

## ORIGINAL ARTICLE

# Novel construction of quasi-cyclic low-density parity-check codes with variable code rates for cloud data storage systems

Vairaperumal Bhuvaneshwari  | Chandrapragasam Tharini

Department of Electronics and Communications Engineering, B. S. Abdur Rahman Crescent Institute of Science & Technology, Ringgold standard institution, Chennai, Tamil Nadu, India

## Correspondence

Vairaperumal Bhuvaneshwari,  
Department of Electronics and Communications Engineering, B. S. Abdur Rahman Crescent Institute of Science & Technology, Ringgold standard institution, Chennai, Tamil Nadu, India.  
Email: [pvbunaperumal@gmail.com](mailto:pvbunaperumal@gmail.com)

## Abstract

This paper proposed a novel method for constructing quasi-cyclic low-density parity-check (QC-LDPC) codes of medium to high code rates that can be applied in cloud data storage systems, requiring better error correction capabilities. The novelty of this method lies in the construction of sparse base matrices, using a girth greater than 4 that can then be expanded with a lift factor to produce high code rate QC-LDPC codes. Investigations revealed that the proposed large-sized QC-LDPC codes with high code rates displayed low encoding complexities and provided a low bit error rate (BER) of  $10^{-10}$  at 3.5 dB  $E_b/N_0$  than conventional LDPC codes, which showed a BER of  $10^{-7}$  at 3 dB  $E_b/N_0$ . Subsequently, implementation of the proposed QC-LDPC code in a software-defined radio, using the NI USRP 2920 hardware platform, was conducted. As a result, a BER of  $10^{-6}$  at 4.2 dB  $E_b/N_0$  was achieved. Then, the performance of the proposed codes based on their encoding–decoding speeds and storage overhead was investigated when applied to a cloud data storage (GCP). Our results revealed that the proposed codes required much less time for encoding and decoding (of data files having a 10 MB size) and produced less storage overhead than the conventional LDPC and Reed–Solomon codes.

## KEYWORDS

bit error rate, cloud data storage, erasure correction capability, low-density parity-check codes, parity-check matrix, quasi-cyclic low-density parity-check codes

## 1 | INTRODUCTION

Big data involves handling massive volumes of data within petabytes (PB), exabytes (EB), and, in the future, zettabytes (ZB). Currently, the available cloud storage infrastructure makes it possible to store big data [1] from sources, such as Facebook (FB), Twitter, Instagram, and so on. Local physical storage provides faster data

retrieval, but it is less economical compared with cloud storage. By virtualizing resources (such as CPU, RAM, and software) and employing parallel processing, cloud computing lowers the cost of purchasing, deploying, and maintaining in-house data storage infrastructures. Hence, major IT leaders have adopted virtual data cloud storage for storing their business data (pay-per-use concept), such as Microsoft Inc., which uses Microsoft Azure [2, 3],

Amazon, which uses Amazon Web Services, and Google, which uses the Google Cloud Platform (GCP), etc. Replication and triplication coding technologies have also been used to achieve 100% data availability in cloud storage, such as Google File Systems [4], Amazon Dynamo [5], and Hadoop Distributed File System [6]. Triple replication offers high data availability and better fault tolerance as three copies of each data record are created and dispersed over geographically located storage servers. Still, it has proven to be an expensive technique, as its storage overhead increases heavily (three times the normal overhead). Finally, an erasure coding technique was introduced, which surpassed all disadvantages of replication methods. Facebook (HDFS Raid) [7, 8], Microsoft Windows Azure [2], and Google Colossus [9] all use the erasure coding scheme as it offers high data availability and less than half of the storage overhead produced by the triple replication scheme [8]. The erasure coding technique is classified into maximum distance separable (MDS) and non-MDS codes.

A  $(n, k)$  linear code that meets the singleton bound is defined as an MDS code. MDS codes must meet the following requirements: (a) have a maximum possible distance between the codewords, where " $d_{\min}$ " must always be either less or equal to " $n - k + 1$ " and (b) have a set of  $(n - k)$  parity-check matrix (PCM) columns that should always be linearly independent. Based on these requirements, an  $(n, k)$  MDS erasure code, such as the Reed–Solomon (RS) code, divides an original data file into  $k$  equal-sized data blocks and encodes them into  $n$ -coded blocks. The remaining " $m = (n - k)$ " blocks are the parity blocks. To this end, RS codes are space-optimal codes with an overhead factor of  $f = 1$ .

When data center failure occurs, the recovery of the original file is possible using a required number of parity blocks. Because RS codes can tolerate a total loss of up to " $m$ " parity blocks, they require many or all data blocks for the original data file reconstruction. These codes achieve the singleton bound [10], which are optimal space-efficient codes [11] employing the Galois field (GF) arithmetic operations for encoding and decoding computations. It is well known that adding GF arithmetic operations involves simple XOR operations and multiplication and division are more complex, as they use discrete logarithmic tables. However, they also involve high computational complexities, making them unsuitable for data storage applications.

Non-MDS erasure codes, such as the locally repairable codes (LRC), necessitate more than " $k$ " blocks for the original data reconstruction. For example, Microsoft Windows Azure storage [2] uses  $(6, 2, 2)$  LRC codes for storage. Here local parity blocks are used for a single failure reconstruction, which increases the storage overhead,

whereas, for  $\geq 2$  failures, global parity blocks are used for reconstruction. Nevertheless, research on finding another class of erasure codes for data storage systems with low encoding/decoding complexities and less storage overhead has become an ongoing hot topic.

Recently, low-density parity-check (LDPC) [11] codes are being considered for big data storage applications [3, 12–14], as their encoding and decoding procedures only involve XOR operations. LDPC codes are linear block codes characterized by sparse PCM. Because sparseness refers to a smaller number of one's in the PCM, this PCM sparse nature decides the LDPC code's performance. Moreover, due to the sparse nature of LDPC codes, the speed of encoding and decoding are quick compared with the complex RS codes. Thus, there is still a need to generate sparse LDPC PCM codes for data storage applications (Table 1).

Based on the above background, our major contributions, as shown in this paper, are as follows:

1. Proposing a novel algorithm to design a sparse base PCM that can be further expanded using the lift factor " $\ell$ " to generate large-sized structured systematic QC-LDPC codes. The proposed large LDPC codes should have medium to high code rates, similar to the proposed base matrix, and should be specifically designed for deployment on cloud data storage systems.

TABLE 1 Summary of abbreviations used in this paper

Abbreviations	Description
LDPC codes	Low-density parity-check codes
QC-LDPC codes	Quasi-cyclic LDPC codes
PCM	Parity-check matrix
SDR	Software-defined radio
NI USRP	National Instruments Universal Software Radio Peripheral
MDS codes	Maximum distance separable codes
FPGA	Field-programmable gate array
DAC	Digital-to-analog converter
ADC	Analog-to-digital converter
DOR	Decoding overhead ratio
ADC	Analog-to-digital converter
MDS codes	Maximum distance separable codes
XOR	Exclusive OR
GF	Galois field
BPSK	Binary phase-shift keying
QPSK	Quadrature phase-shift keying
AWGN	Additive white Gaussian noise
BER	Bit error rate

2. Proposing newly constructed codes (2576, 2080), (2100, 2048), (648, 512), (1024, 832), (880, 512), and (88, 56).

Simulations were performed on newly proposed LDPC codes using the binary phase-shift keying (BPSK) modulation and the additive white Gaussian noise (AWGN) channel, after which we evaluated their bit error rate (BER) performances. The results showed that the proposed LDPC codes displayed low BER.

3. Implementing a software-defined radio (SDR) transceiver using the NI LABVIEW software with proposed QC-LDPC code-based wireless communication systems on an NI universal software radio, Universal Synchronous Radio Peripheral (USRP) 2920 hardware platform. Implementation results verified that our proposed codes had better BER than conventional LDPC codes using the quadrature phase-shift keying modulation scheme.
4. Encoding and decoding speeds were determined and compared with their counterparts. This paper also demonstrated an example of applying the proposed QC-LDPC codes for storing video files on a GCP, after which a storage overhead was computed. The results showed that the coding speeds were faster and the storage overhead generated was lower, making the proposed codes the best eligible candidates for data storage systems.

## 2 | RELATED RESEARCH WORK

### 2.1 | Literature review

In a previous study [14], the author proposed a new LDPC code design called the expancodes (a combination of LDPC-convolution codes) for big data storage systems, such as distributed storage systems (DSS). These codes had low computational complexities and repair traffic by keeping the row weight of the PCM constant or low, thereby increasing reliability.

In another study [15], the author used erasure codes for DSS and proved that they have a higher mean time to failure than replication-based systems with similar storage overheads and bandwidth requirements. In another study [16], the author analyzed the performance of three LDPC codes: systematic, nonsystematic, and irregular repeat accumulate LDPC codes for wide-area network storage applications. These codes were constructed using Monte Carlo techniques (brute force).

Although a study [17] reported that a distributed storage cluster uses LDPC codes that show high reliability

and less storage overhead, making it suitable for storage applications, another [18] reported that LDPC codes were used for cloud storage frameworks, requiring high-rate codes, because large amounts of data were being transmitted. In a previous study [19], the author developed small-sized LDPC codes with  $n$  and  $m < 100$ . Although these codes had optimal properties, the author mentioned that for  $n > 2$ , LDPC-decoding techniques could not decode optimally. Finally, in another study [20], the author used systematic LDPC codes for storage applications because the recovery of original blocks does not require decoding. As a result, the structured format of the LDPC codes reduced the memory usage and complexity costs. Based on these studies, we designed systematic and structured LDPC codes using the proposed algorithm for cloud data storage systems.

Apart from using the proposed systematic and structured LDPC codes in data storage systems, good candidates that can be considered for high-rate applications such as wireless communications and even 5G systems exist. Hence, many researchers have worked on new QC-LDPC codes, based on belief propagation decoding algorithms. Furthermore, they implemented them in hardware, such as a layered QC-LDPC decoder architecture in literature [21], cyclically coupled QC-LDPC codes, and their decoder architecture, producing a model that offers high throughput and excellent error performance. In this paper, a faster, simpler, and more reliable approach to evaluate the performance of the proposed LDPC codes in terms of BER is presented using an SDR platform with NI USRP 2920 as the hardware device.

SDR is a programmable hardware platform capable of supporting software implementations of wireless communication protocols for physical layers [22]. Conversely, while a field-programmable gate array (FPGA) chip supports the USRP hardware, NI USRP 2920, it is a device that can be programmed using LABVIEW. To this end, a complete wireless communication system was prototyped using the LABVIEW software, which includes LABVIEW programs for the LDPC encoder, modulator, decoder, and demodulator. Subsequently, an SDR transceiver was implemented using the LABVIEW software with the proposed QC-LDPC codes on the NI USRP 2920 hardware platform, and BER performance versus  $E_b/N_0$  of the proposed LDPC codes was analyzed and compared with conventional QC-LDPC codes for BPSK and QPSK modulation techniques.

### 2.2 | Theoretical background

A basic digital communication system is shown in Figure 1, in which the input data are sent from the source

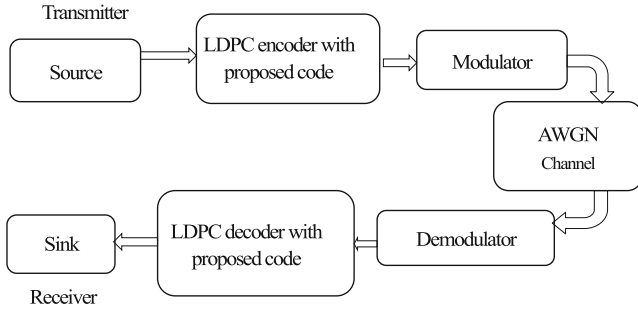


FIGURE 1 Schematic showing the digital communication system model

to the proposed DPC encoder, which has an inbuilt proposed QC-LDPC code. The encoded blocks were then BPSK modulated and sent over an AWGN channel. As a result, a probability density function of the AWGN channel with mean “ $\mu$ ” and variance “ $\sigma^2$ ” was obtained, as shown below (1):

$$f(\omega) = \frac{1}{\sqrt{2\pi\sigma_\omega^2}} \exp\left(\frac{-\omega^2}{2\sigma_\omega^2}\right), \quad (1)$$

where noise “ $\omega$ ” is a Gaussian random process and  $N_0$  is the one-sided power spectral density.

In the BPSK modulation scheme, the carrier phase was shifted  $180^\circ$  for one data symbol and was not shifted for the other data. The binary data symbol, therefore, lasted for  $T_s$  secs, as shown below (2):

$$S(t) = A \cos(2\pi f_c t + \phi), 0 < t < T_s. \quad (2)$$

On the receiver side, however, original data were received after BPSK demodulation and LDPC decoding using the proposed QC-LDPC code. A belief propagation decoding algorithm was used for decoding.

Subsequently, every error probability in the BPSK modulated system was calculated using the expression below:

$$\text{BER} = \frac{1}{2} \text{erfc} \sqrt{\frac{E_b}{N_0}}, \quad (3)$$

where  $E_b$  is energy per information bit and  $N_0$  is the noise power spectral density.

LDPC codes are a class of linear block error correcting codes discovered by R. Gallager in the early 1960s during his PhD thesis preparation. Conventional LDPC codes are defined by sparse PCM and constructed using random construction methods. Their decoding process is complex due to the lack of a proper structure. The other

type of LDPC code is structured LDPC codes, known as quasi-cyclic LDPC codes, which are much easier to encode and decode. Considering an  $(n, k)$  LDPC code having a total block size,  $n = 12$ , number of data blocks,  $k = 8$ , and number of parity blocks,  $m = n - k = 4$ , as shown in Equation 6:

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

Each PCM “ $\mathbf{H}$ ” was visualized using a bipartite graph, also known as a tanner graph. The tanner graph of the  $\mathbf{H}$  matrix in (3), consisting of  $n = 12$  variable nodes and  $m = n - k = 4$  parity nodes, is shown in Figure 2. The left-side nodes are “ $k$ ” message nodes, where  $k = 8$ , and are called data blocks. However, the right-side nodes are called “ $m$ ” parity nodes or parity blocks of sizes  $m = n - k = 4$ .

Encoding was conducted, the connection of a simple XOR on the left nodes to the right nodes. The encoded blocks are shown in Figure 2. To this end, a code rate “ $R$ ” of LDPC codes is given by the expression below (5):

$$R = k/n. \quad (5)$$

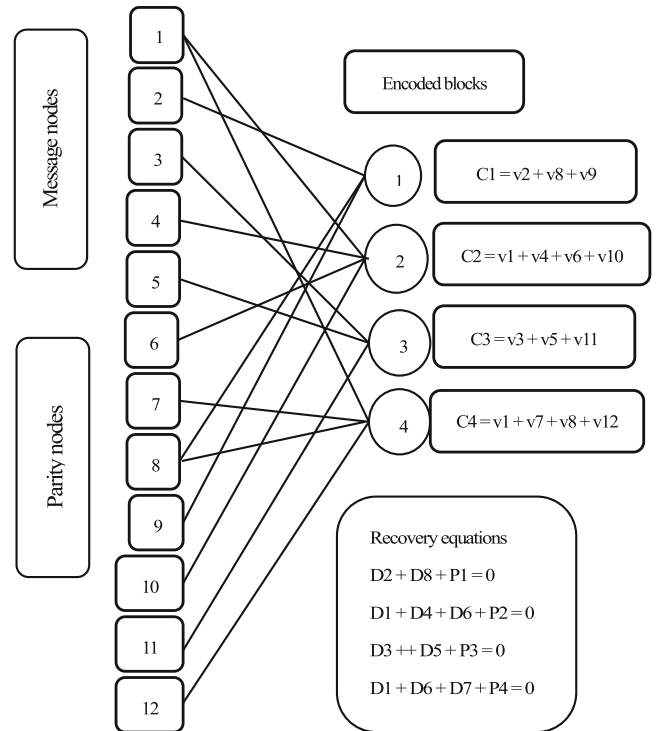


FIGURE 2 Tanner graph showing a PCM with  $n = 12$  variable nodes and  $m = 4$  parity nodes

In contrast, decoding, which is based on the same Tanner graph used for encoding, was also conducted. Because it is an iterative procedure, it is named an iterative decoding algorithm [19, 23]. In this procedure, suppose message block 1 ( $v_1$ ) got erased due to external environment failure conditions; message block  $v_1$  can be obtained using the fourth check node instead of the first check node because all other parity and data blocks were available in the equation. This iterative decoding procedure continued until all erased data blocks or required erased data blocks were repaired. However, when the decoder failed to find corresponding parity blocks to obtain the erased data block, it could not retrieve the original data file.

The remainder of this paper is organized as follows: In Section 3, we proposed a new method for constructing randomly generated base matrices without Cycle 4, which could be further expanded using the lift factor “ $l$ ” to build large-sized QC-LDPC codes with high code rates for data storage applications. This section also includes the structure of the proposed QC-LDPC code and the flowchart of the proposed algorithm.

In Section 4, we compared the BER performance of the proposed codes to those of their counterparts. The implementation results of the proposed codes in the SDR-NI USRP 2920 hardware platform are also presented. Section 5 briefly described the Google Cloud data storage framework, encoding–decoding speeds of the proposed QC-LDPC codes, and the storage overhead. Finally, conclusions were made in Section 6. Table 2 presents other notations used in this paper.

TABLE 2 Other notations used in this paper

Notations	Explanation
$\mathbf{H}_p$	$\mathbf{H}$ matrix with identity diagonal part
$\mathbf{H}_i$	$\mathbf{H}$ matrix with information part
$\epsilon$	Erasur correction capability
$S$	Storage overhead
$\eta$	Storage efficiency
DOR	Decoding overhead ratio
$\mathbf{r}_w, \mathbf{c}_w$	Row weight and column weight of $\mathbf{H}$ matrix
$\mathbf{H}_b$	Base matrix
$l$	Lift factor
$r_n, c_n$	Number of rows and columns
$S_r$	Sum of one's in each row of $\mathbf{H}_b$ matrix
$S_c$	Sum of one's in each row of $\mathbf{H}_b$ matrix
$\epsilon$	Erasur correction capability

### 3 | PROPOSED ALGORITHM TO CONSTRUCT RANDOMLY GENERATED BASE MATRICES FOR FURTHER EXPANSION USING A LIFT FACTOR TO BUILD LARGE-SIZED QC-LDPC CODES HAVING VARIABLE CODE RATES FOR CLOUD DATA STORAGE SYSTEMS

In this section, we propose a novel algorithm to construct a highly sparse base PCM  $\mathbf{H}_b$ , which can be further expanded using lift factor “ $l$ ” to build large-sized QC-LDPC codes of medium- to high code rates ( $0.6 \leq R \leq 0.9$ ). The proposed algorithm's novelty lies in the base matrix  $\mathbf{H}_b$  design. The steps are as follows:

Step 1) Initialize the proposed base “PCM” “ $\mathbf{H}_b$ ” parameters, such as the required block length “ $L$ ” and code rate “ $R$ ”

Let the number of rows be “ $r_n$ ,” number of columns be “ $c_n$ ,” block length (length of data blocks in bits) be “ $L$ ,” and code rate be  $R$ , for example,  $R = \{0.67/0.79/0.81/0.97\}$ .

Choose number of columns that is always greater than the number of rows.

$$c_n > r_n. \quad (6)$$

Step 2) Choose the row weight “ $\mathbf{r}_w$ ” to be a least odd prime number (so that the Tanner graph has larger girth) and generate a random binary matrix “ $\mathbf{B}$ ” with the chosen “ $\mathbf{r}_w$ .” Fill the remaining locations in the “ $\mathbf{B}$ ” matrix with zeroes. Choose

$$\mathbf{r}_w = \begin{cases} 3, & \text{else} \\ 6n + 1, & n \text{ is natural number,} \end{cases} \quad (7)$$

where row weight  $\mathbf{r}_w$  indicates the number of one's (1's) in each row of the base matrix.

This step outputs the “ $\mathbf{H}_L$ ” matrix, which must be checked for linearly independent rows and columns. If independent, go to Step 3, else try making the matrix linearly independent (LI). Suppose the matrix cannot be made LI, then, discard it and return to Step 1.

Step 3) Eliminate short cycles (Girth 4) in the square matrix “ $\mathbf{H}_L$ ” obtained in Step 3.

The base matrix must be free of four cycles to achieve a high performance. The shortest cycle in a Tanner graph is Cycle 4, and the girth indicates the shortest cycle.

Figure 3 and Equation 8 display a PCM with three short cycles of four.

If there are more cycle of 4's in a base matrix, the expanded matrix will contain shorter cycles. Hence, the performance of the expanded QC-LDPC codes reduces based on the BER and decoding complexities [20].

An example of a PCM with three short cycles of 4's is shown in Equation 8:

$$\mathbf{H} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}. \quad (8)$$

Hence, it is necessary to construct a base PCM with a large girth, that is,  $g > 4$ .

Step 4) Construct proposed base matrix  $\mathbf{H}_b$  from the  $\mathbf{H}_L$  matrix obtained in Step 3.

- i. Compute sum of one's in each row, denoted as  $\mathbf{S}_r$ , such that it is always a minimum odd number. Delete the rest of the rows with even sums. This step removes the possibility of double-bit error [24].

$$\mathbf{S}_r = \text{Min odd value}\{2p + 1, p \geq . \quad (9)$$

- ii. Compute the sum of one's in each column  $\mathbf{S}_c$ , such that it is always a minimum odd number. Keep all zero columns and delete the remaining ones with an even sum.

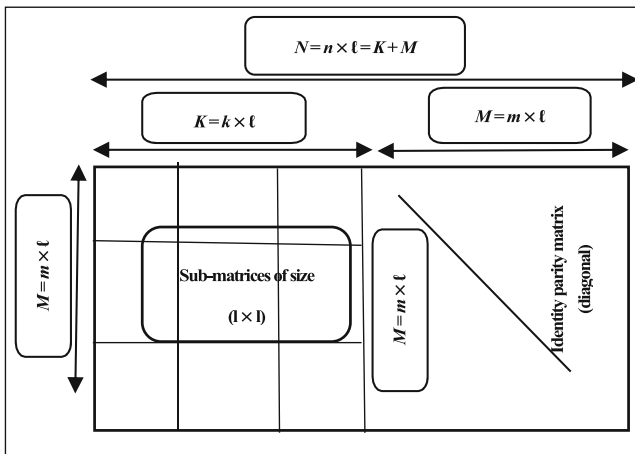


FIGURE 3 Structure of the QC-LDPC code constructed using the proposed  $\mathbf{H}_b$  matrix ( $\ell$  is the lift factor, e.g., 3, 4, 5)

$$\mathbf{S}_c = \text{Min odd value}\{2q + 1, q \geq 1. \quad (10)$$

The reduced matrix obtained from Step 5 is the expected base PCM,  $\mathbf{H}_b$ , which is used to construct the expanded H matrix or larger QC-LDPC code “ $\mathbf{H}$ .” Lifting is applied on  $\mathbf{H}_b$  to generate an expanded “ $\mathbf{H}$ ” matrix (for large block lengths). It is done based on lift factor “ $\ell$ ,” which must always be  $\geq 2$ .

Step 5) Build an expanded and structured LDPC code “ $\mathbf{H}$ ” matrix or QC-LDPC code of size  $(M \times N)$  from the  $\mathbf{H}_b$  matrix.

$$\mathbf{H} = [\mathbf{H}_i \mathbf{H}_p], \quad (11)$$

where  $\mathbf{H}_i$  is the information or replacement matrix constructed by replacing each entry of the base matrix  $\mathbf{H}_b$  with its corresponding submatrix.

Each entry,  $\mathbf{O}$  of  $\mathbf{H}_i$  is replaced with a null matrix of size  $(\ell \times \ell)$ , each “1” is replaced with an identity matrix of size  $(\ell \times \ell)$ , and  $\mathbf{H}_p$  is the identity matrix (diagonal).

### 3.1 | Structure of the proposed QC-LDPC code

Figure 3 displays the structure of the QC-LDPC code “ $\mathbf{H}$ ,” constructed using the proposed  $\mathbf{H}_b$  matrix. The base matrix  $\mathbf{H}_b$  had a block length of size “ $n$ ,” expanded by multiplying with lift factor “ $\ell$ .” As a result, a total block length of the H matrix was obtained with size  $(N = n \times \ell = K + M)$ . The number of  $\mathbf{H}_b$  rows denoted “ $k$ ” was also expanded using the lift factor “ $\ell$ .” Thus, the number of rows of  $\mathbf{H}_i$  matrix became  $(K = k \times \ell)$ . Each sub-matrix was of size  $(\ell \times \ell)$ , where every single entry, either 0 or 1, was replaced by its corresponding expanded zero or identity matrix (right circularly shifted identity matrix). The remaining QC-LDPC code “ $\mathbf{H}_p$ ” was the parity part, consisting of the identity matrix, which was of size  $(m \times \ell)$ .

For example, (12) shows the identity matrix substituted with each nonzero element “1” of  $\mathbf{H}_b$  and each zero element with  $\ell = 4$ , having size  $(4 \times 4)$ .

$$I^1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, I^3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \quad (12)$$

The null and identity parity diagonal matrices are given in (13):

$$0 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad H_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (13)$$

The final expanded QC-LDPC code “H” matrix constructed using the proposed base matrix  $H_b$  is shown in (14):

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \dots & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & \dots & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \dots & 0 & 0 & 0 & 1 & 0 & \dots & \dots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (14)$$

Figure 4 A flowchart showing the proposed algorithm to construct base matrices with medium code rates and a lift factor, which can be expanded to build large-sized QC-LDPC codes of high code rates applicable to cloud storage systems

## 4 | SIMULATION RESULTS

This paper focused primarily on designing sparse base matrices  $H_b$  without Cycle 4, which could be further expanded to generate QC-LDPC codes “H” for applications in cloud storage systems.

A basic communication model was used to test the performance of the proposed QC-LDPC codes with multiple code rates,  $R = \{0.67, 0.7, 0.8, 0.9\}$ .

Code length “n” of the base matrix was chosen as (24, 64, 104, 128, 256, 26), with lift factor “l” = {4, 8}, such that large-sized QC-LDPC codes were constructed with block lengths = {88, 648, 880, 1024, 2100, 2576}. Finally, a BER versus  $E_b/N_0$  curve was plotted to show the communication system and error correction performance based on BER analysis.

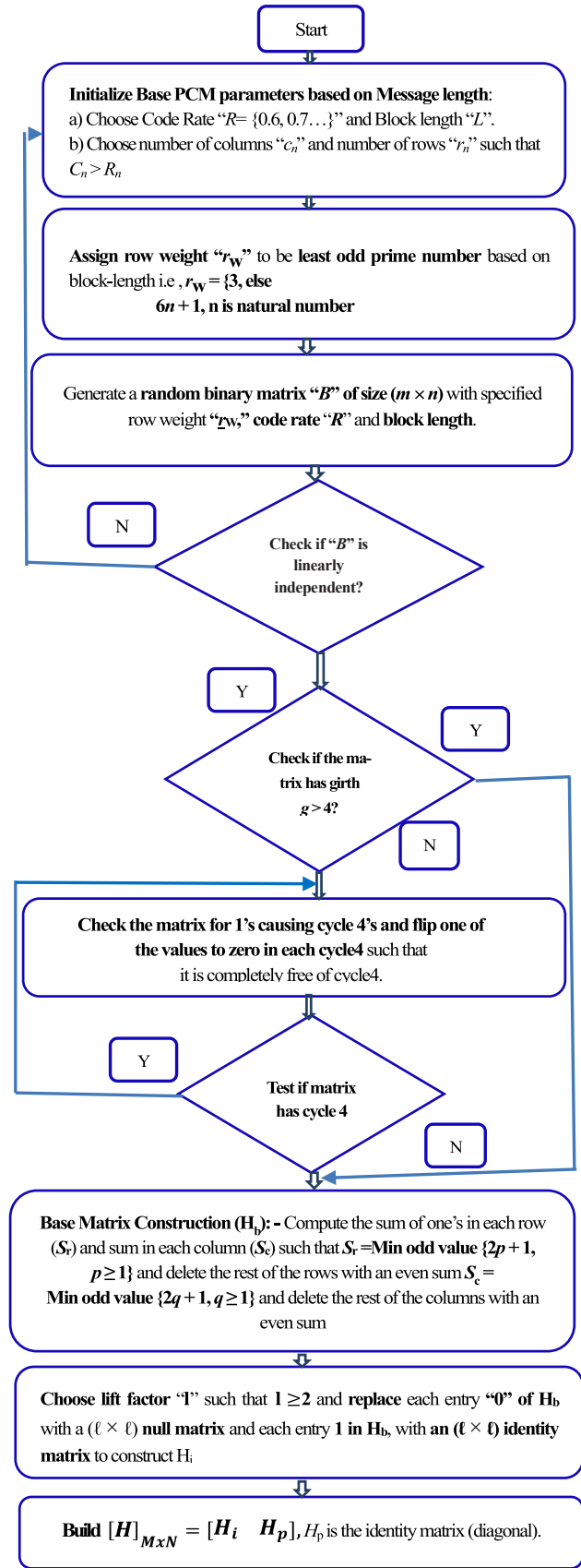


FIGURE 4 Flowchart showing the proposed algorithm

BER results showed that the proposed QC-LDPC codes with high code rates performed better (lowest BER of  $10^{-10}$  at 3.5 dB  $E_b/N_0$ ) than the lower code rate of other proposed QC-LDPC codes (low BER of  $10^{-6}$  at 3 dB  $E_b/N_0$ ). Simulations were performed using the NI LABVIEW software, and newly designed proposed codes were generated using MATLAB. The proposed QC-LDPC codes were inserted into LABVIEW as.ivm files.

Table 3 shows the design specification used for constructing the proposed QC-LDPC codes.

Figure 5 shows the comparative BER performances of the proposed QC-LDPC codes versus other algorithmic QC-LDPC codes with  $R = 0.5(1/2)$  and  $0.8(4/5)$ .

Results from Figure 5 show that the high-rate code ( $R = 4/5 = 0.8$ ) displayed by the proposed QC-LDPC code showed the lowest BER of  $10^{-8}$  at 3 dB  $E_b/N_0$  and the medium-rate (0.6) code achieved the lowest BER of  $10^{-10}$  at 3.5 dB  $E_b/N_0$ . The sparse nature and no cycle four presented in the base matrix  $H_b$  and the expanded proposed QC-LDPC code constructed using the proposed algorithm showed that the BER obtained was far better than the conventional QC-LDPC codes, having similar code rates and sizes.

Conventional QC-LDPC code [25] of high rate (4/5) only attained a BER of  $10^{-7}$  at the same 3.8 dB  $E_b/N_0$  and medium-rate conventional code (0.6) displayed a BER of only  $10^{-6}$  at 3 dB  $E_b/N_0$ . Hence, we conclude that the proposed high-rate QC-LDPC code performed far better than the conventional LDPC code, with higher code rates in terms of BER.

### 4.1 | Implementation and performance evaluation of the proposed QC-LDPC codes in SDR using the NI USRP 2920 hardware

SDR is a radio communication device that uses software for modulation, encoding, demodulation, and decoding

radio signals. It can reconfigure itself at any time using the LABVIEW software. However, USRP 2920 is an SDR within a frequency range of 50 MHz–2.2 GHz, a bandwidth of 20 MHz, and a gain of 12 dB. It allows the experimentation and prototyping of physical layer host-based algorithms, with live signals at similar bands, such as 802.11n and ZigBee, due to its wider frequency coverage, and two radio frequency configurations. These features enable researchers to quickly evaluate their findings using real time data. NI USRP 2920 consists of FPGA, digital-to-analog and analog-to-digital converters, and input/output port.

This paper implemented the proposed QC-LDPC code having a high code rate 0.8(4/5) as well as BPSK and QPSK modulation schemes using USRP 2920 with LABVIEW. Message size was 128 for text and 1024 for images.

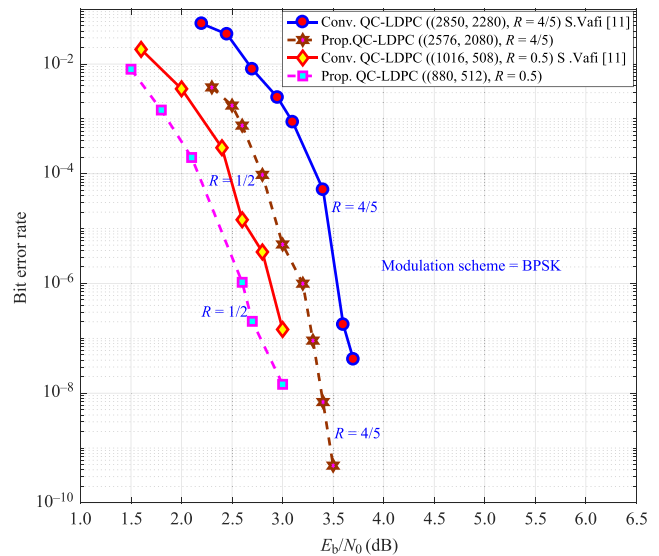


FIGURE 5 BER performances of the proposed QC-LDPC codes versus conventional QC-LDPC codes with  $R = 1/2, 4/5$

TABLE 3 Design specifications for the proposed base matrix and its corresponding expanded QC-LDPC codes

PCM parameters of base matrix $H_b$						
Code Rate <sup>®</sup>	0.5	0.6	0.79	0.8	0.8	0.9
Number of rows	44	8	34	24	62	52
Row weight	3, 5, 7, ...					
Number of Iterations	50					
Code length of $H_b$	64	24	128	104	260	256
Lift factor ( $\ell$ )	8	4	4	8	8	8
LDPC code ( $H$ ) constructed using lift factor, base matrix $H_b$ , and $H_p$ (identity matrix)						
No. of columns ( $K$ )	512	56	512	832	2080	2048
No. of rows of $H$ ( $M$ )	352	32	136	192	496	416
No. of columns postidentity matrix addition ( $N$ )	880	88	648	1024	2576	2100



Figure 6 displays the experimental setup of proposed QC-LDPC codes using USRP 2920.

Table 4 shows the design specifications of the USRP device used in this study.

Figure 7 shows the BER performance of the proposed QC-LDPC codes versus the conventional QC-LDPC codes in SDR using USRP 2920.

From the simulations, our proposed QC-LDPC of high-rate codes (0.8) tested in NI USRP 2920 showed better BER performances of  $10^{-7}$  at 4.2-dB  $E_b/N_0$  using BPSK modulation than the conventional QC-LDPC codes, which displayed a BER of  $10^{-4}$  at 4.5-dB  $E_b/N_0$  for the same code rate. Similarly, the proposed 0.8 code rate displayed a BER of  $10^{-7}$  at 3-dB  $E_b/N_0$  using QPSK modulation than the conventional QC-LDPC codes, which displayed a BER of  $10^{-6}$  at 3.5-dB  $E_b/N_0$  for the same code rate. Thus, it was established that our proposed QC-LDPC code outperformed the conventional QC-LDPC high-rate codes.



FIGURE 6 Experimental setup of the proposed QC-LDPC code, using USRP 2920

TABLE 4 Design specifications of the USRP device used in this study

Specifications	Transmitter specifications IP:192.168.10.1	Receiver specifications IP:192.168.10.1
Carrier frequency	910 MHz	910 MHz
Gain (dB)	12	10
Ac—ive—tri—band antenna	TX1	RX1

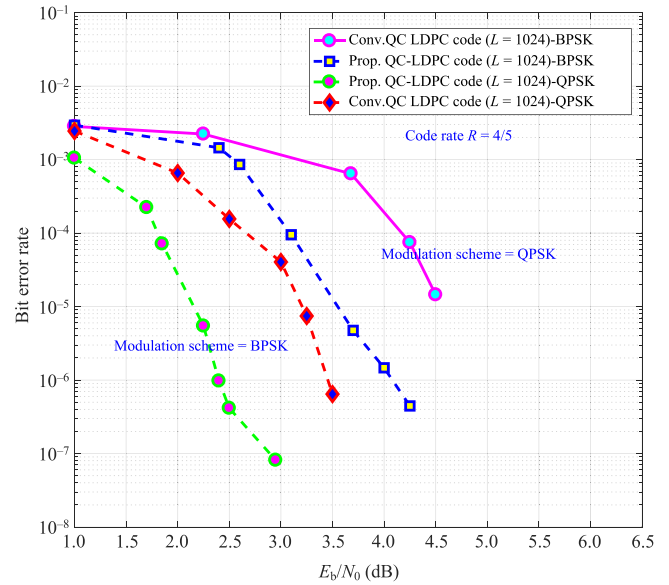


FIGURE 7 BER comparison of the proposed QC-LDPC codes in SDR for high code rate ( $R = 4/5$ )

## 5 | APPLICATION OF THE PROPOSED QC-LDPC CODE-BASED ENCODED DATA IN GCP AND THEIR PERFORMANCE EVALUATION

Google Cloud is well known for providing highly durable data object storage, scaling up to exabytes. Therefore, our study used the Google Cloud's cloud object storage.

### 5.1 | Encoding

Encoding involves computing parity blocks by XOR'ing the data blocks and computing data blocks by XOR'ing the parity blocks. The number of XOR operations involved in this process determines the encoding complexity.

During data loss, a decoding procedure is followed to retrieve all required blocks to decode the lost data block. Encoding the proposed QC-LDPC code involves splitting an original video data file " $V$ " into " $k$ " equal-sized data blocks and generating " $n$ " encoded blocks using QC-LDPC encoding. Then, these encoded blocks are distributed over cloud storage servers across the globe. For example, a proposed LDPC code ( $N = 1024$ ,  $K = 832$ ) generates  $M = 1024 - 832 = 192$  parity blocks using 832 data blocks, such that 1024 blocks are stored in the cloud servers. Figure 8 displays the proposed QC-LDPC code-based encoded data objects stored in GCP buckets.

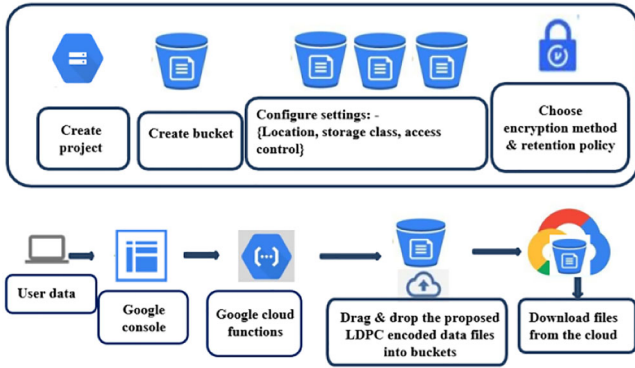


FIGURE 8 View of how data objects are stored in buckets in the cloud

**Steps involved in storing data in cloud:**

- i. A new project named “My First Project” was created in the Google Cloud Console; then 10 new storage buckets with unique names “CloudData\_LDPC\_001-010” were created. All buckets were created with the following configuration: a multi-region storage location, standard storage class, and fine grain access as object-level permissions, after which a Google-managed key for encryption was selected.
- ii. Subsequently, each encoded file using the proposed code was sequentially dragged and dropped into the bucket. For decoding, we downloaded the required files from the bucket in the same sequence and sent them to an LDPC decoder to reconstruct the original file.

In this paper, because systematic QC-LDPC codes were generated using the proposed algorithm, decoding was not required for data reconstruction, but a simple concatenation of data blocks could be used to reconstruct the data file.

**5.2 | LDPC decoding: Recovery equation-based decoding**

The single data block repair process was successfully decoded using the recovery equation-based method [3, 13]. In this method, recovery equations were first formulated, as shown in Figure 1. For example, if (12, 8) the LDPC code with an H matrix, as shown in Figure 1, was used for decoding, then 12 blocks ( $n = 12$ ),  $m = 4$  parity blocks, and  $k = n - m = 8$  data blocks would be obtained. However, if the cloud storage (bucket in the case of Google Cloud) server storing the first data block is lost, the recovery equations need to be used to recover the first

data block. The decoder contacts the bucket where the required blocks are stored to retrieve the respective blocks for reconstructing the original file.

Data sizes considered for performance evaluation in this paper were (a) data size = {10 KB, 100 KB, 1 MB, and 10 MB} files, (b) a basic data file size of 10 KB (text and image), and (c) the a total number of packets of 1024. Figure 9 shows the performance of the proposed QC-LDPC codes versus other algorithmic codes based on encoding speeds.

From Figure 9A, it is evident that the proposed LDPC codes took only 47 s when encoding the 10 MB file, which was much less compared with the 280 s taken by the RS code, and 112 s taken by the conventionally

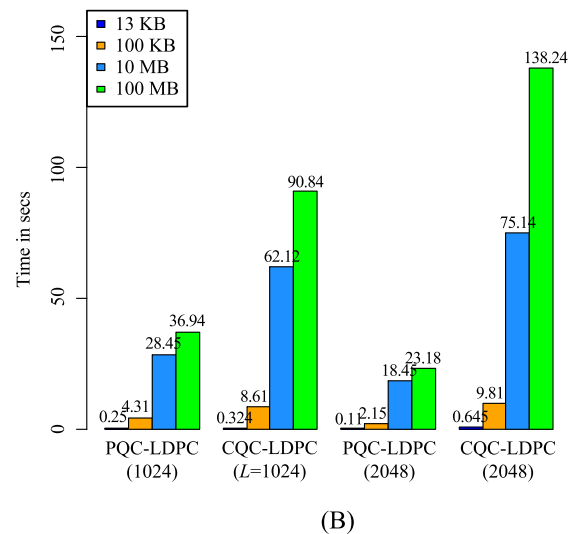
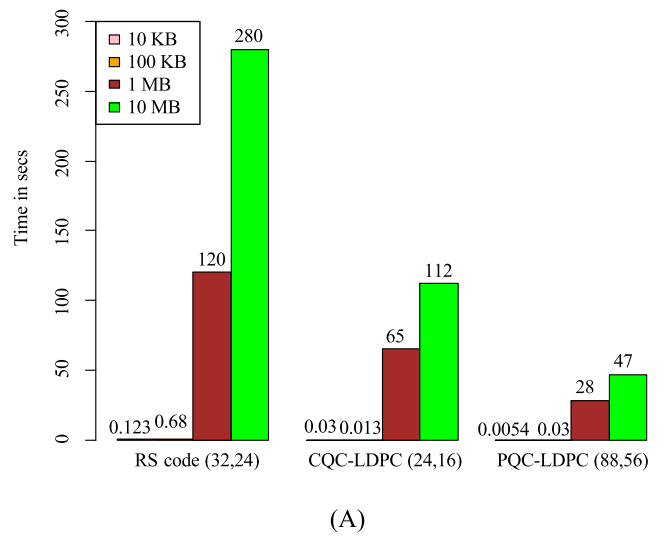


FIGURE 9 (A) Encoding speed in seconds for the proposed QC-LDPC and RS codes and (B) encoding speed in seconds for the proposed QC-LDPC codes of different block lengths versus conventional QC-LDPC codes

structured LDPC code with a high code rate (0.7). However, from Figure 9B, it is evident that the larger block length ( $L = 2048$ ) proposed LDPC codes was faster in terms of encoding speeds compared with shorter block length ( $L = 1024$ ) codes. The proposed code of length ( $L = 2048$ ) required only 23.18 s to encode a 100 MB file compared with the proposed code of length ( $L = 1024$ ), which took only 37 s to encode the same file. Hence, we conclude that the best performing codes were the proposed structured LDPC codes for cloud storage applications.

Figure 10 shows the performance of the proposed LDPC codes versus other algorithmic codes in terms of decoding speeds.

Figure 10A,B reveals that, similar to encoding speeds, proposed QC-LDPC codes of large block lengths outperformed the proposed QC-LDPC codes of shorter block lengths and took much less decoding time than RS codes.

### 5.3 | Storage performance measures and parameters to be considered for the proposed QC-LDPC codes in cloud data storage systems

The performance of the proposed QC-LDPC codes was measured on the basis of certain important factors [26]:

1. Storage overhead ( $S$ )—physical space divided by the actual size of the encoded data stored in the cloud.

$$S = \frac{n}{k}. \quad (15)$$

2. Storage efficiency ( $\eta$ )—number of data blocks divided by the total number of blocks.

$$\eta = \frac{k}{n} \times 100\%. \quad (16)$$

3. Erasure correction capability ( $\varepsilon$ )—number of blocks erased that were tolerated during failures.

$$\varepsilon = (n + k) - fn, \quad (17)$$

where  $f$  is the storage overhead factor.

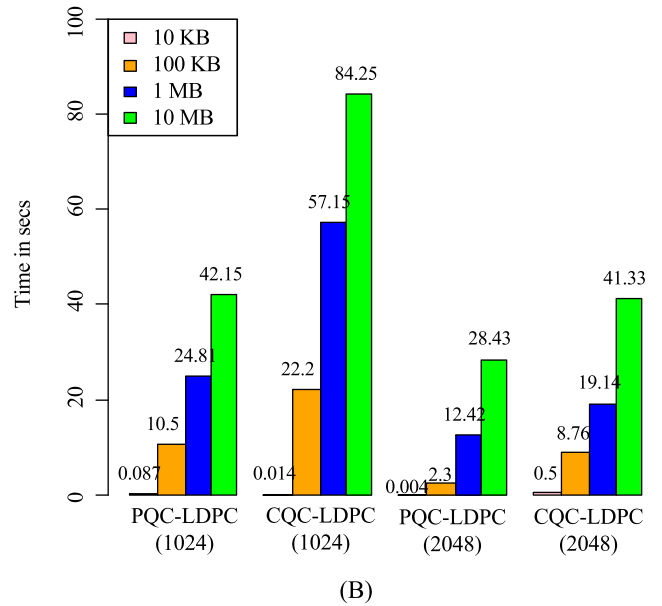
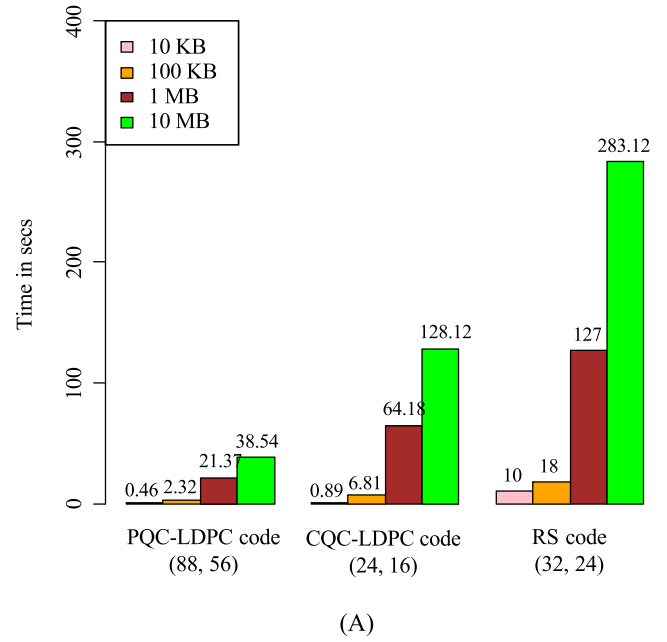


FIGURE 10 (A) Decoding speeds in seconds for the proposed QC-LDPC, conventional QC-LDPC, and RS codes, (B) decoding speed in seconds for the proposed QC-LDPC code of different code lengths and code rates

4. Decoding overhead (DO)—DO was computed using the expression (18) below.

$$\text{DO} = \text{Number of blocks needed to decode} - \text{Actual number of data blocks}. \quad (18)$$

5. Decoding overhead ratio (DOR)—DOR was computed by dividing the number of blocks needed to decode by the number of actual data blocks minus one.

**TABLE 5** Comparative analysis between the decoding performance of conventional LDPC and QC-LDPC codes and the RS versus proposed QC-LDPC codes for a 1 MB file size

Type of code used	Code rate $R$	Storage efficiency	Storage overhead factor ( $f$ )	Dec. overhead (DO)	Dec. overhead ratio (DOR)	For decoding, single erasure	No. of blocks erased to be tolerated ( $\epsilon$ )
Conv. LDPC code (16,8)	0.5	50%	14/8 (1.75)	14	0.75%	14 data blocks	2
Conv. QC-LDPC code (24,16)	0.67	67%	21/16 (1.3125)	21	0.3125%	21 data and 1 parity block	3
Traditional RS code (32,24)	0.75	75%	NA	NA	NA	Nil tolerance	All 32 data blocks are required
Prop. QC-LDPC Code (88,56)	<b>0.63</b>	<b>63%</b>	85/56 (1.517)	85	<b>0.517%</b>	<b>85 data + 1 parity block</b>	<b>3</b>
Prop. QC-LDPC Code (648,512)	<b>0.79</b>	<b>79%</b>	645/512 (1.259)	<b>645</b>	<b>0.259%</b>	<b>645 data + 1 parity block</b>	<b>3</b>
Prop. QC-LDPC code (1024,832)	<b>0.812</b>	<b>81%</b>	1000/832 (1.2019)	<b>1021</b>	<b>0.202%</b>	<b>1021 + 2 parity blocks</b>	<b>4</b>

$$\text{DOR} = \frac{\text{Number of blocks needed to decoded}}{\text{Actual number of data blocks}} - 1. \quad (19)$$

#### 5.4 | Comparative analysis between the decoding performance of conventional LDPC and QC-LDPC codes and RS versus proposed QC-LDPC codes for a 1 MB file size

The results shown in Table 5 show that the proposed QC-LDPC code has better decoding performance when data blocks are lost. We also observed that the proposed QC-LDPC codes showed fewer data blocks required to reconstruct the original data than traditional RS codes.

To summarize, a highly sparse base PCM was constructed using the proposed algorithm with a medium code rate starting from 0.6 after puncturing, the code rate was increased, enabling us to build high-rate and large-length QC-LDPC codes with less complexity. Complexity depends on the number of one's in the rows of the constructed QC-LDPC code. The proposed QC-LDPC code had few one's, which reduced its complexity. Therefore, puncturing is also proposed in the base matrix construction, which helps the code to achieve a high code rate with increased block lengths for high data rate applications. Storage efficiency and fault tolerance were comparatively higher for the proposed QC-LDPC code than the traditional RS, conventional

LDPC, and QC-LDPC codes. Lastly, storage overhead produced after encoding and decoding, which is the key parameter for cloud data storage systems, was also very low.

## 6 | CONCLUSIONS


In this paper, we proposed a new method for constructing randomly generated base matrices, which were then expanded using a lift factor to build large-sized QC-LDPC codes with high code rates for use in cloud data storage systems. The performance of the proposed QC-LDPC codes constructed using the proposed algorithm was analyzed for various  $E_b/N_0$  values, after which we compared BER results using conventional QC-LDPC codes. Simulation results proved that large-sized QC-LDPC codes with high code rates ( $R = 0.8$ ), generated using the proposed algorithm, showed a low BER of  $10^{-9}$  at 3.5 dB  $E_b/N_0$  compared with the other algorithmic QC-LDPC codes with similar code rates and sizes. To evaluate the effectiveness of the proposed algorithm, we implemented the proposed QC-LDPC code using SDR on an NI USRP 2920 hardware platform. Implementation results showed that the proposed QC-LDPC codes of size (1024, 832) and high code rate of 0.8 showed a low BER of  $10^{-7}$  at 2.9 dB  $E_b/N_0$  using the QPSK modulation technique compared with those with similar length and code rate as the conventional QC-LDPC code, which showed a BER of  $10^{-6}$  at 3.5 dB  $E_b/N_0$ .

Storage overhead produced by the proposed high-rate (0.8) and long-length (1024) QC-LDPC codes was 1.20 less than that produced by the proposed QC-LDPC codes with shorter lengths. Hence, the proposed QC-LDPC codes of large block lengths and high code rates were established to be useful for cloud data storage systems and other high data rate applications.

### CONFLICT OF INTEREST

The authors declare that there are no conflicts of interest.

### ORCID

Vairaperumal Bhuvaneshwari  <https://orcid.org/0000-0003-1034-4053>

### REFERENCES

1. R. Nachiappan, J. Bahman, C. Rodrigo, and M. Kenan, *Cloud storage reliability for big data applications: A state-of-the-art survey*, *J Netw Comput Appl.* **97** (2017), 35–47.
2. C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S. Yekhanin, *Erasure coding in windows azure storage*, *USENIX Annual Technical Conference ATC.* 2012, pp. 15–26.
3. Y. Wei, Y. W. Foo, K. C. Lim, and F. Chen, *The auto-configurable LDPC codes for distributed storage*, (IEEE 17th International Conference on Computational Science and Engineering, Chengdu, China), 2014, pp. 1332–1338.
4. S. Ghemawat, H. Gobioff, and S. Leung, *The google file system*, (Proceedings of the ACM Symposium on Operating Systems Principles, Bolton Landing, NY, USA), 2003, pp. 29–43.
5. G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, *Dynamo: Amazon's highly available key-value store*, (Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles, Stevenson WA, USA), 2007, pp. 205–220.
6. D. Borthakur, *The Hadoop distributed file system: Architecture and design*, 2009.
7. Facebook's erasure coded hadoop distributed file system (HDFS-RAID).
8. H. Weatherspoon and J. D. Kubiatowicz, *Erasure coding vs. replication: A quantitative comparison*, In *Peer-to-peer systems*, Springer, 2002, 328–337.
9. F. Fikes, *Storage architecture and challenges*, (Proceedings of the 2010 Google Faculty Summit, CA, USA), July 29, 2010.
10. I. S. Reed and G. Solomon, *Polynomial codes over certain finite fields*, *Soc. Ind. Appl. Math.* **8** (1960), 300–304.
11. M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, *Practical loss-resilient codes*, (29th Annual ACM Symposium on Theory of Computing, El Paso, TX, USA), 1997, pp. 150–159.
12. W. Yongmei, C. Fengmin, and L. K. Cher, *Large LDPC codes for big data storage*, (Proceedings of the ASE Big Data & Social Informatics, Kaohsiung, Taiwan), 2015, pp. 1–6.
13. Y. Wei and Y. W. Foo, *A cost-effective and reliable cloud storage*, (IEEE International Conference on Cloud Computing, Anchorage, AK USA), 2014, pp. 938–939.
14. Y. Wei and F. Chen, *expanCodes: Tailored LDPC codes for big data storage*, (IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, Auckland, New Zealand), 2016, pp. 620–625.
15. R. Sun, X. Cai, J. Liu, and K. S. Kwak, *Distributed SR-LDPC codes over multiple-access relay channel and its applications in cloud storage*, *Concurrency Comput Pract Experience.* **27** (2014), 2064–2077.
16. J. S. Plank and M. G. Thomason, *A practical analysis of low-density parity-check erasure codes for wide-area storage applications*, (DSN-04: International Conference on Dependable Systems and Networks, Florence, Italy), 2004, pp. 115–124.
17. B. Gaidioz, B. Koblitz, and N. Santos, *Exploring high performance distributed file storage using LDPC codes*, *J. Parallel.* **33** (2007), 264–274.
18. S. Hongwei, L. Jingfen, and B. Kumar, *Low complexity LDPC codes for partial response channels*, *IEEE Glob. Commun. Conf.* **2** (2002), 1294–1299.
19. J. S. Plank and L. Collins, *Small parity-check erasure codes - Exploration and observations*, (International Conference on Dependable Systems and Networks, Yokohama, Japan), 2005, pp. 326–335.
20. W. Yongmei and C. Fengmin, *Guided systematic random LDPC for distributed storage system*, (ICIT 2017: Proceedings of the 2017 International Conference on Information Technology, Singapore), 2017, pp. 355–359.
21. C. -W. Sham, X. Chen, W. M. Tam, Y. Zhao, and F.C. M. Lau, *A layered QC-LDPC decoder architecture for high-speed communication system*, (IEEE Asia Pacific Conference on Circuits and Systems, Kaohsiung, Taiwan), 2012, pp. 475–478.
22. Y. Lin, H. Lee, M. Woh, Y. Harel, S. Mahlke, T. Mudge, C. Chakrabarti, and K. Flautner, *Soda: A low-power architecture for software radio*, (Proceedings of the 33rd Annual International Symposium on Computer Architecture, Boston, MA, USA), 2006. <https://doi.org/10.1109/ISCA.2006.37>
23. S. G. Harihara and J. M. Balaji, *SpreadStore: A LDPC erasure code scheme for distributed storage system*, (International Conference on Data Storage and Data Engineering, Bangalore, India), 2010, pp. 154–158.
24. S. Mukherjee and M. Kaufmann, *Error coding techniques*, In *Architecture design for soft errors*, Elsevier, 2008, 161–206.
25. S. Vafi and N. R. Majid, *Combinatorial design-based quasi-cyclic LDPC codes with girth eight*, *Digit Commun Netw.* **4** (2018), 296–300.
26. V. Chouhan and S. K. Peddoju, *Investigation of optimal data encoding parameters based on user preference for cloud storage*, *IEEE Access* **8** (2020), 75105–75118.

### AUTHOR BIOGRAPHIES



**Vairaperumal Bhuvaneshwari** is pursuing her PhD at the BS Abdur Rahman Crescent Institute of Science and Technology, Department of Electronics and Communications Engineering, India. In 2018, she received her MTech degree from the

same department and a BE in Electronics and Communication Engineering from the Karpagam College of Engineering, Coimbatore, India, in 2004. She has more than 8 years of industrial experience as a senior analyst in Verizon Wireless Communications and 1.5 years as an Internet service provider, working with firms such as AT&T and the British Telecom Support in HCL, India. Her research interests range from wireless communication, 5G technology mobile communications, and erasure codes to big data storage systems, including cloud and distributed storage systems.



**Chandrapragasam Tharini** is currently heading the Department of Electronics and Communication Engineering at the School of Electrical and Communication Sciences at the B. S. Abdur Rahman Crescent Institute of Science and Technology, Chennai, India. She received her ME degree in applied electronics from Bharathiar University in 2002 and a PhD in information and communication

engineering from Anna University in 2011. Her current research interests include wireless sensor networks, wireless communication, and signal processing. Notably, she has published more than 25 papers in Scopus-indexed journals and more than 15 in international and national conferences. She is also an active member of the Computer Society of India. She has more than 18 years of teaching experience, with her students presently working on wireless communication systems, wireless sensor networks, and digital signal processing domains.

**How to cite this article:** V. Bhuvaneshwari and C. Tharini, *Novel construction of quasi-cyclic low-density parity-check codes with variable code rates for cloud data storage systems*, ETRI Journal **45** (2023), 404–417. <https://doi.org/10.4218/etrij.2021-0449>