International Journal of Advancements in Electronics and Electrical Engineering –IJAEEE
Volume 3 : Issue 1 [ISSN 2319 – 7498]
Publication Date : 09 January 2014

A Robust Approach for Construction of Irregular LDPC Codes

Rakesh Sharma and Ashish Goswami

***Abstract*—The future communication systems aim at effective error control coding methods with effective error performance and considerable code rates. LDPC codes are again gaining popularity in future communication systems due to ease in encoding design, effective BER and acceptable code rate. This paper presents a new method for constructing an irregular LDPC codes from low column weight LDPC codes. It is discussed that the proposed method has simple encoding process and less memory requirements. It has been shown that proposed method improves BER performance on AWGN channel by approx. 1 dB per extension level with only slight reduction in code rate.**

***Index Terms*—LDPC Codes, Irregular LDPC Codes, Column Weight Code Rate, Parity Check Matrix.**

I. Introduction

ORIGNALLY introduced by Gallager in his doctoral thesis [1], and rediscovered by Mackay and Neal [2], there has been renewed interest in LDPC codes because their bit error rate performance approaches asymptotically the Shannon limit [3], [4]. Much research is devoted to characterizing the performance of LDPC codes and designing codes with good BER performance. LDPC codes are said to be regular if its column and row weights are constant throughout the columns and rows respectively otherwise it is said to be irregular. The irregular codes outperform the regular ones. Some of the methods for generating regular and irregular LDPC codes are discussed in [7], [8]. The performance of the LDPC codes depend on the column weight w_c of its parity check matrix. The large column weight codes show better performance in terms of BER. The codes designed for higher column weights $w_c \geq 3$ have complex designs and hard to implement. On the other side, the codes designed for column weight $w_c = 2$ are less complex to implement. These type of codes can be directly constructed from Regular Graphs [5], [6]. This type of construction is given by Malema [9]. The incidence matrix of the graphs can be used as the parity check matrix for the LDPC Codes. These type of codes have an advantage of high girth but simultaneously having a limitation of fixed column weight 2.

In this paper, a construction method has been proposed to design an irregular LDPC codes by increasing the column weight of the parity check matrix and its concatenation with an Identity matrix. The proposed construction also simplify the encoder design and requires less memory for encoding. We have analyzed the performance of the extended LDPC codes on AWGN channel. The paper is divided into five sections. The introduction is given in section I, the proposed construction method is described in section II. Encoding simplicity and Performance analysis is discussed in III and IV sections respectively. Section V concludes the paper followed by References.

Rakesh Sharma is with the department of Electronics and Communication Enggineering, National Institute of Technology, Hamirpur, INDIA, email: goswami.ashish@rediffmail.com

Ashish Goswami is with the department of Electronics and Communication Enggineering, National Institute of Technology, Hamirpur, INDIA, email: goswami.ashish@rediffmail.com

II. The Construction Method

A. Construction Method

The proposed construction method is explained in following steps,

1. Consider C_0, the incidence matrix of a graph $\mathcal{G}$ (e.g a (k, g)-Cage) or the Regular Matrix having column weight 2. Let the row weight and dimension of C_0 be k and $m \times n$ respectively. Now take a Zero Matrix O_0 of same size as C_0 i.e. having dimension $m \times n$ and an Identity Matrix I_0 of dimension $n \times n$ same as the number of columns in C_0. Now, arrange the matrices C_0, O_0 and I_0 as shown below,

$$C_1 = \begin{bmatrix} C_0 & O_0 & \cdots & O_0 \\ O_0 & C_0 & \cdots & O_0 \\ \vdots & \vdots & \ddots & \vdots \\ O_0 & O_0 & \cdots & C_0 \\ I_0 & I_0 & \cdots & I_0 \end{bmatrix} \quad (1)$$

   The obtained C_1 matrix should have the dimensions $(mk+n) \times (nk)$. The row weight of the obtained matrix remains k, same as in C_0 while column weight gets increased by 1. It can be called as Level-1 extension.
2. Similarly the parity check matrix C_1 can be extended further to Level-2 extended matrix C_2 by replacing C_0, O_0 & I_0 with C_1, O_1 & I_1 respectively. Higher level extension can be carried out in the similar manner. The parity check matrix C_{s+1} at $(s+1)^{th}$-extended Level is given by,

$$C_{s+1} = \begin{bmatrix} C_s & O_s & \cdots & O_s \\ O_s & C_s & \cdots & O_s \\ \vdots & \vdots & \ddots & \vdots \\ O_s & O_s & \cdots & C_s \\ I_s & I_s & \cdots & I_s \end{bmatrix} \quad (2)$$

   The column weight of the obtained parity check matrix has been increased by the number of levels extended while the row weight remained same. We should limit the extension level up to a value less than the row weight. Now this obtained parity check matrix C_{s+1} can be used for generating LDPC codes. The code rate for the above extension code is $r =$

$\frac{n_c - \mathtt{rank}(C_{s+1})}{n_c}$, where n_c is the number of columns. On simulating we found no considerable improvement in BER performance when we extended the LDPC codes beyond column weight 3. Another issue regarding the above codes is with its encoding time, since the parity check matrix contains dependent rows which is to be eliminated at the time of encoding therefore takes longer time for encoding.

3. To avoid the above issues, we modified the matrix by concatenating it with an Identity Matrix of a size t equal to number of rows of C_{s+1}. The final parity check matrix H is given as,

$$H = \left[\begin{array}{c|c} C_{s+1} & I_t \end{array} \right] \tag{3}$$

This type of modification directly converts the matrix into systematic form and the generator matrix can be obtained directly, hence takes less time for encoding. This modification also converts the regular LDPC code into irregular one.

B. Code Rate

The code rate r_{s+1} at $(s+1)^{th}$ level for the proposed LDPC Codes obtained from the above parity check matrix is (the proof is given in Appendix),

$$r_{s+1} = \frac{nk}{(m+n)k + (s+1)n} \tag{4}$$

III. Simple Encoding

Since the designed codes are the higher column weight extension of the low column weight LDPC codes, therefore the encoding of these codes are less complex than their higher column weight counterparts. To encode we need a generator matrix for the extended H matrix. For simplicity only encoder implementation of Level 1 extended codes are discussed. The Generator Matrix of the Level 1 extended code parity check matrix can be given by

$$G_1 = \left[\begin{array}{c|cccc} & C_0' & O_0' & \cdots & O_0' & I_0 \\ & O_0' & C_0' & \cdots & O_0' & I_0 \\ I_g & \vdots & \vdots & \ddots & \vdots & \vdots \\ & O_0' & O_0' & \cdots & C_0' & I_0 \end{array} \right] \tag{5}$$

where I_g is the identity matrix. The size of the matrix is $nk \times (m+n+1)k$.

Let $\mathbf{x}$ be the message vector then the codeword $\mathbf{y}$ can be given by $\mathbf{y} = G_1\mathbf{x}$. The length of the message vector $\mathbf{x}$ should be nk. The main advantage of the method is that we don't require to save the whole G_1 matrix for encoding process but only C_0 matrix need to be saved in the memory. Hence very less memory is required in the encoding process. To calculate $\mathbf{y}$ the steps are as follows:

1. First, we partition the vector $\mathbf{x}$ in to k parts as $[\mathbf{x}_0\ \mathbf{x}_1\ \ldots\ \mathbf{x}_{k-1}]$, each $\mathbf{x}_i,\quad i = 0, \ldots, k-1$ is a vector of n bits.
2. Secondly calculate

$$\mathbf{y}_i = \mathbf{x}_i C_0' \ \forall\ i = 0, \ldots, k-1 \tag{6}$$

and

$$\mathbf{w} = \mathbf{x}_0 + \mathbf{x}_1 + \ldots + \mathbf{x}_{k-1} \tag{7}$$

3. Finally, the codeword $\mathbf{y}$ is formed by the concatenation of the above vectors $\mathbf{x}$, $\mathbf{y}_i$, $\mathbf{w}$ as

$$\mathbf{y} = [\mathbf{x}\ \mathbf{y}_0\ \mathbf{y}_1\ \ldots\ \mathbf{y}_{k-1}\ \mathbf{w}]$$

Hence we see that the encoding is simple and requires less memory since only C_0 matrix needs to be saved. For higher level of extension also only C_0 matrix is required to be saved and the codewords are easily constructed with simple methods.

IV. Performance Analysis And The Required Trade-Offs

We have simulated the constructions based on different codes available in the literature [11], [12] for analyzing the BER performance on AWGN channel. All the simulations are performed in MATLAB. The arbitrary message is encoded with the above obtained method, BPSK modulated and then passed through the AWGN channel. The received waveform was demodulated and decoded by Logarithmic Sum-Product Algorithm [10] with maximum number of 20 iterations and a hard decision at the end. The process was repeated for certain range of SNR and averaged to large number of simulations. Fig. 1 shows BER performance of four different typical LDPC codes taken from literature [11], [12].

From Fig. 1, it is clear that the BER performance of the LDPC codes get improved on extension. For the unextended LDPC code based on Hoffman-singleton graph [11], BER of 10^{-4} can be achieved for 4 dB E_b/N_0. The same can be achieved for approx. 2.7 dB on level-1 extension and approx. 1.5 dB on level-2 extension. Hence it provides approx. 1 dB improvement per extension level. Similarly, a typical Gallager code and High rate Gallager code [12] also shows approx. 1 dB improvement per extension level. In the case of unextended typical Gallager code [12], BER of 10^{-5} can be achieved for 4 dB E_b/N_0. The same can be achieved for approx. 2.4 dB in level-1 and approx. 1.4 dB in level-2 extension. The unextended high rate Gallager code [12] achieves the BER of 10^{-5} for approx. 3.7 dB E_b/N_0 and approx. 2.7 dB E_b/N_0 for level-1 extension. A typical LDPC code generated from Progressive Edge Growth [12] shows 1/2 dB improvement per extension. The BER of 10^{-5} can be achieved for approx. 3.3 dB, 2.5 dB & 1.8 dB E_b/N_0 for unextended, level-1 and level-2 extension respectively.

Hence, we see that on increasing extension levels, the BER performance got improved. We limited our extension level less than or equal to the row weight of the parity check matrix of unextended code. We can see from eq.(4) that on increasing the number of levels, the code rate reduces, which is an another important parameter in the data transmission. The code rates for LDPC Codes generated from different techniques as well as its extended levels are given in Table I. Hence, we have to take a trade-off between the

International Journal of Advancements in Electronics and Electrical Engineering –IJAEEE
Volume 3 Issue 1 [ISSN 2319 – 7498]
Publication Date : 09 January 2014

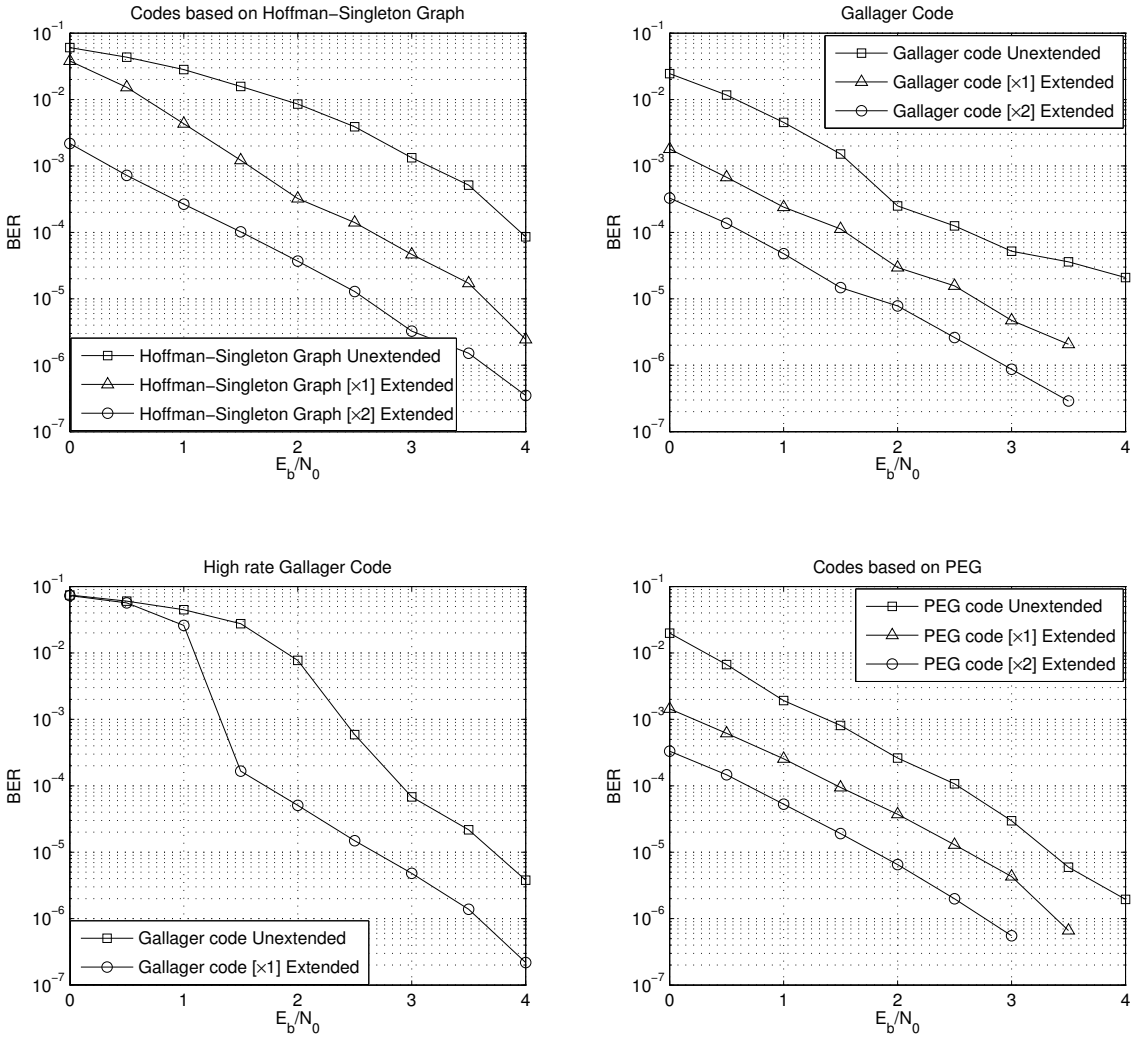

Fig. 1. Simulation results showing BER Performance comparison of original codes with its extended ones.

TABLE I
CODE RATES AT DIFFERENT EXTENDED LEVELS OF LDPC CODES

LDPC code based on ↓	Unextended	Level-1 Ext.	Level-2 Ext.
Hoffman-singleton graph [11] ($k = 7$)	0.78	0.7	0.64
Gallager code [12] ($k = 6$)	0.67	0.6	0.54
High rate Gallager code [12] ($k = 13$)	0.81	0.76	0.72
Progressive Edge Growth [12] ($k = 6$)	0.67	0.6	0.54

BER performance and the code rate of the designed LDPC code.

V. Conclusion

The construction of irregular LDPC codes have complex design while that of column weight 2 regular LDPC codes have simpler design and are easy to implement. The proposed extension method converts the low column weight regular LDPC codes into an irregular one, which not only improves the BER performance but also have simple encoding and require less memory space. On simulation of typical codes [11], [12] and their extensions, we have found approx. 1 dB improvement in their BER performance with only slight reduction in their code rates.

Appendix

[Proof of Eq. 4]

The code rate r for the LDPC Codes obtained from the parity check matrix given by Eq (3) at $(s+1)^{th}$-Level is given by

$$r_{s+1} = \frac{N_c^{(s+1)}}{N_r^{(s+1)} + N_c^{(s+1)}} \tag{8}$$

where, $N_c^{(s+1)}$ is the number of columns in C_{s+1} & $N_r^{(s+1)}$ is the number of rows in C_{s+1}.

Let C_0 matrix has size $m \times n$ i.e, $N_r^{(0)} = m$ & $N_c^{(0)} = n$. Now at $(s+1)^{th}$-Level, the number of rows and columns of C_{s+1} are given by

$$N_r^{(s+1)} = kN_r^{(s)} + N_c^{(s)} \tag{9a}$$
$$N_c^{(s+1)} = kN_c^{(s)} \tag{9b}$$

or

$$N_r^{(s+1)} = k^{s+1}m + k^s n(s+1) \tag{10a}$$
$$N_c^{(s+1)} = k^{s+1}n \tag{10b}$$

Substituting values of Eq. (10) in Eq. (8), we get

$$r_{s+1} = \frac{k^{s+1}n}{k^{s+1}m + k^s n(s+1) + k^{s+1}n} \tag{11}$$

or

$$r_{s+1} = \frac{nk}{(m+n)k + (s+1)n} \tag{12}$$

Hence shown.

References

[1] R.G. Gallager, *Low-Density Parity Check Codes*, Cambridge, MA, MIT Press, 1963.
[2] D.J.C. Mackay and R.M. Neal, *Good codes based on very sparse matrices*, in Cryptography and Coding, 5th IMA Conference (Lecture Notes in Computer Science), C. Boyd, Ed. 1995, vol. 1025, pp. 110-111.
[3] S.-Y. Chung, G. D. Forney, Jr., T. J. Richardson and R. Urbanke, *On the design of low-density parity-check codes within 0.0045 dB from the Shannon limit*, IEEE Commun. Letters, vol. 5, pp. 58-60, Feb. 2001.
[4] D. J. C. MacKay and R. M. Neal, *Near Shannon Limit Performance of Low Density Parity Check Codes*, IEEE Electronics Letters Vol. 32 No. 18, pp. 1645-1646, 29^{th} August 1996.
[5] Geoffrey Exoo and Robert Jajcay, *Dynamic Cage Survey*, The Electronic Journal of Combinatorics, vol. 15, pp. 1-48, 2008.
[6] Meringer, Markus and Weisstein, Eric W, *Regular Graph*, From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/RegularGraph.html
[7] Jose M.F. Moura, Jin Lu, and Haotian Zhang, *Structured Low Density Parity Check Codes*, IEEE signal Processing Magazine, Vol. 21 No. 1, pp. 42-55, Jan 2004.
[8] Chutima Prasartkaew and Somsak Choomchuay, *A Design of Parity Check Matrix for Irregular LDPC Codes*, IEEE 9th International Symposium Communications and Information Technology, 2009. ISCIT 2009, pp. 239-242.
[9] Gabofetswe Malema and Michael Liebelt, *High Girth Column-Weight-Two LDPC Codes Based on Distance Graphs*, EURASIP Journal on Wireless Communications and Networking, Vol. 2007, Article ID 48158, 5 pages.
[10] William E. Ryan and Shu Lin, *Channel Codes: Classical and Modern*, Cambridge University Press, 2009.
[11] Weisstein, Eric W, *Hoffman-Singleton Graph*, From MathWorld–A Wolfram Web Resource. [Online]. Available: http://mathworld.wolfram.com/Hoffman-SingletonGraph.html
[12] David J.C. MacKay, *Encyclopedia of Sparse Graph Codes*. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html

Rakesh Sharma was born in Himachal Pradesh, India, in 1982. He received the B.Tech. degree in Electronics and Communication Engineering with distinction from H.P. University, India, in 2003, and the M.Tech. degree in Electronics and Communication Engineering from National Institue of Technology Kurukshetra, India, in 2007. He is currently working as Asstt. Professor in NIT Hamirpur, India.

Ashish Goswami was born in Junagarh, India, in 1982. He graduated in Electronics and Communication Engineering from Utter Pradesh Technical University, Lucknow, India in 2006 and received M. Tech degree in Communication systems and Networks from NIT Hamirpur, India in 2011. Presently he is working as a Lecturer in NIT Hamirpur, India.