A Threshold-Based Min-Sum Algorithm to Lower the Error Floors of Quantized LDPC Decoders

Homayoon Hatami, *Member, IEEE,* David G. M. Mitchell, *Senior Member, IEEE,*
Daniel J. Costello, Jr., *Life Fellow, IEEE,* and Thomas E. Fuja *Fellow, IEEE*

***Abstract*—For decoding low-density parity-check (LDPC) codes, the attenuated min-sum algorithm (AMSA) and the offset min-sum algorithm (OMSA) can outperform the conventional min-sum algorithm (MSA) at low signal-to-noise-ratios (SNRs), *i.e.*, in the "waterfall region" of the bit error rate curve. This paper demonstrates that, for *quantized* decoders, MSA actually outperforms AMSA and OMSA in the "error floor" region, and that all three algorithms suffer from a relatively high error floor. This motivates the introduction of a modified MSA that is designed to outperform MSA, AMSA, and OMSA across all SNRs. The new algorithm is based on the assumption that trapping sets are the major cause of the error floor for quantized LDPC decoders. A performance estimation tool based on trapping sets is used to verify the effectiveness of the new algorithm and also to guide parameter selection. We also show that the implementation complexity of the new algorithm is only slightly higher than that of AMSA or OMSA. Finally, the simulated performance of the new algorithm, using several classes of LDPC codes (including spatially coupled LDPC codes), is shown to outperform MSA, AMSA, and OMSA across all SNRs.**

***Index Terms*—LDPC codes, min-sum decoding, error floors, absorbing sets, trapping sets, decoder quantization.**

I. Introduction

Low-density parity-check (LDPC) codes [1] are a class of linear block codes for which the performance of iterative message passing (MP) decoding can approach that of much more complex maximum likelihood (ML) decoding. The min-sum algorithm (MSA) [2] is a simplified version of the sum-product algorithm (SPA) [3] that is commonly used for iterative MP decoding of LDPC codes, where the check node computation is approximated and hence is significantly easier to perform. This simplification is particularly desirable for hardware decoder implementations. Moreover, unlike the SPA, no estimation of the channel signal-to-noise ratio (SNR) is needed at the receiver for an additive white Gaussian noise (AWGN) channel.

This material is based upon work supported by the National Science Foundation under Grant Nos. ECCS-1710920 and OIA-1757207. This article was presented in part at the IEEE International Symposium on Information Theory, Paris, France, July 2019.

Homayoon Hatami was with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN, 46556 USA. He is now with Samsung Semiconductor Inc., San Diego, CA 92121 USA (e-mail: hhatami@nd.edu).

David G. M. Mitchell is with the Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM, 88003 USA (email: dgmm@nmsu.edu).

Daniel J. Costello, Jr., and Thomas E. Fuja are with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN, 46556 USA (e-mail: costello.2@nd.edu; tfuja@nd.edu).

In [4], two modifications of the MSA, denoted attenuated MSA (AMSA)[1] and offset MSA (OMSA) were introduced to adjust for the error in the approximation and improve performance. Also, [5] independently introduced the OMSA. Compared to the MSA, these algorithms reduce the magnitudes of the log-likelihood ratios (LLRs) computed at the check nodes of the Tanner graph representation of the parity-check matrix $\mathbf{H}$ of an LDPC code. Both variants have been shown to achieve better *waterfall* (low SNR) performance (closer to the SPA) compared to the conventional MSA [4], [5]. Numerous approaches have been proposed in the literature to improve the performance of the MSA, including adjusting the offset by iteration [6], techniques to approximate the error term [7], [8], and modified quantizer designs [9], [10].

Practical implementations of LDPC decoders require a finite precision (quantized) representation of the LLRs. In [4], quantized density evolution (DE) was used to find the optimum attenuation and offset parameters for the AMSA and OMSA, in the sense that DE calculates the iterative decoding threshold, which characterizes the waterfall performance. Following [4], [5], several papers focused on further improving the waterfall performance of the MSA for quantized [9], [11] and unquantized decoders [12], [13]. At high SNRs, quantization typically causes the early onset of an *error floor*. In [14]–[17], it was shown that certain objects, called *trapping sets*, *elementary trapping sets*, *leafless elementary trapping sets*, or *absorbing sets*, in the Tanner graph cause the iterative decoding process to get stuck, resulting in decoding errors at high SNRs. Hereafter, we refer to the sub-graphs induced by these sets, as well as similar sets, as *problematic graphical objects*. Several methods based on problematic objects have been proposed to estimate the performance of LDPC codes [18]–[24] and a number of strategies have been proposed to lower the error floor of quantized LDPC decoders, including quantizer design [11], [25], [26], modifications to iterative decoding [27]–[30], and post-processing [31]–[34].

In this paper, we propose a novel modification to the check node update of quantized MSA that is straightforward to implement and reduces the error floor when compared to the methods proposed in [4], [5]. First, we show that the AMSA and OMSA exhibit worse (higher) error floors than the MSA with parameters that are optimized for waterfall performance. We then introduce a modification to the MSA that applies the strategies from the AMSA and the OMSA *selectively*, *i.e.*, it applies attenuation/offset when it would be helpful

[1] Attenuated MSA is also known as *normalized MSA*.

and does not apply it otherwise. Assuming that there exist problematic graphical objects that cause most of the decoding failures in the high SNR regime, we further show that the new MSA modification causes these objects to become less prone to decoding failures. As a result, the new algorithm matches the waterfall performance of the AMSA and OMSA, while improving their error floor performance. We note that *no information about the location or structure of the problematic objects is required* to utilize this approach; however, knowledge of the problematic object facilitates determination of the optimal algorithm parameters. Moreover, we note that AMSA (respectively OMSA) can be viewed as a particular case of TAMSA (resp. TOMSA) and, as such, the performance of TAMSA (resp. TOMSA) is at least as good as AMSA (resp. OMSA) with optimal parameter selection. We quantify the complexity of the proposed scheme and show that, since it uses the information that is already generated inside the check node processing unit of the AMSA or OMSA, the new algorithm is only slightly more complex to implement.

Due to the time-consuming process of simulating the high SNR performance of LDPC codes, we utilize the *code-independent* and *problematic object-specific* method of [18] to guide/optimize parameter selection and to evaluate the impact of the proposed algorithm on the performance of LDPC codes containing problematic objects at high SNRs. The results indicate that the new algorithm improves (reduces) the error floor caused by specific problematic objects compared to the MSA, AMSA, or OMSA. Finally, we present the simulated performance of certain randomly constructed and structured LDPC block codes (LDPC-BCs) and spatially coupled LDPC codes (SC-LDPCCs) [35] to verify the effectiveness and versatility of the algorithm. In all cases, the new algorithm performs at least as well as the MSA, AMSA, or OMSA, at both low and high SNRs.

II. Background

Let $V = \{v_1, v_2, \ldots, v_n\}$ and $C = \{c_1, c_2, \ldots, c_m\}$ represent the sets of variable nodes and check nodes, respectively, of a bipartite Tanner graph representation of an LDPC code with parity-check matrix $\mathbf{H}$. Assume that a binary codeword $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ is binary phase shift keyed (BPSK) modulated such that each zero is mapped to $+1$ and each one is mapped to -1. The modulated signal is transmitted over an AWGN channel with mean 0 and standard deviation σ. The received signal is $\tilde{\mathbf{r}} = 1 - 2\mathbf{u} + \mathbf{n}$, where $\mathbf{n}$ is the channel noise. We denote the quantized version of $\tilde{\mathbf{r}}$ as $\mathbf{r} = (r_1, r_2, \ldots, r_n)$.

A. The Min-Sum Algorithm and its Modifications

The MSA is an iterative MP algorithm that is simpler to implement than the SPA. Unlike the SPA, the MSA does not require channel noise information to calculate the channel LLRs. The SPA is optimum for codes without cycles, but for finite length codes and finite precision LLRs, the SPA is not necessarily optimum, particularly with respect to error floor performance [29]. Let $\mathbb{V}_{ij}$ represent the LLR passed from variable node v_i to check node c_j in a given iteration and let $\mathbb{C}_{ji}$ represent the LLR passed from c_j to v_i.[2] The check nodes that are neighbors to v_i are denoted $N(v_i)$, and the variable nodes that are neighbors to c_j are denoted $N(c_j)$. To initialize decoding, each variable node v_i passes r_i to the check nodes in $N(v_i)$, *i.e.*,

$$\mathbb{V}_{ij} = r_i, \tag{1}$$

where the $\mathbb{V}_{ij}$'s computed throughout the decoding process are referred to as the variable node LLRs.[3] The check node operation to calculate the LLRs sent from check node c_j to variable node v_i is given by

$$\mathbb{C}_{ji} = \left(\prod_{i' \in N(c_j) \setminus i} \operatorname{sign}\left(\mathbb{V}_{i'j}\right) \right) \cdot \min_{i' \in N(c_j) \setminus i} \left| \mathbb{V}_{i'j} \right|, \tag{2}$$

where the $\mathbb{C}_{ji}$'s computed throughout the decoding process are referred to as the check node LLRs. After each iteration, the hard decision estimate $\hat{\mathbf{u}}$ is checked to see if it is a valid codeword, where $\hat{u}_i = 0$ iff

$$r_i + \sum_{j' \in N(v_i)} \mathbb{C}_{j'i} > 0. \tag{3}$$

If $\hat{\mathbf{u}}$ is a valid codeword, or if the iteration number has reached $I_{\max}$, decoding stops. Otherwise, the variable node LLRs are calculated as

$$\mathbb{V}_{ij} = r_i + \sum_{j' \in N(v_i) \setminus j} \mathbb{C}_{j'i} \tag{4}$$

and decoding continues using (2).

In [4], two modified versions of the MSA, called attenuated MSA (AMSA) and offset MSA (OMSA), were introduced to reduce the waterfall performance loss of the MSA compared to the SPA. The modified check node computations are given by

$$\mathbb{C}_{ji} = \alpha \left(\prod_{i' \in N(c_j) \setminus i} \operatorname{sign}\left(\mathbb{V}_{i'j}\right) \right) \cdot \min_{i' \in N(c_j) \setminus i} \left| \mathbb{V}_{i'j} \right|, \tag{5}$$

and

$$\mathbb{C}_{ji} = \left(\prod_{i' \in N(c_j) \setminus i} \operatorname{sign}\left(\mathbb{V}_{i'j}\right) \right) \cdot \max\{ \min_{i' \in N(c_j) \setminus i} \left| \mathbb{V}_{i'j} \right| - \beta, 0 \}, \tag{6}$$

respectively, where α, $\beta > 0$ are constants. In both algorithms, the check node LLR magnitudes are modified to be smaller than those of MSA. This reduces the negative effect of overestimating the LLR magnitudes in the MSA, whose larger check node LLR magnitudes compared to the SPA can cause additional errors in decoding at low SNRs.

[2] Note that the iteration indices are dropped for clarity of notation.

[3] In the SPA, computing an LLR value from the received value r_i requires multiplying by $2/\sigma^2$, where σ^2 is the channel noise variance. However, this normalization is not required for min-sum decoding and its variants, so we omit it here.

B. Implementation of the MSA, AMSA, and OMSA

To implement the check node update of (2) in the check node processing unit corresponding to c_j, the sign and magnitude of $\mathbb{C}_{ji}$ to be sent to each v_i are calculated separately as follows. First, for all $i' \in N(c_j)$, the signs of $\mathbb{V}_{i'j}$ are multiplied to form $\prod_{i' \in N(c_j)} \text{sign}(\mathbb{V}_{i'j})$. Then, for each $i \in N(c_j)$, $\text{sign}(\mathbb{V}_{ij})$ is multiplied by $\prod_{i' \in N(c_j)} \text{sign}(\mathbb{V}_{i'j})$ to form $\prod_{i' \in N(c_j) \setminus i} \text{sign}(\mathbb{V}_{i'j})$. Second, the process of calculating $|\mathbb{C}_{ji}|$ involves finding two minimum values, the first and second minimum of all the $|\mathbb{V}_{i'j}|$ at check node c_j, denoted $\mathbb{M}_{1,j}$ and $\mathbb{M}_{2,j}$, respectively. For each $\mathbb{C}_{ji}$, if the variable node v_i corresponds to $\mathbb{M}_{1,j}$, then $|\mathbb{C}_{ji}| = \mathbb{M}_{2,j}$, otherwise, $|\mathbb{C}_{ji}| = \mathbb{M}_{1,j}$. The implementation of (5) or (6) is the same with an extra step of attenuating or offsetting the minimum values.

The process of finding $\mathbb{M}_{1,j}$ and $\mathbb{M}_{2,j}$ is complex to implement. Therefore, several methods have been suggested to reduce the complexity of the process [36] or to avoid calculating $\mathbb{M}_{2,j}$ and instead estimate it based on $\mathbb{M}_{1,j}$ [27]–[29]. The result is that $\mathbb{M}_{1,j}$ plays an important role in the check node processing unit, and the new algorithm introduced in this paper also relies on $\mathbb{M}_{1,j}$, thus making the extension of the algorithm to techniques designed for complexity reduction possible.

C. Quantized Decoders

In a uniform quantized decoder, the operations in (1) – (6) have finite precision, *i.e.,* the values are quantized to a set of numbers ranging from $-\ell_{\max}$ to $\ell_{\max}$, with step size Δ, where the resulting quantizer thresholds are set from $-\ell_{\max} + \frac{\Delta}{2}$ to $\ell_{\max} - \frac{\Delta}{2}$. The attenuation and offset parameters α and β in (5) and (6) that have the best iterative decoding thresholds were found using quantized DE in [4]. In [5], the effects of these modifications on the performance of unquantized, clipped, and quantized versions of the MSA for three different codes were studied using extensive simulations.

D. Trapping Sets and Error Floors

Let A denote a subset of V of cardinality a. Let A_{even} and A_{odd} represent the subsets of check nodes connected to variable nodes in A with even and odd degrees, respectively, where $|A_{\text{odd}}| = b$. Here, A is called an (a, b) *trapping set* [14]. A is defined to be an (a, b) *absorbing set* if each variable node in A is connected to fewer check nodes in A_{odd} than in A_{even} [17]. These sets, along with similar objects such as elementary trapping sets and leafless elementary trapping sets, are known to cause most of the decoding errors at high SNRs in MP decoders [17]. In Fig. 1, the sub-graph $\mathcal{G}(A)$ induced by a $(5, 3)$ absorbing set A is shown. In the next section, we will explain how the check node LLRs in (2) can be modified to improve decoding performance at high SNR, *i.e.*, to lower the error floor, by considering decoder behavior in the presence of such problematic objects.

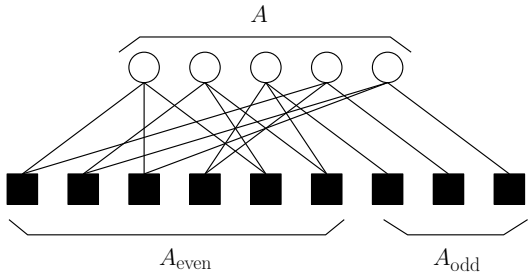


Fig. 1: The sub-graph $\mathcal{G}(A)$ induced by a $(5, 3)$ absorbing set A.

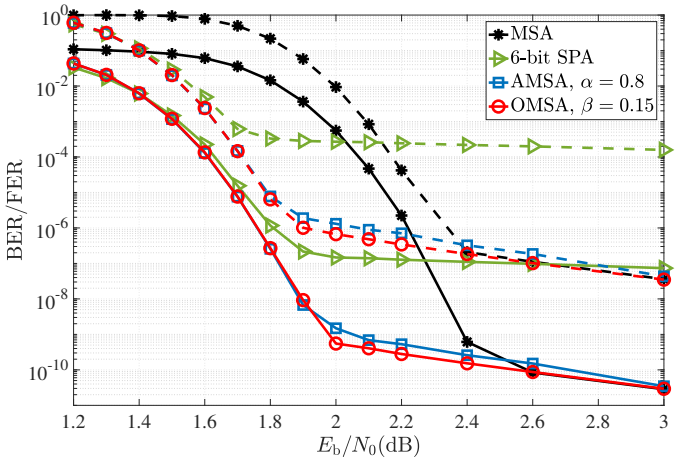


Fig. 2: Simulated performance of an (8000,4000) LDPC code decoded with the MSA, AMSA, and OMSA. Solid curves represent BER, dashed curves represent FER.

III. Threshold Attenuated/Offset MSA

A. Motivation and Rationale

In this section, we present and discuss some empirical observations that motivate our approach in this paper. As discussed in the previous section, it is known that applying attenuation or offset when computing the check node LLRs typically improves performance in the low SNR (waterfall) region of the BER curve for quantized decoders. On the other hand, since high SNR performance is tied to problematic graphical objects, the AMSA and OMSA do not necessarily achieve a good error floor. For example, assuming BPSK modulation on the AWGN channel, Fig. 2 presents the simulated bit-error-rate (BER) and frame-error-rate (FER) performance of the (8000,4000) code of [37] (which was used in [4] and [5]) with a 5-bit uniform quantizer, $\Delta = 0.15$, and $\ell_{\max} = 2.25$, decoded using the MSA, AMSA, and OMSA. We also show the performance of quantized SPA using 6-bit quantization (2-bit integer, 3-bit fractional) for comparison.[4] We see that the AMSA and OMSA gain about 0.7dB in the waterfall compared to the MSA. However, all the algorithms eventually exhibit an error floor at higher SNRs. In the remainder of this section, we discuss the high SNR decoder dynamics of an LDPC code decoded with the MSA, AMSA, or OMSA and the relationship to the problematic object(s) that cause the error floor.

At high SNRs, for a received vector $\mathbf{r}$ of channel LLRs,

[4]As has been pointed out in other publications, *e.g.* [25], the performance of SPA in the error floor is severely affected by quantization.

$$\mathbb{C}_{ji} = \begin{cases} \left(\prod_{i' \in N(c_j) \setminus i} \operatorname{sign}\left(\mathbb{V}_{i'j}\right) \right) \cdot \min_{i' \in N(c_j) \setminus i} \left|\mathbb{V}_{i'j}\right|, & \text{if } \min_{i' \in N(c_j)} \left|\mathbb{V}_{i'j}\right| > \tau, \\ \alpha' \left(\prod_{i' \in N(c_j) \setminus i} \operatorname{sign}\left(\mathbb{V}_{i'j}\right) \right) \cdot \min_{i' \in N(c_j) \setminus i} \left|\mathbb{V}_{i'j}\right|, & \text{otherwise.} \end{cases} \tag{7}$$

$$\mathbb{C}_{ji} = \begin{cases} \left(\prod_{i' \in N(c_j) \setminus i} \operatorname{sign}\left(\mathbb{V}_{i'j}\right) \right) \cdot \min_{i' \in N(c_j) \setminus i} \left|\mathbb{V}_{i'j}\right|, & \text{if } \min_{i' \in N(c_j)} \left|\mathbb{V}_{i'j}\right| > \tau, \\ \left(\prod_{i' \in N(c_j) \setminus i} \operatorname{sign}\left(\mathbb{V}_{i'j}\right) \right) \cdot \max\{ \min_{i' \in N(c_j) \setminus i} \left|\mathbb{V}_{i'j}\right| - \beta', 0\}, & \text{otherwise.} \end{cases} \tag{8}$$

decoding is successful with high probability. In the case of unsuccessful decoding, it is known that a small number of problematic objects are likely to be the cause - objects containing variable nodes with unreliable (small magnitude) LLR values [17]. In this regime, the channel LLRs for the variable nodes outside a problematic object will be, however, mostly reliable and have large magnitudes. In other words, the outside LLRs are typically initially large (with the correct sign) and will continue to grow quickly to even larger values (often $\ell_{\max}$). However, even if some/all of the incorrect sign LLRs inside a problematic object are initially small, they can also be observed to grow quickly to larger values without correcting the errors in sign. This happens because the object contains at least one short cycle, which prevents correction of the sign errors [14], [16], [17].

To improve the probability of correcting errors occurring in a problematic object $\mathcal{G}(A)$ at high SNR, we have observed by extensive numerical simulations that it is helpful if the LLR magnitudes sent from a check node $c_j \in A_{\text{even}}$ to variable nodes $v_i \in A$ grow more slowly (be attenuated) when c_j receives at least one unreliable (small magnitude) LLR from a variable node in A. This ensures that any incorrect LLRs received from the channel in A are not reinforced. If a check node c_j (inside or outside $\mathcal{G}(A)$) receives all large magnitude LLRs, on the other hand, these can be helpful for decoding and hence should not be attenuated. These two factors form the essence of the new threshold-based modification of AMSA/OMSA, presented below, that can lead to correct decoding of a received vector $\mathbf{r}$ that would not otherwise occur.

B. A Threshold Attenuated/Offset MSA

In our modified algorithm, we make use of a relationship observed at high SNRs between the variable node LLR magnitudes $|\mathbb{V}_{ij}|$ received by check node c_j and the likelihood of the check node c_j being inside a problematic object $\mathcal{G}(A)$. This relationship allows us to map the problem of locating errors affected by $\mathcal{G}(A)$ into merely considering the variable node LLR magnitudes $|\mathbb{V}_{ij}|$ received at check node c_j, *i.e.*, we rely on the $|\mathbb{V}_{ij}|$'s to tell us if c_j is likely to be inside $\mathcal{G}(A)$ and has the potential to cause decoding failures. At high SNRs, the check node LLRs outside $\mathcal{G}(A)$ typically grow faster than the LLRs inside $\mathcal{G}(A)$. Therefore, if a check node c_j receives at least one small LLR, *i.e.,* $\min_{i' \in N(c_j)} |\mathbb{V}_{i'j}| \triangleq |\mathbb{M}_{1,j}| \leq \tau$, where τ is some threshold, it is likely that c_j is inside $\mathcal{G}(A)$. Consequently, to improve the error floor performance, we propose the check node computation in (7) to replace (2), where $\alpha' \leq 1$ is an *attenuation parameter* designed to reduce the check node LLR magnitudes sent from a check node c_j inside $\mathcal{G}(A)$ to the variable nodes in A. We denote this modified check node update algorithm as the *threshold attenuated MSA (TAMSA)*. We will see in Sec. IV that, with a proper choice of the parameters τ and α', the TAMSA is capable of correctly decoding some of the errors that occur in the AMSA or MSA due to the problematic objects.

In (7), we use α' to make the check node LLR magnitudes smaller when $\min_{i' \in N(c_j)} |\mathbb{V}_{i'j}| \leq \tau$. As an alternative (or in combination), an offset parameter β' can be used to serve the same purpose, as shown in (8), where $\beta' > 0$ is an *offset parameter* that reduces the check node LLR magnitudes. We denote this modified check node update algorithm as the *threshold offset MSA (TOMSA)*.

Both the TAMSA and TOMSA *selectively*, or *locally*, reduce the magnitudes of the check node LLRs that are likely to belong to a problematic object *without requiring knowledge of its location or structure*. The TAMSA and TOMSA add a simple threshold test compared to the AMSA and OMSA, while the attenuation (offset) parameter only needs to be applied to a few check nodes at high SNRs.

C. Implementation of Threshold Attenuated/Offset MSA

We reviewed the implementation of the MSA, AMSA, and OMSA in Sec. II-B. For the MSA, for some number K of inputs to a check node processing unit, the implementation of sub-units to calculate $\mathbb{M}_{1,j}$ and $\mathbb{M}_{2,j}$ and the index needed to identify which input created $\mathbb{M}_{1,j}$ requires a significant number of multiplexers, comparators, and inverters, which is a function of K (see Table II in [36]). A check node processing unit must include some additional sub-units to generate the proper output and apply the attenuation (offset) parameter for the AMSA (OMSA).

Implementation of the TAMSA (TOMSA) adds just two simple steps to the implementation of the AMSA (OMSA). First, for a check node processing unit corresponding to c_j, after calculating $\mathbb{M}_{1,j}$ and $\mathbb{M}_{2,j}$, the value of $\mathbb{M}_{1,j}$ must be compared to τ. Second, a decision must be made based on the outcome of the comparison to use the attenuated (offset)

or non-attenuated (non-offset) output. Consequently, implementation of the TAMSA (TOMSA) requires just one extra comparator and K extra multiplexers to decide if attenuation (offset) should be applied. If not, the additional multiplication (subtraction) from attenuation (offset) is not necessary. Hence, the extra requirements do not significantly increase the overall area or delay of a check node processing unit.

IV. Results and Discussion

In this section, to illustrate the robustness of the new algorithm, we consider the $(8000, 4000)$ code of [37] (used in [4] and [5]), the progressive edge growth (PEG) $(1008, 504)$ LDPC code of [37] (used in [4]), and the quasi-cyclic $(155, 64)$ Tanner code of [38] decoded with various algorithms, including the proposed TAMSA and TOMSA with different parameters, each using a 5-bit uniform quantizer with $\Delta = 0.15$ and $\ell_{\max} = 2.25$.[5] In Section IV-A, we use a performance estimation method to investigate how different parameter sets can affect the error floor of any code containing a known dominant problematic graphical object. In Section IV-B, we show the simulated performance of the above codes with the TAMSA for different parameter sets, and in Section IV-C we discuss the choice of parameter sets that optimize the performance of TAMSA. Finally, in Section V, we apply the TAMSA to SC-LDPCCs and show the benefits of combining spatial coupling with the TAMSA for improved performance in both the waterfall and error floor relative to LDPC-BCs.

A. Performance Estimation Based on problematic objects

In this section, we utilize the method of [18] to estimate the impact of a problematic object on the performance of an LDPC code decoded with the MSA, AMSA, and TAMSA. In that paper, the authors derived a lower bound on the FER of any LDPC code containing a given problematic object (sub-graph), assuming a particular message passing decoder and decoder quantization. A crucial aspect of the lower bound is that it is code-independent, in the sense that it can be derived based only on a problematic object and then applied to any code containing that object. Given the dominant problematic object, decoder quantization, and decoding algorithm, a performance estimate of the code containing the dominant object is derived. We remark again that the number, type, and location of problematic objects in the Tanner graph do not need to be known to implement the algorithm, but we will see in this section that, if the dominant problematic object is known, then it can facilitate determination of the optimum algorithm parameters. The bounds were shown in [18] to be tight for a variety of codes, problematic objects, and decoding algorithms.

By analyzing the AWGN channel performance simulations of the $(8000, 4000)$ code with a 5-bit quantizer, we find that the $(5, 3)$ absorbing set of Fig. 1 is the major cause of errors in the error floor.[6] Based on this problematic object, high SNR performance estimates of the code (or any code containing this problematic object as the dominant object) can be obtained for various TAMSA parameter sets and for various values of $E_{\rm b}/N_0(\rm dB)$ using the method of [18]. For example, Fig. 3(a) plots the estimated FER performance vs. the parameter sets (α', τ) for $0.5 \leq \alpha' \leq 1$, $0 \leq \tau \leq \ell_{\max}$, and $E_{\rm b}/N_0 = 2\rm dB$.[7] (A contour plot of the same data is shown in Fig. 3(b).) Note that when $\tau = \ell_{\max}$, the TAMSA is equivalent to the AMSA with $\alpha = \alpha'$, since attenuation is always applied. As shown in Fig. 3(a), the line $\tau = \ell_{\max} = 2.25$ has a very high FER, meaning that any code containing this $(5, 3)$ absorbing set is adversely affected in the error floor when decoded using the AMSA. Also, in two special cases, $(\alpha', \tau = 0)$ and $(\alpha' = 1, \tau)$, the TAMSA is equivalent to the MSA, since attenuation is never applied for these parameter sets. From Fig. 3(a), we can therefore predict that the MSA will perform better than the AMSA in the error floor for any code for which the $(5, 3)$ absorbing set is dominant. It is also important to note from Fig. 3(a) that there are values of α and τ that lead to better performance than can be achieved by either the AMSA or the MSA for any code for which the $(5, 3)$ absorbing set is dominant. This observation supports our claim that making use of the parameter τ in the TAMSA will reduce the error probability associated with a specific problematic object compared to either the AMSA or the MSA.

B. Simulated Performance of LDPC Codes with TAMSA and TOMSA Decoders

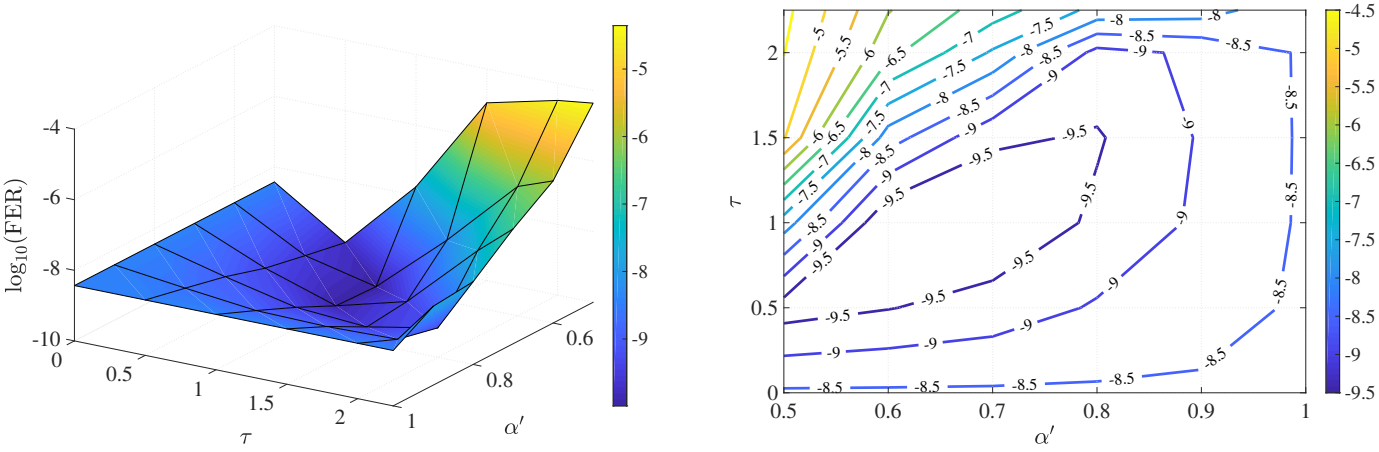
Fig. 4 shows the BER and FER performance of the $(8000, 4000)$ code for the MSA, the AMSA with $\alpha = 0.8$, the OMSA with $\beta = 0.15$, the TAMSA with parameters $(\alpha' = 0.8, \tau = 1.5)$, and the TOMSA with parameters $(\beta' = 0.15, \tau = 2)$. A syndrome-check stopping rule with a maximum number of iterations $I_{\max} = 50$ was employed for all decoders. We see that, for the chosen parameters, the TAMSA and TOMSA exhibit one to two orders of magnitude better error floor performance than the MSA, AMSA, and OMSA while maintaining the same waterfall performance.

Fig. 5 shows the BER and FER performance of the (semi-structured) $(1008, 504)$ PEG-LDPC code for the AMSA, OMSA, and TAMSA with three parameter sets: $(\alpha' = 0.8, \tau = 2)$, $(\alpha' = 0.8, \tau = 1.75)$, and $(\alpha' = 0.75, \tau = 1.75)$. Again we see that the best error floors are achieved with the TAMSA. The parameter set $(\alpha' = 0.75, \tau = 1.75)$ exhibits the most gain, about 1.5 orders of magnitude compared to the AMSA and OMSA for $E_{\rm b}/N_0 = 4\rm dB$, but its waterfall performance is slightly worse than for the parameter sets $(\alpha' = 0.8, \tau = 2)$ and $(\alpha' = 2, \tau = 1.75)$. We note that this behavior allows us

[5] In our simulations, for the check node calculations, we do not quantize α, β, and τ. However, the check node output is quantized. In the case of β (τ), there is a unique quantized β (τ) that generates the same quantized check node output as when β (τ) is unquantized. However, this is not the case for α.

[6] In all of our examples, the dominant problematic objects are determined following the method of [14], *i.e.*, by recording the variable node error locations in the graph after numerical computer simulations over a variety of SNRs, then classifying the sub-graphs and tabulating statistics. At least 50 error frames are collected for each determination.

[7] It was observed through simulations that $E_{\rm b}/N_0 = 2\rm dB$ is the start of the error floor in this case. Similar results, yielding the same conclusions, can also be obtained for larger values of $E_{\rm b}/N_0$.



(a) $\log_{10}(\text{FER})$ performance estimate.

(b) Contour plots of the $\log_{10}(\text{FER})$ performance estimate.

Fig. 3: Performance estimate of the $(5, 3)$ absorbing set vs. different values of α and τ for $E_\text{b}/N_0 = 2$dB.

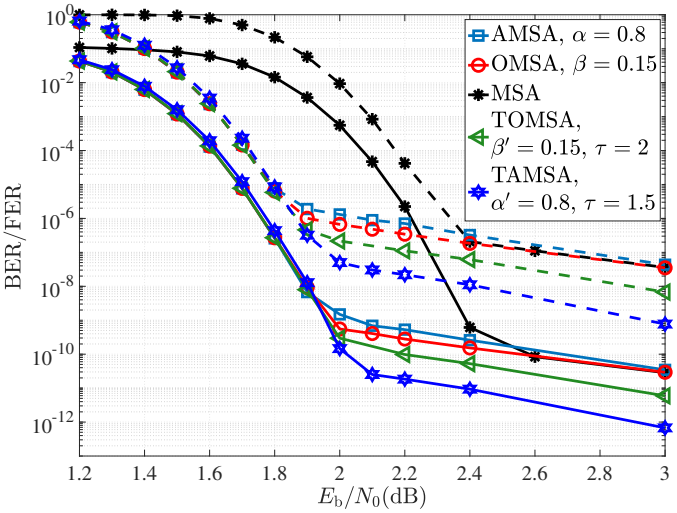


Fig. 4: Simulated performance of an (8000,4000) LDPC code decoded with the MSA, AMSA, OMSA, TAMSA, and TOMSA. Solid curves represent BER, dashed curves represent FER.

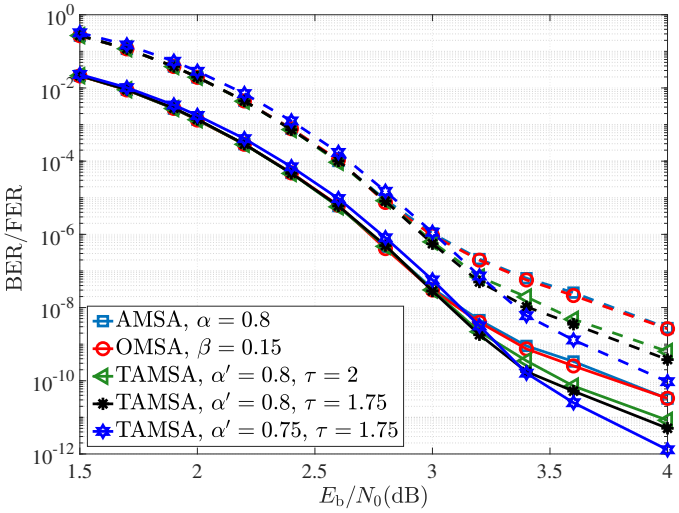


Fig. 5: Simulated performance of the (1008,504) PEG-LDPC code decoded with the AMSA, OMSA, and TAMSA. Solid curves represent BER, dashed curves represents FER.

to tune the performance of the TAMSA to fit a particular application by choosing the values of α' and τ.

Fig. 6 shows the BER performance of the (highly structured) quasi-cyclic $(155, 64)$ Tanner code for the AMSA with different values of α, the TAMSA with parameter set $(\alpha' = 0.8, \tau = 1.5)$, and the MSA. Again we see that, at high SNRs, the TAMSA significantly outperforms both the AMSA and the MSA, gaining about one order of magnitude in the error floor. We further note that an important performance metric for comparison of these algorithms is the *average* number of iterations performed. Table I gives the average number of iterations for the AMSA, MSA, and TAMSA recorded from 1dB to 8dB. We observe that AMSA and TAMSA have an approximately equal number of average iterations. Moreover, both AMSA and TAMSA provide a significant reduction in the average number of iterations when compared to MSA at low SNR. This advantage diminishes as the SNR increases and we observe that all of the algorithms have a similar average number of iterations at high SNR.

E_b/N_0	MSA	AMSA	TAMSA
1dB	68.95	59.28	59.24
2dB	30.4	23.13	22.9
3dB	7.82	6.28	6.2
4dB	3.06	2.95	2.87
5dB	1.97	1.98	1.98
6dB	1.44	1.46	1.46
7dB	1.09	1.1	1.1
8dB	0.85	0.86	0.86

TABLE I: Average number of iterations recorded for the quasi-cyclic $(155, 64)$ Tanner code with the MSA, AMSA, and TAMSA decoding algorithms.

Finally, layered MP decoding of LDPC-BCs [39] has been shown to converge faster than standard MP decoding and is commonly employed in the implementation of quasi-cyclic codes. Fig. 7 shows the BER performance of the quasi-cyclic $(155, 64)$ Tanner code, using both a layered MP decoder and a standard MP decoder with 100 iterations each, for both the MSA and the TAMSA with parameter set $(\alpha' = 0.8, \tau = 1.5)$.

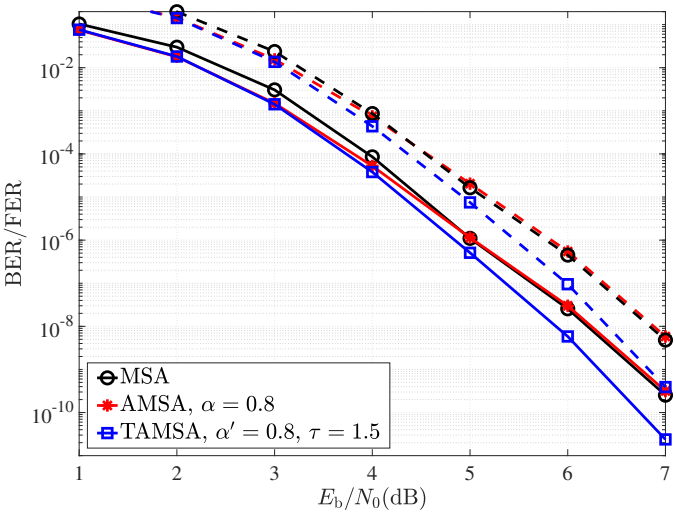


Fig. 6: Simulated performance of the $(155, 64)$ Tanner LDPC code with the MSA, AMSA, and TAMSA. Solid curves represent BER, dashed curves represents FER.

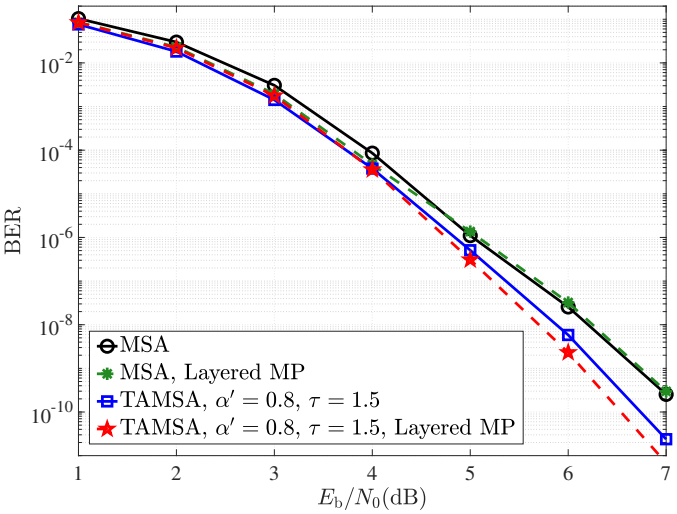


Fig. 7: Simulated BER performance of the $(155, 64)$ Tanner LDPC code with the MSA and the TAMSA with both layered MP and standard MP decoding.

It can be seen that the TAMSA with both standard and layered decoding outperforms the MSA and that the layered TAMSA even is slightly better than the standard TAMSA in the error floor. Taken together, the results of Figs. 4–7 illustrate the robustness of TAMSA decoding.

C. Parameter Set Selection for TAMSA and TOMSA Decoders

We saw in Fig. 5 that the TAMSA parameter sets that lead to the best error floor performance do not necessarily lead to the best waterfall performance. Depending on the application and design goals, the parameter sets can be chosen differently.

If $\alpha' = \alpha_{\text{opt}}$, where α_{opt} is the optimal α for the AMSA derived in [4], [5] using quantized DE, the best waterfall performance is achieved. However, choosing $\alpha' = \alpha_{\text{opt}}$ is best suited to larger values of τ, since for $\tau = \ell_{\max}$ the TAMSA and AMSA are the same, and our empirical results show that there can be a loss in waterfall performance for smaller values of τ.

In the error floor, instead of running time consuming code simulations, the method of [18] can be applied to problematic objects to find the parameter sets (α', τ) that lead to the best error floor performance. From the contour plots in Fig. 3(b) of the FER performance of any code for which the $(5, 3)$ absorbing set is dominant in the error floor, it can be seen that certain parameter sets (α', τ) lead to significantly lower FER values than others. These parameter sets can then be used to guide the selection of the parameters that yield the best error floor performance of any code for which the $(5, 3)$ absorbing set is dominant. According to Fig. 3(b), the best error floor for a code containing the $(5, 3)$ absorbing set should be achieved by choosing a parameter set in the vicinity of $(\alpha' = 0.65, \tau = 1)$.

If the goal is to achieve waterfall performance as good as the AMSA with $\alpha = \alpha_{\text{opt}}$ and to achieve a better error floor than the MSA or AMSA, a good starting point is to set $\alpha' = \alpha_{\text{opt}}$ and then choose the value of τ that leads to the best error floor estimate associated with the dominant problematic object. If there is more than one value of τ that satisfies this condition, the largest should be chosen, since larger values of τ makes the TAMSA perform closer to the AMSA optimized for waterfall performance. In Fig. 3(b), $\alpha_{\text{opt}} = 0.8$, and therefore choosing the parameter set $(\alpha' = 0.8, \tau = 1.5)$ should provide us with good performance in both the waterfall and the error floor. The simulation results in Fig. 4 of the $(8000, 4000)$ LDPC-BC using this parameter set illustrates the advantage of following this approach.

In Figs. 2, 4, and 5, we see that the OMSA slightly outperforms the AMSA at high SNRs. This follows from the fact that, for the chosen values of $\ell_{\max}$, α, and β, the LLR magnitudes for the OMSA grow to larger values than for the AMSA (quantized value of $\ell_{\max} - \beta = 2.10$ vs. quantized value of $\alpha \times \ell_{\max} = 1.8$). Adopting the terminology that check node LLRs larger than τ are *reliable* while those below τ are *unreliable*, the *reliable* check node LLRs of the OMSA (with magnitudes up to 2.10) are more likely to "correct" additional errors inside a problematic object $\mathcal{G}(A)$ than those of the AMSA (with magnitudes only up to 1.8). However, in Fig. 4, we see that the TAMSA has better error floor performance than the TOMSA. While the check node LLRs that satisfy (7) or (8), *i.e.*, the *reliable* LLRs, for both the TAMSA and the TOMSA can grow to $\ell_{\max}$, the check node LLRs that don't satisfy (7) or (8) *i.e.*, the *unreliable* LLRs, are limited to values smaller than τ (a quantized value of $\alpha' \times \tau = 1.2$ for the TAMSA vs. a quantized value of $\tau - \beta' = 1.85$ for the TOMSA). Consequently, for the parameter sets chosen in our examples, the TAMSA makes the *unreliable* check node LLRs smaller than for the TOMSA, which helps TAMSA "correct" more errors by slowing down the check node LLR convergence inside a problematic object.

We note that, as discussed in Section IV-A, AMSA (resp. OMSA) can be viewed as a particular case of TAMSA (resp. TOMSA) and, as such, the performance of TAMSA (resp. TOMSA) is at least as good as AMSA (resp. OMSA) with optimal parameter selection. Moreover, we have demonstrated

significant performance improvements for a variety of code structures and lengths.

V. Application of the TAMSA to Spatially Coupled LDPC Codes

SC-LDPCCs are known to combine the best features of both regular and irregular LDPC-BCs, *i.e.*, they achieve excellent performance both in the waterfall and the error floor regions of the BER (FER) curve [40]. In this section, we use the TAMSA to decode SC-LDPCCs to further verify the effectiveness of the algorithm and to illustrate the benefit of combining the advantages of spatial coupling and the TAMSA.

A. Background

1) SC-LDPC Parity-Check Matrix: Given an underlying LDPC-BC with a $\mu \times \nu$ parity-check matrix $\mathbf{H}_{\mathrm{BC}}$ and rate $R_{\mathrm{BC}} = 1 - \frac{\mu}{\nu}$,[8] a terminated SC-LDPCC with parity-check matrix $\mathbf{H}_{\mathrm{SC}}^{L}$ and *syndrome former memory* m can be formed by partitioning $\mathbf{H}_{\mathrm{BC}}$ into m component matrices $\mathbf{H}_i, i = 0, 1, \ldots, m$, each of size $\mu \times \nu$, such that $\mathbf{H}_{\mathrm{BC}} = \sum_{i=0}^{m} \mathbf{H}_i$, and arranging them as

$$\mathbf{H}_{\mathrm{SC}}^{L} = \begin{bmatrix} \mathbf{H}_0 & & & \\ \mathbf{H}_1 & \mathbf{H}_0 & & \\ \vdots & \mathbf{H}_1 & \ddots & \\ \mathbf{H}_m & \vdots & \ddots & \mathbf{H}_0 \\ & \mathbf{H}_m & \ddots & \mathbf{H}_1 \\ & & \ddots & \vdots \\ & & & \mathbf{H}_m \end{bmatrix}_{\mu(L+m)\times\nu L}, \tag{9}$$

where the *coupling length* $L > m + 1$ denotes the number of block columns in $\mathbf{H}_{\mathrm{SC}}$ and the rate of the terminated SC-LDPCC represented by $\mathbf{H}_{\mathrm{SC}}^{L}$ is given by $R_{\mathrm{SC}}^{L} = \frac{\nu L - \mu(L+m)}{\nu L} = 1 - \frac{\mu}{\nu}\left(1 + \frac{m}{L}\right)$, such that $\lim_{L\to\infty} R_{\mathrm{SC}}^{L} = 1 - \frac{\mu}{\nu} = R_{\mathrm{BC}}$.

2) Sliding Window Decoding of SC-LDPCCs: A sliding window (SW) decoder was introduced in [41] to address the large latency and complexity requirements of decoding SC-LDPCCs with a standard flooding schedule decoder. Fig. 8, adapted from [41], shows an SW decoder with window size $W = 6$ (blocks) operating on the parity-check matrix $\mathbf{H}_{\mathrm{SC}}$ of an SC-LDPCC with $m = 2$ and $L = 10$. All the variable nodes and check nodes included in the window (the blue area) are updated using an MP algorithm that has access to previously decoded symbols (the red area). The goal is to decode the variable nodes in the first block of the window, called the *target symbols* (the green area). The MP algorithm updates the nodes in the window until some maximum number of iterations $I_{\max}$, after which the first block of target symbols is decoded. Then the window slides one block (ν code symbols) to the right and one block down (μ parity-check symbols) to decode the second block, and the process continues until the last block of target symbols is decoded.

[8] In this section, we ignore the slight rate loss that may occur if a parity-check matrix contains a few redundant rows.

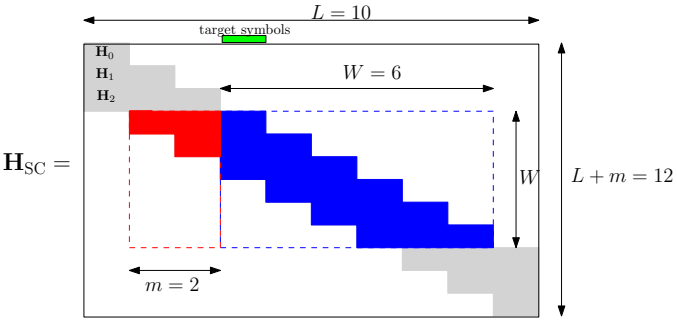


Fig. 8: An SW decoder for an SC-LDPCC operating on the parity-check matrix $\mathbf{H}_{\mathrm{SC}}$.

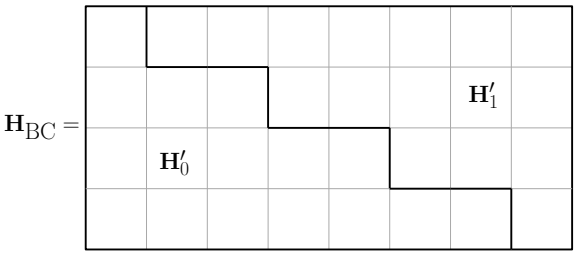


Fig. 9: LDPC-BC with 4000×8000 parity-check matrix $\mathbf{H}_{\mathrm{BC}}$, $\mu' = 4$, $\nu' = 8$, and $\gamma = 1000$, is partitioned by a cutting vector $\mathbf{w} = [1, 3, 5, 7]$ to construct the two component matrices $\mathbf{H}_0$ and $\mathbf{H}_1$ based on $\mathbf{H}_0'$ and $\mathbf{H}_1'$. (Each square represents a $\gamma \times \gamma$ matrix.)

3) Cut-and-Paste Construction of SC-LDPCCs: For the case $m = 1$,[9] the cut-and-paste method [42] of constructing SC-LDPCCs uses a *cutting vector* $\mathbf{w} = [w_0, w_1, \ldots, w_{\mu'-1}]$ of non-decreasing non-negative integers $(0 < w_0 \leq w_1 \leq \cdots \leq w_{\mu'-1} < \nu')$ to form two component matrices $\mathbf{H}_0$ and $\mathbf{H}_1$ from a $\mu \times \nu$ LDPC-BC parity-check matrix $\mathbf{H}_{\mathrm{BC}}$. The cutting vector partitions $\mathbf{H}_{\mathrm{BC}}$, composed of a $\mu' \times \nu'$ array of $\gamma \times \gamma$ blocks such that $\mu \times \nu = \mu'\gamma \times \nu'\gamma$, into two parts, one below and one above the cut, which we represent by $\mathbf{H}_0'$ and $\mathbf{H}_1'$, respectively. Fig. 9 shows an example of a matrix $\mathbf{H}_{\mathrm{BC}}$ of size 4000×8000, where $\mu' = 4$, $\nu' = 8$, and $\gamma = 1000$, being partitioned into $\mathbf{H}_0'$ and $\mathbf{H}_1'$ by the cutting vector $\mathbf{w} = [1, 3, 5, 7]$. $\mathbf{H}_0$ and $\mathbf{H}_1$ in (9) are then obtained by taking $\mathbf{H}_0'$ and setting the $\mathbf{H}_1'$ part to all zeros and by taking $\mathbf{H}_1'$ and setting the $\mathbf{H}_0'$ part to all zeros, respectively, where we note that $\mathbf{H}_0 + \mathbf{H}_1 = \mathbf{H}_{\mathrm{BC}}$. The resulting code rate is given by $R_{\mathrm{SC}}^{L} = 1 - \frac{(L+1)\mu}{L\nu} = 1 - \frac{\mu}{\nu}\left(1 + \frac{1}{L}\right)$, where the underlying LDPC-BC has rate $R_{\mathrm{BC}} = 1 - \frac{\mu}{\nu}$. For quasi-cyclic LDPC-BCs, such as array codes and Tanner codes, the parameter γ is set equal to the size of the circulant permutation matrices in order to maintain the code structure.

B. Simulation Results

We now present simulation results for the SC-LDPCC versions of the $(8000, 4000)$ LDPC-BC and the quasi-cyclic $(155, 64)$ Tanner code decoded with the TAMSA and an SW decoder with $W = 6$, where 50 iterations were performed at each window position. The SC-LDPCCs both have coupling length $L = 50$ and syndrome former memory $m = 1$. For

[9] Syndrome former memory $m = 1$ is chosen for illustration, but the method easily extends to larger values of m.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2020.2969902, IEEE Transactions on Communications

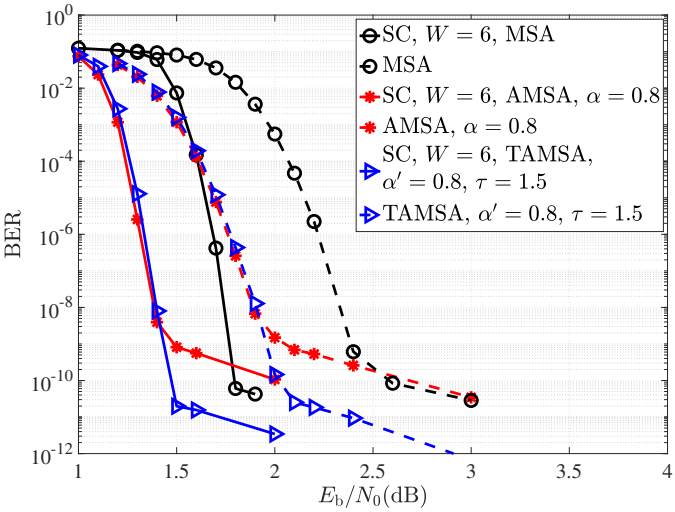


Fig. 10: Simulated performance of an $(8000, 4000)$ LDPC-BC and its spatially coupled version decoded with the MSA, AMSA, and TAMSA. Dashed curves represent the LDPC-BC, solid curves represent the SC-LDPCC decoded with an SW decoder with $W = 6$.

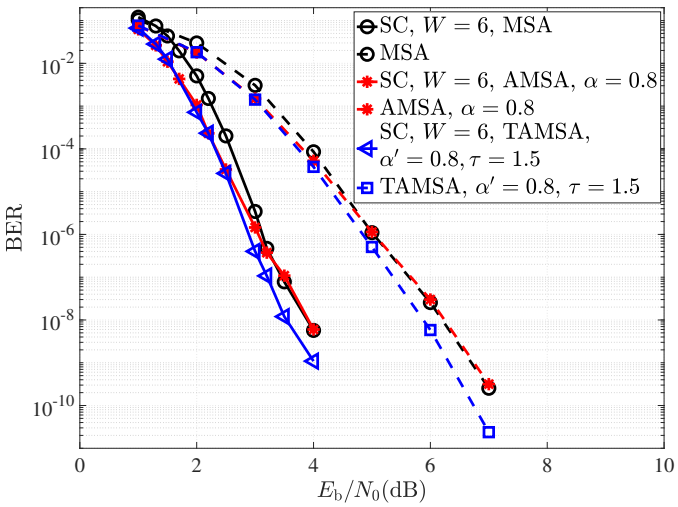


Fig. 11: Simulated performance of the quasi-cyclic $(155, 64)$ Tanner LDPC-BC and its spatially coupled version decoded with an SW decoder with $W = 6$ and the MSA, AMSA, and TAMSA.

the $(8000, 4000)$ code, $\gamma = 1000$ and the cutting vector $\mathbf{w} = [1, 3, 5, 7]$ (as shown in Fig. 9) was chosen, and for the $(155, 64)$ Tanner code, $\gamma = 31$ and $\mathbf{w} = [2, 3, 5]$ was chosen, where the size of the circulant permutation matrices is 31.

Fig. 10 shows the BER performance of the $(8000, 4000)$ LDPC-BC and its spatially coupled version decoded with an SW decoder with $W = 6$ for the MSA, the AMSA with $\alpha = 0.8$, and the TAMSA with parameter set $(\alpha' = 0.8, \tau = 1.5)$. We see that, for the chosen parameters, the TAMSA again has nearly two orders of magnitude better error floor performance than the MSA and the AMSA, for both the LDPC-BC and the SC-LDPCC, and it maintains the same waterfall performance. In addition, the spatial coupling yields a waterfall gain of about 0.5dB for all the decoding algorithms compared to the underlying LDPC-BC. Moreover, the dominant problematic object for both the LDPC-BC and the SC-LDPCC decoded with the algorithms and parameters in Fig. 10 is the $(5, 3)$ absorbing set of Fig. 1. The multiplicity of this object is $N = 14$ and $\hat{N} = 4.92$ for the LDPC-BC and SC-LDPCC, respectively, where $\hat{N}$ is the average multiplicity per block of size $\nu = 8000$ [24]. Therefore, spatial coupling reduces the number of dominant problematic objects by 64%. This explains the almost one order of magnitude gain in the error floor obtained by spatial coupling compared to the underlying LDPC-BC.

Fig. 11 shows the BER performance of the quasi-cyclic $(155, 64)$ Tanner code and its spatially coupled version decoded with an SW decoder with $W = 6$ for the MSA, the AMSA with $\alpha = 0.8$, and the TAMSA with parameter set $(\alpha' = 0.8, \tau = 1.5)$. Again, we see that the TAMSA outperforms the AMSA and the MSA at high SNRs by about one order of magnitude in the error floor, for both the LDPC-BC and the SC-LDPCC.[10] We also note an approximately 2dB gain of the SC-LDPCC compared to its underlying LDPC-BC in the waterfall. The dominant problematic object for the $(155, 64)$ Tanner LDPC-BC decoded with the algorithms and parameters in Fig. 11 is an $(8, 2)$ absorbing set. The multiplicity of this object is $N = 465$, but in this case $\hat{N} = 0$ for the SC-LDPCC, *i.e.*, this object is completely removed by spatial coupling. As a result, we observe almost five orders of magnitude gain at $E_b/N_0 > 3$dB for the SC-LDPCC compared to the underlying LDPC-BC!

[10]The "error floor" for these codes are designated by the change of slope observed at approximately 3dB and 5dB for the SC-LDPCC and LDPC-BC, respectively.

VI. Conclusion

In this paper, a modified version of the MSA was proposed to lower the error floor performance of quantized LDPC decoders. Based on the assumption that a problematic object, such as a trapping set or absorbing set, is the dominant cause of the error floor, the proposed TAMSA (TOMSA) selectively attenuates (offsets) a check node LLR if the check node receives any variable node LLR with magnitude below some threshold τ, while allowing a check node LLR to reach the maximum quantizer level if all the variable node LLRs received by the check node have magnitude greater than τ. This new approach can decode some received vectors of channel LLRs that become stuck, and thus cannot be decoded correctly, using previously proposed versions of the MSA. We note that, even though the method is based on the knowledge that problematic objects are the dominant cause of the error floor, the algorithm requires *no knowledge of the location, type, or multiplicity of such objects* and can be implemented with only a minor modification to the decoder hardware. The complexity of the algorithm was shown to be comparable to the complexity of AMSA or OMSA. As an alternative to running time-consuming simulations, an error floor estimation technique was used to guide the selection of the parameter sets that yield the best error floor performance. Finally, simulation results presented for several codes, including both LDPC-BCs and SC-LDPCCs, demonstrated that the TAMSA (TOMSA) combines the advantages of both the MSA and the AMSA

(OMSA) to offer better error floor performance without sacrificing waterfall performance.

REFERENCES

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, Jan. 1962.

[2] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. Commun.*, vol. 47, no. 5, pp. 673–680, May 1999.

[3] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb 2001.

[4] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288–1299, Aug 2005.

[5] J. Zhao, F. Zarkeshvari, and A. H. Banihashemi, "On implementation of min-sum algorithm and its modifications for decoding low-density parity-check (LDPC) codes," *IEEE Trans. Commun.*, vol. 53, no. 4, pp. 549–554, April 2005.

[6] Meng Xu, Jianhui Wu, and Meng Zhang, "A modified offset min-sum decoding algorithm for LDPC codes," in *3rd International Conference on Computer Science and Information Technology*, vol. 3, July 2010, pp. 19–22.

[7] Zhou Zhong, Yunzhou Li, Xiang Chen, Hanying Hu, and Jing Wang, "Modified min-sum decoding algorithm for LDPC codes based on classified correction," in *Third International Conference on Communications and Networking in China*, Aug 2008, pp. 932–936.

[8] J. Tang, T. Bhatt, and V. Stolpman, "Modified min-sum algorithm for LDPC decoders in UWB communications," in *IEEE International Conference on Ultra-Wideband*, Sep. 2006, pp. 125–130.

[9] X. Zhang and P. H. Siegel, "Quantized min-sum decoders with low error floor for LDPC codes," in *IEEE International Symposium on Information Theory Proceedings*, July 2012, pp. 2871–2875.

[10] D. Oh and K. K. Parhi, "Optimally quantized offset min-sum algorithm for flexible LDPC decoder," in *42nd Asilomar Conference on Signals, Systems and Computers*, Oct 2008, pp. 1886–1891.

[11] D. Oh and K. K. Parhi, "Min-sum decoder architectures with reduced word length for LDPC codes," *IEEE Trans. Circuits Syst., I*, vol. 57, no. 1, pp. 105–115, Jan 2010.

[12] J. Zhang, M. Fossorier, D. Gu, and J. Zhang, "Two-dimensional correction for min-sum decoding of irregular LDPC codes," *IEEE Commun. Lett.*, vol. 10, no. 3, pp. 180–182, March 2006.

[13] X. Wu, Y. Song, M. Jiang, and C. Zhao, "Adaptive-normalized/offset min-sum algorithm," *IEEE Commun. Lett.*, vol. 14, no. 7, pp. 667–669, July 2010.

[14] T. J. Richardson, "Error floors of LDPC codes," in *41st Annu. Allerton Conf. Commun., Control, Comput.*, Monticello, IL, USA, Oct 2003, p. 1426– 1435.

[15] O. Milenkovic, E. Soljanin, and P. Whiting, "Asymptotic spectra of trapping sets in regular and irregular LDPC code ensembles," *IEEE Trans. Inf. Theory*, vol. 53, no. 1, pp. 39–55, Jan 2007.

[16] M. Karimi and A. H. Banihashemi, "On characterization of elementary trapping sets of variable-regular LDPC codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5188–5203, Sep. 2014.

[17] L. Dolecek, Z. Zhang, V. Anantharam, M. Wainwright, and B. Nikolic, "Analysis of absorbing sets and fully absorbing sets of array-based LDPC codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 181–201, Jan. 2010.

[18] H. Hatami, D. G. M. Mitchell, D. J. Costello, and T. E. Fuja, "Performance bounds and estimates for quantized LDPC decoders," *IEEE Transactions on Communications, to appear*, 2019. [Online]. Available: https://arxiv.org/abs/1911.02762

[19] L. Dolecek, P. Lee, Z. Zhang, V. Anantharam, B. Nikolic, and M. Wainwright, "Predicting error floors of structured LDPC codes: deterministic bounds and estimates," *IEEE J. on Sel. Areas in Commun.*, vol. 27, no. 6, pp. 908–917, August 2009.

[20] B. K. Butler and P. H. Siegel, "Error floor approximation for LDPC codes in the AWGN channel," *IEEE Trans. Inf. Theory*, vol. 60, no. 12, pp. 7416–7441, Dec 2014.

[21] J. Sun, O. Y. Takeshita, and M. P. Fitz, "Analysis of trapping sets for LDPC codes using a linear system model," in *Proc. 42nd Annu. Allerton Conf., Monticello, IL, USA,*, Sep./Oct. 2004, pp. 1701–1702.

[22] S. Zhang and C. Schlegel, "Controlling the error floor in LDPC decoding," *IEEE Trans. Commun.*, vol. 61, no. 9, pp. 3566–3575, Sept. 2013.

[23] H. Xiao, A. Banihashemi, and M. Karimi, "Error rate estimation of low-density parity-check codes decoded by quantized soft-decision iterative algorithms," *IEEE Trans. Commun.*, vol. 61, no. 2, pp. 474–484, Feb. 2013.

[24] H. Hatami, D. G. M. Mitchell, D. J. Costello, and T. E. Fuja, "Performance bounds for quantized spatially coupled LDPC decoders based on absorbing sets," in *Proc. IEEE Int. Symp. Inf. Theory*, June 2018, pp. 826–830.

[25] X. Zhang and P. Siegel, "Quantized iterative message passing decoders with low error floor for LDPC codes," *IEEE Trans. Commun.*, vol. 62, no. 1, pp. 1–14, Jan. 2014.

[26] S. Tolouei and A. H. Banihashemi, "Lowering the error floor of LDPC codes using multi-step quantization," *IEEE Commun. Lett.*, vol. 18, no. 1, pp. 86–89, January 2014.

[27] A. Darabiha, A. C. Carusone, and F. R. Kschischang, "A bit-serial approximate min-sum LDPC decoder and FPGA implementation," in *IEEE Int. Symp. Circuits Syst.*, May 2006.

[28] F. Angarita, J. Valls, V. Almenar, and V. Torres, "Reduced-complexity min-sum algorithm for decoding LDPC codes with low error-floor," *IEEE Trans. Circuits Syst., I*, vol. 61, no. 7, pp. 2150–2158, July 2014.

[29] S. Hemati, F. Leduc-Primeau, and W. J. Gross, "A relaxed min-sum LDPC decoder with simplified check nodes," *IEEE Commun. Lett.*, vol. 20, no. 3, pp. 422–425, March 2016.

[30] C. Zhang, Z. Wang, J. Sha, L. Li, and J. Lin, "Flexible LDPC decoder design for multigigabit-per-second applications," *IEEE Trans. Circuits Syst., I*, vol. 57, no. 1, pp. 116–124, Jan 2010.

[31] G. B. Kyung and C. C. Wang, "Finding the exhaustive list of small fully absorbing sets and designing the corresponding low error-floor decoder," *IEEE Trans. Commun.*, vol. 60, no. 6, pp. 1487–1498, June 2012.

[32] N. Varnica, M. P. C. Fossorier, and A. Kavcic, "Augmented belief propagation decoding of low-density parity check codes," *IEEE Trans. Commun*, vol. 55, no. 7, pp. 1308–1317, July 2007.

[33] Y. Han and W. E. Ryan, "Low-floor decoders for LDPC codes," *IEEE Trans. Commun.*, vol. 57, no. 6, pp. 1663–1673, June 2009.

[34] Z. Zhang, V. Anantharam, M. J. Wainwright, and B. Nikolic, "An efficient 10GBASE-T ethernet LDPC decoder design with low error floors," *IEEE J. Solid-State Circuits*, vol. 45, no. 4, pp. 843–855, April 2010.

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

[36] C. Wey, M. Shieh, and S. Lin, "Algorithms of finding the first two minimum values and their hardware implementation," *IEEE Trans. Circuits. Syst., I*, vol. 55, no. 11, pp. 3430–3437, Dec 2008.

[37] D. J. C. MacKay. Encyclopedia of sparse graph codes. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html

[38] R. M. Tanner, D. Sridhara, A. Sridharan, T. E. Fuja, and D. J. Costello, "LDPC block and convolutional codes based on circulant matrices," *IEEE Trans. Inf. Theory*, vol. 50, no. 12, pp. 2966–2984, Dec 2004.

[39] D. E. Hocevar, "A reduced complexity decoder architecture via layered decoding of LDPC codes," in *Proc. IEEE Wksp. Sig. Processing Sys.*, Oct 2004, pp. 107–112.

[40] D. J. Costello, L. Dolecek, T. E. Fuja, J. Kliewer, D. G. M. Mitchell, and R. Smarandache, "Spatially coupled sparse codes on graphs: theory and practice," *IEEE Communications Magazine*, vol. 52, no. 7, pp. 168–176, July 2014.

[41] A. Iyengar, M. Papaleo, P. Siegel, J. Wolf, A. Vanelli-Coralli, and G. Corazza, "Windowed decoding of protograph-based LDPC convolutional codes over erasure channels," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2303–2320, April. 2012.

[42] D. G. M. Mitchell, A. E. Pusane, and D. J. Costello, "Minimum distance and trapping set analysis of protograph-based LDPC convolutional codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 1, pp. 254–281, Jan 2013.