written as

$$
\begin{aligned}
q_{\mathrm{plu},i}^{(t,t-h)} &= f_{\mathrm{plu}}\left(C_{\mathrm{out},i}^{(t)}, 1-\prod_{k=0,k\neq h}^{m}\left(1-q_{\mathrm{eq},i}^{(t-k,t)}\right)\right),\\
p_{\mathrm{eq},i}^{(t,t)} &= f_{\mathrm{eq}}\left(\prod_{k=0}^{m} q_{\mathrm{plu},i}^{(t+k,t)}, \prod_{k=1}^{m} p_{\mathrm{plu},i}^{(t+k,t)}\right),\\
p_{\mathrm{plu},i}^{(t,t-l)} &= f_{\mathrm{plu}}\left(\epsilon, p_{\mathrm{eq},i}^{(t,t)}, 1-\prod_{k=1,k\neq l}^{m}\left(1-q_{\mathrm{eq},i}^{(t-k,t)}\right)\right)
\end{aligned}
\tag{17}
$$

where $1\leqslant l\leqslant m$ and $0\leqslant h\leqslant m$.

For the basic code C_{rc}, we let $f_{\mathrm{rc}}(\cdot)$ denote the transfer function of the information bits of the repetition decoder. Therefore, for the t-th layer, the a posteriori erasure probability of its information bits is computed as

$$
P_{\mathrm{app}} = f_{\mathrm{rc}}(C_{\mathrm{in},i}^{(t)})C_{\mathrm{in},i}^{(t)}
\tag{18}
$$

We can use Eqs. (14)–(17) iteratively to compute the iterative decoding threshold of a BiBMST code ensemble over the BECs. The iterative decoding threshold $\epsilon_{\mathrm{BP}}^{(m)}$ is the maximal erasure probability ϵ such that $P_{\mathrm{app}}<P_b$ when $I_{\max}$ iterations are executed.

We investigate the effect of encoding memory on the iterative decoding thresholds of BiBMST codes. The B-fold Cartesian product of repetition codes $C_{\mathrm{rc}}[2B,B]$, $C_{\mathrm{rc}}[3B,B]$, and $C_{\mathrm{rc}}[4B,B]$ and the single parity-check codes $C_{\mathrm{spc}}[4B,3B]$ and $C_{\mathrm{spc}}[3B,2B]$ are selected as the basic codes. We list the iterative decoding thresholds of the considered BiBMST code ensembles in Table 1. At the same time, for comparison, we list the iterative decoding thresholds of the comparable rBMST code ensembles with the same basic codes. When computing the thresholds, we set $I_{\max}=1000$ and $P_b=10^{-10}$.

From Table 1, we have following observations. Firstly, the iterative decoding thresholds of BiBMST codes and rBMST codes improve with the increase of the encoding memory m, and the gain is marginal when

Table 1 Iterative decoding thresholds of BiBMST codes and rBMST codes.

Rate	BiBMST			rBMST	
	$\epsilon_{\mathrm{BP}}^{(m=1)}$	$\epsilon_{\mathrm{BP}}^{(m=2)}$	$\epsilon_{\mathrm{BP}}^{(m=3)}$	$\epsilon_{\mathrm{BP}}^{(m=2)}$	$\epsilon_{\mathrm{BP}}^{(m=3)}$
3/4	0.2495	0.2497	0.2497	0.2372	0.2468
2/3	0.3327	0.3330	0.3330	0.3195	0.3301
1/2	0.4989	0.4996	0.4996	0.4880	0.4975
1/3	0.6660	0.6663	0.6663	0.6649	0.6663
1/4	0.7494	0.7496	0.7496	0.7494	0.7498

the encoding memory is large enough. Secondly, the thresholds of the BiBMST codes approach the BEC capacities for all considered rates. Particularly, we observe that the thresholds of the BiBMST code ensemble based on the rate half repetition code are within 0.0004 of the BEC capacity when $m = 3$. Thirdly, the iterative decoding thresholds of BiBMST codes are higher than those of the rBMST codes for a given encoding memory.

5 Analytical Result and Performance

In this section, we firstly investigate the impact of various parameters on BiBMST codes, including decoding delay d and encoding memory m. Secondly, we compare the performances of BiBMST codes and rBMST codes under the constraint of an equal decoding latency. The simulation results are obtained under the additive white Gaussian noise (AWGN) channels and binary phase-shift keying (BPSK) modulation. In addition, the interleavers are randomly generated but fixed. In all following examples, the iterative sliding window decoding algorithm with a maximum number of iterations $I_{\max} = 18$ is used for decoding. The coupling length is selected as $L = 1000$. The rate half length repetition code $C_{\rm rc}[2B, B]$ is selected as the basic code. The entropy stopping criterion is employed.

5.1 Fixed *B*, increasing *m* (and hence *d*)

Example 2: In order to study the impacts of encoding memory m on the performances of BiBMST codes, we consider a family of BiBMST codes with encoding memories $m = 1$, 2, 3, and 4. Notice that, to obtain optimal performance, the decoding delay should increase with the increase of the encoding memory. We give the bit error rate (BER) of the considered BiBMST codes for $B = 1000$ and $B = 1666$ in Fig. 4, where $E_{\rm b}/N_0$ denotes the received bits signal-to-noise ratio (SNR) on the AWGN channel in dB.

From Fig. 4, we can find that when the encoding memory increases from 1 to 2, the performance improvement is significant. From the performance point of view, for finite-length BiBMST codes, preferred encoding memory is $m = 2$. We also observe that the performance gain obtained by increasing the encoding memory is negligible when $m \geqslant 2$. Conversely, simulation results show that performance degrades as encoding memory increases. This is partially explained as follows. When m is large, a decoding error in the current layer will contaminate m consecutive layers. This phenomenon is severe when the SNR is low.

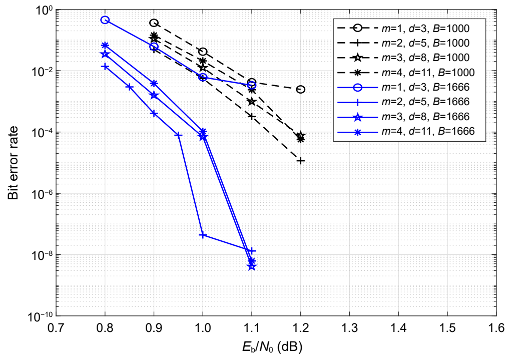

Fig. 4 Simulated decoding performance of BiBMST codes with different encoding memory *m*.

5.2 Fixed *m*, increasing *d*

Example 3: In the previous subsection, the simulation results showed that when the encoding memory of BiBMST codes exceeds 2, negligible performance gain is obtained by increasing m under the finite-length regime. We study the effects of decoding delay d on the performances of BiBMST codes. The simulated decoding delays are $d = 4$, 5, 6, and 7. The repetition codes with $B = 1000$ and $B = 2500$ are selected as basic codes. We show the performances in Fig. 5, where we observe that

(1) In the waterfall region, the performance of BiBMST codes improves as the decoding delay d increases. However, the performance gain saturates when $d \geqslant 6$. It is known that in the literature, in finite-length regime, the decoding delay $d \geqslant 2m - 3m$ is near optimal for iterative sliding window decoder.

(2) The error floor improves as the decoding delay d increases.

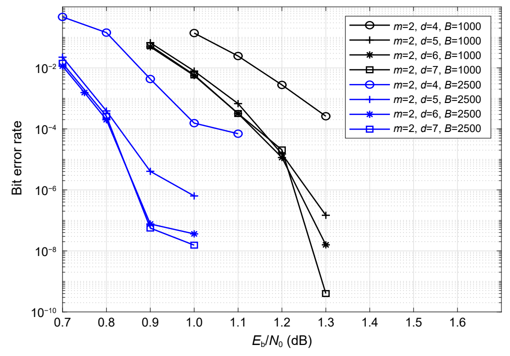

Fig. 5 Simulated decoding performance of BiBMST codes decoded with different decoding delay *d*.

(3) For a given m and a given d, performance improvement can be obtained by increasing the code length of the basic code (increasing B).

5.3 Performance and complexity comparison with rBMST codes

The results of Examples 2 and 3 have shown that for the BiBMST codes based on rate half repetition code, the best performance is obtained with $m = 2$. This is consistent with the iterative decoding thresholds given in Table 1. Furthermore, for BiBMST codes with encoding memory $m = 2$, performance gain in the waterfall region is marginal when the decoding delay exceeds 6. Hence, for performance comparison in the finite-length regime, we select the BiBMST codes with $m = 2$ and $d = 6$.

Example 4: For a BiBMST code with the basic code $C[n,k]$, its decoding latency under the iterative sliding window decoder is given as

$$T_{\mathrm{BiBMST}} = n(d+1).$$

We give the performances of the BiBMST codes based on rate half repetition code for a decoding latency of 20 000 bits in Fig. 6a. The simulated decoding delays are $d = 4$, 5, 6, 7, and 8. From Fig. 6a, we have the following observations.

(1) Under the constraint of a given decoding latency T_{BiBMST}, the BERs of the BiBMST codes in the waterfall region degrade when we increase the decoding delay.

(2) Under the constraint of a given decoding latency T_{BiBMST}, the error floor improves as the decoding delay d increases. Similar observation has been obtained for rBMST codes in Ref. [32].

For further illustration, we also present the performances of the BiBMST codes based on rate half repetition code for a given decoding latency of 30 000 bits. Observations similar to the case with $T_{\mathrm{BiBMST}} = 20\,000$ have been obtained for this case.

In the following, we compare the performances of BiBMST codes and rBMST codes under the constraints of an equal decoding latency. The numerical results in Ref. [32] showed that, in the finite-length regime, the rBMST codes with $m = 3$ have superior performance. In our simulations, the decoding delays for the rBMST codes are chosen as $d = 9$ for the decoding latency of 20 000 bits and $d = 11$ for the decoding latency of 30 000 bits. The simulation results are shown in Fig. 6. We observe that the BiBMST code with $B = 1428$ and $d = 6$ outperforms the rBMST code with $B = 1000$ and $d = 9$. Similarly, for the decoding latency of 30 000 bits, the BiBMST code with $B = 2142$ and $d = 6$ outperforms the rBMST code with $B = 1250$ and $d = 11$.

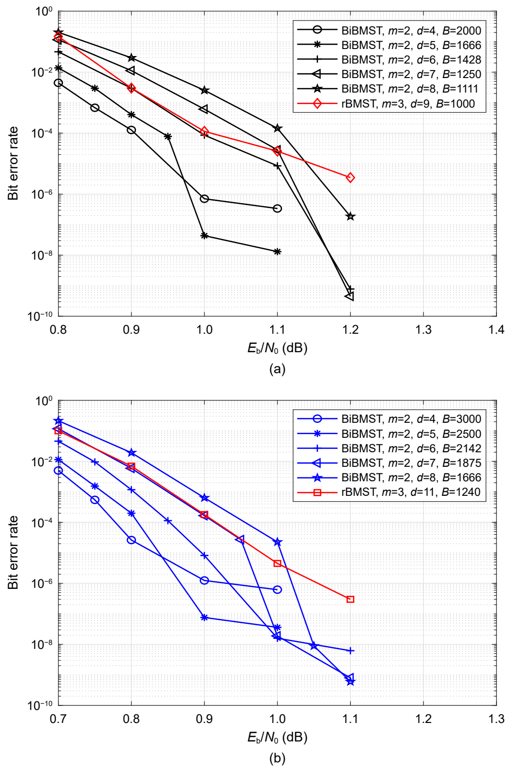

Fig. 6 Error performances of the BiBMST codes with encoding memory m = 2 and the rBMST codes with encoding memory m = 3: (a) a decoding latency of 20 000 bits and (b) a decoding latency of 30 000 bits.

In the following, we compare the decoding complexities of the BiBMST code and the rBMST code. As described in Ref. [32], the average computational complexity of rBMST code within a decoding window is $52mn(d+1)I_{\mathrm{rBMST}}$, where n denotes the length of basic code and I_{rBMST} denotes the average number of iterations needed to obtain a target performance.

Let $\mathrm{Opt}(A)$ denote the number of computation of node A in the normal graph (see Fig. 2 for reference). For a decoding delay d, the numbers of decoding layers in a decoding window is fixed as $W = d+1$. Let I_{BiBMST} denote the average number of iterations required to obtain a target performance. Hence, the average computational complexity within a decoding window is computed as $(d+1)(2n\mathrm{Opt}(\boxplus) + n\mathrm{Opt}(\boxed{=}) + \mathrm{Opt}(\boxed{\mathrm{C}}))I_{\mathrm{BiBMST}}$. Specifically, the average computational complexity of a BiBMST code based on rate half repetition code of length n is given as $104mn(d+1)I_{\text{BiBMST}}$.

Table 2 shows the average computational complexity for the BiBMST code and the rBMST code used in Fig. 6b. The target BER is 10^{-6}. We also give the values of the parameters B, I_{BiBMST}, I_{rBMST}, and E_b/N_0 in Table 2. We observe that the average number of iteration I_{BiBMST} of the BiBMST code is greater than I_{rBMST}. However, since decoding delay used for BiBMST code is smaller than that used for rBMST code, the computational complexity per coded bit for the BiBMST code is lower than that of the rBMST code.

6 Partially-Connected BiBMST Codes

6.1 Encoding of PC-BiBMST codes

The analytical and numerical results have illustrated the superiority of BiBMST codes over rBMST codes in computational complexity and performance. However, in finite-length regime, we observe that increasing the encoding memory may incur performance degradation. In addition, we also observe that, under the constraint of a fixed decoding latency, BiBMST codes cannot perform well in the waterfall and the error floor regions simultaneously. That is, for the BiBMST codes, good performance in the waterfall region is accompanied by high error floor. To address this issue, we further present the class of PC-BiBMST codes.

The PC-BiBMST code has a similar encoding structure as that of the original BiBMST code. The difference is that, in the encoding of PC-BiBMST code, the matrix $\boldsymbol{\Pi}_i$ can be an $n\times n$ all-zero matrix. To make the encoding structure clear, we define the connection pattern $\text{Bi}(\boldsymbol{E}_1,\boldsymbol{E}_2)$ for PC-BiBMST codes, where the $1\times(m+1)$ matrix $\boldsymbol{E}_1$ and the $1\times(m+1)$ matrix $\boldsymbol{E}_2$ denote the non-recursive connection pattern and the recursive connection pattern, respectively. For $i=1,2,\ldots,m$, if $\boldsymbol{\Pi}_i=\boldsymbol{0}$, the i-th element $e_{1,i}$ of $\boldsymbol{E}_1$ is equal to 0; otherwise, $e_{1,i}=1$. We define the matrix $\boldsymbol{E}_2$ similarly. For $i=m+1,m+2,\ldots,2m$, if $\boldsymbol{\Pi}_{m+i}=\boldsymbol{0}$, the i-th element $e_{2,i}$ of $\boldsymbol{E}_2$ is equal to 0; otherwise, $e_{2,i}=1$. Please note that we have $e_{2,0}=e_{1,0}=1$. We point out that there is a one-to-one correspondence between the connection pattern $\text{Bi}(\boldsymbol{E}_1,\boldsymbol{E}_2)$ and a PC-BiBMST encoder. Since the encoding memory of a PC-BiBMST code may be smaller than m, we call the parameter m the maximum allowable encoding memory. The number of nonzero elements in $\text{Bi}(\boldsymbol{E}_1,\boldsymbol{E}_2)$ is referred to as the superposition degree and is denoted as D. The encoding diagram of a PC-BiBMST code with $m=3$ and $\text{Bi}(1101,1011)$ is shown in Fig. 7, where we have $D=6$.

Table 2 Decoding complexity per coded bit of BiBMST code and rBMST code that achieve the BER=10^{-6} with a decoding latency of 30 000 bits.

Code	B	d	I_{BiBMST} or I_{rBMST}	E_b/N_0	Complexity
BiBMST	2142	6	2.41	0.96	3508
rBMST	1250	9	2.01	1.05	3762

For a maximum allowable encoding memory m, the number of possible choices of $\text{Bi}(\boldsymbol{E}_1,\boldsymbol{E}_2)$ is 2^{2m}. For example, when the maximum allowable encoding memory is 5, the number of possible choices of $\text{Bi}(\boldsymbol{E}_1,\boldsymbol{E}_2)$ is 1024. For a given m, it is time-consuming to simulate all of the 2^{2m} possible PC-BiBMST codes for code optimization. In this paper, we use the iterative decoding thresholds analysis to optimize the structure of PC-BiBMST code. The PEXIT functions derived in Section 4 can be easily modified for such a purpose.

6.2 Optimization of PC-BiBMST codes

Consider the PC-BiBMST codes built upon $C_{\text{rc}}[2B,B]$, $C_{\text{rc}}[3B,B]$, and $C_{\text{spc}}[3B,2B]$, where $C_{\text{spc}}[3B,2B]$ denotes the single parity-check code of length $3B$. We let ϵ^* denote the optimal iterative decoding threshold for a given maximum allowable encoding memory m and a given superposition degree D. The optimized iterative decoding thresholds and corresponding connection patterns $\text{Bi}(\boldsymbol{E}_1,\boldsymbol{E}_2)$ are reported in Fig. 8 and Table 3.

The results in Fig. 8 show that, for all considered basic codes and all considered maximum allowable encoding memories, the optimal iterative decoding

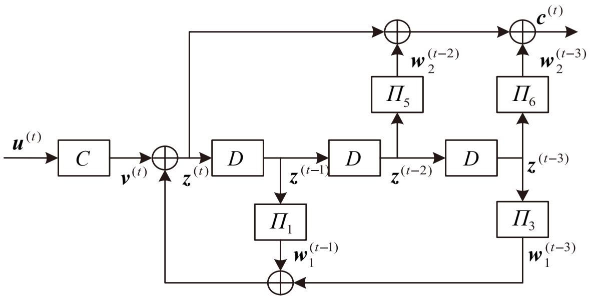

Fig. 7 Encoding structures of PC-BiBMST code with maximum allowable encoding memory $m=3$ and connection pattern Bi(1011, 1101).

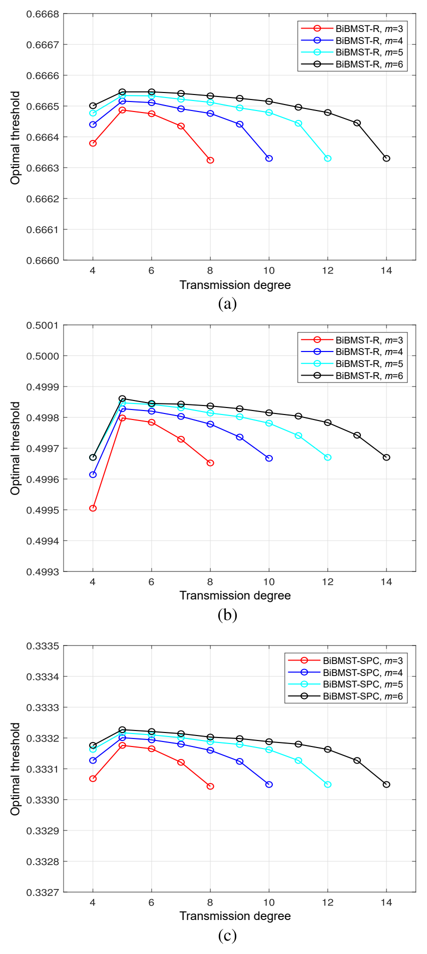

Fig. 8 Optimal iterative decoding thresholds of the BiBMST codes with different maximum allowable encoding memory *m*. (a) The repetition code $C_{\rm rc}[3B, B]$, (b) the repetition code $C_{\rm rc}[2B, B]$, and (c) the single parity-check code $C_{\rm spc}[3B, 2B]$ are selected as the basic codes.

thresholds first increase and then decrease with the increase of the superposition degree D. We point out that, for all cases, the optimal iterative decoding thresholds are always obtained when $D = 5$, and the optimal thresholds are very close to the BEC capacities when $m \geqslant 5$. This shows that the PC-BiBMST codes have near-capacity performance under low complexity decoding.

From Table 3, we find that, for a given maximum allowable encoding memory m and a given superposition degree D, the connection pattern with optimal threshold is not necessarily the same for different basic codes. When the superposition degree D is small, it is preferred to use sparse connection for non-recursive connection pattern $\boldsymbol{E}_1$. The results also show that, in the proposed bidirectional encoding structure, the recursive connection and the non-recursive connection are both indispensable for performance enhancement.

Table 3 Optimal thresholds and the corresponding connection patterns for PC-BiBMST codes under different maximum allowable encoding memories.

Rate	m	D	$\mathrm{Bi}(\boldsymbol{E}_1, \boldsymbol{E}_2)$	ϵ^*
2/3	4	5	Bi(10001, 11010)	0.333 201
		6	Bi(10001, 11011)	0.333 194
		7	Bi(10001, 11111)	0.333 180
	5	5	Bi(100001, 110100)	0.333 217
		6	Bi(100001, 110101)	0.333 210
		7	Bi(100001, 110111)	0.333 201
	6	5	Bi(1000001, 1100100)	0.333 227
		6	Bi(1000001, 1101001)	0.333 221
		7	Bi(1000001, 1101011)	0.333 214
1/2	4	5	Bi(10001, 11001)	0.499 828
		6	Bi(10001, 11011)	0.499 820
		7	Bi(10001, 11111)	0.499 803
	5	5	Bi(100001, 110001)	0.499 847
		6	Bi(100001, 110011)	0.499 842
		7	Bi(100001, 110111)	0.499 831
	6	5	Bi(1000001, 1101000)	0.499 861
		6	Bi(1000010, 1100101)	0.499 845
		7	Bi(1000001, 1101101)	0.499 843
1/3	4	5	Bi(10001, 11001)	0.666 516
		6	Bi(10001, 11011)	0.666 511
		7	Bi(10001, 11111)	0.666 491
	5	5	Bi(100001, 110001)	0.666 534
		6	Bi(100001, 110011)	0.666 533
		7	Bi(100001, 110111)	0.666 522
	6	5	Bi(1000001, 1100010)	0.666 546
		6	Bi(1000001, 1100011)	0.666 546
		7	Bi(1000001, 1100111)	0.666 541

Example 5: To verify the results of iterative decoding thresholds analysis in Fig. 8, we simulate the PC-BiBMST codes over the BEC channels. The basic code is $C_{\rm rc}[2B, B]$, and the maximum allowable encoding memory is $m = 4$. The simulated superposition degrees include $D = 5$, 6, 8, and 10. The corresponding optimal connection patterns are Bi(10001, 11001), Bi(10001, 11011), Bi(10011, 11111), and Bi(11111, 11111). We set the maximum number of iterations as $I_{\max} = 18$ and the decoding window size as $W = 17$. It can be seen that the best performance is

achieved with Bi(10001, 11001), which is consistent with the analytical results in Fig. 8b.

6.3 Simulation result and analysis

6.3.1 Performance and complexity comparison with SC-LDPC codes

In the following, we will compare the performance of BiBMST codes, regular SC-LDPC codes, and PC-BiBMST codes under the assumptions of BPSK modulation and AWGN channels. For PC-BiBMST codes, we choose $m = 4$ for comparison. According to the results in Fig. 7 and Table 3, the optimal connection patterns for superposition degrees $D = 5$ and $D = 6$ are chosen for our simulations. The iterative sliding window decoder with $I_{\max} = 18$ is used for decoding. In our simulations, we choose $L = 1000$. The simulated basic codes are the repetition code $C_{\text{rc}}[2B, B]$ and the single parity-check code $C_{\text{spc}}[3B, 2B]$. For convenience, the PC-BiBMST codes based on $C_{\text{rc}}[2B, B]$ are denoted as PC-BiBMST-R, and the PC-BiBMST codes based on $C_{\text{spc}}[3B, 2B]$ are denoted as PC-BiBMST-SPC. We show the performances of the considered codes in Fig. 9. For comparison, we also show the performances of a (3,6)-regular SC-LDPC code and a (4,8)-regular SC-LDPC code in Fig. 9a, and the performances of the comparable BiBMST codes with $m = 2$ in Fig. 9b. Noting that for (3,6)-regular SC-LDPC codes, the two component submatrices are $\boldsymbol{B}_0 = [2, 1]$ and $\boldsymbol{B}_1 = [1, 2]$, and for the (4,8)-regular SC-LDPC codes, the two component submatrices are $\boldsymbol{B}_0 = [3, 1]$ and $\boldsymbol{B}_1 = [1, 3]$. The decoding latencies 20 000 and 30 000 bits are considered, and the lifting factors M of the considered SC-LDPC codes are selected accordingly. From Fig. 9a, we have the following observations.

(1) The PC-BiBMST-R code with Bi(10001, 11011) and decoding delay $d = 11$ performs better than the BiBMST code with $m = 2$ and $d = 6$ in both the waterfall and the error floor region. Noting that for a given decoding latency of 20 000 bits, the BiBMST code with $m = 2$ and $d = 11$ performs worse than the BiBMST code with $m = 2$ and $d = 6$.

(2) The PC-BiBMST-R code with Bi(10001, 11001) performs better than the PC-BiBMST-R codes with Bi(10001, 11011) in the waterfall region but performs worse in the error floor region. The performance advantage of the PC-BiBMST-R code with Bi(10001, 11001) is consistent with the analytical result in Fig. 10.

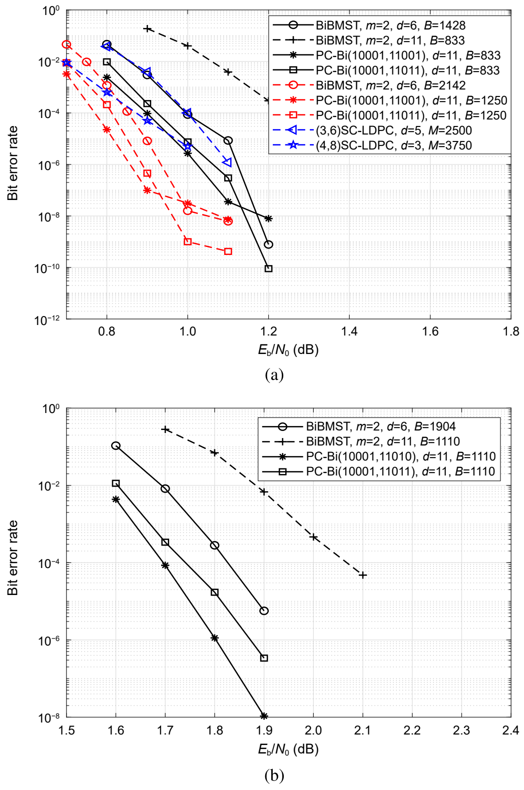

Fig. 9 Performance comparison among PC-BiBMST codes, the BiBMST codes, and SC-LDPC codes. (a) The repetition code $C_{\text{rc}}[2B, B]$ is selected as the basic code. (b) The single parity-check code $C_{\text{spc}}[3B, 2B]$ is selected as the basic code.

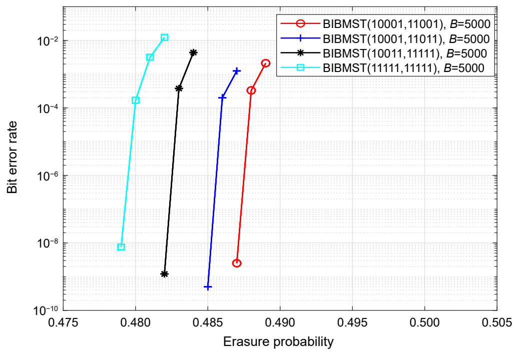

Fig. 10 Error performances of four PC-BiBMST codes with different superposition degrees. The maximum allowable encoding memory is $m = 4$.

In addition, similar observations have been obtained for PC-BiBMST-SPC codes. The above results have shown that optimized PC-BiBMST codes outperform the BiBMST codes in both the waterfall and the error floor region.

We continue to compare the computational complexities of BiBMST codes, PC-BiBMST codes, and regular SC-LDPC codes under an equal decoding latency. For BiBMST codes, its computational complexity per coded bit is given as $104mn(d+1)I_{\text{BiBMST}}$. The computational complexity of the BiBMST codes is related to the superposition degree D and is given as $52n(D-2)(d+1)I_{\text{PC-BiBMST}}$. For a $(J,2J)$-regular SC-LDPC code with the average number of iterations I_{SC} and decoding latency T_{SC}, the average computational complexity in each window is $24JT_{\text{SC}}I_{\text{SC}}$. Since the decoding latency of the sliding window decoder is $T_{\text{SC}}=v_cM(d_{\text{SC}}+1)$ and each window contains v_cM bits to be decoded, the computational complexity per coded bit of a regular SC-LDPC code is $24Jd_{\text{SC}}I_{\text{SC}}$.

Table 4 shows the average computational complexities per coded bit of the BiBMST codes, SC-LDPC codes, and the PC-BiBMST codes used in Fig. 9a with a decoding latency of 30 000 bits. The target BER is 10^{-5}. It can be seen that, the PC-BiBMST code with Bi(10001, 11011) outperforms the considered $(3,6)$-regular SC-LDPC code and $(4,8)$-regular SC-LDPC code, while admiting a slightly lower computational complexity. In addition, the computational complexity per coded bit of the PC-BiBMST code is higher than that of the BiBMST code, which means that the performance gain is obtained at a cost of increased decoding complexity. However, we point out that, for the original BiBMST codes, performance improvement cannot be obtained by increasing the decoding complexity when the decoding latency is fixed. According to Fig. 9b and Table 4, we also observe that the PC-BiBMST-SPC codes perform better than the BiBMST-SPC codes.

6.3.2 Performance comparison with SC-GLDPC codes

In the following, we compare the performance of PC-BiBMST codes and SC-GLDPC codes over the BECs under the constraint of a comparable decoding latency. We select a terminated SC-GLDPC code in Ref. [35] with design rate 1/7 for comparison. This SC-GLDPC code is constructed by coupling a (2, 7)-regular GLDPC code with (7, 4)-Hamming code as the generalized constraint. Following Ref. [35], we employ the generalized peeling decoding (GPD) algorithm[35] for decoding. In the GPD algorithm, the generalized constraint nodes are decoded with the maximum likelihood (ML) decoding algorithm of the Hamming code. For comparison, we consider a PC-BiBMST code with connection pattern Bi(10001, 11001) decoded by the peeling decoding algorithm. To achieve a design rate of 1/7, the repetition code $C_{\text{rc}}[7B,B]$ is chosen as the basic code of the considered PC-BiBMST code. The termination length T is selected as $T=6$. The simulations results are shown in Fig. 11. It can be seen that the PC-BiBMST code with Bi(10001, 11011) outperforms the SC-GLDPC code. We point out that the SC-GLDPC code with $M=500$ and $L=50$, and the PC-BiBMST code with $B=500$ and $L=50$ admit almost the same decoding latencies.

7 Conclusion

In this paper, we have introduced the BiBMST of short codes. Firstly, we derive the generator and the parity-check matrices for BiBMST codes. Secondly, we derive the PEXIT functions for BiBMST codes, which can be used to compute the iterative decoding thresholds of the BiBMST code over the BEC. Thirdly, we investigate the impacts of the parameters on the performance of BiBMST codes, and carry out extensive performance and complexity comparisons. To further enhance the performance of BiBMST codes, we introduce the PC-BiBMST codes. We use the iterative decoding thresholds analysis to optimize the PC-BiBMST codes. We also give extensive numerical results to show the performance advantages of PC-BiBMST codes over the original BiBMST codes, SC-LDPC codes, and SC-GLDPC codes.

Table 4 Decoding complexities per coded bit of BiBMST code and SC-LDPC code that achieve a BER of 10^{-5}.

Code	Bi$(\boldsymbol{E}_1,\boldsymbol{E}_2)$	B or M	d_{BiBMST} or d_{SC}	I_{BiBMST} or I_{SC}	E_b/N_0	Latency	Complexity
BiBMST-R	Bi(11100, 11100)	2142	6	2.559	0.89	30 000	3725.9
	Bi(10001, 11001)	1250	11	2.111	0.81	30 000	3951.8
	Bi(10001, 11011)	1250	11	2.004	0.85	30 000	5002.0
(3, 6) SC-LDPC	–	2500	5	9.650	1.05	30 000	4168.8
(4, 8) SC-LDPC	–	3750	3	10.510	0.97	30 000	4035.8
BiBMST-SPC	Bi(11100, 11100)	1904	6	2.008	1.88	20 000	2923.6
	Bi(10001, 11010)	1110	11	2.004	1.75	20 000	3751.5
	Bi(10001, 11011)	1110	11	2.003	1.80	20 000	4999.5

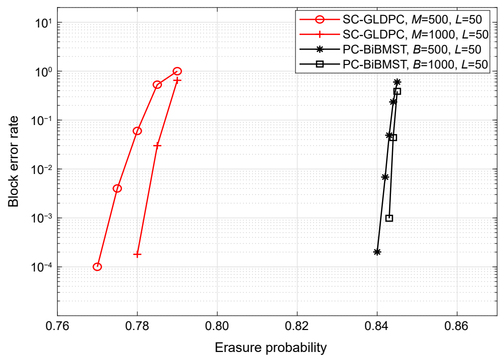

Fig. 11 Finite-length performances of a PC-BiBMST code with Bi(10001, 11001) and a (2, 7)-regular SC-GLDPC code with (7, 4) Hamming constraint codes.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Nos. 62271233, 12271215, and 62261003), the Basic Research Program of Guangzhou Municipal Science and Technology Bureau (No. 202201020036), and the Guangdong Provincial Natural Science Foundation (Nos. 2022A1515010029 and 2021A1515011906).

References

[1] J. Du, J. Song, Y. Ren, and J. Wang, Convergence of broadband and broadcast/multicast in maritime information networks, *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 592–607, 2021.

[2] P. Wang and S. Wang, A fairness-enhanced intelligent MAC scheme using Q-learning-based bidirectional backoff for distributed vehicular communication networks, *Tsinghua Science and Technology*, vol. 28, no. 2, pp. 258–268, 2023.

[3] Z. Chen, Z. Sun, Y. Pei, and L. Yin, Generalized sparse codes for non-Gaussian channels: Code design, algorithms, and applications, *Fundam. Res.*, vol. 2, no. 2, pp. 284–295, 2022.

[4] E. Arikan, Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels, *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[5] C. Li, H. Chen, Z. Wang, Y. Sun, X. Li, and T. Qin, Two-stage constructions for the rate-compatible shortened polar codes, *Tsinghua Science and Technology*, vol. 28, no. 2, pp. 269–282, 2023.

[6] R. Gallager, Low-density parity-check codes, *IEEE Trans. Inform. Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[7] D. J. MacKay, Good error-correcting codes based on very sparse matrices, *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, 1999.

[8] P. Wang, L. Yin, and J. Lu, Efficient helicopter-satellite communication scheme based on check-hybrid LDPC coding, *Tsinghua Science and Technology*, vol. 23, no. 3, pp. 323–332, 2018.

[9] A. J. Felstrom and K. S. Zigangirov, Time-varying periodic convolutional codes with low-density parity-check matrix, *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 2181–2191, 1999.

[10] S. Kudekar, T. J. Richardson, and R. L. Urbanke, Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC, *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 803–834, 2011.

[11] M. Lentmaier, A. Sridharan, D. J. Costello, and K. S. Zigangirov, Iterative decoding threshold analysis for LDPC convolutional codes, *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 5274–5289, 2010.

[12] A. Yedla, Y. Y. Jian, P. S. Nguyen, and H. D. Pfister, A simple proof of threshold saturation for coupled scalar recursions, in *Proc. 2012 7th Int. Symp. on Turbo Codes and Iterative Information Processing* (*ISTC*), Gothenburg, Sweden, 2012, pp. 51–55.

[13] A. Yedla, Y. Y. Jian, P. S. Nguyen, and H. D. Pfister, A simple proof of maxwell saturation for coupled scalar recursions, *IEEE Trans. Inform. Theory*, vol. 60, no. 11, pp. 6943–6965, 2014.

[14] S. Kudekar, T. J. Richardson, and R. L. Urbanke, Wave-like solutions of general 1-D spatially coupled systems, *IEEE Trans. Inf. Theory*, vol. 61, no. 8, pp. 4117–4157, 2015.

[15] A. Ashikhmin, G. Kramer, and S. T. Brink, Extrinsic information transfer functions: Model and erasure channel properties, *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2657–2673, 2004.

[16] C. Measson, A. Montanari, T. J. Richardson, and R. Urbanke, The generalized area theorem and some of its consequences, *IEEE Trans. Inf. Theory*, vol. 55, no. 11, pp. 4793–4821, 2009.

[17] V. Aref, N. Macris, R. Urbanke, and M. Vuffray, Lossy source coding via spatially coupled LDGM ensembles, in *Proc. 2012 IEEE Int. Symp. on Information Theory Proceedings*, Cambridge, MA, USA, 2012, pp. 373–377.

[18] V. Aref, N. Macris, and M. Vuffray, Approaching the rate-distortion limit by spatial coupling with belief propagation and decimation, in *Proc. 2013 IEEE Int. Symp. on Information Theory*, Istanbul, Turkey, 2013, pp. 1177–1181.

[19] S. Cai, W. Lin, X. Yao, B. Wei, and X. Ma, Systematic convolutional low density generator matrix code, *IEEE Trans. Inf. Theory*, vol. 67, no. 6, pp. 3752–3764, 2021.

[20] S. Moloudi, M. Lentmaier, and A. G. I. Amat, Spatially coupled turbo-like codes, *IEEE Trans. Inf. Theory*, vol. 63, no. 10, pp. 6199–6215, 2017.

[21] M. Lentmaier, S. Moloudi, and A. G. I. Amat, Braided convolutional codes—A class of spatially coupled turbo-like codes, in *Proc. 2014 Int. Conf. Signal Processing and Communications* (*SPCOM*), Bangalore, India, 2014, pp. 1–5.

[22] A. G. I. Amat, S. Moloudi, and M. Lentmaier, Spatially coupled turbo codes: Principles and finite length performance, in *Proc. 2014 11th Int. Symp. on Wireless Communications Systems* (*ISWCS*), Barcelona, Spain, 2014, pp. 883–887.

[23] W. Zhang, M. Lentmaier, K. S. Zigangirov, and D. J.

Costello, Braided convolutional codes: A new class of turbo-like codes, *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 316–331, 2010.

[24] M. Qiu, X. Wu, J. Yuan, and A. G. I. Amat, Generalized spatially coupled parallel concatenated convolutional codes with partial repetition, in *Proc. 2021 IEEE Int. Symp. on Information Theory* (*ISIT*), Melbourne, Australia, 2021, pp. 581–586.

[25] X. Ma, C. Liang, K. Huang, and Q. Zhuang, Block Markov superposition transmission: Construction of big convolutional codes from short codes, *IEEE Trans. Inf. Theory*, vol. 61, no. 6, pp. 3150–3163, 2015.

[26] C. Liang, X. Ma, Q. Zhuang, and B. Bai, Spatial coupling of generator matrices: A general approach to design good codes at a target BER, *IEEE Trans. Commun.*, vol. 62, no. 12, pp. 4211–4219, 2014.

[27] C. Liang, J. Hu, X. Ma, and B. Bai, A new class of multiple-rate codes based on block Markov superposition transmission, *IEEE Trans. Signal Process.*, vol. 63, no. 16, pp. 4236–4244, 2015.

[28] N. Lin, S. Cai, and X. Ma, Block Markov superposition transmission of BCH codes with iterative hard-decision decoding, in *Proc. 2017 IEEE Int. Symp. on Information Theory* (*ISIT*), Aachen, Germany, 2017, pp. 1598–1602.

[29] J. Hu, X. Ma, and C. Liang, Block Markov superposition transmission of repetition and single-parity-check codes, *IEEE Commun. Lett.*, vol. 19, no. 2, pp. 131–134, 2015.

[30] X. Ma, K. Huang, and B. Bai, Systematic block Markov superposition transmission of repetition codes, *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1604–1620, 2018.

[31] Q. Wang, S. Cai, Y. Wang, and X. Ma, Free-ride feedback and superposition retransmission over LDPC coded links, *IEEE Trans. Commun.*, vol. 71, no. 1, pp. 13–25, 2023.

[32] S. Zhao, X. Ma, Q. Huang, and B. Bai, Recursive block Markov superposition transmission of short codes: Construction, analysis, and applications, *IEEE Trans. Commun.*, vol. 66, no. 7, pp. 2784–2796, 2018.

[33] K. Huang and X. Ma, Performance analysis of block Markov superposition transmission of short codes, *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 362–374, 2016.

[34] G. Liva and M. Chiani, Protograph LDPC codes design based on EXIT analysis, in *Proc. IEEE GLOBECOM 2007 - IEEE Global Telecommun. Conf.*, Washington, DC, USA, 2007, pp. 3250–3254.

[35] D. G. M. Mitchell, P. M. Olmos, M. Lentmaier, and D. J. Costello, Spatially coupled generalized LDPC codes: Asymptotic analysis and finite length scaling, *IEEE Trans. Inf. Theory*, vol. 67, no. 6, pp. 3708–3723, 2021.

Gaoyan Li received the BS degree in communication engineering from Nanchang University, China in 2021. He is currently pursuing the MS degree at College of Information Science and Technology, Jinan University, Guangzhou, China. His current research interests include spatially coupled codes.

Jinming Wen received the bachelor degree in information and computing science from Jilin Institute of Chemical Technology, Jilin, China in 2008, the MSc degree in pure mathematics from Jilin University, Jilin, China in 2010, and the PhD degree in applied mathematics from McGill University, Montreal, Canada in 2015. He was a postdoctoral research fellow at Laboratoire d’Informatique de Paris (LIP) from March 2015 to August 2016, University of Alberta from September 2016 to August 2017, and University of Toronto from September 2017 to August 2018. He has been a full professor in Jinan University, Guangzhou, China since September 2018. His research interests are in the areas of lattice reduction and sparse recovery. He has published around 60 papers in top journals and conferences. He is an associate editor of *IET Quantum Communication* and *Alexandria Engineering Journal*.

Haiqiang Chen received the MS degree from Guangxi University, Nanning, China in 2002 and the PhD degree from Sun Yat-sen University, Guangzhou, China in 2011. He is currently a professor and also the instructor of graduate student at Guangxi University, Nanning, China. His research interests include coding theory and relay system.

Shancheng Zhao received the bachelor degree in software engineering and the PhD degree in communication and information systems from Sun Yat-sen University, Guangzhou, China in 2009 and 2014, respectively. From 2013 to 2014, he was a graduate visiting student with University of California at Los Angeles, CA, USA. He is currently a professor at College of Information Science and Technology, Jinan University, Guangzhou, China. His current research interests include spatially coupled codes and their applications. He was a co-recipient of the Best Paper Award at the IEEE GlobeCom in 2015. He serves as an associate editor for *Physical Communication* (*Elsevier*), *IET Quantum Communication*, and *Alexandria Engineering Journal*.