Science Journal of Circuits, Systems and Signal Processing
2014; 3(5): 31-34
Published online November 28, 2014 (http://www.sciencepublishinggroup.com/j/cssp)
doi: 10.11648/j.cssp.20140305.11
ISSN: 2326-9065 (Print); ISSN: 2326-9073 (Online)

Construction of flexible type II and III QC-LDPC codes

Gabofetswe Malema, Nkwebi Motlogelwa

Department of Computer Science, University of Botswana, Gaborone, Botswana

Email address:

malemag@mopipi.ub.bw (G. Malema), motlogel@mopipi.ub.bw (N. Motlogelwa)

To cite this article:

Gabofetswe Malema, Nkwebi Motlogelwa. Construction of Flexible Type II and III QC-LDPC Codes. *Science Journal of Circuits, Systems and Signal Processing.* Vol. 3, No. 5, 2014, pp. 31-34. doi: 10.11648/j.cssp.20140305.11

Abstract: Type II and III low-density parity-check codes (QC-LDPC) codes have been shown to have better minimum distance compared to Type I QC-LDPC codes. This article presents a highly flexible method for constructing high-girth type II and III QC-LDPC codes. The proposed algorithm establishes constraints to be observed in creating a bipartite graph of a given girth. The algorithm is by far more flexible in constructing a wide range (rates and lengths) of type II and III QC-LDPC codes compared to existing methods. Although the proposed algorithm uses a search approach to construct codes, it generally successfully constructs a code even at low code lengths. Constructed codes show better bit error rate performances compared to type I codes as expected.

Keywords: QC-LDPC Codes, Tanner Graph, Girth Code Rate and Length

1. Introduction

The design, construction and implementation of low-density parity-check (LDPC) codes involve a range of contradicting factors such as error correcting performance and ease of implementation. Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) codes are structured codes that have been shown to be easily implementable in both encoder and decoder because of their block and cyclic properties [1][2] They have also been shown to perform close to Shannon's capacity limit[3]. Performance of LDPC codes could be improved by constructing codes with larger girths [4][5]. Performance could also be improved further by optimizing the base matrix of the code. It has been shown that codes with circulant-weights high than one result in higher minimum distances than when only weight-1 circulants are used [6[7]]. Codes with better minimum distances tend to show better bit error rate performances especially at high signal to noise rations. Circulant-1 codes are referred to as type-I codes. Type I QC-LDPC codes have single connections between row (check nodes) and column (variable nodes) in a protograph (base matrix).Type II QC-LDPC codes have base or seed matrices with circulants weights of 0 to 2 whereas Type III codes have circulant weights of 0 to 3.

In this paper we are considering codes with circulant-weights 0 to 3. That is, type II and III QC-LDPC codes.

Multi-edge protographs are those that have multiple connections between nodes. Figure 1 shows a base or seed matrix with a corresponding protograph. Each matrix entry specifies the number of connections between two nodes (rows and columns) in the graph. There are a few construction methods for type II QC-LDPC codes. In [8] a deterministic method is presented for girth-six type II QC-LDPC codes. In [6][7] ways to construct base matrices with maximum minimum distances for type II and type III codes are presented. The derived matrices have upper bounds on the maximum girth. However, the presented methods do not derive constraints to construct codes that will maintain the girth upper bounds.

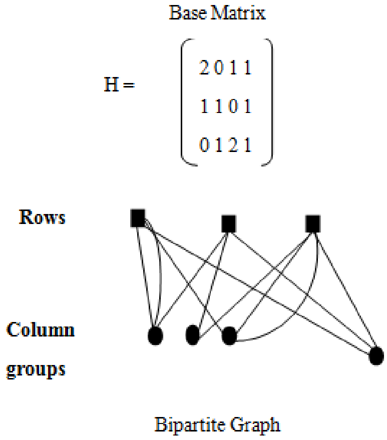

Figure 1. Protograph base matrix and corresponding graph.

2. Proposed Algorithm

In [9] a search algorithm is proposed that constructs type I flexible QC-LDPC codes. The algorithm connects rows (variable nodes) and columns (check nodes) one at a time provided the targeted girth and rate are not violated. The algorithm's main steps are as listed below [9]

Search algorithm four main steps:

1) Divide rows and columns of the constructed code into *j* (column weight) and *k* (row-weight) or more equal size groups respectively. The division of rows and columns into groups creates sub-matrices of the code.
2) The row-groups (RG) and column-groups (CG) are then paired such that each row-group appears *k* times and each column-group *j* times. The number of each row-group or column-group appearances determines the rate of the code. If the number of group appearances (connections) varies an irregular code is obtained. The pairing of RG and CG is according to predesigned protograph.
3) For each row-column group (RCG) pair select a row, *i,* in the row-group, and search for a column, *x,* in the column-group that is at a desired distance (shortest path between nodes) from row *i*. Desired distance is g-1 where g is targeted girth. Connect rows in the row-group to columns in the column-group according to the connection of row *i* and column *x*. That is, if row *i* is connected to column *x*, then row *i+a* is connected to column *x+a*. The connections are *modulo* of the size of row and column groups, *p*. These connections create a cyclic shift in the sub-matrices (shifted identity sub-matrices) of the constructed code. Check if there are smaller cycles formed after connections. If there are, choose another column *x*.
4) Use the obtained Tanner graph to form a parity check matrix.

In the four steps of the proposed algorithm outlined above, smaller cycles are avoided by connecting the reference row *i* to a column *x* that satisfies a desired distance. This condition does not guarantee girth of six if there are multiple connections between nodes even if the distance from *i* to *x* is satisfied.

Figure 2 shows how 4-cycles could be formed when the targeted girth is at least six. In both graphs of Figure 2 the distance from row *i* to column *x* is initially infinity (before proposed dotted lines are drawn). However, if the difference (smallest number of nodes between two nodes in the same group) from *i* to *a* (*i*→ *a*) is the same as that of from *x* to *b* (*x*→*b*) then *a* is mapped to *b* forming a cycle of 4 with *i* and *x*. A 4-cycle is also formed in the second graph of the figure if *i*→*a* is the same as *x*→*b*. The differences are modulo the size of row and column groups (*p*) and rows *i* and *a* are in the same group and columns *x* and *b* are also in the same group. A group in this case means the same sub-matrix or a single node in a protograph. To avoid 4-cycles in the shown graphs of Figure 2, the proposed algorithm searches for column *x* that does not map *a* onto *b* forming 4-cycles or less than desired cycles. The algorithm searches for *x* that is at a desired distance from *i* then checks if by connecting the two nodes smaller cycles are not going to be formed. If smaller cycles are going to be formed, *x* is discarded and another *x* is searched.

In the next subsections we list and describe the sub-graph connections that could result in less than desired cycles for girth-six, eight and ten codes when the searched column *x* satisfies the desired distance from reference row *i*. To obtain desired girths the listed sub-graphs are avoided.

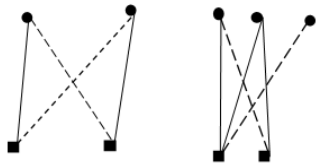

Figure 2. *4-cycle formations (avoiding 4-cycles).*

2.1. Type III QC-LDPC Codes

Type III QC-LPDC codes have at least one triple connection between two row and column groups. That is, there is at least one triple connection between two nodes of a protograph. Such configurations have been shown to have a maximum girth of six [6]. To obtain girth-six codes cycles of four have to be avoided in the construction of a code. The girth condition in the proposed algorithm does not guarantee girth of six. That is, smaller cycles could be obtained even if the distance between row *i* and column *x* is at least six. Smaller cycles could be formed as illustrated in Figure 2. Although, row *i* and column *x* do not form a six-cycle initially, when they are connected, they may map row *a* onto column *b* which leads to a four-cycle between rows *i*, *a* and columns *x*, *b*. To avoid 4-cycles sub graphs shown in Figure 2 are avoided by searching for *x* that does not map *a* onto *b* when connected to *i*. Smaller cycles are avoided as described above.

As explained above connecting *i* and *x* maps *a* onto *b* if the difference from *i* to *a* is the same as that from *x* to *b*. To avoid this mapping the differences must not be equal. When the algorithm is implemented it checks these differences to determine if smaller cycles will be formed or not. The numbers below the sub graphs show the length of the path with existing edges. The first number is the number of edges from *i* and the second number is the number of edges from *x* or the length of the second branch from *i*. The cycle will be completed by one or two proposed edges indicated as dotted lines.

2.2. Girth-8 Type II QC-LDPC Codes

Type II QC-LDPC codes have circulant-weights of 0 to 2 with at least one circulant-weight of 2. Depending on the combination of weights they could have a maximum girth of eight or ten. Type II QC-LPDC codes base matrices with a weight configuration of [2 2] have a maximum girth of eight [6][7]. [2 2] means two double connections in a protograph.To obtain girth-eight four and six-cycles have to be avoided in the construction of the Tanner graph. Four cycles are avoided as shown in the above sub-section.

Figure 3 shows all the combinations of connections that could result in the formation of six-cycles in a [2 2] configuration in the base matrix. In each case the distance from row *i* to column *x* is assumed to be at least 7 (g-1) such that *i* and *x* form at least eight-cycles, where g is desired girth. However, if the difference from row *i* to row *b* is the same as from column *x* to *b* then six-cycles are formed. Therefore, to avoid six-cycles, configurations shown in the figure should be avoided.

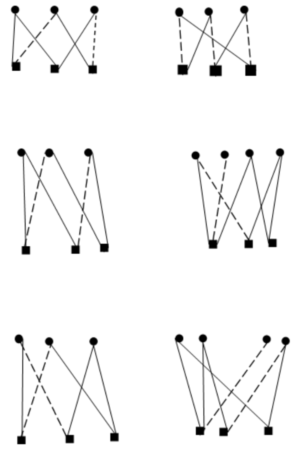

Figure 3. *Avoiding six-cycles in Tanner graphs.*

It is important to note that all these cases of sub-graphs in Figure 3 are a result of double connections between two nodes (row and column groups). If there are no double connections then girth-eight is guaranteed by find *x* that is at least a distance of 7 from *i* [9].

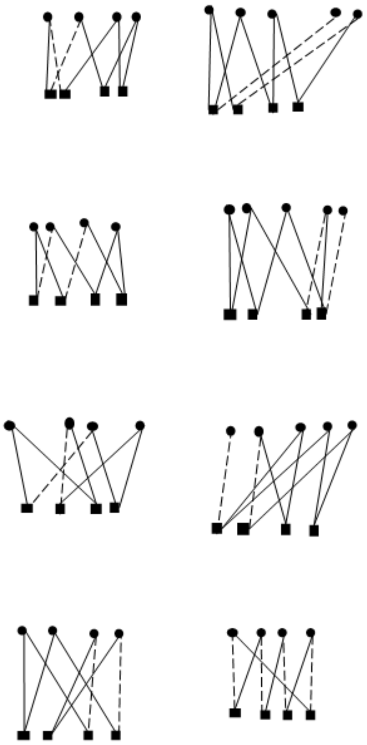

Figure 4. *Avoiding eight-cycles in Tanner graphs.*

2.3. Girth-Ten Type II QC-LDPC Codes

For type II codes to have a girth of ten they must have a base matrix configuration of [1 1;1 2] [6,7]. We avoid formation of less than ten cycles by looking at all the connection configurations that may result in smaller cycles as in the above cases. Four and six-cycles are avoided as in the above subsections. Figure 4 shows how eight-cycles could be avoided in order to obtain girth-ten codes. As in the case of girth-eight, less than ten cycles are avoided by not mapping row *a* onto column *b* when row *i* and column *x* are connected. This is achieved by searching for *x* that avoids all these cases. If all these sub graphs are avoided girth-ten codes are obtained. Similar to avoiding 6-cycles, the shown distances under the sub graphs add up to 6 (*g*-2 or 8-2) and 7 (*g*-1 or 8-1).

2.4. Algorithm Analysis

Row-column connections could be made either from the row or column side in the proposed algorithm. We formally describe the algorithm below with connections made from the row side.

QC-LDPC Code Search Algorithm [9]

```
1) Divide code rows into j' equal groups of size p, (RG1...RGj') and columns into k' groups of size p, (CG1...CGk' ), where j' ≥ j and k' ≥ k. k is code row-weight and j is column-weight. rx is row x. Urx is a set of rows and columns within a desired distance from row rx. cx is column x. Ucx is a set of rows and columns within a desired distance from column cx. Distance is the shortest path between any two nodes (rows or columns).
2) Make row-column group (RCG) pairings according to your specifications such that each row-group appears k times and each column-group j times for regular codes. The number of such group pairings is j'k or k'j for regular codes. The row-column groups are (RCG1 ...RCGkj'). Connections are made based on a predesigned protograph.
3) For t = 1 to kj' {
        Select ri from RG in RCGt, where 1 ≤ i ≥ p.
        Sequentially or randomly search for cx from CG
        in RCGt, where cx notin Uri and avoid smaller
        than desired cycles by using the difference
        condition to choose for suitable x. Else the
        algorithm
        fails.
            for z = 0 to p-1 {
                ri+z is connected to cx+z. }}
4) Use the obtained Tanner graph to form a parity-check matrix.
```

The proposed algorithm goes through all the possible sub graphs that could result in smaller cycles. The possible sub graphs configurations are determined by looking at the number of edges needed to complete a cycle. Generally we need two edges to complete a cycle. The two edges are from i to x and from a to b. Therefore we look for all subgroups configurations with total length of (g-2) where g is the targeted girth. Some of the numbers below the sub graphs add up to (g-2). One length is from i, the other from x. There are other cases where a single line is needed to complete a cycle and others where multiple lines would be required. These cases are also shown in the sub graphs. Using this observation, we could systematically calculate all the sub-graphs for a given girth as was demonstrated in our Matlab code in [10].

As shown in [9] the computational complexity of the algorithm is O(M) where M is the number of rows.

Although each execution of the algorithm does not guarantee that we will get a code (that is, algorithm may fail in step 3), from our experiments the algorithm constructs a code most of the time even for small groups sizes (sub matrix size).

The proposed algorithm is flexible in that the number of sub-matrices and their sizes is not fixed. Protographs could be designed for better performance, to suit a particular decoder architecture and decoder architecture techniques such as overlapping to speedup decoding computations. Irregular codes could be obtained by having an unequal number of row and column groups' connections. The length of a code can easily be varied by changing the size of row and column groups and the rate by changing the values of *j* and *k* accordingly.

3. Performance Simulations

Bit error rate (BER) performances of constructed codes were simulated on an AWGN channel with BPSK modulation. Performance curves for girth-six and ten codes are shown in Figure 5. The figure compares BER performance of type I and type II and Type III (786, 3, 6) QC-LDPC codes. Type I codes were constructed using the method in [9] and type II and III codes using the proposed method. Type II and III codes slightly outperform type I codes at 3 SNR. Performance of type II and III codes is expected to be better at low SNR. However, performance simulations shown did not go further to the floor area. Further simulations need to be performed reaching the floor area to investigate the extent to which type II and III codes are better compared to type I codes.

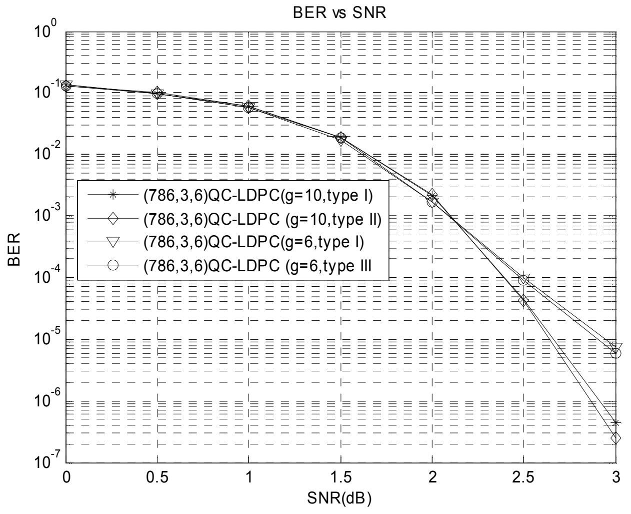

Figure 5. *BER performances of type I, II and III qc-ldpc codes*

4. Conclusions

A search algorithm for constructing high-girth type II and type III Quasi-Cyclic LDPC codes has been presented. The algorithm is flexible in that, it could be used with any protograph configuration, a wide range of rates and lengths can easily be obtained for both regular and irregular codes. Experiments from the implemented Matlab programs obtained codes reasonably fast and do not require very large sub matrices to find codes. The algorithm offers more flexibility compared to previous developed QC-LDPC construction algorithms. Obtained codes show good BER performances. LDPC codes are being considered by a number of communication standards. The presented algorithm is well suited to construct a wide variety of codes in the proposed standards.

References

[1] H. Fujita and K. Sakaniwa, "Some Classes of Quasi-Cyclic LDPC Code: Properties and Efficient Encoding Method", *IEICE Fundamentals*,Vol. E88-A, No.12, pp. 3627 – 3635, 2005.

[2] S. Olcer, "Decoding Architecture for Array-code-based LDPC Codes", *Proc. IEEE GLOBECOM*, pp. 2046 – 2050, December 2003.

[3] L. Chen, J. Xu, I. Djurdjevic, and S. Lin, "Near Shannon-Limit Quasi-Cyclic Low-Density parity-Check Codes," *IEEE Transactions on Communications.*, vol. 52, pp. 1038–1042, July 2004

[4] M. O' Sullivan,J. Brevik and R. Wolski, "The Performance of LDPC Codes with Large Girth," Proc.of the 43rd Annual Allerton Conference; Communication, Control and Computing, Septem-ber 2005.

[5] Y. Mao and A. Banihashemi, "A Heuristic Search for Good Low-Density Parity-Check Codes at short Block Lengths," *Proceedings of IEEE International Conference on Communications,* Vol. 1, pp,41-44, June 2001.

[6] R. Smarandache and P.O Vontel, "Quasi-Cyclic LDPC Codes: Influence of Proto and Tanner Graph Structure on minimum Hamming Distance upper Bounds", *IEEE Transactions on Information Theory*, 2009.

[7] B.K Butter and P.H Siegel, "On Distance Properties of Quasi-Cyclic Protograph-Based LDPC Codes", *ISIT 2010*, pp. 809 – 813, Austin, Texas, June 13-18, 2010.

[8] K. Lally, "Explicit Construction of type-II QC LDPC Codes with Girth at least 6", I*SIT2007*, pp. 2371 – 2375, Nice, France, June 24-29 2007.

[9] G. Malema, "Flexible Construction of High-Girth QC-LDPC Codes", *International Journal of Computer Science and Application*, Vol. 1 Issue 1 August 2012 pp. 19-25

[10] G. Malema, "Construction of Type II and III QC-LDPC codes", *http://www.mathworks.de/matlabcentral/fileexchange/authors/30162*