Nonbinary LDPC Decoding by Min-Sum with Adaptive Message Control

Weiguo Tang, Jie Huang, Lei Wang and Shengli Zhou
Department of Electrical and Computer Engineering
University of Connecticut, Storrs, CT 06269
Email: {weiguo.tang, jhuang, leiwang, shengli}@engr.uconn.edu

Abstract—A new decoding algorithm, referred to as Min-Sum with Adaptive Message Control (AMC-MS), is proposed to reduce the decoding complexity of nonbinary LDPC codes. The proposed algorithm adaptively trims the message length of belief information to reduce the amount of arithmetic operations. Exploiting the fact that during the decoding iteration, the distribution of belief information will become more concentrated around the correct element in the case of convergence, the messages can be truncated accordingly by considering only a few entries with large likelihood. Simulation results with a GF(2^m) nonbinary LDPC code indicate that the proposed approach can reduce arithmetic operations by up to 65% compared with non-truncated cases. Compared with the state-of-the-art extended MS (EMS) decoding, the proposed AMC-MS algorithm can reduce the computation by up to 50%, thereby enabling low-complexity decoding of nonbinary LDPC codes.

I. INTRODUCTION

Low-density parity-check (LDPC) codes [1], [2] are considered as one of the most powerful capacity-approaching codes. LDPC codes can be constructed in both binary and high-order Galois fields (i.e., GF(2^m)), where m > 1). Binary LDPC codes have been studied extensively [3]–[5] and adopted in many communication protocols, such as DVB-T2, WiMax, etc. In general, a very long code length is required for binary LDPC codes to approach the channel capacity. Nonbinary LDPC codes constructed in high-order Galois fields [7], [9] offer improved performance at moderate code lengths. In addition, nonbinary LDPC codes can be seamlessly combined with high order modulation [17]. Due to these attractive features, nonbinary LDPC codes have become a new research frontier.

A key challenge in the application of nonbinary LDPC codes is the high decoding complexity, as each symbol in the codeword is decoded using a very long message (e.g., 2^m) in GF(2^m), see Section II-A for the definition of message). A lot of research effort aims at reducing the decoding complexity of nonbinary LDPC codes [6], [11]–[13], [15]. The FFT-BP [6], [11] implemented convolution operations with FFT, reducing the complexity from \(O(2^m)\) to \(O(m2^m)\). Log-domain representation of messages in [13] avoided multiplication operations and achieved better numerical stability. Mixed domain approach [14] was also proposed to benefit from both FFT and logarithmic computation. Although most of the work done in the past can reduce the decoding complexity significantly, the complexity is still formidable as m increases. To deal with this problem, Declercq et al. [12] proposed the extended Min-Sum (EMS) where only the most significant \(n_m\) entries in a message were used in the computation. This keeps the complexity at a much lower level because \(n_m\) is usually much smaller than \(2^m\). A decoding technique developed in [15] conducted the EMS algorithm with a reduced complexity of \(n_m \log_2 n_m\) with minor performance degradation.

In this paper, we propose a new technique that exploits adaptive message control (AMC) with the Min-Sum (MS) updating to reduce the decoding complexity. Different from EMS which maintains a constant message length for every symbol, the proposed AMC-MS adjusts the message length adaptively by exploiting two features. First, the distribution of belief information (see Section II-A for the definition of belief information) in a message can be different across different symbols due to the random nature of channel noise. Therefore, different symbols can use messages with different lengths while maintaining the same amount of belief information. Second, we observe that during the decoding iteration, each message gradually converges towards the correct result, which indicates that the distribution of belief information becomes more concentrated. Thus, a shorter message is sufficient to represent most of the information. In the proposed approach, each message has a different length and becomes shorter in the decoding iteration rather than being kept constant as that in the EMS. By dynamically adjusting the message length for each symbol, the proposed approach effectively reduces the number of arithmetic operations, thereby enabling a low-complexity nonbinary LDPC decoding with negligible or minor performance loss.

To facilitate the AMC, we propose a message truncation criterion, i.e., to maintain the least number of entries in each message with a total probability larger than a threshold. A low-complexity approximation method is developed to conduct message truncation while minimizing the performance impact. Simulation results demonstrate that the proposed AMC-MS can reduce arithmetic operations by up to 50% as compared with the EMS.

II. REVIEW OF NONBINARY LDPC CODES

A nonbinary LDPC code is defined by its parity check matrix (PCM) \(H = [h_{ij}]\), which is an \(M \times N\) sparse matrix with low density of nonzero entries. The nonzero entries of \(H\) take values from a Galois field \(GF(2^m)\). A length-N vector \(x\) with entries having values from \(GF(2^m)\) is a codeword if and only if \(Hx = 0\). Each entry in the codeword is called a symbol. An LDPC code with PCM \(H\) can be represented by a bipartite graph called Tanner graph [8], which consists of two categories of nodes; that is, \(N\) variable nodes \(v_i\), \(1 \leq i \leq N\) and \(M\) check nodes \(c_j\), \(1 \leq j \leq M\). A variable node \(v_i\) is connected with a check node \(c_j\) if and only if \(h_{ij}\) in the PCM \(H\) is nonzero.

The decoding process of nonbinary LDPC codes operates on the messages that represent the probability distribution of symbols. A message is a length-\(2^m\) vector recording the \(2^m\) belief information of a symbol subject to channel noise, where each belief information indicates the probability of this noisy symbol to be one of the \(2^m\) elements in \(GF(2^m)\). For example, the belief information \(q^m\) in the message \(q\) is the probability that the true value of the noisy symbol is \(\alpha \in GF(2^m)\). The decoding process is essentially iterative message exchange and update between the check nodes and variable nodes in the Tanner graph representation. The messages on the Tanner graph exchange and update in two directions. Variable node messages (VNMs), denoted by \(q\), pass from the variable nodes to the check nodes; and check node messages (CNMs), denoted by \(r\), pass from the check nodes to the variable nodes. VNMs are initialized with channel messages, which are the input to the decoder. Then, VNMs are sent to the check nodes to update the CNMs. The new CNMs are sent back to the variable nodes and the process is repeated until convergence.
are then sent back to the variable nodes and used together with the channel messages to update the VNMs. This procedure is known as the belief propagation (BP) [7], [10].

There are mainly two types of decoding algorithms – sum-product algorithm (SPA) and min-sum (MS), of which the latter one is a mathematical approximation of SPA where the sum of product is replaced by the maximum product term. Due to its relatively simple operation, MS is often chosen for practical applications. Further reduction in the complexity of MS is desirable; one such approach is the extended MS (EMS) algorithm [12], [15].

III. MIN-SUM WITH ADAPTIVE MESSAGE CONTROL

In this section, we develop an adaptive message control method that dynamically adjusts the message length of a symbol during the decoding iteration. A truncation scheme is proposed for the proposed AMC-MS algorithm.

A. Adaptive Message Control (AMC)

Intuitively, when the distribution of belief information is more concentrated, a message with a smaller number of entries might be sufficient to retain the same amount of the belief information. Exploiting this fact, we propose to adaptively control the message length during the decoding iteration. Our approach leads to two major advantages. First, due to the random nature of channel noise, channel messages for different symbols may have different statistics. In comparison with the EMS which maintains a fixed number of entries for all channel messages, the proposed AMC can lower the average message length while retaining the same amount of belief information. Second, as the decoding proceeds, the belief information will gradually concentrate around the correct element in the case of convergence. Thus, fewer entries are needed in the message of a symbol, i.e., the message can be truncated even more. These features are essential for reducing the computational complexity of decoding nonbinary LDPC codes.

The basic idea of AMC is to keep as few entries in a message as possible without incurring much information loss. The proposed method works as follows. Assume that a length-2^n message \( \tilde{q} \) is sorted in the descending order of the magnitude of belief information, then the number \( n_m \), i.e., the truncated message length, is equal to the minimal value of \( n \) subject to

\[
\sum_{k=1}^{n} \tilde{q}(k) \geq \sum_{k=1}^{2^m} \tilde{q}(k), \tag{1}
\]

where \( \tilde{q}(k) \) is the k-th entry in the sorted message \( \tilde{q} \) and \( \zeta \in [0, 1] \) is the confidence factor. For example, \( \zeta = 0.999 \) indicates that after the message truncation, the correct element, while yet to be decoded, still remains in the message with a probability of 99.9% (as \( \tilde{q}(k) \) is a statistical measure by itself). In this way, \( n_m \) is chosen such that the very first \( n_m \) entries of \( \tilde{q} \) possess more than \( \zeta \) fraction of the total belief information. As the probability distribution of the message changes during the iteration, the message length \( n_m \) will vary dynamically for a given \( \zeta \).

Usually log domain representation is preferred for implementation due to its low complexity and numerical stability. Thus, the direct computation of (1) would require domain changes between the probability domain and log domain, which involves extra computation such as additions and multiplications. Here we propose a low-complexity approximation. Subtracting \( \zeta \sum_{k=1}^{n} \tilde{q}(k) \) from both sides of (1) and approximating each summation with the largest term, i.e.,

\[
\tilde{q}(1) \approx \sum_{k=1}^{n} \tilde{q}(k) \text{ and } \tilde{q}(n+1) \approx \sum_{k=n+1}^{2^m} \tilde{q}(k),
\]

(1) can be simplified as

\[
(1 - \zeta)\tilde{q}(1) \geq \zeta\tilde{q}(n+1). \tag{2}
\]

Simulation results in Section IV-A show that the result obtained from (2) is very close to the exact value of \( n_m \). Thus, most of the belief information is retained while the complexity of truncation is greatly reduced.

In the log domain, multiplications become additions; thus the truncation problem of (1) can be solved by finding the minimal \( n \) that satisfies

\[
q(n+1) \leq q(1) + \ln \left( \frac{1 - \zeta}{\zeta} \right), \tag{3}
\]

where \( q \) is the log domain representation of the message \( \tilde{q} \).

In general, a small \( \zeta \) is desirable to reduce the average message length. However, when \( \zeta \) is too small, truncation will incur extra performance loss. Also, the decoding complexity may not decrease monotonically as a function of the confidence factor \( \zeta \). This is because over-truncation may cause a large information loss, and the number of decoding iterations will increase, thereby offsetting the effect of message length reduction. Section IV discusses the tradeoffs between decoding complexity and performance with more details.

B. AMC-MS

The proposed AMC-MS algorithm employs AMC to truncate channel messages and variable node messages (VNMs). As check node messages (CNMs) are a function of VNMs, their lengths are also controlled in an indirect way. The operations of AMC-MS such as variable node and check node updates are quite different from the MS algorithm due to the adaptive message control. In this subsection, we present the complete decoding algorithm first and then explain some of the key operations.

The AMC-MS algorithm can be summarized as

**Initialization**
- **Channel message**: \( f_{j}^* = \ln(p(x_j = \alpha \mid \text{channel})) \), where \( p(x_j = \alpha \mid \text{channel}) \) is the probability that the received symbol \( x_j = \alpha \in GF(2^n) \) for a given channel condition.
- **Channel message truncation**: \( f_{j} = \text{AMC}(f_{j}) \)
- **Variable node message**: \( q_{ij} = f_{j} \), where \( q_{ij} \) is the VNM from the variable node \( v_{j} \) to the check node \( c_{i} \).

**Iterations**
- **Permutation**: \( \tilde{q}_{ij}^{*} \rightarrow \tilde{q}_{ij}^{*+h_{ij}} \), where \( h_{ij} \) is the nonzero PCM element, and the multiplication is conducted in \( GF(2^n) \) field.
- **Check node update**: \( r_{ij} = \bigoplus_{k \in M(i) \setminus j} q_{ik} \), \( \bigoplus \) is the elementary operation that will be explained later.
- **Inverse permutation** It’s an inverse operation of the permutation.
- **Variable node update**: \( q_{ij} = \text{AMC}(f_{j} + \sum_{k \in N(j) \setminus i}^{\text{AMC}} r_{kj}) \), \( \sum^{\text{AMC}} \) means that AMC is applied to all the intermediate results.

**Tentative decoding**: \( \hat{c}_{j} = \max_{\alpha} (f_{j}^* + \sum_{k \in N(j)} r_{kj}^*) \).

The details of the key operations are explained as follows.

1) **Truncation of channel messages**: Channel messages are truncated by applying AMC once at the initialization step. The truncation of channel messages affects the decoding performance as it determines the amount of external information input to the decoder.
2) Variable node update: The elementary operation of variable node update is the summation of two messages. The belief information of all the elements shared between the two messages (i.e., $\alpha \in \Gamma_1 \cap \Gamma_2$, where $\Gamma_1$ and $\Gamma_2$ are the sets of Galois field elements in messages $r_1$ and $r_2$, respectively) are generated by adding up the two entries corresponding to the same Galois field element (i.e., $q^n = r_1^n + r_2^n$). Compensation factors are used for elements not shared between the two messages. For example, if the entry corresponding to $\alpha \in GF(2^m)$ is in $r_1$ but is truncated in $r_2$, the belief information of $q$ concerning $\alpha$ is computed as

$$q^n = r_1^n + \xi,$$

where $\xi$ is the compensation factor. The optimal compensation factor depends on several factors such as code structure and Galois field order. In [12], this value is obtained through simulations. In this paper, we use the same compensation scheme for both EMS and AMC-MS as [15]

$$\xi = r_2(n_m + 1) - \ln(2^m - n_m),$$

where $n_m$ is the length of the truncated message $r_2$ and $r_2(n_m + 1)$ is the largest belief information among the truncated entries of $r_2$.

Note that each updated variable node message should be normalized with respect to its largest entry to maintain numerical stability. This is also required in EMS [15].

3) Check node update: For check node update, the elementary operation is $\oplus$ with truncated messages of different lengths, which is defined as

$$r^n = \max_{\alpha, \beta, \gamma \in GF(2^m)} (q_1^\alpha + q_2^\beta),$$

where $r^n$, $q_1^\alpha$, $q_2^\beta$ are the entries (i.e., belief information) of three messages $r_1$, $q_1$, $q_2$ to $\alpha, \beta, \gamma \in GF(2^m)$, respectively. As the concentration level of the result $r$ is mainly determined by the longer input, i.e., the less concentrated one, we propose to maintain the result with the same length as the longer input.

Note that we do not explicitly use AMC to truncate the check node messages because the elementary operation $\oplus$ is essentially a simplified convolution which makes the probability distribution more dispersive. Directly applying AMC to check node messages will lead to longer messages than the input messages, where the insignificant entries will be truncated again during the variable node update. To avoid redundant computation of these insignificant entries, we generate the same number of entries as the longer input to retain most of the useful entries.

IV. SIMULATION RESULTS AND DISCUSSIONS

We choose to evaluate an irregular GF(16) nonbinary LDPC code with mean column weight 2.3, codeword length 672 and rate 1/2. This code was used in a multicarrier underwater acoustic system to boost the communication performance [17]. In all simulations the codewords are transmitted through the binary AWGN channel. The maximum number of decoding iterations is set at 10.

Without loss of generality, we consider the zigzag belief propagation scheme [16], which is a horizontally sequential decoding scheme and exploits alternative forward-backward scheduling to reduce the number of elementary operations. The major arithmetic operations in the decoding algorithm include real number additions in check node and variable node update, and the Galois field multiplications and divisions (i.e., the permutation and inverse permutation). For simplicity, they are referred to as additions and Galois field operations, respectively. The number of these arithmetic operations are collected to evaluate the effectiveness of the proposed AMC-MS. Tentative decoding is conducted at each iteration step.

A. Decoding Performance

Fig. 1 shows the decoding performance measured by the block-error-rate (BLER) of the proposed AMC-MS (with different confidence factor $\zeta$) and the EMS (with different message length $n_m$). Note that EMS with $n_m = 16$ is equivalent to MS.

From Fig. 1, both AMC-MS and EMS incur only minor performance loss by appropriate message truncation. Specifically, we observe less than 0.1 dB loss at BLER of $10^{-3}$ for the AMC-MS with $\zeta = 0.9999$, and the performance loss becomes about 0.35 dB when $\zeta = 0.999$. For EMS, reducing $n_m$ from 16 to 12 and 10 results in the performance losses of 0.15 dB and 0.3 dB, respectively, at BLER of $10^{-3}$. We will compare the decoding complexity of these two algorithms under the configurations with a similar BLER performance at Section IV-C.

B. Message Size Reduction by AMC

Fig. 2 shows the reduction of message length of the proposed AMC-MS as the decoding iteration proceeds. The message length is obtained by averaging more than 1000 block transmissions with a fixed SNR of 2.3 dB. When the decoding terminates before 10 iterations, the message lengths of the rest iterations are counted as zero because no message exchange will happen. A closer look reveals that for $\zeta$ equal to 0.999 and 0.9999, the average message length decreases monotonically with the iteration; while this is not the case for $\zeta$ being 0.99. This difference is caused by the fact that, when the confidence factor is too small (e.g., $\zeta = 0.99$), over-truncation happens and decoding may not converge consistently.

There exists an optimal $\zeta$ in terms of the average message length. For this specific example, the average message lengths over all the iterations for the three confidence factors $\zeta = 0.99$, 0.999, 0.9999 are 4.68, 3.37 and 3.82, respectively. Thus, $\zeta = 0.999$ is the best in terms of the message length reduction.

C. Complexity Reduction of AMC-MS and EMS

The reductions in additions and Galois operations achieved by the AMC-MS and EMS are depicted in Fig. 3 and Fig. 4. These results are obtained by comparing with the MS algorithm. As shown, the proposed AMC-MS achieves much smaller decoding complexity than the EMS under the similar decoding performance. Specifically, from Fig. 3 we have the following observations:

- The AMC-MS achieves the reduction in additions from 40% to 55% with $\zeta = 0.9999$, whereas for EMS with a similar decoding
performance ($n_m = 12$), it ranges from 12% to 28%, only about $1/3$ to $1/2$ of the reduction achieved by the AMC-MS.

- A similar trend is shown in the AMC-MS with $\zeta = 0.999$, where the reduction in additions ranges from 50% to 65%. The EMS only achieves 20% to 35% with $n_m = 10$.

The reduction in Galois field operations in terms of the multiplications and divisions performed at the Galois field follows similar trend as the reduction in additions (see Fig. 4). This is reasonable as both of them are approximately linear to the length of messages.

Furthermore, we can see from Figs. 3 and 4 that the advantages of AMC-MS over EMS become more obvious as SNR increases. For example, AMC-MS with $\zeta = 0.999$ achieves reduction in additions from 38% to about 50% over EMS with $n_m = 10$ when SNR changes from 1.9dB to 3.0dB. This is because the complexity reduction of EMS comes mainly from the reduced number of iterations, whereas the complexity reduction of AMC-MS comes from a combination of fewer iterations and shorter message lengths as SNR increases.

V. CONCLUSIONS

In this paper, we propose a new nonbinary LDPC decoding algorithm – Min-Sum with Adaptive Message Control (AMC-MS), which can significantly reduce the decoding complexity by adaptively adjusting the message length of belief information while maintaining the required performance. A truncation scheme operating in the log domain is presented to facilitate the AMC-MS. Simulation results demonstrate that the proposed AMC-MS can reduce the number of arithmetic operations greatly with negligible or minor performance loss. Future work is directed towards adaptive confidence factor selection.

REFERENCES