Journal of Computer Science and Cybernetics, V.37, N.2 (2021), 91–106
DOI 10.15625/1813-9663/37/2/15917

ONE-MINIUM-ONLY BASIC-SET TRELLIS MIN-MAX DECODER ARCHITECTURE FOR NONBINARY LDPC CODE

THE CUONG DINH[1,*], HUYEN PHAM THI[1], HUNG DAO TUAN[1], NGHIA PHAM XUAN[2]

[1]*National Laboratory for Securing Information, Ha Noi, Viet Nam*

[2]*Le Quy Don Technical University, 236 Hoang Quoc Viet Street, Bac Tu Liem District, Ha Noi, Viet Nam*

Crossref Similarity Check
Powered by iThenticate

Abstract. Nonbinary low-density-parity-check (NB-LDPC) code outperforms their binary counterpart in terms of error-correcting performance and error-floor property when the code length is moderate. However, the drawback of NB-LDPC decoders is high complexity and the complexity increases considerably when increasing the Galois-field order. In this paper, an One-Minimum-Only basic-set trellis min-max (OMO-BS-TMM) algorithm and the corresponding decoder architecture are proposed for NBLDPC codes to greatly reduce the complexity of the check node unit (CNU) as well as the whole decoder. In the proposed OMO-BS-TMM algorithm, only the first minimum values are used for generating the check node messages instead of using both the first and second minimum values, and the number of messages exchanged between the check node and the variable node is reduced in comparison with the previous works. Layered decoder architectures based on the proposed algorithm were implemented for the (837, 726) NB-LDPC code over GF(32) using 90-nm CMOS technology. The implementation results showed that the OMO-BS-TMM algorithm achieves the almost similar error-correcting performance, and a reduction of the complexity by 31.8% and 20.5% for the whole decoder, compared to previous works. Moreover, the proposed decoder achieves a higher throughput at 1.4 Gbps, compared with the other state-of-the-art NBLDPC decoders.

Keywords. NB-LDPC, basic-set, trellis min-max, VLSI design.

1. INTRODUCTION

Nonbinary low-density parity-check (NB-LDPC) codes, which are defined over Galois Fields GF(q) ($q > 2$), outperform their binary counterpart in terms of error-correcting performance, burst error correction capability, and performance improvement in the error-floor region when the code length is moderate [1]. Nonetheless, the NB-LDPC decoding algorithms require complex computations, and their architectures have very high complexity and large memory requirements.

The works [2, 3] show that NB-LDPC codes provide superior performance compared with the best optimized binary LDPC code over fading channels, data transmission channel,

*Corresponding author.
E-mail addresses: cuongdt@mta.edu.vn (T. C. Dinh), phamhuyenmta87@gmail.com (H. P. Thi), daotuanhung@gmail.com (H. D. Tuan); nghiapx@mta.edu.vn (N. P. Xuan)

and the combination of NB-LDPC code with high-order modulations improves not only the bandwidth efficiency but also the error-correction capability. Furthermore, the NB-LDPC codes demonstrate much promise for multilevel flash memory applications [4] because of the elimination of the error floor. However, the main disadvantage of NB-LDPC codes is their highly complex decoding algorithms and NB-LDPC decoder architecture.

For practical NB-LDPC decoder implementations, suboptimal algorithms such as extended min-sum (EMS) [5] and the min-max [6] algorithm have been proposed to reduce the complexity of the CNU as the main bottleneck of the NB-LDPC decoder. Recently, the relaxed trellis min-max (R-TMM) algorithm [7] has been proposed to improve both the throughput and the complexity. The R-TMM algorithm introduced the trellis representation and the minimum basis for check node processing to remove computing the forward-backward messages in [6]. However, the check node processing is sequentially processed which requires a large number of clock cycles. In [8], a simplified trellis min-max (STMM) algorithm was proposed to improve the throughput of the min-max decoders with less complexity by means of an extra column inserted to the original trellis.

In [9], the one-minimum-only TMM (OMO-TMM) algorithm was introduced on the basis of the STMM algorithm to reduce the CNU complexity by obtaining only one minimum and estimating the second one. In these works [8, 9], $q \times d_c$ check node output messages are exchanged between the check node and the variable nodes. For high-order GFs or high-rate NB-LDPC codes, there are two main drawbacks in the previous works [8, 9]. First, the amount of exchanged messages increases, which causes wiring congestion, and thus limits the maximum throughput of the decoders. Second, the check node output messages are stored in the memory for the next decoding iteration in the layered decoders. Therefore, the memory requirement becomes large, which leads to a significant growth in the decoder area for NB-LDPC codes.

To overcome the above drawbacks, the work in [10] proposed to simplify the CNU architecture and reduce the exchanged messages with the almost similar error-correcting performance. In [11], the approximated TMM algorithms are introduced to further decrease the number of intrinsic information at the cost of some error-correcting performance loss. In [12], a basic-set trellis min-max (BS-TMM) algorithm, which is especially efficient for high-order Galois Fields, has been introduced to reduce the exchanged messages to a factor of $\log_2 q$ with a negligible performance loss.

In this paper, an one-minimum-only basic-set TMM (OMO-BS-TMM) algorithm is proposed for the check node processing to further reduce the check node unit complexity as well as the whole decoder with the almost similar error-correcting performance, compared to the existing decoding algorithms. The OMO-BS-TMM algorithm is implemented by using (837, 726) NB-LDPC code over GF(32) to demonstrate the efficiency of the proposal.

2. REVIEW OF NB-LDPC DECODING ALGORITHM

2.1. NB-LDPC codes

NB-LDPC codes, which are a kind of linear block code, are defined by a sparse parity-check matrix $\mathbf{H}$ having M rows and M columns. Let h_{mn} be a nonzero element of the matrix

H that belongs to the GF($q = 2^p$). Let d_v and d_c be the variable node degree and the check node degree of matrix **H**, respectively. A regular NB-LDPC code is considered in this paper with the fixed values of d_c and d_v.

2.2. NB-LDPC decoding algorithm

Algorithm 1 presents the layered basic-set trellis min-max (BS-TMM) decoding algorithm for the NB-LDPC codes [12]. Symbols c_n and z_n define the n-th reference symbol of a received codeword and the n-th hard-decision symbol with the highest reliability, respectively. Starting the decoding process is implemented by obtaining the log-likelihood ratio (LLR) vectors $L_n(a)$ with a size of q that are the channel information. At the first layer of the first iteration, the a posteriori information as $Q_n(a)$ corresponding the variable node n is equal to $L_n(a)$. The check node to variable node (C2V) messages $R_{mn}(a)$ are equal to zero. The numbers k and l define the loop index for k-th iteration and the layer index for l-th layer, respectively. In addition, the decompression network (DN) in step 3 and step 8 is implemented in the variable node processor to generate the C2V messages $R_{mn}(a)$ from outputs of the CNU architecture. It is noted that two DNs are required in the variable node processor. However, the proposed decoder area is much lower than that of the conventional decoders [8,9]. Then, the variable node to check node (V2C) messages $\tilde{Q}_{mn}(a)$ are calculated from the $Q_n(a)$ messages permuted using the nonzero element h_{mn} of matrix **H**, as shown in step 4. The normalization of V2C messages are performed in steps 5 and 6. Step 7 presents the computation of the basic-set messages and the information used to update the C2V messages using the BS-TMM function applied for the check node processing. Step 9 calculates the updated messages $Q_n(a)$, which is undergone the reverse permutation before processing a new layer. The decoding process is repeatedly implemented until the maximum number of iteration I_{max} is reached. Finally, the output codeword $\tilde{c}_n(a)$ is the most reliable symbol corresponding to $Q_n(a)$ message.

Algorithm 1: Layered min-max decoding algorithm [12]

Input: $L_n(a)= \ln(\Pr(c_n = z_n|channel)/\Pr(c_n = a|channel))$

$Q_n^{1,0}(a) = L_n(a)$; $R_{mn}^0(a) = 0; k = 1$

1: **While** $k \leq I_{max}$ **do**

2: **for** $l = 1$ to M **do**

3: $R_{mn}^{k-1,l}(a) =$ DN $\{z_n^*, E(a), B^*\}$

4: $\tilde{Q}_{mn}^{k,l}(a) = Q_n^{k,l-1}(h_{mn}a) - R_{mn}^{k-1,l}(a)$

5: $\tilde{Q}_{mn}^{k,l} = \min_{a \in GF(q)}(\tilde{Q}_{mn}^{k,l}(a))$; $z_n = \arg\min(\tilde{Q}_{mn}^{k,l}(a))$

6: $Q_{mn}^{k,l}(a) = \tilde{Q}_{mn}^{k,l}(a) - \tilde{Q}_{mn}^{k,l}$

7: $\{z_n^*, E(a), B^*\} =$ BS-TMM $\{Q_{mn}^{k,l}(a), z_n\}_{n \in N(m)}$

8: $R_{mn}^{k,l}(a) =$ DN $\{z_n^*, E(a), B^*\}$

9: $Q_n^{k,l}(h_{mn}^{-1}a) = Q_{mn}^{k,l}(a) + R_{mn}^{k,l}(a)$

10: **end for**

11: **end while Output:** $\tilde{c}_n = \arg\min(Q_n^{k,l}(a))$

2.3. Basic-set trellis min-max algorithm

Algorithm 2: Basic-set TMM algorithm [12]

Input: $\mathbf{Q_{mn}}, z_n = \arg\min_{a \in GF(q)} Q_{mn}(a); \forall n \in N(m)$

1: $\Delta Q_{mj}(\eta = a \oplus z_j) = Q_{mj}(a);\ (0 \leq j < d_c)$

2: $\beta = \sum_{j=0}^{d_c-1} z_j \in GF(q)$

3: $\{m1(a), I_{col}(a), m2(a)\} = \Psi\{\Delta Q_{mk}(a)|_{k=0}^{d_c-1}\}$

4: $B^* = \{m1_l^*, I_l^*, a_l^*\}_{1 \leq l \leq p} = \Phi\{m1(a), I_{col}(a)\}_{1 \leq a < q}$

5: $E(a) = \begin{cases} m2(a) & \text{if} \quad a = a_l^* (1 \leq l \leq p) \\ m1(a) & \text{otherwise} \end{cases}$

Output: $\begin{cases} B^* \\ E(a) \\ z_n^* = z_n \oplus \beta \end{cases}$

In this section, the BS-TMM algorithm [12] is illustrated as Algorithm 2. Without loss of generality, the Galois-field GF(q) with $q = 2^p$ including q elements such as $\{0, \alpha^0, \alpha^1, ..., \alpha^{q-2}\}$ is considered in our work. For each Galois-field GF(2^p) any field element is uniquely represented by the linear addition of p independent field elements. To take advantage of this, a set of only $p = \log_2 q$ independent field elements with the smallest LLRs, called the basic set B*, are generated in the check node processing. Then, construction of the $\Delta Q(a)$ is implemented in the variable node processing based on the basic set B^*.

The first step transforms the input messages from the normal domain $Q_{mn}(a)$ to the delta domain $\Delta Q_{mn}(a)$ to ensure that the most reliable symbols are always in the first index corresponding to the GF symbol 0, and the rest of the indexes are in order of $\{\alpha^0, \alpha^1, ..., \alpha^{q-2}\}$. Step 2 relates to the computation of the syndrome β using the most reliable symbols z_n from V2C messages. In step 3, the first minimum value $m1(a)$, its column index $I_{col}(a)$, and the second minimum value $m2(a)$ for each trellis row are calculated using the function Ψ. Step 4 computes the basic set $B^* = \{m1_l^*, I_l^*, a_l^*\}_{1 \leq l \leq p}$ including $3 \times p$ values (p LLR values, p column indexes, and p field elements), based on the minimum values $m1(a)$ and their column indexes $I_{col}(a)$ $(1 \leq a < q)$. Finding the basic set B^* is given by the Φ function in Algorithm 2. Step 5 calculates the complement values in set $E(a)$. The complement values for p field elements, which belong to the basic-set B^*, are assigned to the second minimum values $m2(a)$. For the remaining field elements, the complement values are assigned to the minimum values $m1(a)$. Finally, the output of the check node processing includes three sets B^*, $E(a)$, and z_n^* with a size of $3 \times p + (q-1) + d_c$ values, which are used for generating the C2V messages in the variable node processing.

3. ONE-MINIMUM-ONLY BASIC-SET TRELLIS MIN-MAX DECODING ALGORITHM

3.1. One-minimum-only basic-set trellis min-max (OMO-BS-TMM) decoding algorithm

In this section, the OMO-BS-TMM algorithm is proposed to significantly reduce the complexity and the memory requirement for check node processing as well as the exchanged messages between check nodes and variable nodes with a negligible error-correcting performance loss. To take advantage of the work in [9] which used only the first minimum values for generating the check node messages, the OMO-BS-TMM algorithm is proposed to generate the check node messages based on only the first minimum values in our work.

Algorithm 3: One-minimum-only basic-set TMM algorithm

Input: $\mathbf{Q_{mn}}, z_n = \arg\min_{a \in GF(q)} Q_{mn}(a); \forall n \in N(m)$

1: $\Delta Q_{mj}(\eta = a \oplus z_j) = Q_{mj}(a);\ (0 \leq j < d_c)$

2: $\beta = \sum_{j=0}^{d_c-1} z_j \in GF(q)$

3: $\{m1(a), I_{col}(a)\} = \Psi\{\Delta Q_{mk}(a)|_{k=0}^{d_c-1}\}$

4: $B^* = \{m1_l^*, I_l^*, a_l^*\}_{1 \leq l \leq p} = \Phi\{m1(a), I_{col}(a)\}_{1 \leq a < q}$

5: $E(a) = m1(a)$; if $a \notin a_l^* (1 \leq l \leq p)$

Output: $\begin{cases} B^* \\ E(a) \\ z_n^* = z_n \oplus \beta \end{cases}$

The proposed OMO-BS-TMM algorithm is depicted in Algorithm 3. Steps 1 to 2 are similar to steps 1 to 2 in Algorithm 2. Step 3 relates to calculate only the minimum values $m1(a)$ and their column indexes $I_{col}(a)$ $(1 \leq a \leq p)$, which reduces significantly complexity by removing the second minimum values. Step 4 computes the basic set $B^* = \{m1_l^*, I_l^*, a_l^*\}_{1 \leq l \leq p}$ including $3 \times p$ values (p LLR values, p column indexes, and p field elements), based on the minimum values $m1(a)$ and their column indexes $I_{col}(a)$ $(1 \leq a < q)$, as Algorithm 2. In step 5, the only $(q-1)-p$ elements, which are not belong to the basic-set, have the complement values assigned to the minimum values $m1(a)$. For the p elements in the basic set, the complement values are proposed to calculate approximately based on the values of the basic set in the variable node processing. Finally, the output of the check node processing includes three sets B^*, $E(a)$, and z_n^* with a size of $3 \times p + (q-1) - p + d_c$ values, which are used for generating the C2V messages in the variable node processing. From our observation, in [12], the complement values for p elements in the basic set are assigned to the second minimum values $m2(a)$ which is higher than the first minimum values $m1(a)$. Furthermore, p values in the basic-set B^* are in increasing order, and the last value $m1_p^*$ is the highest value. As the result, an approximation is proposed to generate the complement values for all p elements in the basic set, which does not affect to the efficient of the algorithm. In this work, the complement value, which is proposed to replace for p second minimum values, have to be higher than the last value $m1_p^*$. Therefore, a proximate value, which is a scale of the last

Table 1: Comparison of exchanged messages between check node and variable node with $d_c = 27$ and $w = 6$.

Algorithm	Number of exchanged bits			
	GF($q = 2^p$)	GF(2^5)	GF(2^6)	GF(2^7)
[8]	$q \times d_c \times w$	5184	10368	24192
[13]	$3 \times (q-1) \times w + 2 \times (q-1) \times \lceil \log(d_c) \rceil + d_c \times p$	1003	1926	3745
[10], [14]	$2 \times (q-1) \times (w + \lceil \log(d_c) \rceil) + d_c \times p$	817	1548	2983
[15]	$2 \times (q-1) \times \lceil \log(d_c) \rceil + (q+1) \times w + (d_c+2) \times p$	653	1194	2247
[12]	$p \times (w + \lceil \log(d_c) \rceil + p) + (q-1) \times w + d_c \times p$	401	642	1077
Proposed	$p \times (w + \lceil \log(d_c) \rceil + p) + ((q-1)-p) \times w + d_c \times p$	371	612	1008

value $m1_p^*$, is selected to satisfy the condition

$$E(a) = \beta \times m1_p^*. \tag{1}$$

This approximation is proposed to reduce the number of output messages in the check node processing.

Table 1 shows the number of bits exchanged between check node and variable node in the proposed algorithm and previous works for the general GF($q = 2^p$) and w quantization bits for the LLR values and high-order GFs such as GF(32), GF(64) and GF(128) with $d_c = 27$ and $w = 6$ quantization bits. It is clear that the proposed algorithm greatly reduces the exchanged bits, compared to previous works. In [8], all C2V messages generated in the check node processing are exchanged, which causes an extremely high number of check node output bits. It can be seen that the exchanged bits are reduced by factors of almost 13, 16, and 22.46 for GF(32), GF(64), and GF(128), respectively. In [10,13–15], a small number of fixed sets, in which the size of each set is proportional to either q or d_c, is exchanged. Compared to the original compression technique [13], the proposed work reduces the number of exchanged bits by factors of almost 2.7 and 3.72 for GF(32) and GF(128), respectively. In comparison with [15], the reduction of the exchanged bits is 43.18% and 55.14% for GF(32) and GF(128), respectively. Moreover, the OMO-BS-TMM algorithm decreases a considerable number of exchange messages, compared to the latest work [12].

In the variable node processing, the extra column $\Delta Q(a)$ is recovered and the C2V messages $R_{mn}(a)$ are generated on the basis of the output sets of the check node processing, including B^*, $E(a)$ and z_n^*, as shown in Algorithm 4. First, the extra column $\Delta Q(a)$ and the path information $d(a)$ are calculated in steps 1 to 7. For p field elements, which belong to the basic set B^*, the $\Delta Q(a)$ value is the most reliable LLR $m1_l^*$, and the path information $d(a)$ has one deviation at the column index I_l^* with $1 \le l \le p$. It is remarked that $\Delta Q(a)$ values of $(q-1)-p$ remaining field elements are calculated approximately as the LLR value of the last field element $m1_p^*$, as shown in Step 5 of Algorithm 4 [16]. Updating the C2V messages is implemented in steps 8 to 14. For each row, if the column index j does not belong to the part information $d(a)$, the C2V message $\Delta R_{mj}(a)$ is assigned to the extra column $\Delta Q(a)$. Otherwise, the C2V message $\Delta R_{mj}(a)$ is assigned to the complement set $E(a)$. Finally, the

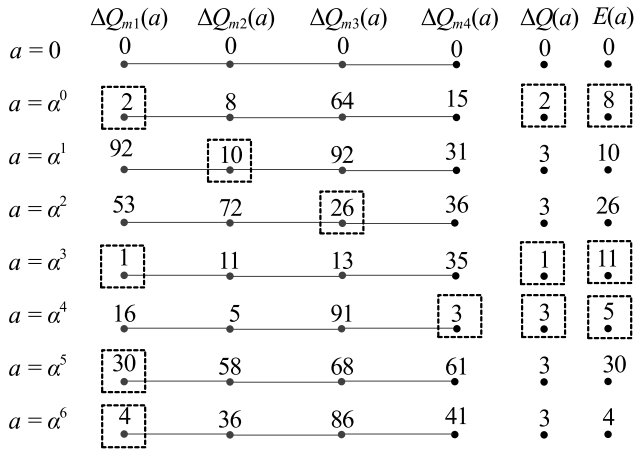

Figure 1: Example of the trellis based on GF(8) with $d_c = 4$

C2V messages in the delta domain are converted to the normal domain in step 15.

Algorithm 4: Construct extra column $\Delta Q(a)$ and $R_{mn}(a)$
Input: $B^* = \{m1_l^*, I_l^*, a_l^*\}_{1\le l\le p}$; $E(a)$; $z_n^*; \forall n \in N(m)$
1: **for** $a = 1$ to $q-1$ **do**
2: **if** $a = a_l^* (1 \le l \le p)$ **then**
3: $\Delta Q(a) = m1_l^*$; $d(a) = \{I_l^*\}$
4: **elseif** $a = a_1^* \oplus a_2^* \oplus ... \oplus a_s^* (2 \le s \le p)$ **then**
5: $\Delta Q(a) = m1_p^*$; $d(a) = \{I_1^* \cup I_2^* \cup ... \cup I_s^*\}$
6: **end if**
7: **end for**
8: **for** $j = 0$ to $d_c - 1$ **do**
9: **if** $(j \notin d(a))$ **then**
10: $\Delta R_{mj}(a) = \Delta Q(a)$
11: **else**
12: $\Delta R_{mj}(a) = E(a)$
13: **end if**
14: **end for**
15: $R_{mj}(a \oplus z_j^*) = \lambda \Delta R_{mj}(a), a \in GF(q)$
Output: $\mathbf{R_{mn}}$

In Figure 1, an example of the delta trellis for GF(8) with $d_c = 4$ is presented, where the minimum values in each row are marked with a dashed square. The extra column $\Delta Q(a)$ is constructed on the basis of basic set B^* , as shown in steps 1 to 7 of the Algorithm 4. This example demonstrates the method of building the extra column $\Delta Q(a)$ and $R_{mn}(a)$. Firstly, the basic set $B^* = \{(1, 1, \alpha^3), (2, 1, \alpha^0), (3, 4, \alpha^4)\}$ including $p = 3$ independent field elements with the most reliable messages is calculated as shown in [12]. Then, the extra column $\Delta Q(a)$ is constructed using the simplified extra column construction in [16]. For p field elements in the extra column, which belong to the basic set B^* such as $\{\alpha^3, \alpha^0, \alpha^4\}$, their LLR values $\Delta Q(a)$ and the path information $d(a)$ are the same as the LLR values and column indexes in the basic set B^*. For other field elements, all combinations of the field

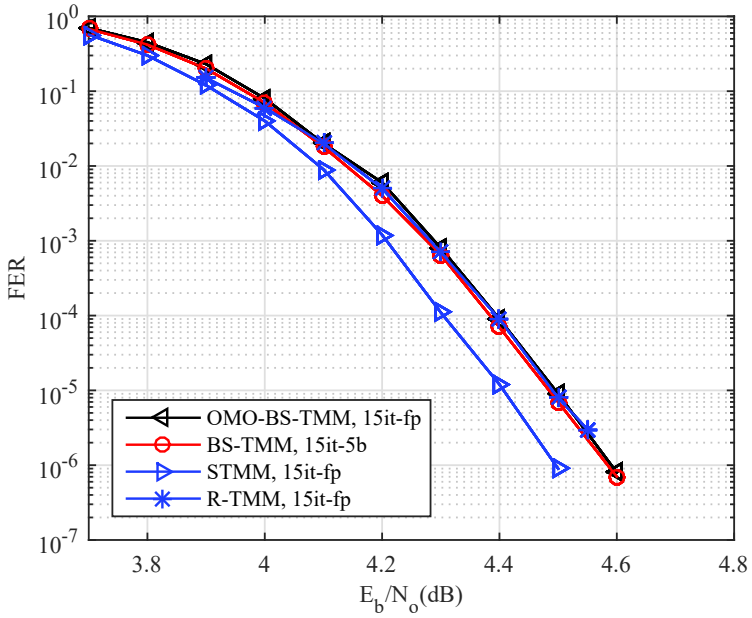

Figure 2: FER performance of the (837, 726) NB-LDPC code over GF(32) under the AWGN channel at 15 iterations.

elements in B^* are considered as follows:

$$\Delta Q(\alpha^3+\alpha^0=\alpha^1)=m1(\alpha^4)=3 \text{ and } d(\alpha^3+\alpha^0=\alpha^1)=\{1,1\};$$
$$\Delta Q(\alpha^0+\alpha^4=\alpha^5)=m1(\alpha^4)=3 \text{ and } d(\alpha^0+\alpha^4=\alpha^5)=\{1,4\};$$
$$\Delta Q(\alpha^3+\alpha^4=\alpha^6)=m1(\alpha^4)=3 \text{ and } d(\alpha^3+\alpha^4=\alpha^6)=\{1,4\};$$
$$\Delta Q(\alpha^3+\alpha^0+\alpha^4=\alpha^2)=m1(\alpha^4)=3 \text{ and } d(\alpha^3+\alpha^0+\alpha^4=\alpha^2)=\{1,1,4\}.$$

The complement values $E(a)$ for field elements excepted $p=3$ independent field elements in the basic set B^* are assigned the minimum values as $E(\alpha^1)=10, E(\alpha^2)=26, E(\alpha^5)=30$ and $E(\alpha^6)=4$. In [12], the complement values $E(a)$ for $p=3$ independent field elements in the basic set B^* are assigned the second minimum values as shown in the rightmost column of Figure 1. It is clear that the second minimum value of the last field element α^4 in the basic set B^* equals to 5 always greater than the minimum values in the basic set B^*. Therefore, an approximate value, which is proposed for the complement values of the $p=3$ independent field elements in the basic set B^*, is a scale of the last value as $\beta \times m1_p^*$ or $\beta \times m1(\alpha^4)$.

3.2. Performance analysis

To demonstrate the error-correcting performance of the proposed OMO-BS-TMM decoding algorithm, we performed the simulations for GF(32). Figure 2 illustrates the frame error rate (FER) performance for (837, 726) NB-LDPC code over GF(32) with $d_v=4$ and $d_c=27$ under the additive white Gaussian noise (AWGN) channel and binary phase shift keying (BPSK) modulation. As shown in Figure 2, the floating-point simulation result of the OMO-BS-TMM algorithm with 15 iterations shows a minor performance loss of 0.1 dB, compared to the STMM algorithm [8]. Furthermore, the proposed OMO-BS-TMM algorithm

provides low computation complexity, a large area reduction, and a significant improvement in throughput. This is explained by the fact that $(q-1)$ messages in the extra column $\Delta Q(a)$ in [8, 10] are constructed directly from all reliable messages of the configuration set $conf(1,2)$ using $(q-1)$ processors, whereas these are constructed on the basis of only p reliable messages in the basic set in our work.

Compared to the R-TMM algorithm [7], in which C2V messages are generated on the basis of minimum basic sets, the FER performance of the OMO-BS-TMM algorithm is almost the same as that of the R-TMM algorithm. It is noted that d_c minimum basic sets are required in the R-TMM algorithm [7] to generate the C2V messages, whereas the proposed OMO-BS-TMM algorithm requires only one basic set to construct the extra column. Moreover, the sequential design implemented in [7] causes a throughput problem, whereas the proposed OMO-BS-TMM algorithm based design performs all calculations in one clock cycle.

Compared to [12], the OMO-BS-TMM obtains the almost similar error-correcting performance and a significant reduction in the computation and hardware complexity because of removing the second minimum calculating process and decreasing the number of storage bits in the check node processing. This result demonstrates that the proposed OMO-BS-TMM algorithm provided a good FER performance, a significantly reduced computation complexity and hardware complexity for the high-order GF.

4. OMO-BS-TMM DECODER ARCHITECTURE

In this section, the proposed quasi-cyclic NB-LDPC decoder architectures and design technologies for the BS-TMM algorithm are described. The quasi-cyclic NB-LDPC codes over GF(q) are constructed by the algebraic construction method based on array-dispersions of matrices in [17], where a $(q-1)\times(q-1)$ submatrix is generated first. Then, a submatrix with size (d_v, d_c) is selected from the $(q-1)\times(q-1)$ submatrix. Each field element from the (d_v, d_c) submatrix is dispersed in either a zero matrix or a circulant permutation matrix (CPM) of size $(q-1)\times(q-1)$. As a result, the $\mathbf{H}$ matrix generated from the (d_v, d_c) submatrix has $M=(q-1)\times d_v$ rows and $N=(q-1)\times d_c$ columns.

4.1. CNU architecture

The top-level CNU architecture for the OMO-BS-TMM algorithm is shown in Figure 3, where each module corresponds to a step in Algorithm 3. The transformation module converts V2C messages from normal to delta domain using the control signals z_j. This module is constructed by means of dc reordering networks, as shown in [18], where each reordering network requires $q\times\log_2 q$ w-bit multiplexers. The check node syndrome β is generated by a tree adder structure. The delta-to-normal domain transformation is derived later using d_c reordering networks with the control signals $z_j^* = z_j + \beta$. The Ψ function is responsible for finding the first minimum values and the first minimum value's index from d_c inputs using the One-min finder. The One-min finder is adopted by applying the technique in [19], which provides a good tradeoff between the area and latency. Because $(q-1)$ rows in the delta trellis except the first row must perform the Ψ function, a total of $(q-1)$ One-min

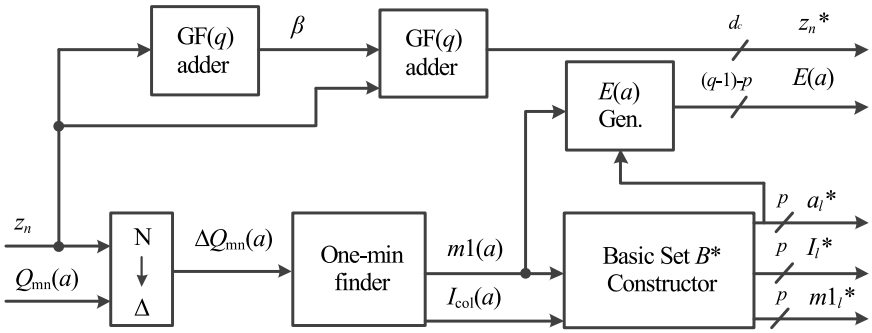

Figure 3: Proposed top-CNU architecture for OMO-BS-TMM algorithm

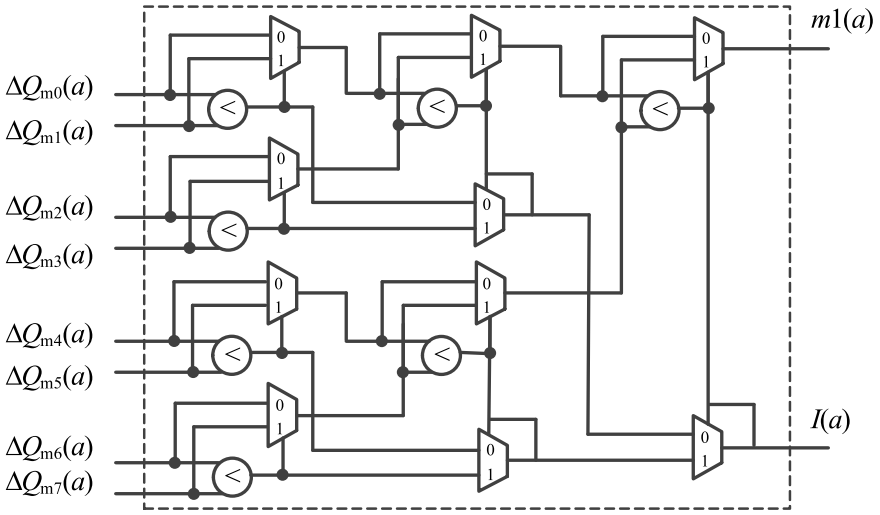

Figure 4 One-min Finder architecture with eight inputs

finders is required. Figure 4 shows an example of the One-min finder architecture with eight inputs. Compared to Two-min Finder architecture in Figure 5, the complexity of the One-min Finder architecture is greatly reduced. It can be seen that the number of comparisons are 7 and 13, and the number of multiplexors are 10 and 23 in One-min Finder and Two-min Finder architecture, respectively. As the results, the proposed decoding algorithm using only the minimum values can greatly reduce the complexity of the CNU architecture when increasing the order of the GF(q).

4.2. Decoder architecture

The proposed OMO-BS-TMM algorithm demonstrates a performance improvement in the computation and hardware complexity with the almost similar error-correcting performance, compared to recent works. In this part, the hardware design and implementation will be performed to clarify the efficient of the proposed OMO-BS-TMM algorithm. Figure 6 shows the top-level decoder architecture for the proposed layered decoding algorithm, where one row of H corresponding to one layer is processed in one clock cycle. It can be seen that the decoder architecture is divided into a variable node processor and check node processor. To start the

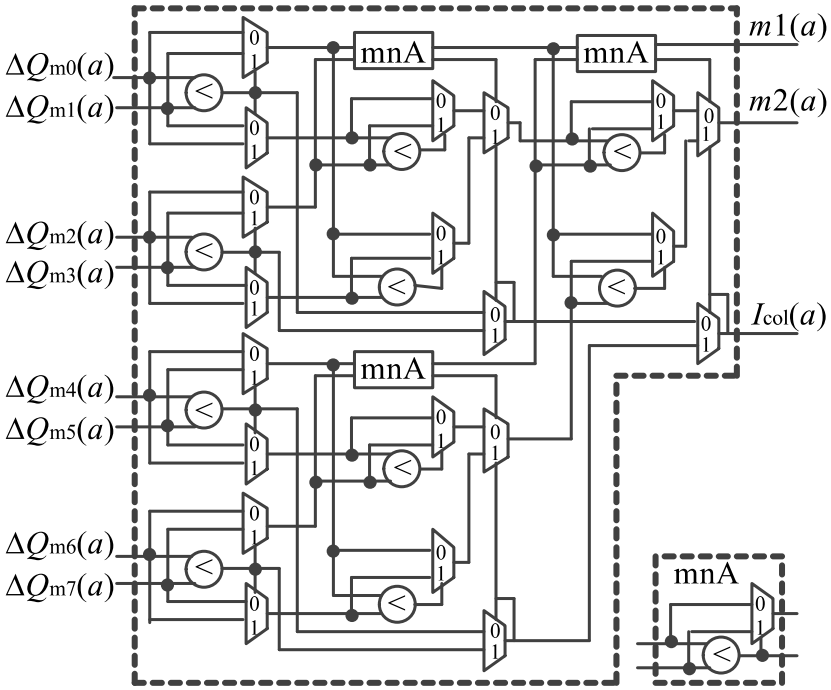

Figure 5: Two-min Finder architecture with eight inputs

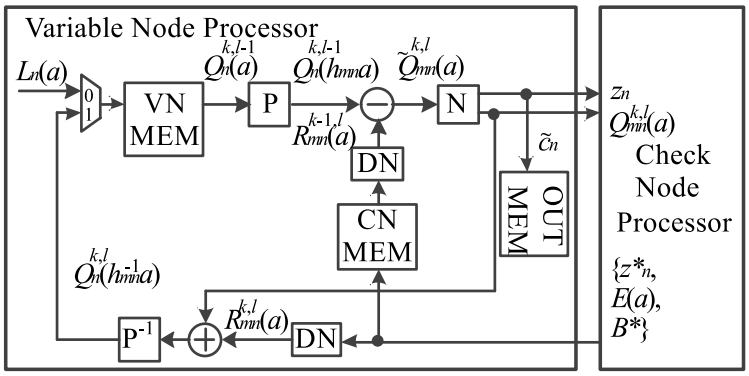

Figure 6: Top-level decoder architecture based on the OMO-BS-TMM algorithm.

decoding process, the LLR messages from channel information $L_n(a)$ are loaded in variable node memory (VNMEM). From the next layer and next iteration, the output messages of the variable node processor $Q_n^{k,l}(a)$ are stored in the VNMEM. The VNMEM includes dc memories with a depth of $(q-1)$ as the size of the circulant permutation matrix [17] and a width of $q \times w$ bits. For each decoding time, one address is read and one address is written from each memory.

The permutation and de-permutation of the variable messages in steps 4 and 9 in Algorithm 1 are implemented by modules $\mathbf{P}$ and $\mathbf{P}^{-1}$, respectively. Each module requires $d_c \times (q-1) \times \log_2 q$ multiplexers of w bits to permute or de-permute d_c vectors of $(q-1)$ messages, and the control signals are based on the h_{mn} nonzero values of $\mathbf{H}$.

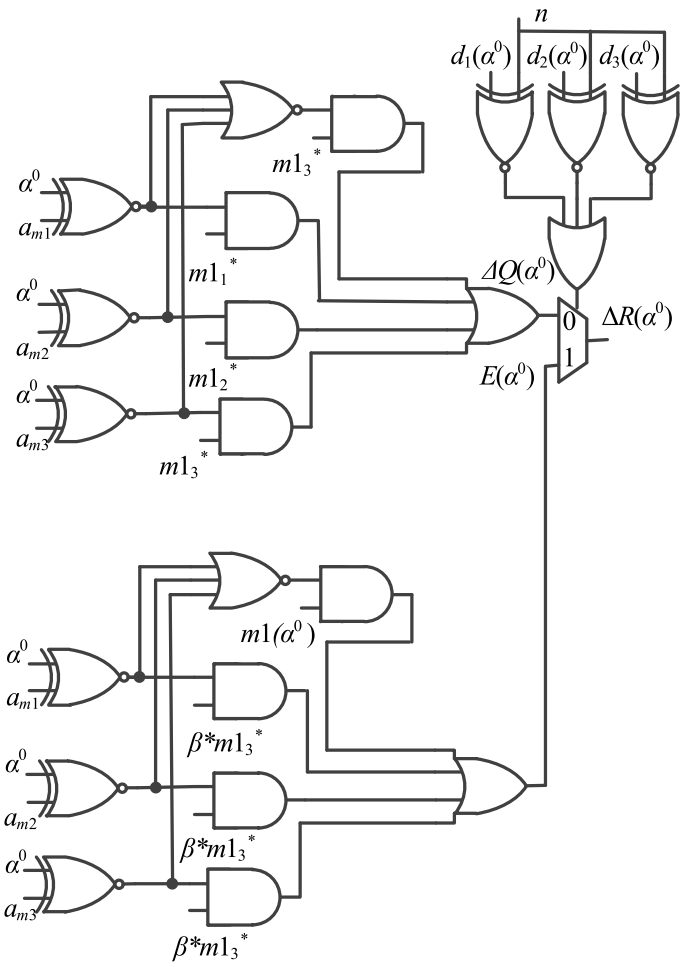

Figure 7: Proposed C2V generator for GF(8)

The normalization module **N** is responsible for finding the most reliable messages and their locations z_n, and generating the $Q_{mn}^{k,l}(a)$ messages for the inputs of the check node processor. In addition, normalization ensures that the smallest value in each LLR vector $Q_{mn}^{k,l}(a)$ is always equal to zero. At the last decoding iteration, the z_n values are the hard-decision symbols $\tilde{c}_n$ stored in the output memory (OUTMEM), and the **P** module and subtractor are inactive during this process.

It is remarked that the decompression network (DN) corresponding to Algorithm 4 is implemented in the variable node processor to generate the C2V messages $R_{mn}(a)$ from outputs of the CNU architecture. Figure 7 shows the proposed C2V generator in the DN module, which is based on the OMO-BS-TMM algorithm for each C2V message vector in GF(8). Since both the extra-column constructor and the complement sets are eliminated, the complexity of the proposed C2V generator is significantly reduced. For three field elements in the basic set, the C2V messages are either the LLR values in the basic set or the complement values $E(a) = \beta \times m1_p^*$, which depend on the path information. It is clear that for the remaining field elements, the C2V messages are either the LLR value of the last field element

Table 2: Comparison of the proposed decoder with other works for the (837, 726) NB-LDPC code over GF(32).

Algorithm	STMM [8]	TMM [13]	mT-MM [15]	TEC -TMM [10]	TMM [11]	BS-TMM [12]	OMO-BS-TMM
Report	Post.	Post.	Post.	Syn.	Post.	Post.	Syn.
Quantization	6	6	6	6	6	5	5
(d_v, d_c)	$(4, 27)$	$(4, 27)$	$(4, 27)$	$(4, 27)$	$(4, 27)$	$(4, 27)$	$(4, 27)$
Gate count (NAND)	3.28M	1.25M	1.17M	800K	1.06M	756K	601K
f_{clk} (MHz) (Synthesis)	238	300	345	370	393	395	405
Iteration	9	8	8	8	8	8	8
Throughput (Mbps)	660	981	1080	1274	1071	1261	1404
Efficiency (Mbps/Mgates)	201.2	784.8	932.07	1592.5	1010.4	1668	2336

in the basic set as $m1_3^*$ or the complement values $E(a) = m1(a)$.

Since a layered decoding scheme is used, the outputs of the check node processor in one iteration must be stored in the check node memory (CNMEM) for the next iteration process. Thus, the CNMEM in the proposed decoder has a depth of **M** and a width of $p \times (w + \lceil \log(d_c) \rceil + p) + ((q-1) - p) \times w + d_c \times p$ bits corresponding to the output bits of the check node processor. A total of $M \times [p \times (w + \lceil \log(d_c) \rceil + p) + ((q-1) - p) \times w + d_c \times p]$ bits are stored in one iteration. Compared to the $M \times q \times d_c \times w$ bits stored in CNMEM in the conventional approach [8], the memory requirement for CNMEM in the proposed decoder is greatly reduced, which leads to a large reduction in decoder area.

5. IMPLEMENTATION RESULTS AND COMPARISON

To illustrate the efficiency of our proposal for NB-LDPC codes, the complete decoder architectures were implemented for (837, 726) NB-LDPC code over GF(32). A Verilog HDL was used to model the architectures, and Synopsys design tools with the TSMC 90-nm CMOS standard cell library were used to implement the proposed decoder architectures. The throughput T_p of the decoders is archived as shown in the equation

$$Tp = \frac{f_{clk}[\text{MHz}] \times (q-1) \times d_c \times p}{I_{max} \times (M + d_v \times seg) + (q-1)}[\text{Mbps}], \tag{2}$$

where seg is the number of pipeline stages used in the decoder architecture to improve the timing. In the proposed decoder architectures, $seg = 9$ was chosen to obtain a balance between throughput and area.

Table 2 shows the implementation results of the proposed decoder in comparison with the other state-of-the-art works for the (837, 726) NB-LDPC code over GF(32). It can be seen that the proposed decoder outperforms the other approaches in both area and throughput. Compared to the STMM algorithm with uncompressed messages [8], our work has almost 11.6 times higher efficiency, and reduces gate count by a factor of 5.45. This significant

improvement is achieved by the great reduction in both the storage bits in the check node memory and the CNU complexity, as explained previously. In [11], a reduced-complexity NB-LDPC decoder was proposed on the basis of reducing the size of the intrinsic information and the path coordinates to $L \ll q$ values, and the decoder performance depends on the selected L value, whereas our approach reduces the size of these sets to $p = \log_2 q$ values for any GF. Because the complexity of the proposed CNU is reduced, the efficiency of the proposed decoder with $p = 5$ is almost 2.3 times higher than that in [11] implemented with $L = 4$. Compared to the decoders in [10, 13, 15], the proposed decoder reduces the gate count by 52%, 48.6%, and 24.8%, and achieves 66.4%, 60%, and 31.8% higher efficiency, respectively. Compared to the work using the basic sets of the reliable messages BS-TMM [12], the proposed decoder improves not only the gate count but also the throughput because of a significant reduction of the complexity in the CNU as well as the whole decoder architecture. Therefore, the proposed decoder reduces the gate count by 20.5%. Moreover, the proposed decoder exhibits almost 29% higher efficiency compared to the work in [12].

6. CONCLUSION

In this paper, we proposed an one-minimum-only basic-set Trellis min-max algorithm for decoding NB-LDPC codes to reduce the complexity of the CNU architecture, the messages exchanged between the check node and the variable node, and the storage bits in the CNMEM, compared with previous works. The error-correcting performances, which is illustrated by the frame error rate (FER) performance of (837, 726) NB-LDPC code over GF(32) under the additive white Gaussian noise (AWGN) channel and binary phase shift keying (BPSK) modulation, demonstrate that the proposed OMO-BS-TMM algorithm obtains a good error-correcting performance, and a significantly reduced computation complexity and hardware complexity for the high-order GF. The implementation results show that the decoder architecture based on the proposed algorithm provides a great area reduction and throughput improvement compared with the other state-of-the-art works.

ACKNOWLEDGMENT

This work was supported by National Laboratory of Information Security, Ha Noi, Viet Nam.

REFERENCES

[1] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over gf(q)," in *1998 Information Theory Workshop (Cat. No.98EX131)*, 1998, pp. 70–71.

[2] R. Peng and R. Chen, "Wlc45-2: Application of nonbinary ldpc codes for communication over fading channels using higher order modulations," in *IEEE globecom 2006*, 2006, pp. 1–5.

[3] M. Arabaci, I. B. Djordjevic, L. Xu, and T. Wang, "Nonbinary ldpc-coded modulation for high-speed optical fiber communication without bandwidth expansion," *IEEE Photonics Journal*, vol. 4, no. 3, pp. 728–734, 2012.

[4] Z. Cui, Z. Wang, and X. Huang, "Multilevel error correction scheme for mlc flash memory," in *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2014, pp. 201–204.

[5] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary ldpc codes over gf(q)," *IEEE Transactions on Communications*, vol. 55, no. 4, pp. 633–643, 2007.

[6] V. Savin, "Min-max decoding for non binary ldpc codes," in *2008 IEEE International Symposium on Information Theory*, 2008, pp. 960–964.

[7] F. Cai and X. Zhang, "Relaxed min-max decoder architectures for nonbinary low-density parity-check codes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 11, pp. 2010–2023, 2013.

[8] J. O. Lacruz, F. Garcia-Herrero, D. Declercq, and J. Valls, "Simplified trellis min-max decoder architecture for nonbinary low-density parity-check codes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 9, pp. 1783–1792, 2015.

[9] J. O. Lacruz, F. Garcia-Herrero, J. Valls, and D. Declercq, "One minimum only trellis decoder for non-binary low-density parity-check codes," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 1, pp. 177–184, 2015.

[10] H. P. Thi and H. Lee, "Two-extra-column trellis min-max decoder architecture for nonbinary ldpc codes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 5, pp. 1787–1791, 2017.

[11] J. O. Lacruz, F. Garcia-Herrero, M. J. Canet, and J. Valls, "Reduced-complexity nonbinary ldpc decoder for high-order galois fields based on trellis min-max algorithm," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 8, pp. 2643–2653, 2016.

[12] H. P. Thi and H. Lee, "Basic-set trellis min-max decoder architecture for nonbinary ldpc codes with high-order galois fields," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 3, pp. 496–507, 2018.

[13] J. O. Lacruz, F. Garcia-Herrero, and J. Valls, "Reduction of complexity for nonbinary ldpc decoders with compressed messages," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 11, pp. 2676–2679, 2015.

[14] J. Lacruz, F. Garcia-Herrero, M. Canet, J. Valls, and A. Perez-Pascual, "A 630 mbps non-binary ldpc decoder for fpga," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, pp. 1989–1992.

[15] J. O. Lacruz, F. Garcia-Herrero, M. J. Canet, and J. Valls, "High-performance nb-ldpc decoder with reduction of message exchange," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 5, pp. 1950–1961, 2016.

[16] H. P. Thi, C. D. The, N. P. Xuan, H. D. Tuan, and H. Lee, "Simplified variable node unit architecture for nonbinary ldpc decoder," in *2019 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2019, pp. 213–216.

[17] B. Zhou, J. Kang, S. W. Song, S. Lin, K. Abdel-Ghaffar, and M. Xu, "Construction of non-binary quasi-cyclic ldpc codes by arrays and array dispersions," *IEEE Transactions on Communications*, vol. 57, no. 6, pp. 1652–1662, 2009.

[18] J. Lin, J. Sha, Z. Wang, and L. Li, "Efficient decoder design for nonbinary quasicyclic ldpc codes," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 5, pp. 1071–1082, 2010.

[19] C. Wey, M. Shieh, and S. Lin, "Algorithms of finding the first two minimum values and their hardware implementation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 11, pp. 3430–3437, 2008.

Received on March 07, 2021
Accepted on May 08, 2021