Spatially Coupled LDPC Codes via Partial Superposition

Qianfan Wang*, Suihua Cai†, Wenchao Lin‡, Li Chen*, and Xiao Ma‡§

*School of Electronics and Communication Engineering, Sun Yat-sen University, Guangzhou 510006, China
†School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China
‡School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China
§Guangdong Key Laboratory of Information Security Technology, Sun Yat-sen University, Guangzhou 510006, China
Email: {wangqf6, caish5, linwch7}@mail2.sysu.edu.cn, {chenli55, maxiao}@mail.sysu.edu.cn

Abstract—In this paper, we present a new class of spatially coupled low-density parity-check (SC-LDPC) codes, which are constructed by sending codewords of LDPC block code (LDPC-BC) in a block Markov superposition transmission (BMST) manner. Different from the conventional SC-LDPC codes, the proposed SC-LDPC codes can have encoder/decoder implemented with the basis of the hardware components of the corresponding LDPC-BCs. The proposed SC-LDPC codes are also a special class of BMST-LDPC codes. Distinguished from other types of BMST codes, BMST-LDPC codes have lower error floors even with an encoding memory of one and hence have lower decoding latency. Also different from the original BMST codes, partial superposition is implemented to alleviate error propagation. To analyze the bit error rate (BER) performance, we present the genie-aided (GA) bounds, which can be obtained by simulation or estimated from the performance of the basic code. Numerical results are presented to validate our analysis and demonstrate the performance advantage of the BMST-LDPC codes over the LDPC-BCs.

I. INTRODUCTION

Low-density parity-check convolutional codes, also known as spatially coupled low-density parity-check (SC-LDPC) codes, were first introduced in [1] and have capacity-approaching performance over binary-input memoryless symmetric-output (BMS) channels under iterative belief propagation (BP) decoding [2]. This important feature is due to threshold saturation [3] that the decoding performances of SC-LDPC codes under BP decoding approach the maximum a posteriori (MAP) decoding performances of uncoupled low-density parity-check block codes (LDPC-BCs). The threshold saturation has been proven for binary erasure channels [3] and generalized to BMS channels [2, 4]. The SC-LDPC codes can be decoded using a sliding window (SW) decoder [5, 6], which usually has low-complexity and yields a low message recovery latency. As a result, the construction of SC-LDPC codes has received significant attention in recent years. The construction in [1] relies on a matrix-based unwrapping procedure, while the construction in [7] exploits similarities between quasi-cyclic block codes and time-invariant convolutional codes. These two constructions were shown in [8] to be tightly connected via (proto)graph-cover construction. The protograph-based construction was further investigated in [9], where a series of uncoupled LDPC-BCs protographs are first coupled into a single chain by the edge spreading procedure and then lifted to a covering graph. All constructions above can be carefully tailored to avoid short cycles and hence to improve the performance especially in the error-floor region. For example, a systematic protograph-based construction was proposed in [10] such that the Tanner graph can have a girth of eight, while the replicate-and-mask construction was proposed in [11] to improve the performance in the finite-length regime. In brief, most constructions focus on how to derive convolutional parity-check matrices from those of the LDPC-BCs. The performance of the SC-LDPC codes is closely related to the underlying LDPC-BCs. However, the hardware implementation of the encoder and decoder of the SC-LDPC codes is usually loosely related to those of the underlying codes. This somehow brings inconvenience for real design and practical implementation.

In this paper, we propose a new class of SC-LDPC codes, which can be implemented with the basis of the hardware of the underlying LDPC-BCs. The basic idea is to send the codewords of an LDPC-BC in a block Markov superposition transmission (BMST) manner, resulting in BMST-LDPC codes. The construction is universal in the sense that it applies to any existing LDPC-BCs. Distinguished from other BMST codes [12], the BMST-LDPC codes have lower error floor even with an encoding memory of one and hence inherit a lower decoding latency. To avoid catastrophic error propagation caused by the abrupt performance of the basic LDPC-BCs, we propose to use partial superposition instead of full superposition, in which only a portion of coded bits are randomly selected and superimposed onto the adjacent codewords. To analyze the bit error rate (BER) performance, we present the genie-aided (GA) bounds, which can be obtained by simulating an equivalent GA system or estimated from the performance curve of the basic LDPC-BC. Numerical results show that the proposed SC-LDPC codes can yield extra coding gain over the underlying LDPC-BCs. Numerical results also reveal the
trade-offs between the performance and the decoding latency.

II. BMST-LDPC CODES

A. Preliminaries

Let \( \mathcal{C}[n, k] \) be a binary LDPC code of length \( n \) and dimension \( k \), whose parity-check matrix and generator matrix are denoted by \( H \) and \( G \), respectively. Let \( u = (u^{(0)}, u^{(1)}, \ldots, u^{(L-1)}) \) be \( L \) blocks of information sequences to be transmitted, where \( u^{(t)} \in \mathbb{F}_2^n \) for \( 0 \leq t \leq L-1 \). At time slot \( t \), the encoder of \( \mathcal{C}[n, k] \) takes as input the information sequence \( u^{(t)} \) and delivers as output the coded sequence \( v^{(t)} = u^{(t)}G \in \mathbb{F}_2^m \). Assume that \( v^{(t)} \) is modulated by BPSK signaling and transmitted over an additive white Gaussian noise (AWGN) channel, resulting in a received sequence \( y^{(t)} \). Upon receiving \( y^{(t)} \), the decoder delivers an estimation \( \hat{u}^{(t)} \) of \( u^{(t)} \).

Evidently, for LDPC-BC coded system, the transmission of codewords at different time slots is independent and the decoding latency is fixed to \( n \) (by definition). A question arises: if the constraint on the decoding delay is relaxed, can we improve the performance without changing too much the structure of the encoder and the decoder? This can be realized by introducing memory between the adjacent transmissions. In the rest of this paper, the considered code \( \mathcal{C}[n, k] \) is referred to as the basic code, which is assumed to have an efficient encoding algorithm and a soft-in-soft-out (SISO) decoding algorithm.

B. Encoding of BMST-LDPC Codes

Different from the conventional BMST codes, we focus on the case of encoding memory one. Such a setup is assumed due to the following reasons. First, our construction is based on well-constructed LDPC-BCs, which already have capacity-approaching performance. Further increasing memory will only lead to marginal performance improvement. Second, we attempt to keep the complexity as low as possible. Also note that, we consider BMST with partial superposition to alleviate the error propagation.

Let \( \alpha (0 < \alpha \leq 1) \) be a parameter (referred to as superposition fraction) to be optimized. We define a random matrix \( S \) as follows. First, a permutation matrix \( S \) of size \( n \times n \) is sampled uniformly at random from the set of all possible permutation matrices. Second, \( n(1-\alpha) \) out of \( n \) columns of \( S \) are randomly chosen and set to all-zero columns. For a binary vector \( v \in \mathbb{F}_2^n \), \( vS \) is a vector that is an interleaving version of \( v \) but with some bits being set to zero. Therefore, we may call \( S \) a selection matrix, since it selects some bits for partial superposition. Given \( u = (u^{(0)}, u^{(1)}, \ldots, u^{(L-1)}) \) with \( u^{(t)} = (u_0, u_1, \ldots, u_{k-1}) \in \mathbb{F}_2^k \), the encoding algorithm of the BMST-LDPC codes with partial superposition is described in Algorithm 1, see Fig. 1 for illustration.

C. Algebraic Description of BMST-LDPC Codes

In this subsection, the relations between the presented codes and the existing codes are revealed by their algebraic descriptions. With termination, a BMST of \( \mathcal{C}[n, k] \) LDPC code with a superposition fraction \( \alpha \) can be treated as a linear block code with dimension \( kL \) and length \( n(L+1) \). The generator matrix of the BMST-LDPC code can be specified as an upper banded block matrix,

\[
G_{SC} = \begin{bmatrix}
G & GS \\
G & GS \\
& \ddots & G & GS \\
& & G & GS \\
& & & G & GS
\end{bmatrix},
\]

where \( G \) is the generator matrix of the basic LDPC-BC and \( S \) is a selection matrix.

In the case of full superposition (\( \alpha = 1 \)), the presented construction is nothing more than a special class of the BMST codes which take LDPC-BCs as basic codes and fix the encoding memory to one. In this paper, we are more interested in the case of partial superposition (\( \alpha < 1 \)), which is the main difference as compared with the conventional BMST codes. Introducing partial superposition is to alleviate the effect of error propagation during decoding, where the superposition fraction \( \alpha \) can be optimized for this aim.

Similar to the derivation of [12], we can determine the parity-check matrix of the BMST-LDPC code as a lower block...
LDPC codes are also a special class of SC-LDPC codes.

A. Sliding Window Decoding Algorithm

and a node of type \( S \) given by

\[
H_{SC} = \begin{bmatrix}
H & H & \cdots & H \\
H_S & H_{S_1} & \cdots & H_{S_L} \\
H_{S_{L-1}} & H_{S_{L-2}} & \cdots & H_{S_1}
\end{bmatrix},
\]

where \( H \) is the parity-check matrix of the basic LDPC-BC and \( S_i = (S^i)^T \) for \( i \geq 1 \). By construction, we know that the selection matrix \( S \) has \( n(1-\alpha) \) zero columns and hence \( n(1-\alpha) \) zero rows. So, \( S_i \) has at least \( n(1-\alpha) \) zero columns, and \( HS_i \) is a low-density matrix. Hence, the presented BMST-LDPC codes are also a special class of SC-LDPC codes.

Throughout this paper, we refer the presented codes to as BMST-LDPC codes and SC-LDPC codes interchangeably.

III. DECODING ALGORITHM AND PERFORMANCE ANALYSIS

A. Sliding Window Decoding Algorithm

With a low-density parity-check matrix \( H_{SC} \) (see Sec. II-C), BMST-LDPC codes can be decoded as an LDPC-BC. However, such a schedule has high decoding latency and is not suitable for some applications. More awkwardly, such an implementation is loosely related to the decoder of the basic LDPC-BC. In this subsection, we introduce the SW decoding algorithm, which integrates the SISO decoder of the basic LDPC-BC in an iterative manner.

The SW decoding algorithm for the BMST-LDPC codes can be described as an iterative message passing algorithm over a normal graph [13], in which edges represent variables and nodes represent constraints. Associated with each edge is a message that is defined in this paper as the probability mass function (pmf) of the corresponding variable. Fig. 1 shows the high-level normal graph of a BMST-LDPC code, where an edge represents a sequence of random variables and its associated messages are collectively written in a sequence. The iterative SW decoding algorithm with decoding window of size \( d \) can be described as a message passing algorithm over a subgraph containing \( d \) layers, where the decoding latency is given by \( dn \) in terms of bits. Each decoding layer consists of a node of type \( \text{LDPC} \), a node of type \( \text{S} \), and a node of type \( + \). The decoding algorithm starts from nodes of type \( + \) which connect to the channel.

- **Node [ ]**: It represents the constraint that the sum of all connecting variables must be zero over \( F_2 \). The message updating rule at the node [ ] is similar to that at the check node in an LDPC-BC. The only difference is that the messages associated with the half edges need to be calculated from the channel observations.

- **Node [S]**: It represents the selection matrix, which simply transfers the messages associated with the selected bits between the node [ ] and the node [+].

- **Node [ ]**: It represents the constraint that all connecting variables must take the same value. The message updating rule at the node [ ] is the same as that at the variable node in an LDPC-BC.

- **Node [LDPC]**: It represents the basic LDPC-BC encoding constraint, where \( v^{(t)} \) must be a codeword corresponding to \( u^{(t)} \). The message updating at the node [LDPC] can be implemented based on certain SISO decoder, say, the iterative sum-product algorithm (SPA) with a maximum iteration number \( I_{max} \) of the basic LDPC-BC. The extrinsic messages associated with \( u^{(t)} \) can be used to make decisions on the transmitted data.

The decoding algorithm is summarized in Algorithm 2.

**Algorithm 2 Iterative Sliding Window Decoding of the BMST-LDPC Codes (window size \( d \geq 1 \))**

- **Global initialization**: Set a maximum global iteration number \( I_{max} \). For \( 0 \leq t \leq d-2 \), compute the a posteriori probabilities associated with \( c^{(t)} \) from the received vector \( y^{(t)} \). All messages over the other edges within and connecting to the \( t \)-th layer (\( 0 \leq t \leq d-2 \)) are initialized as uniformly distributed variables.

- **Sliding window decoding**: For \( t = 0, 1, \cdots, L-1 \),

1) **Local initialization**: If \( t + d - 1 \leq L \), compute the a posteriori probabilities from the received vector \( y^{(t+d-1)} \) and all messages over other edges within and connecting to the \( (t+d-1) \)-th layer are initialized as uniformly distributed variables.

2) **Iteration**: For \( j = 1, 2, \cdots, I_{max} \),

   a) **Forward recursion**: For \( i = 0, 1, \cdots, d-1 \), the \( (t+i) \)-th layer performs a message passing algorithm scheduled as

   \[ + \rightarrow \text{LDPC} \rightarrow \text{S} \]

   b) **Backward recursion**: For \( i = d-1, d-2, \cdots, 0 \), the \( (t+i) \)-th layer performs a message passing algorithm scheduled as

   \[ \text{LDPC} \rightarrow + \rightarrow \text{S} \]

   c) **Decisions**: Decisions are made on \( v^{(t)} \) resulting in \( \hat{v}^{(t)} \). If \( H\hat{v}^{(t)} = 0 \), exit the iteration. Notice that the parity-check stopping criterion is also used in the above recursions, where the node [LDPC] performs an iterative BP decoding with a preset maximum iteration number \( I_{max} \).

3) **Successive cancellation**: Remove the effect of \( \hat{v}^{(t)} \) on \( y^{(t+1)} \) and output \( \hat{u}^{(t)} \) based on \( \hat{v}^{(t)} \).
of all \( \mathbf{v}' = (v^{(0)}, \cdots, v^{(t-1)}, v^{(t+1)}, \cdots, v^{(L-1)}) \). The GA bound can be obtained by simulating the GA decoder, or equivalently, simulating a BMST-LDPC code with \( L = 1 \).

The above simulated GA bound is also adaptable to BMST codes with higher order modulations. However, it does not characterize explicitly the relation to the performance of the basic code. For BPSK signaling over AWGN channels, we have an estimated GA bound, which is simple and derived from the performance of the basic LDPC-BC. Let \( p_b = f_{\text{basic}}(\gamma_b) \) and \( p_b = f_{\text{BMST-LDPC}}(\gamma_b) \) be the BER performance functions of the basic LDPC-BC and the BMST-LDPC code, respectively, where \( p_b \) is the BER and \( \gamma_b = E_b/N_0 \) in dB. Since the assumption of GA decoder is equivalent to assuming that each coded bit is transmitted on average \( 1 + \alpha \) times, we then have the following estimated GA bound

\[
\Rightarrow f_{\text{BMST-LDPC}}(\gamma_b) \geq f_{\text{basic}}(\gamma_b + 10\log_{10}(1 + \alpha)),
\]

which can be obtained by shifting the BER performance curve of the basic code to left by \( 10\log_{10}(1 + \alpha) \) dB.

C. Complexity Analysis

To analyze the complexity, we take the basic LDPC-BC as the comparison benchmark. We assume that the encoder/decoder of the basic LDPC-BC have been implemented. For encoding of the SC-LDPC code, \( \alpha \) extra binary additions per coded bit are required to complete the superposition. For decoding, let \( J_{\text{max}} \) be the maximum number of iteration and let the decoding window size be \( d \). Without considering earlier stopping, it is required to perform the decoding of LDPC-BC \( dJ_{\text{max}} \) times. From simulation, we see that both \( J_{\text{max}} \) and \( d \) can be small. We also note that, for the SC-LDPC code, the component decoder in each layer can have less iterations than that for independent transmission. So the increase in complexity can be small compared with the basic LDPC-BC.

IV. NUMERICAL RESULTS

In this section, we give the numerical results of the BMST-LDPC codes. We use rate 1/2 (3,6)-regular LDPC codes, which are constructed by the progressive-edge-growth (PEG) algorithm, as the basic LDPC-BCs in the BMST-LDPC codes. The SW decoding algorithm (Algorithm 2) is employed for the decoding of the BMST-LDPC codes, in which the maximum global iteration number \( J_{\text{max}} = 3 \). The embedded basic LDPC-BCs are decoded by the SPA with a maximum iteration number 20. For independent LDPC-BCs, the maximum iteration number is 50. In all examples, earlier stopping is activated with the parity-check based criterion. The encoder terminates every \( L = 98 \) sub-blocks in all of the simulations.

Example 1 (GA Bounds): Consider an LDPC-BC with length 1024 as the basic code. Codewords \( \mathbf{c}^{(t)} \) are transmitted with BPSK modulation over AWGN channels. The BER performance curve of the BMST-LDPC code with superposition fraction \( \alpha = 0.1 \) is depicted in Fig. 2. We also give the simulated GA bound and the estimated GA bound. From the figure, we see that the simulation results match well with the GA bound in the high SNR region.

Example 2 (Fixed \( d \) and \( k \), Varying \( \alpha \)): Consider an LDPC-BC with length 1024 as the basic code. Codewords \( \mathbf{c}^{(t)} \) are transmitted with BPSK modulation over AWGN channels. The BER performance curves of the BMST-LDPC codes with different superposition fraction \( \alpha \) are shown in Fig. 3. For comparison, we also give the performance of the rate 1/2 LDPC-BC with length 1024. From the figure, we see that the superposition fraction \( \alpha \) affects the performance of the BMST-LDPC codes. The BMST-LDPC codes with properly selected superposition fraction \( \alpha \) can achieve good performance. We also see that the BER performance of BMST-LDPC (\( \alpha = 0.3 \)) code exhibits 0.7 dB coding gain over the LDPC-BC, at the BER of \( 10^{-5} \).

Example 3 (Comparison with Other LDPC Codes): Con-
consider the LDPC-BCs with length 1152 and 2400 as the basic codes. Codewords \( c^{(i)} \) are transmitted with BPSK modulation over AWGN channels. The BER performance comparison is shown in Fig. 4. The rate 1/2 LDPC-BC with length 2304 and the SC-LDPC code [10] are chosen to be compared. Fig. 4 shows that the BMST-LDPC code can outperform the benchmark codes with the same decoding latency.

**Example 4 (Fixed \( k \) and \( \alpha \), Increasing \( d \)):** Consider an LDPC-BC with length 2304 as the basic code. Codewords \( c^{(i)} \) are modulated using the 16-ary quadrature amplitude modulation (16-QAM) constellation with Gray labelling and transmitted over AWGN channels. For comparison, we also give the performance of the bit-interleaved coded modulation with LDPC (BICM-LDPC) code and delayed bit-interleaved coded modulation with LDPC [14] (DBICM-LDPC) code with the same LDPC-BC. The comparison is shown in Fig. 5. The performance of the BMST-LDPC code can be improved by increasing the window size \( d \). In comparison with the BICM-LDPC code, the BMST-LDPC code achieves 0.5 dB, 0.6 dB, and 0.7 dB coding gains at the BER of \( 10^{-5} \), by using windows of sizes 2, 3, and 4, respectively.

V. CONCLUSIONS

This paper presented a new class of SC-LDPC codes, which are constructed by sending codewords of an LDPC-BC in a BMST manner with partial superposition. The proposed SC-LDPC codes can be implemented with the basis of the hardware of the corresponding LDPC-BCs. To analyze the BER performance, we presented the GA bounds, which can be obtained by simulation or estimated from the performance of the basic code. Numerical results show that, the proposed SC-LDPC codes can obtain a coding gain up to 0.7 dB compared to the underlying LDPC-BCs.

**REFERENCES**


