Construction of Codes Protographes LDPC Quasi-Cycliques Based on an Arithmetic Progression

I. Diop, S.M Farssi, MBA.H.B Diouf
Polytechnic School of Cheikh, Anta Diop University
Dakar, Senegal
idydiop@yahoo.fr

ABSTRACT: *In this document, we study the construction of codes LDPC quasi cyclic based on a protograph. In addition to their large perimeters, the presented codes take profit from the advantages of bass coding and decoding complexity. The aim of this article is to make an improvement on the implementation of quasi-cyclic LDPC codes. Thus to respect the structure of the basic protograph we use an arithmetic progression to determine in advance the positions of certain nodes of control in the derived graph, which turn to some extent amounts generating a new model while proceeding to an enlarging of the basic protograph. Once this new model conceived we apply the usual techniques to build in an optimal way the derived graph.*

Keywords: Regular LDPC, QC-LDPC, PEG, Matrix of Vandermonde, Protograph

Received: 24 June 2012, Revised 31 July 2012, Accepted 4 August 2012

1. Introduction

The low density parity check codes (LDPC) were proposed for the first time in 1963 by Gallager [1]. At that time, even if it was found very interesting on the theoretical level, these work found little echo, that was mainly due to the limits of material architectures at the time. Put aside for a long time, it is thirty years later, with the advent of the turbo-codes [2] and the description of the "*turbo*" principle related to iterative decoding that these codes re-appeared in 1996 [3]. It is especially the work of Thomas J. Richardson and Rudiger Urbanke [4, 5] who allowed a notorious projection on the knowledge and the optimization of the structure of these codes. For ten years now, the whole of undertaken research have made it possible to reach the same degree of maturity as the turbo-codes [6] and make them good candidates in the standards. Currently, LDPC codes from their increasingly powerful capacity and their low complexity of decoding continue to draw the attention of the scientific community of research on coding. However, the implementation of such codes turn out very complex, and in general the complexity of coding develops quadratically (or slower [7]) with the length of the block. A LDPC code is described by its hollow matrix of parity check (P.C.M) or by its tan graph[8].This document is mainly based on the construction of quasi-cyclic LDPC codes proposed by Nicholas Bonello and al [9]. More explicitly we propose a new approach to generate a regular PCM built on blocks matrix of Vandermonde [10]. The advantage of studied construction is to obtain a quasi-cyclic (QC) form [11] which significantly reduces the complexity of coding in the length of the block [12]. In addition the associated complexity of decoding is reduced by adopting a construction based on a descriptor, which is also indicated under the name of protograph by J. Thorpe [13]. These codes can be decoded by semi-parallel, as proposed by Lee and al. in [14]. The contribution brought in this document is to propose a new construction of the PCM based on an new approach which not only makes it possible to build codes regular quasi-cyclic protograph, but in addition conforms to the nature even of architectures used in particular those of the shift registers.

The document is structured as follows: section 2 introduced the basic principles of structured codes LDPC and codes LDPC built on a protograph. Section 3 described and explains construction proposed. Finally the section 4 presents the results obtained.

2. Structured Binary LDPC Codes

For LDPC codes the structure of the PCM (matrix of parity check) can be regular or irregular.

A code is regular if the number of elements per line (respectively by column) is constant. A code is irregular if it is not, by definition, regular. We consider a binary LDPC code of PCM H built on GF (2), then, suppose the PCM made up of m lines and of N columns, the density of this code becomes R = 1 - m/n. This can also be represented by means of a binodal graph, commonly called factorial graph [15] or bipartite Tanner graph [16, 17], being composed of the m nodes of control and N nodes of variable. More explicitly, we consider a regular code having a uniform degree of edges emerging from each node of control and variable. The degrees of the nodes of variable and control will be respectively indicated by γ and ρ, which also corresponds to the weight of column and line of the PCM. LDPC Codes are typically decoded by using the algorithm nap-product (SPA) [18], where messages are permuted between the nodes residing on the two sides of the graph. The independence of these messages is characterized by the length of the shortest cycle on the graph, which is typically indicated under the name of *"girth"* G.

2.1 Codes *LDPC* based on the protographs

The protograph codes were introduced by J. Thorpe [13]. By definition, a protograph is a bipartite graph of small size from which we build a larger graph by a procedure known as "*copies and permutations*", during which the protograph is duplicated in certain number of times, then the branches of the duplicated graphs are permuted by complying with certain structural rules. The protograph is generally described by its matrix of H adjacency, also called basic matrix [19] [20], where the coefficients H (I, J) represent the number of branches between the i-th node of control Ci and the j-th node of Vj variable of the protograph. The procedure of construction of a binary LDPC code based on a protograph illustrated by figure 1. This procedure perhaps recapitulated as follows:

• **Stage of copy:** The protograph is recopied T time to obtain T counterparts. T is selected in an arbitrary way to obtain the desired size of the word of code. On the example illustrated by figure 2 T = 3.

• **Stage of permutation:** The branches of the various counterparts are permuted between the various counterparts in order to obtain a larger graph. Figure 3 illustrates perfectly the operation of permutation between T = 3 copies of the protograph.

The set of permutations must satisfy the topological structure of the initial protograph. Let us note that that build code, can be seen like a big size projection of the basic model from where the acronym "*projected graph*". Consequently, the branches of the graph obtained correspond to the not null entries of the matrix of parity H associated o this bipartite graph, and thus define a code LDPC said "*structured* ".

Let us consider the basic protograph, G, described by the whole of the nodes of control C ={cti: t =1 ; i =1,…, m}, the whole of variable nodes V= {vti: t =1; i = 1, … n} and the whole of edges E, where |E|= m.ρ = n. γ; m and n respectively indicate the number of nodes of control and variables of the basic protograph.

The basic photograph will t have a PC size (m, n). After permutation of T copies, we obtain the graph of the resulting code, G', defined by the units It, V' and E', where each unit has a size, which is T time larger than the corresponding whole in the basic protograph. The permutations of the edges of the nodes in the derived graph obey certain constraints, which will be discussed in section 3.

2.2 Construction of LDPC *codes* based on a protograph

Two big branches of construction are then possible. The first family returns to techniques of algebraic construction of structured cyclic or quasi-cyclic codes [12]. The second rests algorithms of pseudo-random construction such as algorithms PEG (Progressive Edge Growth) [21].

• Construction by PEG: This method simply consists in considering pseudo-random permutations by assigning the branches of the graph, group's nodes of variables by nodes group of variables.

• Construction based on circulating matrices This method uses matrices of circulating permutations; the assignment is done either by branch (edge) but by group of T branches via the assignment of the matrix of permutation.

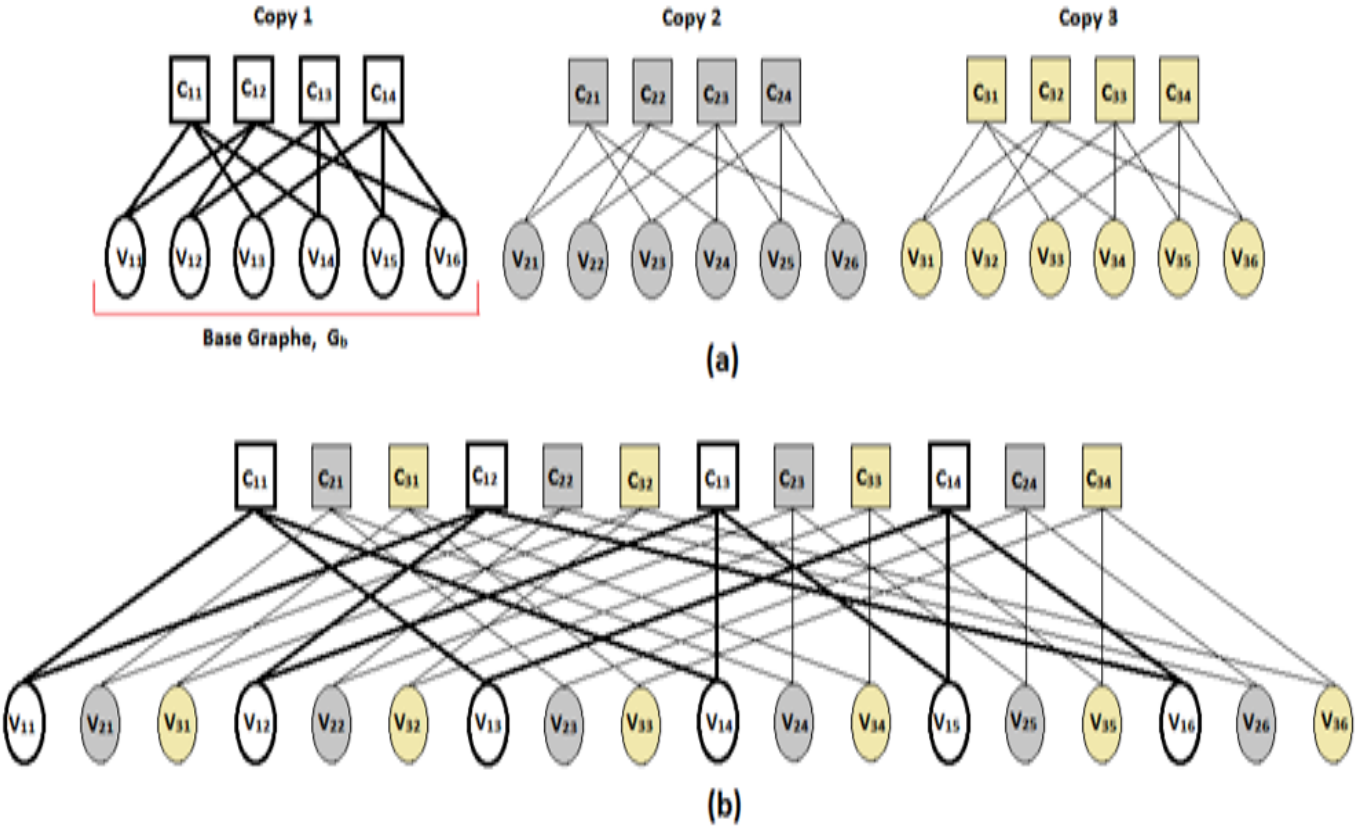

Figure 1. Example of *"copies and permutation"*

The matrix H resultant of this last method is a version known as "*lifting /enlarging"* (increased) and the parameter T is often indicated like the order of enlarging. The matrix H resultant of this last method is a version known as "*lifting*" and the parameter T is often indicated like the order of enlarging. Moreover resulting code LDPC will be quasi-cyclic: indeed, any shift of a word of code is again a word of code. That allows an easy encoding by shift registers what has as a consequence a good degree of parallelism of architectures of decoding, from where the attraction for these codes known as structured. The structure of quasi-cyclic codes LDPC builds on a protograph is subjected to two major constraints:

•Drawback (1): The fact of using a regular PCM as basic graph imposes a certain structural regularity on the derived graph with

•Drawback (2): With this first drawback a second is added which is the quasi-cyclicity induced by a construction based on protograph in other words the manner of permuting the edges.

3. Construction of regular code protograph QC_LDPC

3.1 Construction of the basic protograph

Since we want to build quasi-cyclic codes protograph, we use a construction based on the matrix of Vandermonde. This method stays primarily on the construction of the matrix of noted permutation P. The matrix P is a matrix whose Pmn elements are such as:

$$P_{mn} = \begin{cases} 1 & \text{if } m = (n-1) \bmod q \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

with $q > (\rho - 1)(\gamma - 1), 0 \le m \le q$ *et* $0 \le n \le q$

Example, for q = 3, the p_q and p_q^2matrixes of permutations are given by:

$$p_q = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \text{ and } p_q^2 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad (2)$$

The matrix of permutation as the example shows it above has the following properties:

- It is a square matrix of order q
- If $r = q$ then $(p_q)^T = I_q$

The basic protograph (PCM based on the construction of the matrix of Vandermonde) with the following configuration:

$$H = \begin{bmatrix} I_q & I_q & I_q & \cdots & I_q \\ I_q & P_q & P_q^2 & \cdots & P_q^{(\rho-1)} \\ I_q & P_q^2 & P_q^4 & \cdots & P_q^{2(\rho-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ I_q & P_q^{(\gamma-1)} & P_q^{2(\gamma-1)} & \cdots & P_q^{(\gamma-1)(\rho-1)} \end{bmatrix} \quad (3)$$

Thus the basic matrix H is of size ($\gamma q \times \rho q$).

$$\text{where} \quad P_q^x = \begin{cases} I_q & \text{if } x \bmod q = 0 \\ P_q^{x \bmod q} & \text{otherwise} \end{cases} \quad (4)$$

3.2 Construction of the derived graph

The construction of the derived graph is determined by the basic protograph and can be described in two principal stages:

- The widening of the basic graph "*lifted matrix*"

In the approach suggested, the choice of the enlarging term is not free, because we seeks for widenning the graph to tan basic while respecting the two constraints (cf 2.3) related on the construction and the structure even of the basic graph.

- Copy and permutation of the widened graph.

The new basic model obtained after enlarging is projected to give the derived graph. It is important to notice a fundamental property of the enlarging which imposes systematically on the widened graph the same number of nodes of variables as the basic graph. Thus to build the derived graph it will occur to parallel place (T-1) copies of new the model conceived. The procedure to be followed is recapitulated in algorithm 2.

3.3 Description and analyzes of construction's algorithm

The algorithm is composed of two distinct parts:

The first part (line 1 with line 15) consists of carrying out the enlarging of the basic graph (represented by the matrix G of size (mT,nN). It is a stage impossible to circumvent of construction suggested. To start one adopts the principle of algorithm PEG [21], i.e. for each node of variable, one traverses all the incidental nodes of control to this node of variable. This amounts generating the vicinity (under graph) for each node of variable considered [13].

Thus under graphs on the basis of each variable node are obtained by unfolding the graph To tan basic according to the width [21]. The first constraint makes it possible to pre-empt the positions of the nodes of incidental controls to the node of variable concerned in the derived graph.

In fact the determination of the whole nodes of variables in the final configuration is established according to an arithmetic progression of reason T and so that the first term of this progression is the incidental node of control having the low degree.

Algorithme 2. Construction of derived graph

```
Inputs : H, T
Output : G'
H (m, n), G'= matrix (mT, nT)
P = matrix (mT, nT), G = matrix (mT, n)
BEGIN
1 for j =1 à n
    C = {i } où C(1) < C(2) < ... C(n)
2    and 1 ≤ i ≤ m / H_b (i, j ) ← 1
3    for k = [(j-1).T+1] à jT
4      for i =1 à mT
5        if k = (j-1) T+1 alors
6          for x =1 à cardinal (C)
7              P ([1 + T(C(x) - 1)], k) ← 1
8          End
9        End
10      End
11    End
12    for i = 1 à mT
13      g(i, j) ← P(i , [(j-1) T + 1])
14    End
15  End
16  for j = 1 à n
17     for k ← ((j-1) T+1) à jT
18        for i =1 à mT
19           if k = (j-1) T+1 alors
20              G' (i, k) ← g (i, j)
21           End
22        End
23        End
24     End
25   establishing connections between nodes
     of control and variables not yet connected
   similarly to PEG (modified) [9]
END
```

The fundamental idea is to find the node of the most distant control then to place a new edge formed by the node of symbol and this node of control while taking account of the constraints. The interest of such a process is to determine a new structure which the form approaches more that of the basic protograph in an environment where MT nodes of control are present instead of m nodes.

Thus in a general way the new position of the node of control is given by the following arithmetic progression:

$$C_n = \begin{cases} C_0 = 1 \\ C_n = C_0 + T(n - 1) \end{cases} \quad (5)$$

Where T represents the reason of the continuation, N is the position of the node of incidental control to the node of variable concerned in the basic graph and Cn indicates the new position of the node of control in the derived graph. By fixing the edges of the top spin graph we obtain a derived graph which respects the structure of the basic PCM initially built on the matrix of Vandermonde. Thus the top spin graph counts n.γ edges which will be fixed, in other words n.γ nodes of control from which the positions in the graph derived are given by the arithmetic progression defined by (Cn).

If we take the example of the node of V13 variable of the basic graph G the whole of the nodes of control which are incidental for him is C = {C11, C14} respectively occupying the positions n =1 and n = 4.

Thus the positions of these nodes in the derived graph are given by: C1 = 1 and C4 = 10 what is illustrated perfectly by figure 1 (b).

The second part of the algorithm (line 16 with line 30) allows projecting the basic graph top spin. This projection consists in parallel laying out T-1 counterparts of the new model. Since the final graph represents an enlarging of order T of the basic graph it is necessary to pre-empt the position of the nodes of variable in the final configuration. These new positions are given by considering T blocks of N nodes of variables where the j-th node of variable of the basic graph occupies the first position of the j-th block. The last stage of construction consists in placing the remaining edges. Since the top spin graph count $n.\gamma$ fixed edges and that the derived graph must be made up of $(n.\gamma.T)$ then the number of connection to establish will be equal to $n.\rho.(T-1)$. These connections will be established between the $n.\gamma.(T-1)$ free nodes of variables and nodes of control remaining by using the algorithm of the PEG modified [9] in order to optimize the parameter of the derived graph.

In short this new approach allows a construction not requiring traversing all the edges to determine the "*edges of them allowed* " and the "*prohibited edges*" it is based on the combination of the principles of the PEG and construction per block of circulating matrices.

4. Results and Simulations

4.1 Construction of the matrix of parity

The results of simulation presented were obtained by implementing algorithm 2 proposed in section 3. We consider a basic protograph H of size (27, 45) having for weight $\gamma = 3$, $\gamma = 5$ and which will be copied 5 times to obtain a quasi-cyclic PCM noted G'. H is the PCM of the code protograph LDPC, G' is obtained by copy and permutation of 5 identical protographs H (figure 2) thus the number of edges of G' (many edges = 675) is 5 times more important than that of H (many edges = 135). H and G' have the same structure (figure 3).

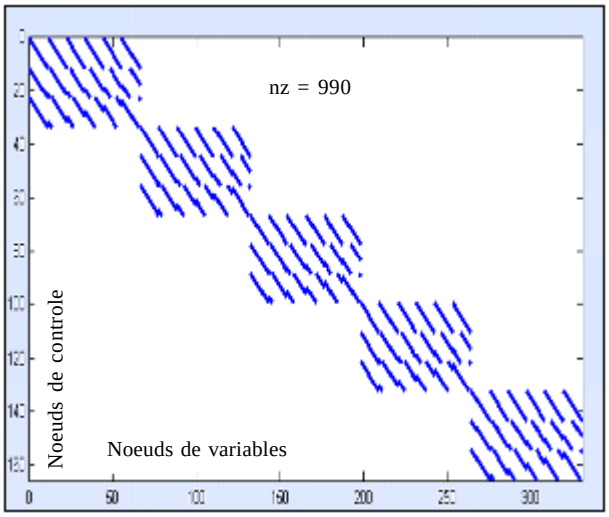

Figure 2. Fives copies of H (γ=3; ρ = 5)

4.2 Evaluation of Complexity

To evaluate the complexity of the algorithm 2 several measurements were taken starting from a computer equipped with a processor Intel Pentium 4 CPU 3.06GHz and equipped with a physical memory total of 1015 Mo.

The results obtained are represented by curves of complexity. Each curve specifically represents the execution time of N derived graphs having jointly the same basic protographe.

Thus to each value of N ($1 \leq N \leq 15$) corresponds a graph derived built starting from N copies from a basic graph whose matrix is noted H, the execution time of the process is expressed in second. The results of simulation show a linear complexity for protographes of weight column $\gamma \leq 4$ and one quadratic complexity for a weight of column $\gamma \geq 4$. The longest the execution time (about 6s) corresponds to that of the implementation of a graph derived from perimeter 8 of size (2790, 3255) made up of 19530 edges.

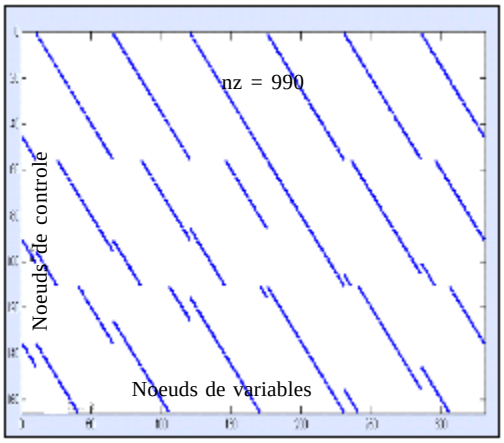

Figure 3. Derived graph G'(γ=3; ρ = 5)

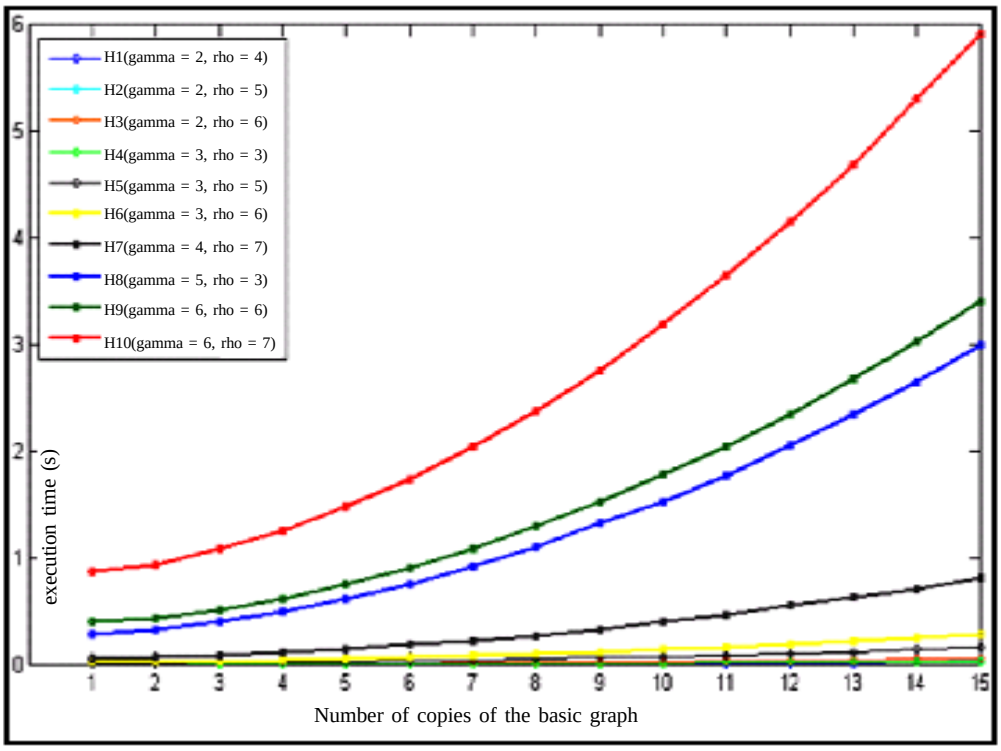

Figure 4. Plots of complexity

References

[1] Gallager, R. G. (1963). Low-Density Parity-Check Codes. MIT Press.

[2] Berrou, C., Glavieux, A. (1996). Near optimum error correcting coding and decoding. *IEEE Transactions on Communications*, 44 (10)1262–1271, January.

[3] David. J. C., MacKay. (1999). Good error correcting codes based on very sparse matrices. *IEEE Trans. Inform. Theory*, 45 (2) 399–431, March.

[4] Thomas J. Richardson, Rudiger Urbanke. (2001). The capacity of low-density parity check codes under message-passing decoding. *IEEE Trans. Inform. Theory*, 47(2) 599–618, February.

[5] Thomas J. Richardson, Amin Shokrollahi, RudigerUrbanke. (2001). Design of capacity approaching irregular low-density parity-check, codes. *IEEE Trans. Inform. Theory*, 47(2) 619–637, February.

[6] Pyndiah, R. M. Near-optimum decoding of product codes: block turbo codes. *Communications, IEEE Transactions on*, 46 (8) 1003 –1010, aug.

[7] Richardson, T., Urbanke, R. (2001). Efficient encoding of low-density parity check codes, *IEEE Trans. Commun*., 47 (6) 808–821, Feb.

[8] Tanner, R. M. (1981). A recursive approach to low complexity codes, *IEEE Trans. Inf. Theory*, IT-27 (5) 533–54, Sep.

[9] Bonello, N., Sheng Chen, LajosHanzo. (2008). Construction of Regular Quasi-Cyclic Protograph LDPC codesbased on Vandermonde Matrices, *IEEE Trans. on Vehicular Technology*, 57(4), july.

[10] Fan, J. L. (2000). Array codes as low density parity check codes, *In*: Proc. 2nd Int. Symp. Turbo Codes, Brest, France, 3, p. 543–546.

[11] Tanner, R. M. (1999). On quasi-cyclic repeat-accumulate codes, *In*: Proc.37th Annu. Allerton Conf. Commun., *Control Comput., Monticello*, IL, Sep, p. 249–259.

[12] Tanner, R. M. (2001). Spectral graphs for quasi-cyclic LDPC codes, *In*: Proc. IEEE Int. Symp. Inf. Theory, Washington, DC, Jun. p. 226

[13] Jeremy Thorpe. (2003). Low-density parity-check (LDPC) codes constructed from protographs. Technical report, *JPL Interplanetary Network Progress* (INP) Report 42-154, August.

[14] Chen, L., Xu, J., Djurdjevic, I., Lin, S. (2004). Near-Shannonlimit quasi-cyclic low-density parity-check codes. *IEEE Trans. Commun*., 52 (7) 1038–1042.

[15] Kschischang, F., Frey, B., Loeliger, H. A. (2001). Factor graphs and the sum product algorithm. *IEEE Transactions on Information Theory*, 47 (2) 498–519, February .

[16] Tanner, R. M. (1981). A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, 27:533–547, September.

[17] Wiberg, N. (1996). Codes and decoding on General graphs. Diisertation nr 440, Dept. Of Electrical Engineering, Link¨oping, Sweden, October.

[18] Lee, J. K. S., Lee, B., Thorpe, J., Andrews, K., Dolinar, S., Hamkins, J. (2004). A scalable architecture of a structured LDPC decoder, *In*: Proc. IEEE Int. Symp. Inf. Theory, Jun. 27–Jul. 2, 292.

[19] Ryan, W. E., Lin, S. (2009). Channel Codes: Classical and Modern. Cambridge University Press.

[20] Richardson, T., Urbanke, R. (2008). Modern Coding Theory. Cambridge University Press.

[21] Xiao-Yu Hu, Evangelos Eleftheriou, Dieter M Arnold. (2005). Regular and irregular progressive edge-growth tanner graphs. *IEEE Trans. Inform. Theory*, 51(1) 386– 398, January.

[22] Divsalar, D., Dolinar, S., Jones, C. R., Andrews, K. (2009). Capacity-approaching protograph codes. Selected Areas in Communications, *IEEE Journal on*, 27 (6) 876–888, aug.

[23] Venkiah, A. (2008). Analysis and Design of Raptor Codes for Multicast Wireless Channels. PhD thesis, Universit´e de Cergy-Pontoise, November.

[24] Fossorier, M. P. C. (2000). Quasi-cyclic low-density paritycheck codes from circulant permutation matrices, *IEEE Trans. Inf. Theory*, 50 (8) 1788–1793, Aug.