EAI Endorsed Transactions
on Industrial Networks and Intelligent Systems
Research Article EAI.EU

Efficient LDPC Code Design based on Genetic Algorithm for IoT Systems

Thanh-Loc Nguyen-Van[1], Tan Do-Duy[1,*], Thien Huynh-The[1]

[1]Department of Computer and Communication Engineering, Faculty of Electrical and Electronics Engineering, Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam

Abstract

In this paper[†], we propose a low-density parity check (LDPC) code design scheme that improves the performance of the existing genetic algorithm-based LDPC scheme. In particular, we enhance the performance of the LDPC code by removing the girth-4 property of the parity check matrix and utilizing the min-sum decoding algorithm instead of the belief propagation decoding algorithm. In addition, we consider various short block-length scenarios, specifically focusing on 64-bit and 128-bit lengths, which are well-suited for IoT systems. Then, we evaluate the block error rate (BLER) of the LDPC code over the binary input additive white Gaussian noise (BI-AWGN) channel. Finally, extensive simulation results indicate that our proposed approach achieves more than 11% gain in terms of BLER compared with the benchmarked schemes.

Received on 11 02 2024; accepted on 01 07 2024; published on 01 08 2024

Keywords: LDPC, genetic algorithm, short block length, Internet of Things.

doi:10.4108/eetinis.v11i4.5843

1. Introduction

Ultra-reliable low-latency communication (URLLC) is one of the key features of the fifth-generation (5G) and sixth-generation (6G) wireless networks, as well as Internet of Things (IoT) systems [2]. URLLC caters to applications with critical needs for extremely high transmission success rates, minimal delays in data delivery from source to destination, and the adaptability to handle packets of diverse lengths. These requirements make URLLC a challenging technology to implement, particularly for the design of forward error correction (FEC) encoding and decoding schemes in the physical layer [3].

FEC is a powerful technique used to control errors in data transmission over unreliable or noisy communication channels where redundancy codes are added to the information data to allow receivers to detect and correct possible errors, thus enhancing the data's reliability. Consequently, by using strong FEC codes in short-packet transmission systems, we can achieve high reliability and low latency requirements for URLLC applications [4]. There are two main types of FEC, namely block codes and convolutional codes. One of the most popular types of block codes is low-density parity check (LDPC) codes, which play a vital role in improving the reliability of the data [4, 5]. In particular, LDPC codes first proposed by Robert Gallager in the 1960s [6], are known for their high error-correcting capabilities and have been shown to approach the Shannon limit, which is the theoretical maximum rate at which data can be transmitted over a noisy channel without errors. Advantageously, LDPC codes lend themselves well to hardware implementation because of their inherent simplicity. This stems from the structure of their parity-check matrix, which is characterized by sparsity. In simpler terms, this matrix contains only a few non-zero elements in each row and column. This sparsity translates to a reduction in the complexity of the hardware required to implement LDPC code processing. This makes them highly appropriate for hardware resource-constrained devices in IoT systems [7, 8]. This shift is also reflected in the decision by the 3GPP (3rd

*Corresponding author: Tan Do-Duy. Email: tandd@hcmute.edu.vn
[†]The paper has been presented in part at the International Conference on System Science and Engineering (ICSSE), Ho Chi Minh, Vietnam [1].

Generation Partnership Project) to adopt LDPC codes instead of Turbo codes within their New Radio (NR) technology standard [9]. Furthermore, beyond cellular applications, LDPC codes are even used in near-earth and deep-space communication systems [10].

Prominent researchers from both academic institutions and private businesses have conducted extensive research on LDPC codes for short-block transmission, but these studies still have some limitations. For example, while the method proposed in [11] delivers good performance, it requires the user to have the experience to be effective. In another work [12], the proposal uses a combination of cyclic redundancy check (CRC) code and LDPC code for error correction. However, research in [12] doesn't work well for cases where the decoding algorithm utilizes classical iterative decoding with numerous iterations. In another work [13], this proposal suggests replacing ordinary LDPC codes with NB-LDPC codes (Non-Binary Low-Density Parity Check), because NB-LDPC codes offer significant benefits, particularly when dealing with variable information block lengths or transmission environments prone to burst errors. However, complex computation and high memory requirements of NB-LDPC codes make them less suitable for applications that use minimal hardware, like the IoT applications [14]. Consequently, it appears that no research has yet discovered an optimal method for enhancing LDPC code performance in ultra-short block transmissions within ultra-reliable low-latency communication scenarios.

Referring to the recent LDPC-related surveys, the work in [15] provides an overview as well as a performance comparison between three error-correcting codes, including LDPC, Polar, and Turbo codes in terms of implementing their decoding algorithms in application-specific integrated circuits (ASIC) for mobile communication systems. In another research [7], the authors present a comprehensive survey that summarizes and compares different L DPC d ecoding algorithms based on key features, like error-correcting performance. In addition, the authors emphasize the necessity for versatile and computationally efficient decoding schemes, which can be utilized for both fixed and varying channel conditions. Furthermore, the authors propose to utilize optimization algorithms, linear programming, and parameter estimation to enhance LDPC decoding performance in the future. On the other hand, the authors in [16] present a comprehensive brief of the genetic algorithm (GA) as well as its benefits and drawbacks. In addition, the authors in [17] propose to utilize the genetic algorithm to optimize the whitelist of the industrial firewall for industrial control systems. In this system, the genetic algorithm plays a crucial role in supporting the vector machine (GA-SVM) algorithm to automatically learn the rules, which improves the efficiency of in dustrial co ntrol sy stems. Additionally, in [18], an LDPC code design scheme for the short-packet transmission systems is proposed. This proposal is constructed based on the well-known genetic algorithm, similar to the proposal for polar codes in [19]. Specifically, the proposed scheme in [18] directly optimizes the parity check matrix of the LDPC code. In the short-packet transmission scenario over both the binary input additive white Gaussian noise (BI-AWGN) channel and Rayleigh channel, the LDPC code designed using a genetic algorithm in [18] demonstrates better performance than other LDPC codes such as CCSDS Up-Link LDPC, Regular LDPC, and 5G LDPC [18]. Furthermore, the LDPC code designed using a genetic algorithm exhibits adaptability to practical decoding requirements and channel constraints since it does not have any special graph structure (i.e., due to crossover and mutation of the genetic algorithm).

To the best of the authors' knowledge, the genetic algorithm-based LDPC code design scheme in [18] can be enhanced to attain better performance while slightly increasing the system's complexity. The main contributions of our paper can be summarized as follows:

1. We propose an improved LDPC code design scheme built based on the design scheme in [18]. Specifically, we eliminate the worst fitness case of the genetic algorithm-based optimization process by removing the girth-4 property of the parity check matrix of LDPC codes and propose the utilization of the min-sum decoding algorithm to increase the decoding performance of the LDPC code simultaneously.

2. We evaluate the performance of our proposed design with different short block length scenarios, including 64-bit and 128-bit block length. In terms of the block error rate (BLER), our proposed approach can achieve a gain of more than 11% compared to the existing scheme.

The remainder of this paper is organized as follows. In Section 2, we review the fundamentals of LDPC code design, taking into account LDPC decoding algorithms as well as the effect of the girth on the decoding performance. In Section 3, we propose an LDPC code design scheme that improves the performance of the existing genetic algorithm-based LDPC scheme. In Section 4, we evaluate the performance of our proposed genetic algorithm-based LDPC design scheme in comparison with the conventional schemes. In Section 5, we conclude the paper.

2. Overview of LDPC codes

LDPC codes are a type of FEC codes that can be used to detect and correct errors in data transmission. In LDPC codes, the parity check matrix ($m \times n$) is denoted

by $H = \left[h_{ji}\right]_{m\times n}$ where n is the number of variable nodes (VNs) (i.e., n is the number of codewords) and m is the number of check nodes (CNs) (i.e., m is the number of redundancy codes that is added to the data before transmission to improve the reliability of data transmission). The low density implies that the size of the parity check matrix is usually very large, but the density of nonzero elements is very low [20]. Let us denote $k = n - m$ as the number of data bits. The code rate is then defined as the ratio of the number of data bits k and the total number of bits in the codeword n and is expressed as

$$R_c = \frac{k}{n}. \tag{1}$$

A higher code rate means that more bits of data can be transmitted in a given amount of time, but the code is less able to correct errors i.e., lower data reliability. Therefore, according to [20], to achieve a balance between reliability performance and the number of data bits, the code rate is usually chosen to be $R_c = \frac{1}{2}$.

LDPC codes can be classified into two types: regular LDPC codes and irregular LDPC codes. Regular LDPC codes are LDPC codes that the number of ones in each row and column of the parity-check matrix is the same. In contrast, irregular LDPC codes contain different numbers of ones in each row and column of the parity-check matrix [20]. In this paper, we explore a random construction of the parity check matrix for the irregular LDPC codes.

The parity check matrix of LDPC codes can also be presented by the Tanner graph, respectively, in which VNs v_i are connected to CNs c_j if $h_{ij} = 1$ with $i \in \{1, ..., n\}$ and $j \in \{1, ..., m\}$. For example, a (3×6) irregular LDPC code is presented by parity check matrix that is depicted in Matrix (2) and the Tanner graph, respectively, as depicted in Figure 1. During the LDPC decoding process, Log-Likelihood Ratio (LLR) values are passed back and forth between check nodes and variable nodes on the Tanner graph, iteratively refining the estimates until the correct data is recovered. It can be realized that the design of LDPC codes is the process of determining the link $h_{ji} = \{0, 1\}$ to achieve specific goals for different applications, including minimizing errors (target error floor) or fitting hardware limitations. Performance and complexity metrics for various LDPC decoding algorithms are presented in Table 1.

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}. \tag{2}$$

In addition, according to [20], in the Tanner graph, a sequence of connected nodes starting and ending at the same node is called a cycle and the number of edges in a cycle is called the cycle length. The minimum cycle length is called girth. The minimum lower bound distance for LDPC code with girth-g is written as:

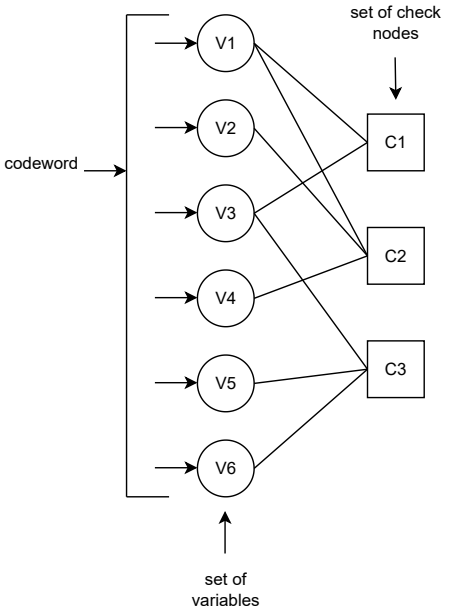

Figure 1. An illustration of (3×6) irregular LDPC code using Tanner Graph.

Table 1. Comparison of performance and complexity of different LDPC decoding algorithms

Parameter	Bit-Flipping	Sum-Product	Min-sum
Check node operation	XOR	$tanh$ and $tanh^{-1}$	XOR and comparison
Variable node operation	Comparison	Addition	Addition
LLR quantization	N-bit	N-bit	N-bit
Extrinsic message	1-bit	N-bit	N-bit
BER performance	Poor	Best	Good
Complexity	Simple	Complex	Complex for long codes
Clock per decoding iteration	1	1	1

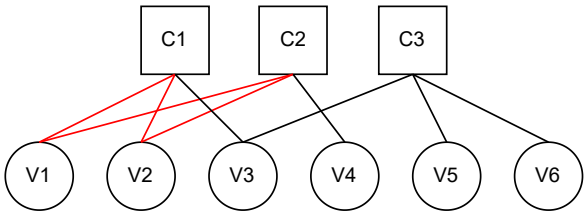

Figure 2. A Tanner graph with girth-4.

$$d_{min} \geq \begin{cases} 1 + w_c + w_c(w_c-1) + ... + w_c(w_c-1)^{\frac{g-6}{4}}, & \text{for odd } \frac{g}{2}, \\ 1 + w_c + w_c(w_c-1) + ... + w_c(w_c-1)^{\frac{g-8}{4}}, & \text{otherwise,} \end{cases} \quad (3)$$

where w_c is the column weight. Therefore, the minimum distance can be increased by raising the girth or the column weight [20]. This means that the effect of girth on the performance of LDPC codes can be mitigated by selecting LDPC codes whose corresponding Tanner graphs exhibit larger girth values. Girth-6 is sufficient; hence, the removal of girth-4 is mandatory [20]. The girth-4 is illustrated in the Tanner graph by the blue line in Figure 2 or depicted by a 1-bit 4-square in the parity check matrix as follows:

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}. \quad (4)$$

3. THE PROPOSED GENETIC ALGORITHM-BASED LDPC CODE DESIGN

We consider the LDPC code design process as an optimization process to reduce BLER at some fixed signal-to-noise ratio (SNR). Moreover, the code rate is set at $R_c = \frac{1}{2}$ to achieve a balance between performance and the number of data bits [20]. The number of variable nodes and check nodes is also fixed. In addition, each variable node (check node) must be connected to at least one check node (variable node). Our proposed genetic algorithm-based LDPC code design scheme can be summarized in Figure 3, in which a set of candidate LDPC codes is referred to as a population.

First, our proposed LDPC code design scheme begins with an initial population of some randomly constructed LDPC codes (i.e., population 1). An error rate computation framework is used to evaluate the error-rate performance of the LDPC code in a fixed-SNR environment. Next, the best-performing LDPC codes, which have the gain of BLER better than others, are selected to be parents in the new population. These LDPC codes are the input values for evolutionary transformations (mutations and crossovers), which are performed in the Update Population (Mutations & Crossovers) block. Then, girth-4 of the parity check

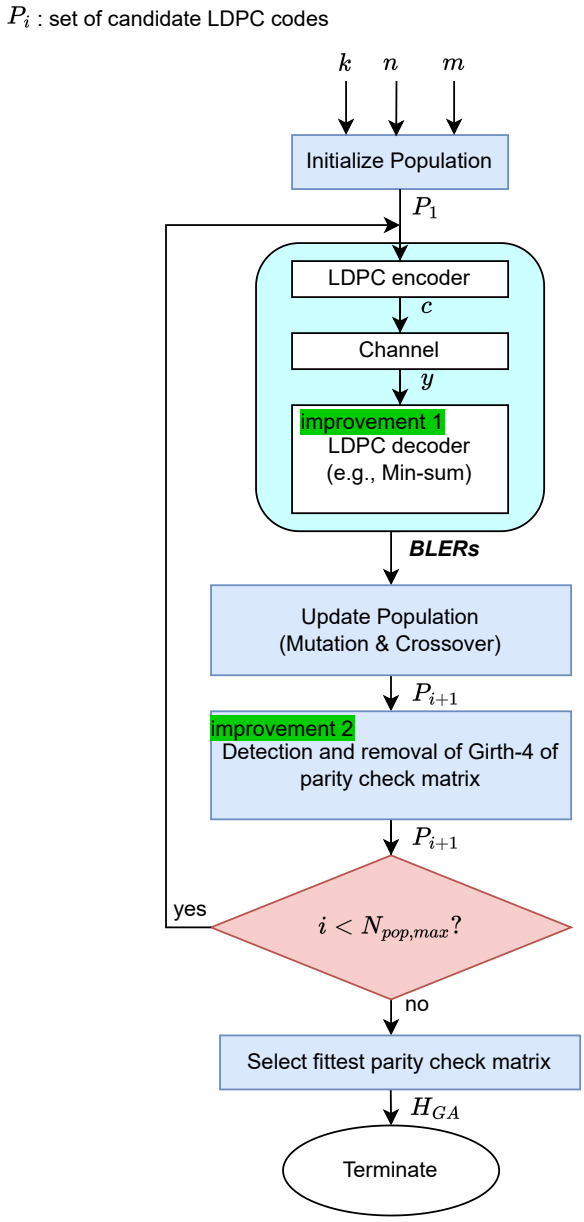

Figure 3. A summary of the proposed LDPC code design scheme.

matrix is detected and removed from the parity check matrix of these LDPC codes, which is carried out in the "Detection and Removal of Girth 4 of Parity Check Matrix" block. Finally, the above process is repeated until a target BLER is satisfied or the maximum population size is reached.

In genetic algorithms, choosing the right selection method is critical. It allows convergence towards optimal solutions because selected values are input values for evolutionary transformation. In our research, we pick the best offspring solutions to be parents in the

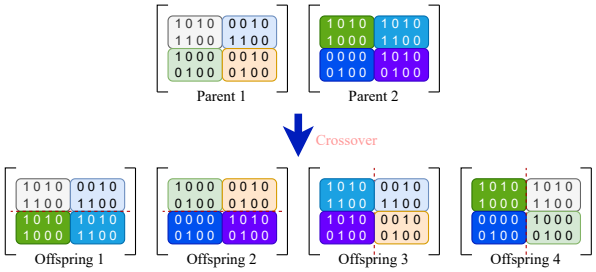

Figure 4. Illustration of the crossover operation.

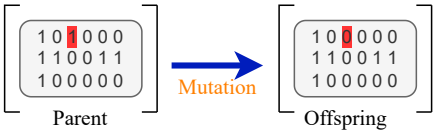

Figure 5. Illustration of the mutation operation.

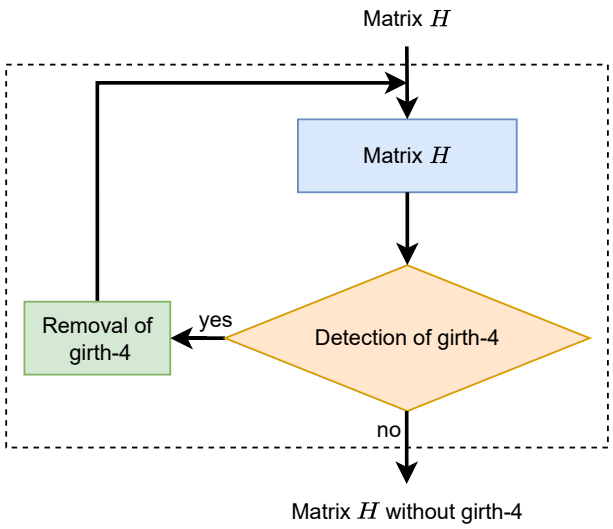

Figure 6. Block diagram of "Detection and Removal of Girth 4 of Parity Check Matrix" block.

new population based on the improvement of the BLER of the LDPC code as the fitness value.

The "Update Population (Mutations & Crossovers)" block encompasses two vital evolutionary transformations: crossovers and mutations. Combining crossover and mutation in genetic optimization algorithms offers a potent synergy. As illustrated in Figure 4, crossover operation leverages the strength of the parent, which operates like tools that shuffle the features of these good solutions (parents) to create entirely new solutions (offspring), that inherit their desirable features. This ensures continuity and refinement of promising solutions. Therefore, the crossover probability should be large, about 0.5. Conversely, as depicted in Figure 5, mutation operation introduces randomness and diversity into the population, fostering exploration of uncharted territories within the search space. Because mutation operates based on random changes, the probability should be small, around 0.0001. This ensures that only a small part of the individuals is changed. As a result, the balance between exploitation and exploration allows the algorithm to escape local optima and ultimately achieve superior performance [21, 22].

This work introduces two key enhancements over the prior method described in [18]. First, to achieve better decoding performance, we implement the min-sum decoding algorithm within the LDPC decoder, replacing the BP decoding algorithm used in [18]. Second, we incorporate a "Detection and Removal of Girth 4 of Parity Check Matrixes" block which functions as a penalty function, filtering out undesirable scenarios (specifically, identifying and eliminating girth-4 within each parity check matrix). This additional step helps reduce the time and computational cost required for the optimization process, as referenced in [20].

According to [20], in the parity check matrix H, the existence of a square with 4 sides of 1s is called girth-4. As mentioned, girth-4 has the worst negative impact on the performance of the LDPC code for short-length code. Conversely, the larger girths tend to have less impact. Therefore, it is crucial to identify and remove the girth-4 in the parity check matrix before the encoding and decoding processes. This can be achieved by strategically rearranging elements within the parity-check matrix to break up the squares with 4 sides of 1s while preserving the overall structure of the matrix.

As a crucial step in the process, detailed in Figure 6, this task involves identifying and eliminating a specific structural issue within the parity-check matrix, known as girth-4. The process can be broken down into two key stages:

1. Detection: The system meticulously examines the parity-check matrix to pinpoint the presence of girth-4.

2. Correction: If girth-4 is detected, the system takes corrective measures to remove it from the matrix. Only matrices free of girth-4 are transmitted for further processing.

To improve practicality, researchers developed the min-sum algorithm as a simpler alternative to the sum-product algorithm (SPA) by assuming $M_{ji} = y_i$ instead of $M_{ji} = \frac{4y_i}{N_0}$ as in SPA [20]. Min-sum algorithm significantly reduces the computational complexity of the SPA while achieving nearly the same performance [20]. The LDPC decoding process using the min-sum algorithm is illustrated in Figure 7 through five steps as follows.

Step 1: Initialization

$$L_i = L_{c_i|y_i} = y_i. \tag{5}$$

If $h_{i,j} = 1$, assign $M_{ji} = L_i$, where $i \in (0, .., n-1)$ and $j \in (0, .., m-1)$.

Step 2: Update Check Nodes

Calculating the prediction of each variable node by the following equations

$$\alpha_{ji} = sign(M_{ji}), \tag{6}$$

$$\beta_{ji} = |M_{ji}|, \tag{7}$$

$$E_{ji} = \prod_{i'} \alpha_{ji'} \times \min_{i'} \beta_{ji'}. \tag{8}$$

Step 3: The sum of LLR

$$L_i^{total} = L_i + \sum_{j \in A_i} E_{ji}, \tag{9}$$

where A_i is set of parity check equations of ith bit in received vector y.

Step 4: Code recovery

$$c_i = \begin{cases} 1 & if \quad L_i^{total} < 0 \\ 0 & else. \end{cases} \tag{10}$$

If $cH^T = 0$ or the number of iterations is larger than the limit ($ite > N_{ite,max}$), the decoding process stops; otherwise, proceed to Step 5.

Step 5: Update Variable Nodes

$$M_{ji} = L_i + \sum_{j' \in A_i, j' \neq j} E_{ji}. \tag{11}$$

Assign $ite = ite + 1$ and go back to Step 2.

4. PERFORMANCE EVALUATION

In this section, we conduct extensive simulation results using Matlab to evaluate the performance of our proposed genetic algorithm-based LDPC code design compared with the conventional schemes as follows:

(i) Scheme 1: LDPC code design scheme in [18].

(ii) Scheme 2: the original scheme in [18] added "Detection and removal of Girth 4 of parity check matrix" block.

(iii) Scheme 3: our proposed scheme where both "Detection and Removal of Girth 4 of Parity Check Matrices" block and "Min-sum algorithm-based decoder" block are added as illustrated in Figure 3.

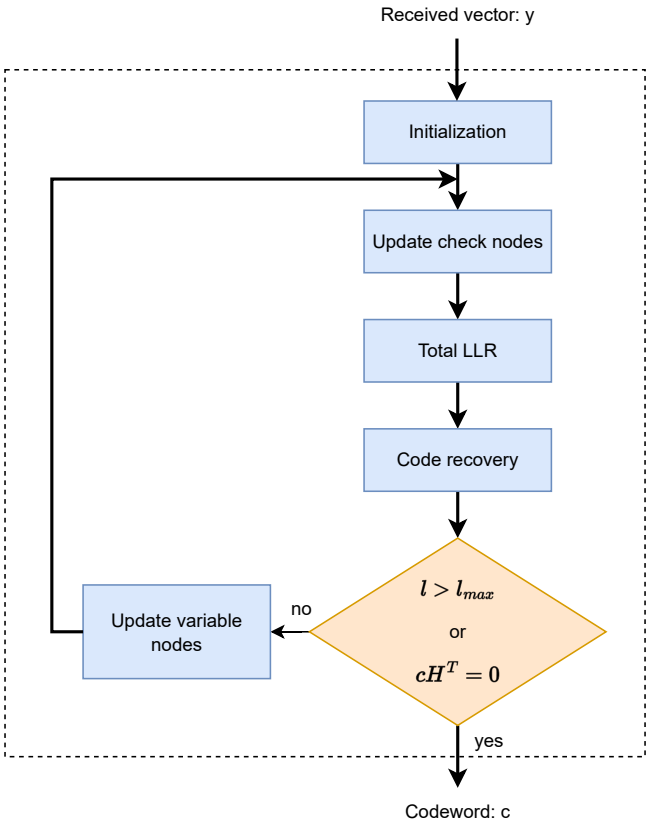

Figure 7. Summary of LDPC decoding process using min-sum algorithm.

4.1. Scenario of 64-bit block length code

BLER versus the number of populations. The performance evaluation of our proposed LDPC code design scheme is performed according to the simulation settings as summarized in Table 2. In particular, we design LDPC codes with the number of variable nodes $n = 64$, the number of check nodes $m = 32$, and code rate $R_c = \frac{1}{2}$. In addition, all LDPC codes considered in this section are simulated over the Binary Additive White-Gaussian-Noise (BI-AWGN) channel.

Table 2. Summary of simulation settings with 64-bit block length code

Parameter	Value
The number of variable nodes	64
The number of check nodes	32
SNR (dB)	3
Noise model	BI-AWGN
$N_{pop,max}$	50
$N_{ite,max}$	200

In Figure 8, we assess the changing of the BLER as a function of the population index in the BI-AWGN channel with $SNR = 3\ dB$. Simulation results indicate

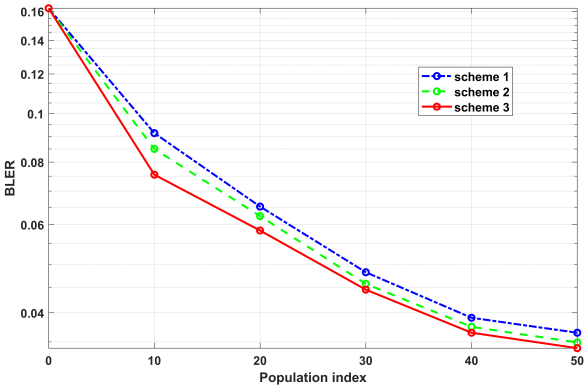

Figure 8. BLER as a function of population index with 64-bit block length code

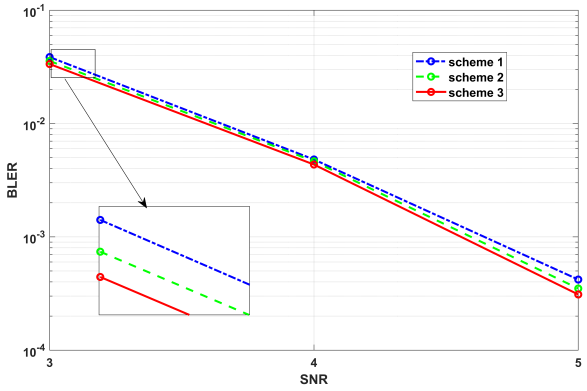

Figure 9. BLER as a function of SNR in BI-AWGN channel with 64-bit block length.

that our proposed scheme (scheme 3) outperforms scheme 1 and scheme 2 with BLER gains up to 11.1% and 4%, respectively. This is because the min-sum decoding algorithm has better decoding performance and a more optimized process due to eliminating worse fitness c ases, s pecifically by id entifying and removing girth-4 cycles in each parity check matrix H. The detailed comparison of the BLER gain obtained with the proposed scheme in comparison with the conventional schemes is summarized in Table 3. It is noted that the performance of a genetic algorithm is measured in terms of the number of required fitness function evaluations until the optimum is found or approximated with the desired accuracy. For the genetic algorithm to find the best possible solution (optimal solution), we need to run it numerous repetitions. This increases the chance of the GA discovering the best option. Research suggests that at least 25 repetitions are beneficial, with 50 or 100 repetitions being even more recommendable [21]. Therefore, to achieve a balance between performance and computational cost, the population size, which represents repetitions, is chosen at 50.

Table 3. BLER gain obtained with the proposed scheme according to 64-bit block length.

Scheme	SNR	Population index	Gain
1	3 dB	50	11%
2	3 dB	50	4%

BLER versus a range of SNR. In this subsection, we evaluate the BLER performance for a range of SNR in the BI-AWGN channel with a 64-bit block length code. As depicted in Figure 9, we can observe that as SNR increases, the BLER obtained with our proposed schemes is significantly improved when compared with the original work in [18]. This improvement comes from the utilization of the min-sum decoding algorithm, which is known as a soft-decision message-passing algorithm that accepts the probability of each received bit as input, to enhance the performance of the LDPC code [20]. In particular, Table 4 depicts the BLER gain of scheme 3 compared with scheme 1 and 2, respectively, for different values of SNR ratio.

Table 4. BLER gain obtained with the proposed scheme for a range of SNR according to 64-bit block length code.

Scheme	Population index	3 dB	4 dB	5 dB
1	50	11%	17.7%	33.3%
2	50	4%	8%	12.2%

4.2. Scenario of 128-bit block length code

BLER versus the number of populations. In the subsection, we evaluate the performance of our proposed LDPC code design scheme with 128-bit block length. To be specific, we consider the LDPC code with the number of variable nodes $n = 128$, the number of check nodes $m = 64$, and code rate $R_c = \frac{1}{2}$. The simulation settings are summarized in Table 5. As shown in Figure 10, scheme 3 obtains a significant i mprovement i n t erms of BLER compared to the benchmarked schemes at $SNR = 3dB$ for a range of population index.

BLER versus a range of SNR. We also evaluate the BLER according to different v alues o f S NR i n t he BI-AWGN channel with 128-bit block length. Figure 11 indicates that as SNR increases, the BLER of our proposed scheme is significantly i m proved w h en c o mpared w i th the benchmarked schemes.

We also compare the performance of our proposed LDPC code design in the scenario of 64-bit block-length code and the scenario of 128-bit block-length code with different SNR levels. As shown in Figure 12,

Table 5. Summary of simulation settings with 128-bit block length code.

Parameter	Value
The number of variable nodes	128
The number of check nodes	64
SNR (dB)	3
Noise	BI-AWGN
$N_{pop,max}$	50
$N_{ite,max}$	200

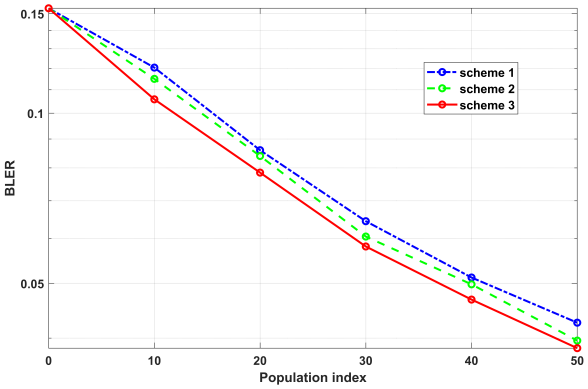

Figure 10. BLER as a function of population index with 128-bit block length.

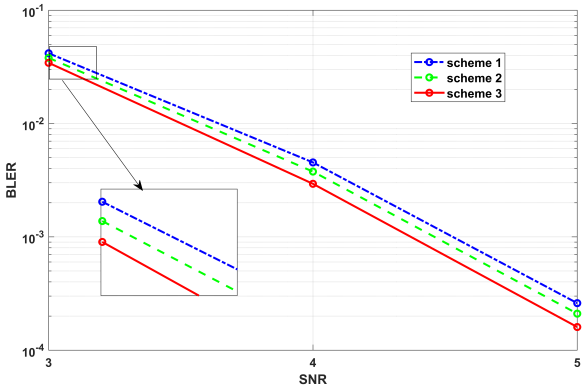

Figure 11. BLER with different schemes as a function of SNR in AWGN channel with 128-bit block length.

we can see that the BLER performance of the 128-bit block length code exhibits a substantial enhancement over that of the 64-bit block length code. In particular, Table 6 highlights the BLER gain of the 128-bit block-length code compared to the 64-bit block-length code. This observation aligns with the theoretical framework proposed by Robert Gallager in his seminal work during the 1960s [6] that the typical decoding error probability decreases exponentially with increasing coding block length.

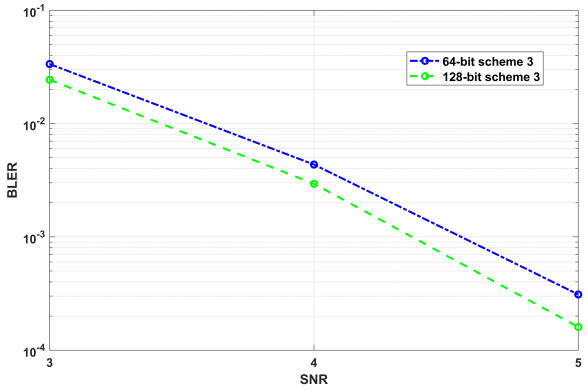

Figure 12. BLER with our proposed code design as a function of SNR according to different block lengths.

Table 6. BLER gain obtained with our proposed scheme according to 128-bit block length and 64-bit block length.

SNR	3 dB	4 dB	5 dB
Gain	27%	32%	48%

5. CONCLUSION

In this paper, we proposed an improved LDPC code design scheme that is built based on the conventional genetic algorithm. In particular, first, for the LDPC decoding algorithm, we applied the min-sum decoding algorithm to improve the decoding performance. Second, we incorporated a module for detecting and removing girth-4 within the parity-check matrix. This module functions as a penalty function, prioritizing the elimination of these unfavorable configurations. We then evaluated the BLER performance of the proposed LDPC code design scheme over the BI-AWGN channel with different short block length scenarios, including 64-bit and 128-bit block length. Extensive simulation results indicated that our proposed design achieves a significant improvement in terms of the BLER reaching 11% compared to the conventional schemes.

As part of future work, we intend to apply the self-adaptive genetic algorithm, which can further improve the performance of the genetic algorithm for the LDPC design scheme.

References

[1] Nguyen-Van-Thanh, Loc and Do-Duy, Tan (2023) *Efficient Genetic Algorithm-based LDPC Code Design for IoT Applications* (International Conference on System Science and Engineering (ICSSE)), 598–603.

[2] Siddiqui, M. U. A., Abumarshoud, H., Bariah, L., Muhaidat, S., Imran, M. A. and Mohjazi, L. (2023) *Urllc in beyond 5g and 6g networks: An interference management perspective* (IEEE Access), 11, 54639-54663.

[3] Yue, C., Miloslavskaya, V., Shirvanimoghaddam, M., Vucetic, B. and Li, Y. (2023) *Efficient decoders for short block length codes in 6G URLLC* (IEEE Communications Magazine), 61(4), 84-90.

[4] Moon, T. K. (2020) *Error correction coding: mathematical methods and algorithms* (John Wiley & Sons).

[5] Yang, K. and Du, W. (2022, November) *LLDPC: A low-density parity-check coding scheme for LoRa networks* (In Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems), 193-206.

[6] Gallager, R. (1962) *Low-density parity-check codes* (IRE Transactions on information theory), 8(1), 21-28.

[7] Roberts, M. K. and Anguraj, P. (2021) *A comparative review of recent advances in decoding algorithms for low-density parity-check (LDPC) codes and their applications* (Archives of Computational Methods in Engineering), 28(4), 2225-2251.

[8] Chander, B. and Gopalakrishnan, K. (2023) *An ECC-based enhanced and secured authentication protocol for IoT and cloud server* (International Journal of Communication Networks and Distributed Systems), 29(4), 407-425.

[9] Richardson, T. and Kudekar, S. (2018) *Design of low-density parity check codes for 5G new radio* (IEEE Communications Magazine), 56(3), 28-34.

[10] Liu, J. and Feng, Q. (2021) *A miniaturized LDPC encoder: Two-layer architecture for CCSDS near-Earth standard* (IEEE Transactions on Circuits and Systems II: Express Briefs), 68(7), 2384-2388.

[11] Ebada, M., Elkelesh, A., Cammerer, S. and ten Brink, S. (2018, May) *Scattered EXIT charts for finite length LDPC code design* (In 2018 IEEE International Conference on Communications (ICC)), 1-7.

[12] Van Wonterghem, J., Alloum, A., Boutros, J. J. and Moeneclaey, M. (2016, November) *Performance comparison of short-length error-correcting codes* (In 2016 Symposium on Communications and Vehicular Technologies (SCVT)), 1-6.

[13] Wijekoon, V. B., Viterbo, E., Hong, Y., Micheloni, R. and Marelli, A. (2019) *A novel graph expansion and a decoding algorithm for NB-LDPC codes* (IEEE Transactions on Communications), 68(3), 1358-1369.

[14] Ferraz, O., Subramaniyan, S., Chinthala, R., Andrade, J., Cavallaro, J. R., Nandy, S. K. and Falcao, G. (2021) *A survey on high-throughput non-binary LDPC decoders: ASIC, FPGA, and GPU architectures* (IEEE Communications Surveys & Tutorials), 24(1), 524-556.

[15] Shao, S., Hailes, P., Wang, T. Y., Wu, J. Y., Maunder, R. G., Al-Hashimi, B. M. and Hanzo, L. (2019) *Survey of turbo, LDPC, and polar decoder ASIC implementations* (IEEE Communications Surveys & Tutorials), 21(3), 2309-2333.

[16] Katoch, S., Chauhan, S. S. and Kumar, V. (2021) *A review on genetic algorithm: past, present, and future* (Multimedia tools and applications), 80, 8091-8126.

[17] Zhou, X., and Shi, W. (2024) *Research on the optimisation of whitelisting technology for network firewall in industrial control system using genetic algorithm* (International Journal of Communication Networks and Distributed Systems), 30(1), 30-41.

[18] Elkelesh, A., Ebada, M., Cammerer, S., Schmalen, L. and Ten Brink, S. (2019) *Decoder-in-the-loop: Genetic optimization-based LDPC code design* (IEEE access), 7, 141161-141170.

[19] Elkelesh, A., Ebada, M., Cammerer, S. and Ten Brink, S. (2019) *Decoder-tailored polar code design using the genetic algorithm* (IEEE Transactions on Communications), 67(7), 4521-4534.

[20] Rao, K. D. (2015) Channel coding techniques for wireless communications (Berlin, Germany: Springer India).

[21] Kramer, O. and Kramer, O. (2017) *Genetic algorithms* (Springer International Publishing).

[22] Eremia, M., Liu, C. C. and Edris, A. A. (2016) *Advanced solutions in power systems: HVDC, FACTS, and Artificial Intelligence* (John Wiley & Sons).