"Science Stays True Here"
Journal of Mathematics and Statistical Science, Volume 2015, 75-88 | Science Signpost Publishing

Message Passing Decoding of Codes from Complete Graphs

Francisco Chamera

Lilongwe University of Agriculture and Natural Resources, P. O Box 219, Lilongwe. Malawi.

Khumbo Kumwenda

Mzuzu University, Private Bag 201, Luwinga. Mzuzu 2. Malawi.

Abstract

We describe iterative decoding of binary codes from incidence matrices of complete graphs. Parameters for these codes are well known. The codes are also known to be low density parity-check (LDPC). We determine cases where they are decodable by bit flipping (BF) and sum product (SP) decoding algorithms.

Let c be a codeword from the binary code from an incidence matrix of a complete graph. Suppose c is sent through the binary symmetric channel (BSC) with parameter p. Let N and k be the length and dimension of the code respectively. We show that errors occurring in the first k positions are correctable by SP while those occurring in the last $N-k$ positions are correctable by BF.

Keywords: Binary Symmetric Channel, Bit flipping; Complete graphs; Incidence matrix; LDPC codes; Linear code; Sum product; Tanner graph.

Introduction

Low-density parity-check (LDPC) codes were introduced by Robert Gallager [4] in 1960. This is a class of linear block codes fully described by their parity check matrices. A parity check matrix for an LDPC code contains only a small number of nonzero entries. These codes are represented by Tanner

Corresponding author:

Francisco Chamera, Lilongwe University of Agriculture and Natural Resources, P. O Box 219, Lilongwe. Malawi. E-mail: fchamera@gmail.com.

graphs introduced in [12].

In this paper we describe binary LDPC codes from incidence matrices of complete graphs. These codes are decoded by permutation decoding in literature [9]. However, the permutation decoding sets (PD-sets) found are not close enough to the Gordon bound described in [8], [10] and [11]. Therefore, it makes sense to consider other decoding algorithms.

LDPC codes are preferred because of their iterative decoding algorithms. Some examples of the general message-passing decoding algorithms are bit flipping (BF) and sum product (SP). We have been successful in generalising that both BF and SP algorithms work for binary codes from incidence matrices of complete graphs. In both cases we have used the binary symmetric channel with parameter p.

In iterative decoding, messages are passed between bit nodes and check nodes of the Tanner graph until a valid codeword is found or the number of iterations is exhausted and the decoder gives up. We show that only one iteration is required to converge to a sent codeword.

The rest of this paper is organised as follows. In Section 2 we present background review of graphs and related codes which are of interest to us. We introduce LDPC codes and their decoding. Finally in Section 3 we describe the codes from incidence matrices of complete graphs. We give the result on the girth of the Tanner graphs of the codes before we conclude with a look at the decoding of the codes.

Preliminaries

A **graph** is a pair $\Gamma = (V, E)$ of sets such that $E \subseteq V^{\{2\}}$, where $V^{\{2\}}$ is the set of 2-element subsets of the set V [2]. $V = V(\Gamma)$ is the vertex-set of Γ while $E = E(\Gamma)$ is its edge-set. If $e = \{u, v\} \in E$ then we write $e = [u, v]$. In this case u and v are **adjacent** and are **incident** with the edge $[u, v]$. The **degree** (or **valency**) of a vertex v is the number of vertices adjacent to it. A k **-regular** (or simply regular) graph is a graph in which all vertices have the same degree k.

A **walk** of length n in a graph Γ is a sequence $v_1, v_2, ..., v_n$ of vertices such that $[v_{i-1}, v_i] \in E$ for $1 \le i \le n$ [1]. A walk is **closed** if it starts and ends at the same vertex. It is a **trail** if all edges appearing in it are distinct. If all vertices in a walk are different, then a walk is called a **path**. If there exists a path between any two vertices of a graph Γ, then the graph is said to be **connected**. A **cycle** is a closed trail with no repeated vertices other than the starting and ending vertices. The **girth** of a graph is the size of its smallest cycle.

Let X be a non-empty set. A **complete graph** K_X is a graph on X in which every pair of

vertices is adjacent. If $|X|=n$ then K_X is denoted by K_n. The complete graph K_n has $\binom{n}{2}$ edges.

It is well known that a complete graph is super-λ. If the vertex-set of a graph can be partitioned into two non-empty subsets U and V such that each edge has one endpoint in U and another endpoint in V, then the graph is **bipartite**. Bipartite graphs are characterised by the property that they have no odd cycles.

An **incidence matrix** $B=(b_{ij})$ of a graph Γ is a matrix whose rows and columns are indexed by vertices and edges respectively and $b_{ij}=1$ if the ith vertex is incident with the jth edge and $b_{ij}=0$ otherwise.

A binary $[N,k,d]$ **linear code** C is a k-dimensional subspace of $\mathbb{F}_2^N$, the vector space of N-tuples over a finite field $\mathbb{F}_2$ [5]. The **hamming distance** between any two codewords of C is at least d. A generator matrix of a linear code is a matrix whose rows form a basis for the code. A generator matrix G is in **standard form** if it is written in the form

$$G=\left(I_k|X\right) \tag{1}$$

where I_k is a $k\times k$ identity matrix and X is some matrix.

The **dual** of C is the set $C^{\perp}$ of vectors which are orthogonal to elements of C, i.e.,

$$C^{\perp}=\{v\in\mathbb{F}_q^N:(c,v)=0 \text{ for all } c\in C\}$$

where (,) denotes the standard inner product in $\mathbb{F}_2^N$ [6]. A generator matrix for $C^{\perp}$ is called a **parity-check matrix** for C. If G is in form (1), then a parity check matrix is found to be

$$H=\left(X^T|I_{N-k}\right). \tag{2}$$

For all $c=(c_1,...,c_N)\in C$

$$Hc^T=0^T. \tag{3}$$

This follows from the definition of H.

The matrix multiplication in Equation (3) gives rise to **check equations** in the form

$$c_l+...+c_m=0 \tag{4}$$

where $1 \le l \le .... \le m \le N$ depending on non-zero positions of the entries in each row of H.

The binary symmetric channel (BSC), with **crossover probability** (parameter) $p < 1/2$, is a memoryless communication channel with channel alphabet $\{0,1\}$ and transition probabilities

$$\text{P(1 received}|\text{0 sent)} = \text{P(0 received}|\text{1 sent)} = p,$$

$$\text{P(1 received}|\text{1 sent)} = \text{P(0 received}|\text{0 sent)} = 1-p.$$

A low-density parity-check **(LDPC)** code is a linear code with a sparse parity check matrix. It is represented by a bipartite graph called **Tanner graph**. A Tanner graph has two sets of vertices: N vertices for the codeword bits (called **bitnodes**), and $N-k$ vertices for the parity check equations (called **check nodes**). An edge joins a bit node to a check node if that bit is included in the corresponding parity-check equation.

An LDPC code is (w_c, w_r)**-regular** if in each column of its parity-check matrix there are w_c non-zero entries and each row has w_r non-zero entries, otherwise it is **irregular**.

LDPC codes are decoded by passing of messages along the edges of a Tanner graph. This is why the decoding algorithms are collectively called **message-passing** algorithms.

The first step in BF decoding is to make a binary (hard) decision about each received bit before passing it to the decoder. This means changing each symbol to binary. In this conversion a negative number corresponds to the bit 1 while a positive number corresponds to the bit 0

Each binary bit of the received word is initialised to the corresponding bit node. Each bit node sends a message to all its connected check nodes. This message, labelled M_i for i-th bit node, is the value of the bit itself, i.e., each bit node sends a '0' or a '1'. Upon the receipt of messages from various bit nodes, each check node determines that its parity-check equation is satisfied if the XOR of the incoming bit values is zero.

The notation B_j is used to represent the set of bit nodes connected to the j-th check node. Each check node sends a message to each connected bit node, declaring what value the bit is based on the information available to the check node. This message, labelled $E_{j,i}$ for the message from j-th check node to i-th bit node, declares the value of the i-th bit '1' or '0' as determined by the j-th check node. The j-th check node with parity-check equation $c_i + c_l + ... + c_m = 0$ sends the message

$$E_{j,i} = \sum_{i' \in B_j, i' \neq i} M_{i'}$$

to each i-th bit node. As usual, the addition here is done over the binary field.

We use A_i to represent the set of check nodes connected to the i-th bit node. After receiving the messages from all the check nodes connected to it, each bit node determines the number of messages that differ with its received value. If the majority of messages received are different from its received value, the bit node **flips** its current value.

The process described above is repeated until all the parity-check equations are satisfied which means that bits in the bit nodes form the most likely codeword sent. Often times the number of maximum iterations, which is believed to produce a valid codeword, is set. If after such iterations, no valid codeword is produced, the decoder gives up.

The bit flipping decoding algorithm is presented in [7]. Input is the hard decision on the received vector and the maximum number of iterations.

The sum product (SP) algorithm is similar to BF algorithm described above. The major difference between the two is the type of messages passed between nodes. Unlike BF, SP is a soft decision message-passing algorithm which accepts real numbers as inputs. These numbers are expressed as log likelihood ratios (LLRs).

Given the received vector y, before passing it to the decoder, the a priori LLR r_i is calculated for each received bit y_i. For the BSC with crossover probability p,

$$r_i = \begin{cases} \ln \dfrac{p}{1-p} & \text{if } y_i = 1, \\ \ln \dfrac{1-p}{p} & \text{if } y_i = 0. \end{cases}$$

To begin decoding, the i-th bit node sends the message $M_{j,i} = r_i$ to the j-th check node connected to it.

After receiving various messages from its connected bit nodes, each check node sends messages depending on the information available to it. The j-th check node sends the following message to the i-th bit node:

$$E_{j,i} = \ln \left(\frac{1 + \prod_{i' \in B_j, i' \neq i} \tanh(M_{j,i'} / 2)}{1 - \prod_{i' \in B_j, i' \neq i} \tanh(M_{j,i'} / 2)} \right). \tag{5}$$

After receiving messages from all the neighbouring check nodes, LLR value L_i is calculated on each bit node. Since each bit has access to the input a priori LLR, r_i, and the LLRs from every connected

check node, the total LLR of the i th bit is the sum of these LLRs:

$$L_i = r_i + \sum_{j \in A_i} E_{j,i}. \tag{6}$$

At this stage we have a vector L whose length is the length of the code. This vector is converted to a binary vector Z using hard decision. To see if this is a valid codeword, we check whether $ZH^T = 0$ is satisfied. If it is not satisfied, we begin another iteration.

For the second and subsequent iterations, the messages sent from the bit nodes to the check nodes, $M_{j,i}$, are not the full LLR value for each bit. To avoid sending back to each check node information which it already has, the message from the i th bit node to the j th check node is the sum in Equation (6) minus the component $E_{j,i}$ which was just received from the j th check node:

$$M_{j,i} = r_i + \sum_{j' \in A_i, j' \neq j} E_{j',i}. \tag{7}$$

SP decoding algorithm is presented in [7]. Input is the LLRs for the a priori message probabilities.

The Binary LDPC Codes $C_2(B_n)$ from Incidence Matrices of Complete Graphs K_n

We give properties of the code $C_2(B_n)$ from an incidence matrix of a complete graph K_n $n \geq 4$.

The following theorem is a summary of our results.

Theorem 3.1. Let $C_2(B_n)$ be the binary code from an incidence matrix of a complete graph K_n.

1. $C_2(B_n)$ is an irregular LDPC code.

2. The Tanner graph for $C_2(B_n)$ has girth 6.

3. Let $c = c_1, ..., c_N \in C_2(B_n)$ be sent through the binary symmetric channel (BSC) with parameter p. Then

   (a) sum product decoding algorithm corrects the errors if they occur in the first $k = dim(C_2(B_n))$ positions of c and $p \leq 0.22$. Furthermore, correction of errors occurs after only a single iteration.

   (b) any errors occurring in the last $N - k$ positions are not correctable by sum product.

   (c) any errors are correctable by bit flipping if and only if they occur in the last $N - k$ positions. Successful decoding is also achieved after performing only a single iteration.

The proof of Theorem 1 follows from Propositions 2 and 3 and Theorems 4, 5, 6 and 7.

Let B_n be an incidence matrix of a complete graph K_n. We write B_n as follows. Rows of B_n are indexed by vertices $1,2,...,n$, in that order, while the columns of B_n are indexed by edges $[1,2],...,[1,n],[2,3],...,[2,n],...,[n-1,n]$, in that order. Hence

$$B_3 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

For $n \geq 4$, the matrix B_n has size $n \times \binom{n}{2}$ and takes the form

$$B_n = \left(\begin{array}{c|c} 11....1 & 00....0 \\ \hline I_{n-1} & B_{n-1} \end{array} \right) \tag{8}$$

where I_{n-1} is an identity matrix of size $n-1$ and B_{n-1} is an incidence matrix of K_{n-1}.

Since B_n is made up of weight-2 column vectors, its rows are linearly dependent over $\mathbb{F}_2$. However, removing any single row gives a matrix whose rows are linearly independent. We remove the first row and let $C_2(B_n)$ be generated by the remaining $n-1$ rows. Hence the remaining rows of B_n have the form

$$B_n^* = \left(I_{n-1} | B_{n-1} \right). \tag{9}$$

The code $C_2(B_n)$ is an $[\binom{n}{2}, n-1, n-1]_2$ linear code [9].

By Equation (2) a parity-check matrix H for $C_2(B_n)$ has the form

$$H = \left(B_{n-1}^T | I_{\binom{n-1}{2}} \right). \tag{10}$$

Observe that the weights of columns of parity check matrix H are different. Hence the following proposition is immediate.

Proposition 3.2. The binary code from incidence matrix of a complete graph $C_2(B_n)$ is an irregular LDPC code.

However, note that each of the two sub-matrices B_{n-1}^T and $I_{\binom{n-1}{2}}$ in H is regular. It is also clear that every row of H is a weight-3 vector.

Existence of a 4-cycle in a Tanner graph comes as a result of two columns of a parity check matrix overlapping in two positions [3]. This implies that after performing column (or row) permutations in the parity-check matrix, we have a sub-matrix of the form

$$D = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}. \tag{11}$$

Proposition 3.3. Let $\mathcal{T}_n$ be the Tanner graph for $C_2(B_n)$. Then $\mathcal{T}_n$ has girth 6.

Proof.

We will show that $\mathcal{T}_n$ does not have a 4-cycle. The result follows from the fact that $\mathcal{T}_n$ has a 6-cycle.

It is sufficient to show that no column or row permutations of H as given in Equation (10) give a sub-matrix of the form D given in Equation (11).

Observe that

$$D^T = D. \tag{12}$$

If the sub-matrix D exists then it must come from columns of B_{n-1}^T. With Equation (12), this means that D is also a sub-matrix of B_{n-1}.

Since B_{n-1} is an incidence matrix of K_{n-1}, this implies that two distinct edges of K_{n-1} are incident with the same pair of vertices of K_{n-1} which is impossible. Hence H does not have a sub-matrix of the form D.

We now show that T_n has a 6-cycle. Notice that a sub-matrix

$$E = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \tag{13}$$

of any parity-check matrix of an LDPC code results into a 6-cycle in the corresponding Tanner graph. We show that E is a sub-matrix of H. Observe that

$$E^T = E. \tag{14}$$

Since $E = B_3$, we conclude that E is a sub-matrix of both B_{n-1} and B_{n-1}^T for any $n \geq 4$. This implies that E is a sub-matrix of H which completes the proof. □

3.1. Message Passing Decoding of $C_2(B_n)$

We look at the extent to which either BF or SP can be used in decoding $C_2(B_n)$. Consider the first $k = n-1$ positions and the last $N-k$ positions of the sent codeword $c_1,...,c_N \in C_2(B_n)$. On whether a particular algorithm corrects errors or not depends on the position in which the errors occur.

Theorem 3.4. Let $c_1,...,c_N \in C_2(B_n)$ be sent through binary symmetric channel with parameter p. Then any errors occurring in the last $N-k$ positions are not correctable by SP decoding.

Proof. Let $y_1,..,y_N$ be the received vector. Without loss of generality let c_m be the symbol received in error. We consider the case where $c_m = 0$ and $y_m = 1$ since the proof of the case where $c_m = 1$ and $y_m = 0$ is similar. In this case the value of r_m is negative. We consider the value of L_m in Equation (6) after the first iteration. Recall that the m-th bit node is connected to only one check node so that we have

$$L_m = r_m + E_{j,m}. \tag{15}$$

If L_m is positive, the hard decision gives symbol '0' in the vector Z described above. This would mean that the error has been corrected. We claim that L_m is negative so that the symbol y_m does not change.

To see that this is true, observe that by Equation (5) we have

$$| E_{j,m} | < | M_{j,m} | = | r_m | \tag{16}$$

in the first iteration. Since r_m is negative, the sign of L_m is negative regardless of whether $E_{j,m}$ is positive or negative. In the next iteration the decoder starts with the same received symbol y_m which will again not be corrected. This continues until maximum number of iterations is attained. This completes the proof. □

Suppose $c_1,...,c_k,c_{k+1},....,c_N \in C_2(B_n)$ is sent through the binary symmetric channel. Let $c_i + c_l + c_m = 0$ be a typical check equation. In the first iteration of SP decoding we have

$$| r_i | = | M_{j,i} | = | M_{j,l} | = | M_{j,m} | = \ln \frac{1-p}{p}.$$

Write Equation (5) as

$$E_{j,i} = \ln\left(\frac{1+\tanh(M_{j,m}/2)\tanh(M_{j,l}/2)}{1-\tanh(M_{j,m}/2)\tanh(M_{j,l}/2)}\right) = \ln\left(\frac{1+\tanh(r_i/2)\tanh(r_i/2)}{1-\tanh(r_i/2)\tanh(r_i/2)}\right). \quad (17)$$

In Table 1 we give a few values of p and corresponding values of $|r_i|$ and $|E_{j,i}|$. All values are expressed to 4 decimal places. From the table the following Equation appears to hold for values of p satisfying $p \le 0.22$:

$$2|E_{j,i}| > |r_i|. \quad (18)$$

The following Theorem is based on the assumption that the parameter of the BSC satisfies $p \le 0.22$.

p	$\lvert r_i \rvert$	$\lvert E_{j,i} \rvert$	$2\times\lvert E_{j,i} \rvert$
0.4	0.4055	0.08	0.16
0.35	0.6190	0.1805	0.361
0.3	0.8473	0.3228	0.6456
0.25	1.0986	0.5108	1.0216
0.23	1.2083	0.6006	1.2012
0.22	1.2656	0.6491	1.2982
0.21	1.3249	0.7001	1.4002
0.20	1.3863	0.7538	1.5075
0.1	2.1972	1.5164	3.0328

Table 1. Values of p, $|r_i|$ and $|E_{j,i}|$

Theorem 3.5. Let $c_1,...,c_k,c_{k+1},....,c_N \in C_2(B_n)$ be sent through the binary symmetric channel with parameter $p \le 0.22$. Then sum product decoding corrects the errors that occur in the first k positions after a single iteration.

Proof. Let $y_1,..,y_N$ be a received vector. Without loss of generality let c_i be the distorted symbol where $1 \le i \le k$. We only consider the case where $c_i = 0$ and $y_i = 1$ since the proof of the case where $c_i = 1$ and $y_i = 0$ is very similar. In this case $r_i = \ln\frac{1-p}{p}$ is negative, since $p < 1-p$. We need to

determine the sign of

$$L_i = r_i + \sum_{j \in A_i} E_{j,i}.$$

Since each check node has degree 3, let

$$c_i + c_l + c_m = 0$$

be the parity-check equation at the j-th check node. Since $c_i = 0$, either $c_m = c_l = 0$ or $c_m = c_l = 1$. If both c_m and c_l are correctly transmitted, then either $y_m = y_l = 0$ or $y_m = y_l = 1$. By Equation (17), $E_{j,i}$ is positive (a sign different from that of r_i). The situation is the same if both c_m and c_l are received in error. However, if one of c_m and c_l is received in error the sign of $E_{j,i}$ is negative.

Since $|A_i| = n-2$, the number of $E_{j,i}$'s is $n-2$. Two situations make L_i to be negative:

a. $\sum_{j \in A_i} E_{j,i}$ is negative

b. $|r_i| > |\sum_{j \in A_i} E_{j,i}|$.

The first case is impossible, otherwise the number of $E_{j,i}$s that are negative exceeds the number of those that are positive. This would mean that more than half of $n-2$ symbols are received in error.

By Equation (18) the second case is only possible if the difference between negative $E_{j,i}$s and positive ones is less than 2. We claim that the difference is at least 2. To see that this is true let a be the number of positive $E_{j,i}$s and b be the number of negative $E_{j,i}$s. If $n-2$ is even, then clearly $a-b$ is not 1. Also this difference can not be equal to 0 since this would mean that half of $n-2$ c_ms (or c_ls) are received in error, and plus c_i we have more than a correctable number of errors. Similarly for odd $n-2$, $a-b$ can not be equal to 0 or 1.

This shows that L_i is positive. Hence the hard decision applied to the vector L changes the bit y_i from one back to zero. This completes the proof. □

While SP algorithm corrects errors occurring in the first k positions, BF algorithm does the opposite. As in the following Theorem, the errors are not corrected until all iterations are exhausted.

Theorem 3.6. Let $c_1, ..., c_N \in C_2(B_n)$ be sent through binary symmetric channel with crossover probability p. Then errors occurring in any of the first k positions are not correctable by bit flipping decoding algorithm.

Proof. Let $y_1, .., y_N$ be received. Without loss of generality suppose c_i is received in error where $1 \le i \le k$. We only consider the case where $c_i = 0$ and $y_i = 1$ since the proof of the case where $c_i = 1$

and $y_i = 0$ is similar. Let

$$c_i + c_l + c_m = 0$$

be a check equation involving c_i. Recall that there are $n-2$ such equations for each bit c_i. $c_i = 0$ means $c_l = c_m = 0$ or $c_l = c_m = 1$.

If both c_l and c_m are sent correctly, then $y_l = y_m = 0$ or $y_l = y_m = 1$. Consider the message

$$E_{j,i} = y_l + y_m$$

sent from the j-th check node to the i-th bit node. Clearly $E_{j,i}$ is zero, a value different from the received value y_i. However c_m has degree 1 which means it receives only one message

$$E_{j,m} = y_i + y_l.$$

If both y_l and y_m are 1, $E_{j,m} = 1 + 1 = 0$ and if both y_l and y_m are 0 $E_{j,m} = 1 + 0 = 1$. This means that c_m flips its current value, as $E_{j,m}$ is the only message received.

Suppose the bit node c_i receives more messages which are different from its originally received value forcing it to change its value. Then after the first iteration the new vector will have zero which is the originally sent symbol. However, because of the change in value at c_m, the resulting vector will not be a valid codeword. After the next iteration c_m returns to its original value while c_i will have a wrong value. The situation continues in the other iterations until a maximum number of iterations is achieved.

If one of c_l and c_m is received in error, then $E_{j,i} = 0+1 = 1 = y_i$ which is still a wrong value.

If both c_l and c_m are received in error then $E_{j,i} = 0$ and $E_{j,m} = 0$ if $y_l = y_m = 1$ and $E_{j,m} = 1$ if $y_l = y_m = 0$. In this case the error at c_m is corrected.

We claim that in more than $\frac{n-2}{2}$ of the check equations, both of c_l and c_m are correctly transmitted. Otherwise there are more errors than a decoder can correct. These parity check equations make the error not to be corrected as explained above. We conclude that such an error is always not corrected. This completes the proof. □

The following Theorem states that BF corrects errors occurring in any of the last $N-k$ positions. However the error symbol is corrected if the other symbols in its check equation are both transmitted either correctly or incorrectly.

Theorem 3.7. In the transmitted codeword $c_1,...,c_N \in C_2(B_n)$ let c_m, $k+1 \le m \le N$ be received in error. Suppose $c_i + c_l + c_m = 0$ is the check equation involving c_m. If both c_i and c_l are either transmitted correctly or incorrectly, then bit flipping decoding algorithm corrects c_m after a single iteration.

Proof. Let $y_1,..,y_N$ be received. It suffices to consider the case where $c_m = 0$ and $y_m = 1$. Since c_i and c_l are both transmitted either correctly or incorrectly, we have

$$c_i + c_l = y_i + y_l = 0.$$

Hence the message from the j-th check node to the m-th bit node in the first iteration is

$$E_{j,m} = y_i + y_l = 0.$$

The m-th bit node, therefore, changes its value from the received 1 to 0 since $E_{j,m}$ is the only message it receives. This completes the proof. □

References

[1]. A. Dickson, Introduction to Graph Theory, Available from http://www.math.utah.edu/mathcircle/notes/MC_Graph_Theory.pdf, 2006.

[2]. R. Diestel, Graph Theory, New York: Springer, 1997.

[3]. J. Fan, K. Kim and Y. Xiao, Design LDPC codes without cycles of length 4 and 6, Hindawi Publishing Cooperation, Beijing, 2008.

[4]. R. G. Gallager, Low-Density Parity-Check codes, Cambridge, MA: MIT Press, 1963.

[5]. R. Hill, A First Course in Coding Theory, Oxford: Oxford University Press, 1986.

[6]. W. C. Huffman and V. S. Pless, Fundamentals of Error-Correcting codes, Cambridge: Cambridge University Press, 2003.

[7]. S. J. Johnson, Introducing Low-Density Parity-Check Codes, Available from http://sigpromu.org/sarah/SJohnsonLDPCintro.pdf.

[8]. J. D Key, D. Moori and B. G Rodrigues, Permutation decoding sets for the binary codes from triangular graphs, European J. Combin. 25 (2004), 113-123.

[9]. J. D. Key, P. Dankelmann and B. Rodrigues, Codes from Incidence Matrices of graphs, Available from http://www.ces.clemson.edu/ keyj/Key/JDK_3ICTMA.pdf .

[10]. H. Kroll and R. Vincenti, Antiblocking decoding, Discrete Appl. Math. 158 (2010) 1461-1464.

[11]. F. J. MacWilliams, Permutation decoding of systematic codes, Bell system Tech. J., 43 (1964) 485-505.

[12]. M. Tanner, A recursive approach to low complexity codes, IEEE Trans. Inform. Theory, 27, (1981), 533-547.