# Simplified Variable Node Unit Architecture for Nonbinary LDPC Decoder

1st Huyen Pham Thi
*National Laboratory of Information Security*
Ha Noi, Viet Nam
phamhuyenmta87@gmail.com

2nd Cuong Dinh The
*National Laboratory of Information Security*
Ha Noi, Viet Nam
cuongdt@mta.edu.vn

3rd Nghia Pham Xuan
*Le Quy Don Technical University*
Ha Noi, Viet Nam
nghiapx@mta.edu.vn

4th Hung Dao Tuan
*National Laboratory of Information Security*
Ha Noi, Viet Nam
daotuanhung@gmail.com

5th Hanho Lee
*Dept. Information and Communication Engineering*
*Inha University*
Incheon, Korea
hhlee@inha.ac.kr

***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, a simplified basic-set trellis min-max (sBS-TMM) algorithm that is especially efficient for high-order Galois Fields, is proposed for the variable node processing to reduce the complexity of the variable node unit (VNU) as well as the whole decoder. The decoder architecture corresponding to the proposed algorithm is designed for the (837, 726) NB-LDPC code over GF(32). The implementation results using 90-nm CMOS technology show that the proposed decoder architecture reduces the gate count by 21.35% and 9.4% with almost similar error-correcting performance, compared to the up-to-date works.**

***Index Terms*—NB-LDPC, Basic-set, Trellis min-max, VLSI design**

## I. 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.

For practical NB-LDPC decoder implementations, suboptimal algorithms such as extended min-sum (EMS) [2] and the min-max [3] 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 [4] 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 [3]. However, the check node processing is sequentially processed which requires a large number of clock cycles. In [5], 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 this work [5], $q \times d_c$ check node output messages are exchanged between the check node and the variable nodes. For high-order GFs or high-rate NB-LDPC codes, the amount of exchanged messages increases and the memory requirement is large, which limits the maximum throughput of the decoders and leads to a significant increase in the decoder area.

To overcome the above drawbacks, the work in [6] proposed to simplify the CNU architecture and reduce the exchanged messages with the almost similar error-correcting performance. In [7], the approximated TMM algorithms are introduced to further decrease the number of intrinsic information at the cost of some error-correcting performance loss. In [8], 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, a simplified basic-set trellis min-max (sBS-TMM) algorithm is proposed for the variable node processing to further reduce the decoder complexity with the almost similar error-correcting performance, compared to the existing decoding algorithm. The decoder architecture of a (837, 726) NB-LDPC code over GF(32) was performed using the sBS-TMM algorithm to demonstrate the efficiency of the proposal.

## II. Review of NB-LDPC Decoding Algorithm

### A. 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 $N$ columns. Let $h_{mn}$ be a nonzero element of the matrix $\mathbf{H}$ 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 $\mathbf{H}$, respectively. A regular NB-LDPC code is considered in this paper with the fixed values of $d_c$ and $d_v$.

**Algorithm 1** Layered Decoding Algorithm [8]

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

### B. NB-LDPC Decoding Algorithm

*Algorithm* 1 presents the layered basic-set trellis min-max (BS-TMM) decoding algorithm for the NB-LDPC codes [8]. 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. $k$ 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 [5], [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 $\mathbf{H}$, as shown in step 4. The normalization of V2C messages are performed in step 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.

### C. Basic-set Trellis Min-Max Algorithm

In this section, the BS-TMM algorithm [8] 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 $B^*$.

**Algorithm 2** Basic-Set TMM Algorithm [8]

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

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 $\{\alpha^0, \alpha^1, ..., \alpha^{q-1}\}$. Step 2 relates to the computation of the syndrome $\beta$ using the most reliable symbols $z_n$ from V2C messages. In step 3, the first minimum value $m1(a)$, its column index $m1_{col}(a)$, and the second minimum value $m2(a)$ for each trellis row are calculated using the function $\Psi$. 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 $\Phi$ 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 ones are assigned to the minimum values $m1(a)$. Finally, the output of the check node processing includes three sets $B^*$, $E(a)$, and $z_n^*$, which are used for generating the C2V messages in the variable node processing.

## III. Simplified Basic-set Trellis Min-Max Decoding Algorithm

### A. Simplified Basic-set Trellis Min-Max Decoding Algorithm

In the variable node processing, *Algorithm* 3 shows the simplified extra column construction $\Delta Q(a)$ that the $\Delta Q(a)$ and the path information $d(a)$ are calculated based on the basic set $B^*$ as one of output sets of the check node processing. 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 $1 \leq l \leq p$.

From our observation, a proximate approach is proposed for calculation of the remaining field elements. In [8], the remaining field elements are computed on the basis of all

**Algorithm 3** Simplified Extra Column Construction $\Delta Q(a)$

**Input:** $B^* = \{m1_l^*, I_l^*, a_l^*\}_{1 \le l \le p}$;
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: **else if** $a = a_1^* \oplus a_2^* \oplus ... \oplus a_s^* (2 \le s \le p)$ **then**
5: $\Delta Q(a) = m1_p^*$;
$d(a) = \{I_1^* \cup I_2^* \cup ... \cup I_s^*\}$
6: **end if**
7: **end for**

possible combinations of the field elements in the basic set $B^*$. Their $\Delta Q(a)$ values are the maximum LLR value from the LLR values corresponding to the combined field elements as $\Delta Q(a) = \max(m1_1^*, m1_2^*, ..., m1_s^*)$ with $(2 \le s \le p)$, and their path information $d(a)$ has more than one deviation and a maximum of $p$ deviations. It is remarked that there are $(q-1)-p$ remaining field elements, which are generated on the basic of combinations of the $p$ field elements in the basic set $B^*$. For each remaining field element corresponding to each combination, $\Delta Q(a)$ value is the maximum LLR value from the LLR values of the combined field elements. Furthermore, the number of possible combinations including the last field element $a_p^*$ is the largest in comparison to the other field elements in the basic set. It is clear that the LLR value $m1_p^*$ corresponding to the last field element $a_p^*$ accounts for a larger factor than other field elements. Therefore, in this work, an approximate method is proposed to assign the LLR value of the last field element $m1_p^*$ to the LLR values of the remain field elements in the extra column $\Delta Q(a)$, as shown in Step 5 of *Algorithm* 3.

For example, in GF(8), the basic set $B^*$ consists of three field elements such as $\{a_1^*, a_2^*, a_3^*\}$. Other nonzero field elements are constructed as $\Delta Q(a_1^* \oplus a_2^*) = \max(m1_1^*, m1_2^*) = m1_2^*$; $\Delta Q(a_1^* \oplus a_3^*) = \max(m1_1^*, m1_3^*) = m1_3^*$; $\Delta Q(a_2^* \oplus a_3^*) = \max(m1_2^*, m1_3^*) = m1_3^*$; $\Delta Q(a_1^* \oplus a_2^* \oplus a_3^*) = \max(m1_1^*, m1_2^*, m1_3^*) = m1_3^*$. It is obvious that most of LLR values of the remaining field element in the extra column are equal to $m1_3^*$ that is the LLR value of the last field element in the basic set. Thus, in this work, $m1_3^*$ value is assigned to $\Delta Q(a)$ values of the all remaining field elements.

### *B. Performance Analysis*

To demonstrate the error-correcting performance of the sBS-TMM decoding algorithm, Fig. 1 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 Fig. 1, the floating-point simulation result of the sBS-TMM algorithm with 15 iterations shows the almost similar error-correcting performance, compared to the BS-TMM algorithm [8], and a performance loss at almost 0.1 dB, compared to the two-extra-column TMM (TEC-TMM) algorithm [6] and the STMM algorithm

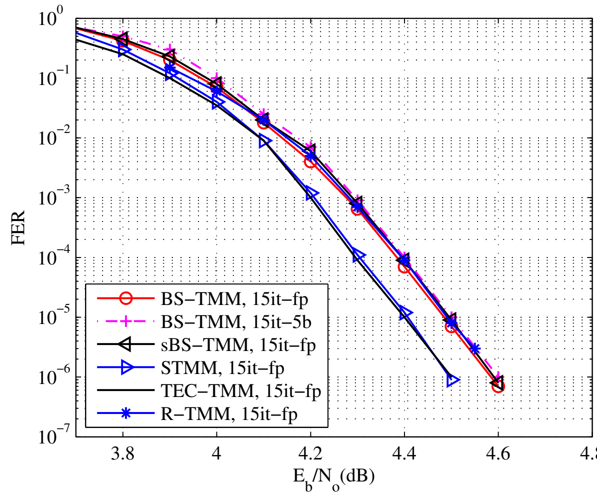


Fig. 1. FER performance of the (837, 726) NB-LDPC code over GF(32) under the AWGN channel at 15 iterations.

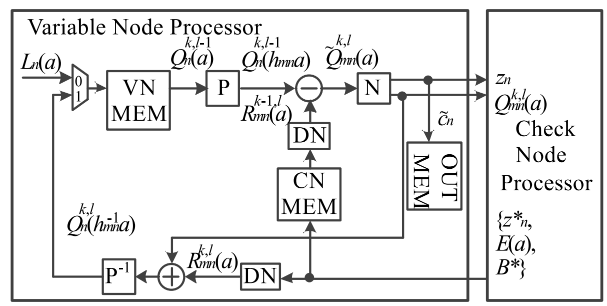


Fig. 2. Top-level NB-LDPC decoder architecture based on the sBS-TMM algorithm [8].

[5]. Because the proposed sBS-TMM algorithm discards the computation of the $((q-1)-p)$ values of the remaining field elements in the extra column $\Delta Q(a)$, the decoder architecture corresponding to the sBS-TMM algorithm obtains a low computational complexity, a large area reduction, and a significant improvement in throughput.

## IV. Reduced-complexity decoder architecture

Fig. 2 shows the top-level decoder architecture for the proposed layered decoding algorithm, where one row of $\mathbf{H}$ 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. The decoding process and modules such as permutation $\mathbf{P}$, depermutation $\mathbf{P}^{-1}$, and normalization $\mathbf{N}$ are similar to the ones in [8]. It is remarked that the decompression network (DN) corresponding to *Algorithm* 3 is implemented in the variable node processor to generate the C2V messages $R_{mn}(a)$ from outputs of the CNU architecture.

Fig. 3 shows the proposed C2V generator in the DN module, which is based on the sBS-TMM algorithm for each C2V message vector in GF(8). Since the extra-column constructor is eliminated, the complexity of the proposed C2V generator

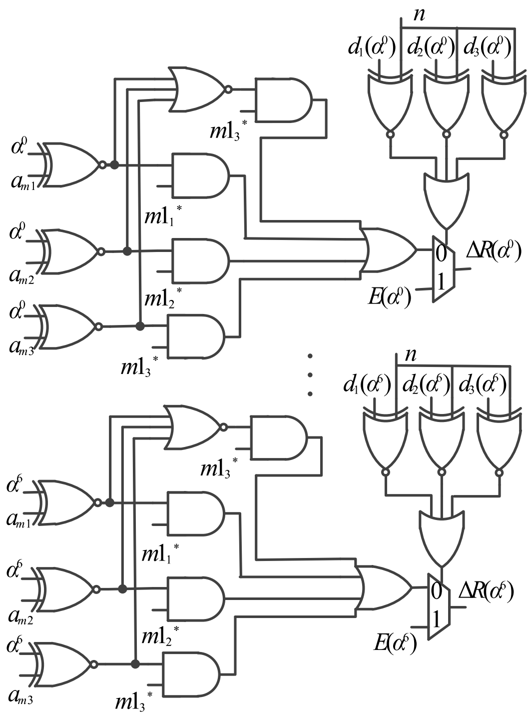


Fig. 3. Proposed C2V generator based on sBS-TMM algorithm for GF(8).

TABLE I
IMPLEMENTATION RESULTS OF THE PROPOSED DECODER FOR THE (837, 726) NB-LDPC CODE OVER GF(32) IN A 90-NM CMOS PROCESS

| Algorithm | RMM [4] | T-MM [7] | BS-TMM [8] | sBS-TMM [proposed] |
|---|---|---|---|---|
| Report | Syn. | Post-layout | Post-layout | Syn. |
| Quantization | 5 bits | 6 bits | 5 bits | 5 bits |
| Gate count (NAND) | 871K | 1.06M | 756K | 685K |
| $f_{clk}$ (MHz) (Synthesis) | 200 | 345 | 393 | 397 |
| Iterations | 8 | 8 | 8 | 8 |
| Throughput (Mbps) (Layout) | 154 | 1071 | 1261 | 1264 |
| Efficiency (Mbps/M gates) | 176.8 | 1010.4 | 1668 | 1845 |

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 $E(a)$, which depend on the path information. It is clear that for remaining field elements, the C2V messages are either the LLR value of the last field element in the basic set as $m1_3^*$ or the complement values $E(a)$.

## V. Complexity and Comparison

In this work, the (837, 726) NB-LDPC code over GF(32) is constructed by the submatrix $(d_v, d_c) = (4, 27)$ and a CPM of size $(q-1) \times (q-1)$ [10]. 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). The synthesis results of the proposed decoder for the (837, 726) NB-LDPC code and the comparison with previous works are presented in Table I. It can be seen that the proposed decoder reduces the gate count by 35.38% and achieves almost twice times higher efficiency, compared to the work from [7]. Compared to the works with using the basic sets of the reliable messages [4], [8], the proposed decoder improves not only the gate count but also the throughput because of a significant reduction of the complexity in the VNU as well as the whole decoder architecture. Therefore, the proposed decoder reduces the gate count by 21.35% and 9.4%, respectively. Moreover, the proposed decoder exhibits almost 10.61% higher efficiency, compared to the work in [8].

## VI. Conclusion

In this paper, a simplified BS-TMM algorithm is proposed for the NB-LDPC codes to reduce the decoder complexity. The decoder architecture corresponding to the proposed algorithm is designed to demonstrate the efficiency of the proposal. The implementation results show a gate count reduction of 35.38% and 21.35% with the almost similar error-correcting performance.

## Acknowledgment

This work was supported by National Laboratory of Information Security, Ha Noi, Viet Nam and, in part, by the MSIT, Korea, under the ITRC support program (IITP-2018-2014-0-00729) supervised by the IITP.

## References

[1] H. C. Davey and D. J. MacKay, "Low-density parity check codes over GF(q)," in *Information Theory Workshop*, Jun. 1998, pp. 165–167.

[2] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over GF(q)," *IEEE Trans. Commun.*, vol. 55, no. 4, pp. 633–643, Apr. 2007.

[3] V. Savin, "Min-max decoding for non binary LDPC codes," in *in Proc. IEEE Int. Symp Inf. Theory*, Toronto, ON, Canada, Jul. 2008, pp. 960–964.

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

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

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

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

[8] 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.

[9] J. O. Lacruz, F. García-Herrero, J. Valls, and D. Declercq, "One minimum only trellis decoder for non-binary low-density parity-check codes," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 1, pp. 177–184, Jan. 2015.

[10] B. Zhou, J. Kang, S. Song, S. Lin, K. Abdel-Ghaffar, and M. Xu, "Construction of non-binary quasi-cyclic LDPC codes by arrays and array dispersions," *IEEE Trans. Commun.*, vol. 57, no. 6, pp. 1652–1662, Jun. 2009.