Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 567–570

ISBN 978-83-60810-51-4

On LDPC Codes Corresponding to Infinite Family of Graphs $A(k, K)$

Monika Polak
Institute of Mathematics,
Maria Curie-Sklodowska University,
pl. M. Curie-Sklodowskiej 5,
20-031 Lublin, Poland
Email: monika.katarzyna.polak@gmail.com

Vasyl Ustimenko
Institute of Mathematics,
Maria Curie-Sklodowska University,
pl. M. Curie-Sklodowskiej 5,
20-031 Lublin, Poland
Email: ustymenko_vasyl@yahoo.com

***Abstract*—In this paper we investigate correcting properties of LDPC error correcting codes obtained from new infinite family of special extremal graphs. We describe how to construct these codes and compare our results with codes obtained by Guinand and Lodge, corresponding to family of graphs $D(k, q)$.**

I. Introduction

INFORMATION is always transmitted through the communication channels with interferences, which can be an air, a telephone line,a beam of light or a cable. An interference could cause errors in the transmited messages. It is very important for the recipient to receive exactly the same message as was sent, in order to minimize the number of errors in the transmission we can use error correcting codes.

All information in a computer are represented as zeros-ones sequences. Coding of information using linear error correcting codes means adding to the sequences of k elements some extra bits in a certain way. Such codes are called redundant codes, extra bits don't carry any information and are used for error detection and correction. We denote by $[n, k]$ the code, which has a length of code words n and k information bits. Error correcting code is $A \subset \mathbb{F}_2^n$, where $\mathbb{F}_2 = \{0, 1\}$ and codewords are in classical Hamming metric:

$$d(x, y) = |\{i : x_i \neq y_i\}|.$$

In that code we have $r = n - k$ parity checks. The ratio r/n is called *code rate* and is denoted by R_C. It is interesting to look for codes with the best correction properties at the lowest code rate for economic reasons. In 1948 Claude Shannon in his works defined the concept of capacity and proved that there exists code allowing the transfering of information from any small error probability if the rate of information transmission is below the capacity. Let T be the time of transmition of a single bit. Then *the rate of information transmission* is $R_t = \frac{k}{nT}$. Unfortunately, he didn't show a way of constructing such codes. The most known classes of error correcting codes are Turbocodes and Low-Density Parity-Check Codes (LDPC codes). In this article we are only interested in LDPC codes. They were introduced in 1963 by Robert G. Gallanger. These codes have a high possibility of selection of parameters n and r, making it possible to create codes with a large block size and excellent correction properties. Their advantage is the existence of efficient decoding algorithms of linear complexity of the block length n.

LDPC codes can be obtained by few methods but a very good codes can be obtained from families of graphs with certain specific properties. The ability to use graphs to construct error correcting codes was first discussed by Tanner [8]. This is the area where we can work because only specified graphs are suitable for creating a good code. Usually for this purpose, *simple graphs* are used, which means undirected graphs and containing no graph loops or multiple edges. The graph should be bipartite, sparse, without small cycles and biregular or regular with the possibility to obtain biregularity.

There are three ways to represent linear error correcting code allowing us to obtain LDPC codes: generator matrix G, parity check matrix H or Tanner graph $\Gamma(V, E)$. *Parity check matrix* for $[n, k]$ code is $r \times n$ matrix, which words are zeros or ones. Rows of this matrix correspond to the parity checks and the column to codeword bits. If a bit number j in codeword is checked by a parity check number i then the number on a position (i, j) in matrix H is one, otherwise the number is zero. Each bit is checked by a unique set of control equations. In regular LDPC code every row has the same constant weight a and every column has the same constant weight b. Switching columns doesn't change code properties and gives an equivalent code. We assume that every codeword is from the set:

$$\mathcal{C} = \{y \in \mathbb{F}_2^n |\ Hy^T = 0\}.$$

Generator matrix G for $[n, k]$ code is $k \times n$ zeros-ones matrix, which rows create code base. G creates a codeword y for information vector x of length k: $y = x \cdot G$. Each information vector corresponds to exactly one codeword. Parity check matrix and generator matrix are dependent. It is known that if $G = [I_k|A]$ is generator matrix in standard form for the $[n, k]$ code $\mathcal{C}$, then $H = [-A^T|I_{n-k}]$ is a parity check matrix for $\mathcal{C}$.

Bipartite graph we call graph $\Gamma(V, E)$, in which a set of nodes V can be divided into two subsets $V = V_1 \cup V_2$ in such a way that no two vertices from each set V_i, $i = 1, 2$ are connected by an edge. The only connections by edge is from V_1 to V_2.

Tanner graph we call bipartite graph in which one subset V_1 corresponds to codeword bits and the second subset V_2 corresponds to the parity checks. Vertex from the subset V_1 is connected to a vertex from the subset V_2 if and only if a bit corresponding to vertex from V_1 is controled by the parity check corresponding to vertex from V_2. There is a standard way to create LDPC codes depending on adjacency matrix of bipartite, biregular Tanner graph. Parity check matrix H is a part of the adjacency matrix for a graph used to create the code. Adjacency matrix has the form:

$$\begin{pmatrix} 0 & H \\ H^T & 0 \end{pmatrix}$$

Determination of the matrix H is an equivalent code designation.

Codes which have sparse parity check matrices H we call Low-Density Parity-Check Codes (LDPC). Matrix is called a *sparse matrix* if the ratio of number of ones to the number of zeros in each row and column is small compared to the length of the rows and columns. A very primary example of LDPC code is $[7, 4]$ Hamming code with $R_C = \frac{3}{7}$. Code has a sparse matrix H if and only if when it has a representation as sparse Tanner graph. Sparse graph has a small number of edges in comparison to the number of vertices. A simple relationship describing the density of the graph $\Gamma(V, E)$ is

$$g = \frac{2|E|}{|V|(|V| - 1)},$$

where $|E|$ is the number of edges of graph Γ and $|V|$ is the number of verteces. The parameter g can take values from the interval $[0, 1]$. If $g = 1$, then the graph is totally connected.

II. Description of $A(k, K)$

Missing definitions from the Simple Graphs Theory can be find in [2].

A distance between vertices v_1 and v_2 in the graph is the length of minimal path from v_1 to v_2. A graph is connected if for arbitrary pair of vertices v_1, v_2 there is a path from v_1 to v_2. The diameter in a connected, simple graph is the maximum of distances between vertices in the graph and a length of the shortest cycle in a graph is called a *girth*.

We refer to bipartite graph $\Gamma(V_1 \cup V_2, E)$ as biregular one if the number of neighbors for representatives of each partition sets are constants $a+1$ and $b+1$ (bidegrees). We call a graph regular in a case $a = b$.

Let K be a commutative ring. $A(k, K)$ is connected, biregular, bipartite family of graphs of a large girth. The following interpretation of $A(k, K)$ the can be find, for example in [11].

Traditionally one subset of vertices in graphs from family $A(k, K)$ is denoted by $V_1 = P_k$ and called a set of points and another one $V_2 = L_k$ is called a set of lines. Let P and L be two copies of Cartesian power K^n, where n is a positive integer. Brackets and parenthesis allow us to distinguish points and lines.

If $z \in K^n$, then $(z) \in P$ and $[z] \in L$. Let us use the notions for points and lines introduced in [7]:

$$(p) = (p_{0,1}, p_{1,1}, p_{1,2}, p_{2,2}, p_{2,3}, ..., p_{i,i}, p_{i,i+1}, ...),$$
$$[l] = [l_{1,0}, l_{1,1}, l_{1,2}, l_{2,2}, l_{2,3}, ..., l_{i,i}, l_{i,i+1}, ...].$$

The elements of P and L can be thought as infinite ordered tuples of elements from K, such that only a finite number of components is different from zero. We now define an incidence structure (P, L, I) in a following way. We say that the point (p) is incident with the line $[l]$, and we write $(p)I[l]$, if the following relations between their coordinates hold:

$$\begin{cases} l_{i,i} - p_{i,i} = l_{1,0} p_{i-1,i} \\ l_{i,i+1} - p_{i,i+1} = l_{i,i} p_{0,1} \end{cases} \quad (1)$$

We denote this infinite incidence structure (P, L, I) as $A(K)$ and it can be identified with the bipartite incidence graph of (P, L, I), which has the vertex set $P \cup L$ and the edge set consisting of all pairs $\{(p), [l]\}$ for which $(p)I[l]$. $A(K)$ is a infinite tree. For each positive integer $k > 2$ we obtain a finite incidence structure (P_k, L_k, I_k) in a following manner. Firstly, P_k and L_k are obtained from P and L, respectively, by simply projecting each vector onto its k initial coordinates with respect to the natural order. The incidence I_k is then defined by imposing the first k incidence equations and ignoring all others. The incidence graph corresponding to the structure (P_k, L_k, I_k) is denoted by $A(k, K)$. Let $\mathbb{F}_q$ be the finite field containing q elements, where q is prime power. If $K = \mathbb{F}_q$ then we denote $A(k, \mathbb{F}_q)$ simply as $A(k, q)$.

$A(k, K)$ is q-regular but has a structure that allows us to remove points and lines in such a way that we can obtain arbitrary bidegree. We can make it exactly the same as it was done with $D(k, q)$ in [9], when L is a set of all lines and P is a set of all points. To obtain the desired bidegree (a, b) we must put restriction on coordinates. Let $A \subset \mathbb{F}_q$ and $B \subset \mathbb{F}_q$ be an a-element and b-element subsets respectively and let V_P and V_L be sets of points and lines in new bipartite graph. They are the following sets:

$$V_P = \{(p) \in P | c_p \in A\},$$
$$V_L = \{[l] \in L | c_l \in B\},$$

where c_p is fixed coordinate of point and c_l is fixed coordinate of line. If the set of points is bigger than the set of lines then points correspond to codeword bits and lines correspond to parity checks. Otherwise, lines correspond to codeword bits and points correspond to parity checks.

III. Code construction

To create LDPC code with codeword of length n we use $A(k, q)$ such that $k^q > n$. Graph is connected and q-regular. The graph determines how many parity check equations per bit can be used. Each bit should be checked by at least 2 parity check equations. We denote the number of parity checks by d. We reduce the bidegree to (d, q) by the method shown above. We can put restrictions on the value of first or other coordinates of lines and points. Bidegree reduction can only

increase the girth so there is no risk that short cycles occur. After bidegree reduction the graph can become disconnected and divide into several components. Choose one vertex and take the component containing this chosen vertex (point or line) and find all other vertices, for which there is a path to the chosen one. We use this component to create a parity check matrix. If $|V_P| > |V_L|$ then points correspond to codeword bits and lines to parity checks, if not then lines correspond to codeword bits and points to parity checks. We decide to put one or zero on a position (i, j) in parity check matrix by checking if the relations (1), between vertex number i from set V_P and vertex number j from set V_L, are satisfied.

For $k \leq 3$ graphs $D(k, q)$ and $A(k, q)$ are isomorphic. They are different for $k \geq 4$ and lead us to different codes. For example, graphs $D(k, q)$ are disconnected for $k \geq 6$ when $A(k, q)$, $q \neq 2$ have compact structure. The structure of $A(k, q)$ after biregularity reduction allows us to obtain codes of bigger block size than for codes obtained from $D(k, q)$. For example, graph $A(8, 5)$ after the reduction of bidegrees to 2 and 5 splits into 125 components and $D(8, 5)$ into 625. $A(10, 3)$ after the reduction of bidegrees to 2 and 3 splits into 81 components and $D(10, 3)$ into 243. When we use bigger field we obtain better code rate (more economic). Reducting bidegrees to 2 and q gives code rate $\frac{2}{q}$. Obviously, for each code we can change the bidegrees of a graph to $\frac{a}{b}$, where $2 \leq a, b \leq q$, but then the code rate can increase. A good example is a case of bidegrees 3 and q.

IV. Results

Transmission quality depends mainly on code, decoding algorithm and the level of noise in a communication channel. Properties of error correcting codes are tested by determining the relationship between noise level and bit error rate. Bit error rate (BER) is the ratio of number of error bits to the total number of transferred bits. Simulation usually is carried out for Gaussian Channel where noise is modeled by White Gaussian Noise. Our simulations were done using BPSK modulation over AWGN channel and simple MAP decoder implementation. Let y be the received codeword. MAP decoder works according to the rule which returns an output value $\hat{x}$ of a code word x for which the *a posterior* probability $P = (x|y, H)$

TABLE I
Properties of graphs after receiving bidegree $(2, s)$ used for presented sample codes

Initial graph	Biregularity	Number of lines in fixed component	Number of points in fixed component	Code rate
$A(6, 6)$	$(2, 6)$	648	216	$0.(3)$
$A(6, 7)$	$(2, 7)$	2401	686	≈ 0.286
$A(8, 5)$	$(2, 5)$	3125	1250	0.4
$A(10, 3)$	$(2, 3)$	729	486	$0.(6)$
$D(6, 6)$	$(2, 6)$	216	72	$0.(3)$
$D(6, 7)$	$(2, 7)$	2401	686	≈ 0.286
$D(8, 5)$	$(2, 5)$	625	250	0.4
$D(10, 3)$	$(2, 3)$	243	162	$0.(6)$

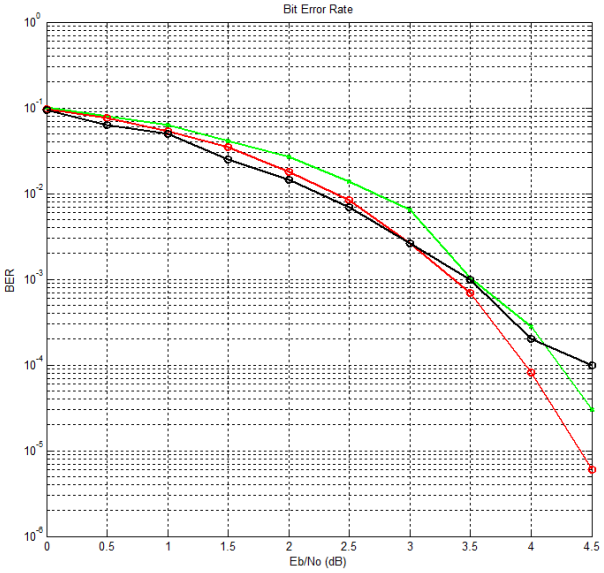

Fig. 1. Bit error rate for $[2401, 686]$ code (green) based on $A(6, 7)$, $[3125, 1250]$ code (red) based on $A(8, 5)$ and $[729, 486]$ code (black) based on $A(10, 3)$

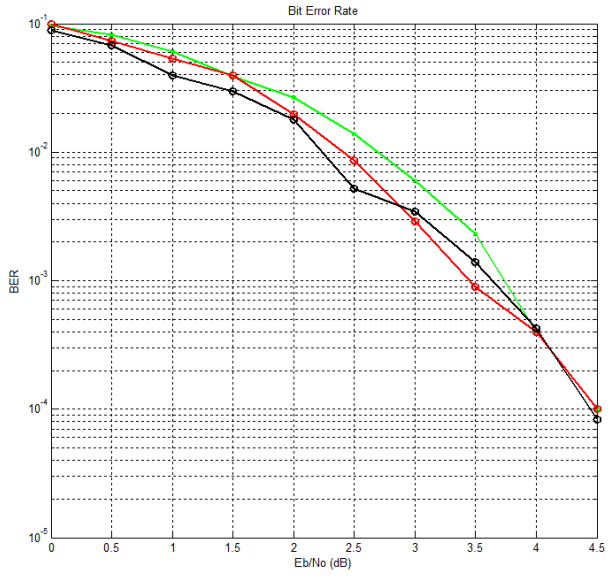

Fig. 2. Bit error rate for $[2401, 686]$ code (green) based on $D(6, 7)$, $[625, 250]$ code (red) based on $D(8, 5)$ and $[243, 162]$ code (black) based on $D(10, 3)$

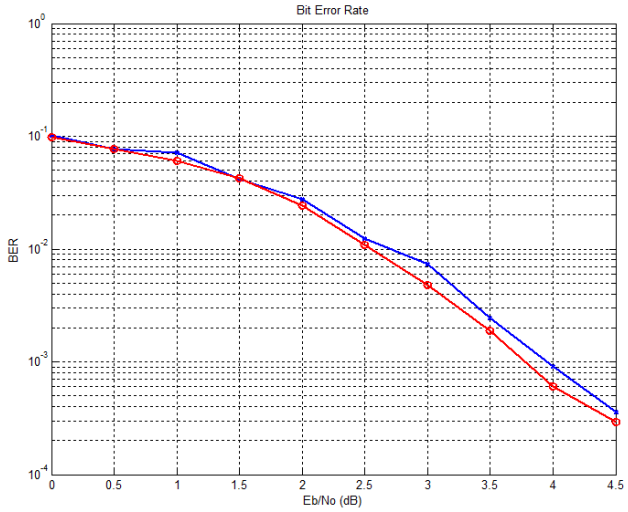

Fig. 3. Bit error rate for $[648, 216]$ code (red) based on $A(6, 6)$, $[625, 250]$ code (blue) based on $D(6, 6)$

is maximized. Table 1. shows properties of sample described codes. In order to compare quality of codes Fig. 1 shows codes, which based on some representatives of family $A(k, q)$ and Fig. 2 shows codes based on some representatives of family $D(k, q)$, with the same parameters accordingly. We see that codes based on representatives of family $A(k, q)$ have better error correcting properties. This fact is supported by a dozen other conducted simulations.

V. Remarks

Instead of using $\mathbb{F}_q$ as K we can use ring $\mathbb{Z}_n$ and modulo operations. Modified codes where rings are used, based on subgraph of $A(k, \mathbb{Z}_n)$ give better code than those based on subgraph of $D(k, \mathbb{Z}_n)$ (Fig. 3 shows results for fixed parameters).

References

[1] B. Bollobas, Extremal Graph Theory. Academic Press, 1978.

[2] A. Brower, A. Cohen, A. Nuemaier, Distance regular graphs, Springer, Berlin, 1989.

[3] R. G. Gallager, Low-Density Parity-Checks Codes, IRE Trans of Info Thy 8 (Jan 1962):21–28.

[4] P. Guinand, J. Lodge, Graph theoretic construction of generalized product codes, IEEE International Symposium on Information Theory ISIT'97 Ulm, Germany (June 29-July 4 1997):111.

[5] P. Guinand, J. Lodge, Tanner type codes arising from large girth graphs, Canadian Workshop on Information Theory CWIT '97, Toronto, Ontario, Canada (June 3-6 1997):5–7.

[6] W. C. Huffman, V. Pless, Fundamentals of error correcting codes, first edition, Cambridge University Press, Cambridge, 2003.

[7] F. Lazebnik, V. A. Ustimenko, A. J. Woldar, A characterization of the components of the graphs $D(k, q)$, Discrete Mathematics Vol. 157 (1996):271–283.

[8] F. Lazebnik, V. A. Ustimenko and A. J. Woldar, A new series of dense graphs of high girth , Bulletin (New Series) of the AMS Vol. 32, Number 1 (1995):73–79.

[9] F. Lazebnik, V. A Ustimenko, Explicit construction of graphs with an arbitrary large girth and of large size, Discrete Applied Mathematics Vol. 60 (1995):275–284.

[10] F. Lazebnik, V. A. Ustimenko, New examples of graphs without small cycles and of large size, European Journal of Combinatorics Vol. 14 (1993):445–460.

[11] U. Romańczuk , V. A. Ustimenko, On Extremal Graph Theory, explicit algebraic constructions of extremal graphs and corresponding Turing encryption machinesŤ, in "Artificial Intelligence, Evolutionary Computing and Metaheuristics , In the footsteps of Alan Turing Series: Studies in Computational Intelligence Vol. 427, Springer, June, 2012.

[12] C. E. Shannon, A Mathematical Theory of Communication, Bell System Technical Journal Vol. 27 (1948):379–423, 623–656.

[13] C. E. Shannon, W. Warren, The Mathematical Theory of Communication, University Of Illinois Press, 1963.

[14] T. Shaska, W. C. Huffman, D. Joener and V. Ustimenko (editors), Advances in Coding Theory and Cryptography, Series on Coding and Cryptology Vol. 3 (2007):181–200.

[15] T. Shaska, V. A. Ustimenko, On some applications of graph theory to cryptography and turbocoding, Special issue of Albanian Journal of Mathematics: Proceedings of the NATO Advanced Studies Institute "New challenges in digital communications" Vol. 2, Issue 3, University of Vlora (2008):249–255.

[16] A. Shokrollahi, LDPC Codes: An Introduction, Digital Fountain Inc, Fremont (2002), available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.110.1008.

[17] R. M. Tanner, A recursive approach to low density codes, IEEE Transactions on Information Theory IT 27(5) (1984):533–547.

[18] V. A. Ustimenko, On the extremal regular directed graphs without commutative diagrams and their applications in coding theory and cryptography, Albanian. J. of Mathematics, Special Issue Algebra and Computational Algebraic Geometry Vol. 1, Number 4 (2007):387–400.

[19] V. A. Ustimenko, A. Woldar, Extremal properties of regular and affine generalized polygons as tactical configurations, European Journal of Combinatorics Vol. 24 (2003):99–111.