APPLICATIONS OF GRAPH THEORY TO ERROR CORRECTING CODES

by

Yeşim İmamoğlu

B.S. in Math. Eng., İstanbul Teknik Üniversitesi, 1997

Submitted to the Institude for Graduate Studies in
Science and Engineering in Partial fulfillment of
the requirements for the degree of
Master of Science
in
Mathematics

Boğaziçi University

2001

ACKNOWLEDGEMENTS

I would like to thank my thesis supervisor Prof. Haluk Oral for his support, guidance and encouragement thruogh this study.

I would also like to thank Prof. Erol Balkanay and Prof. Yılmaz Akyıldız for their partipication in my thesis comitee.

I am deeply grateful to my friends Dr. Selda Küçükçiftçi and Hayri Ardal for their editorial comments on the text. I would also like to express my gratitude to Yunus Dönmez and Sinan Işık for their technical support.

Finally, I want to thank my family, my friends Betül Çelik and Müge Taşkın and all the teaching assistants in Mathematics Department for their endless support and understanding during the completion of this thesis.

ABSTRACT

APPLICATIONS OF GRAPH THEORY TO ERROR CORRECTING CODES

Graph Theory has applications in many different fields, especially in combinatorics. In this study, we investigate the methods developed for obtaining error-correcting codes using graphs.

First, the codes obtained from cycle and cut-set spaces of a graph are considered. After constructing the codes and giving the decoding schemes, methods for increasing the dimensions of these codes are examined. Then decoding schemes for these new codes are given.

Next, a method for obtaining self-dual codes using cubic planar bipartite graphs is examined.

The last method covered is to obtain perfect one error-correcting codes using some graphs that are constructed from the Tower of Hanoi Puzzle.

ÖZET

GRAF TEORİSİNİN HATA DÜZELTEN KODLARA UYGULANMASI

Graf teorisinin başta kombinatorik olmak üzere birçok değisik alanda uygulamaları bulunmaktadır. Bu çalışmada, graflar yardımı ile hata düzelten kodlar elde etmek için geliştirilen bazı metodları araştırıyoruz.

İlk olarak bir grafın döngü ve kesen küme uzaylarından elde edilen kodlar üzerinde duruluyor. Kodlar kurulduktan ve hata düzeltme algoritmaları verildikten sonra, bu kodların boyutlarının arttırılması ile ilgili metodlar inceleniyor. Son olarak bu yeni elde edilen kodların hata düzeltme algoritmaları veriliyor.

İncelenen ikinci method kübik düzlemsel iki parçalı graflardan kendi duali olan kodlar elde edilmesidir.

Son olarak mükemmel tek hata düzelten kodlar elde etmek için geliştirilmiş bir metod inceleniyor. Bunun için Hanoi Kulesi probleminden yola çıkarak olusturulmuş bazı graflar kullanılmıştır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

$A_c(x, y)$	Weight enumerator of C
$B(G)$	Cut-set space of G
B_f	Fundamental cycle matrix
C	Code
$C^{\perp}$	Dual code of C
$C_E(G)$	Cycle space G
$d_i(G)$	Degree of the vertex v_i in G
$d(x, y)$	Distance between the vertices x and y in the graph
$d(\boldsymbol{u}, \boldsymbol{v})$	Distance between the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$
G	Graph
H	Hamming code
T	Spanning tree of a graph
Q_f	Fundamental cut-set matrix
$w(\boldsymbol{x})$	Weight of the vector $\boldsymbol{x}$

1. INTRODUCTION

Certain amount of work has been done about the relationship between codes and graphs and some methods have been developed for constructing codes using graphs. Three of these methods are covered here, where the codes obtained from the cut-set and cycle spaces of a graph are of main interest.

These codes are referred as the *graphical codes* and they were mainly examined by Hakimi and Frank in [1], Hakimi and Bredson in [2], [3] and [4], Bobrow in [5], and Bobrow and Hakimi in [6] in 1960's. They found methods for increasing the dimensions of the codes, and also considered decoding the resulting codes. Then they expanded their studies to obtain ternary and in general, q-ary codes, and applied the results of the binary case to these codes.

In late nineties, Jungnickel and Vanstone (in [7], [8] and [9]) reconsidered graphical codes. They generalized the methods of Hakimi and Bredson and had a new approach; the augmented codes would still be graphical, i.e., the codewords would still be spanning subgraphs of the graph used. They also gave decoding algorithms with higher efficiencies than the ones of Hakimi and Bredson. They also worked on some problems related to graphical codes in [10], [11] and in [12] with De Besimini.

Other methods for obtaining codes from graphs include the study of Haluk Oral, where he obtained self-dual codes in [13], and Cull and Nelson in [14] used the Tower of Hanoi puzzle to obtain perfect one error-correcting codes via constructing a family of graphs.

2. CODES OBTAINED FROM CYCLE AND CUT-SET SPACES OF A GRAPH

2.1. Constructing Binary Codes

Let G be a connected graph with n edges and m vertices. Consider the subgraphs of G as binary n-tuples in the following manner: Assume the edges have an arbitrary but fixed labeling. The i^{th} coordinate is one if edge e_i is contained in the subgraph, and zero otherwise.

If we fix a spanning tree T of G, then adding any edge e to T will result in a cycle consisting of the path in T between the endpoints of e and e itself. Label the edges of G in such a way that the first $n-m+1$ edges are from $G-T$ and the rest are the $m-1$ edges of T.

Definition 2.1 The $n-m+1$ cycles that are formed by each edge that is not in T and its unique tree path are called as the *fundamental cycles* with respect to T. Similarly, the *fundamental system of cut-sets* with respect to T is the $m-1$ cut-sets, where every cut-set includes exactly one edge of T. We can find such a cut-set, since every cut-set must contain at least one edge from every tree of G.

Definition 2.2 The *fundamental cycle matrix*, denoted by B_f, is defined to be the matrix having edges of G as columns and the fundamental cycles with respect to T as rows. Since each cycle is obtained by adjoining one edge to T, we can arrange the rows in such a way that the first $n-m+1 = N(G)$ (nullity of G) columns will give the identity matrix of order $N(G)$. So the rank of this matrix is $N(G)$ and $B_f = [I_{N(G)} \; B_{f\,12}]$ is a $(N(G), n)$ matrix. For the same tree T and the same labeling of the edges, the *fundamental cut-set matrix*, denoted by Q_f, is defined in a similar manner. The columns are again the edges of G and the rows of the matrix consist of the $m-1 = R(G)$ (rank of G) fundamental cut-sets obtained from T. This time we can arrange the rows to give the identity matrix of order $R(G)$ on the last $m-1$ columns. So the rank is $R(G)$ and $Q_f = [Q_{f_{11}} \; I_{R(G)}]$ is a $(R(G), m)$ matrix. For

details on fundamental cycle and cut-set matrices, the reader is referred to Seshu and Reed [15].

Example 2.1 Consider the graph G, shown in Figure 2.1. Let the edge set of G be $E(G) = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$

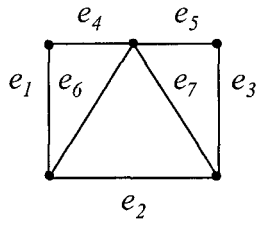


Figure 2.1. The graph used to demonstrate the construction of fundamental cut-set and cycle matrices.

Let $T = \{e_4, e_5, e_6, e_7\}$. Then the fundamental cycle matrix with respect to T is:

$$B_f = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

And the fundamental cut-set matrix with respect to T is:

$$Q_f = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Definition 2.3 A spanning subgraph of G in which every vertex has even degree is called as an *even subgraph* of G. It can be shown that an even subgraph is either a cycle or union of disjoint cycles. A *seg* is defined to be a cut-set or union of edge disjoint cut-sets.

Let $C_E(G)$ be the set of all even subgraphs of G. It can be shown that $C_E(G)$ is a vector space of dimension $N(G)$ under the symmetric difference of subgraphs. The symmetric difference of subgraphs corresponds to *mod 2* sum of the binary n-tuples representing these graphs. For example, the symmetric difference of two subgraphs g_1 and g_2, denoted by $g_1 \oplus g_2$, result in a subgraph consisting of the edges in either g_1 or g_2 but not in both. $C_E(G)$ is generally called the *cycle space* of G. Similarly, the set of all segs, denoted by $B(G)$, is a vector space of dimension $R(G)$ under the symmetric difference of subgraphs. This vector space is called as the *cut-set space* of G.

The cycle space and the cut-set space, which clearly are subspaces of the vector space of binary n-tuples under *mod 2* addition, form binary codes having the parameters *(n, N(G), d)* and *(n, R(G), d)* respectively. Here, d is the minimum number of edges in a cycle (or a cut-set) of G. The fundamental cycle and cut-set matrices B_f and Q_f serve as the generating matrices of these codes.

Proposition 2.1 Any cut-set has an even number of edges in common with any cycle.

Proof. Let C be a cycle and S be a cut-set of a connected graph G. Assume they have edges in common (if not the statement follows). Let X and Y be the components of G-S. Then C contains vertices from both X and Y. Starting with a vertex in X, we get to a vertex in Y using a common edge of C and S. To go back to X, we need another common edge. Since C is a cycle, we need to go back to X. Thus we must have even number of common edges.

From this, we can conclude that the cycle space is the nullspace of the cut-set space and vice versa. So these codes are dual codes of each other.

Example 2.2 Consider the graph G in Example 2.1. Then the cycle code $C_E(G)$ is a code with parameters *(7, 3, 3)* and generating matrix B_f. The cut-set code $B(G)$ has parameters *(7, 4, 3)* and generating matrix Q_f. It can easily be checked that the rows of B_f are orthogonal to the rows of Q_f.

2.2. Decoding Cut-set and Cycle Codes

Hakimi and Bredson show in [2] that the codes obtained from cycle and cut-set matrices are majority decodable. Majority decoding is the method of decoding a received vector to the closest code vector.

Lemma 2.1 If G is a connected graph with n edges in which every cut-set contains at least d edges and e_k is any edge of G, then there exist at least $d-1$ cut-sets (or segs) $C_1(k), C_2(k), \cdots, C_{d-1}(k)$ such that $C_i(k) \cap C_j(k) = \{e_k\}$ for all $i \neq j$ where $1 \leq i, j \leq d-1$. Considering $\boldsymbol{C}_1(k), \boldsymbol{C}_2(k), \cdots, \boldsymbol{C}_{d-1}(k)$ as n-coordinate vectors corresponding to these cut-sets, the following is shown:

Theorem 2.1 If $\boldsymbol{g}_t$ is the vector transmitted and $\boldsymbol{g}_r$ is received, assuming no more than $\left\lfloor \frac{d-1}{2} \right\rfloor$ errors have occurred, the k^{th} component of $\boldsymbol{g}_r$ is correctly received if and only if $\sum_{j=1}^{d-1} \delta_j \leq \left\lfloor \frac{d-1}{2} \right\rfloor$ (the summation is over the real field) where $\delta_j = \boldsymbol{C}_j(k) \cdot \boldsymbol{g}_r^t$ (*mod 2* inner product of two vectors)

For the cut-set code, the same theorem can be applied by giving a lemma similar to above:

Lemma 2.2 If G is a connected graph with n edges in which every cut-set contains at least d edges and e_k is any edge of G, then there exist at least $d-1$ even subgraphs $C_1(k), C_2(k), \cdots, C_{d-1}(k)$ such that $C_i(k) \cap C_j(k) = \{e_k\}$ for all $i \neq j$ where $1 \leq i, j \leq d-1$.

Definition 2.4 An *odd degree pattern* of a subgraph of G is defined to be a binary vector of length m in such a way that, for a fixed labeling of the vertices of G, the i^{th} coordinate position is 1 if and only if v_i has odd degree in that subgraph.

It is shown by Jungnickel and Vanstone in [7] that the odd degree patterns of subgraphs of G form the subspace $V_e(m,2)$ of $V(m,2)$ consisting of all even vectors of $V(m,2)$, where $V(m,2)$ is the vector space of m-tuples with entries from $GF(2)$. Using the odd degree patterns, they develop an algorithm for decoding cycle codes.

Algorithm 2.1 For decoding a t-error correcting graphical code $\boldsymbol{C} = \boldsymbol{C}_E(G)$ with *parameters (n, N(G), d)* let $X = C + S$ be the received word, where $C \in \boldsymbol{C}$ and S is an unknown subgraph consisting of at most t edges. Here, it is assumed that at most t errors have occurred during transmission. Then the odd degree pattern W of S, which obviously is also the odd degree pattern of X, has weight $w \leq 2t$. The algorithm is given as follows:

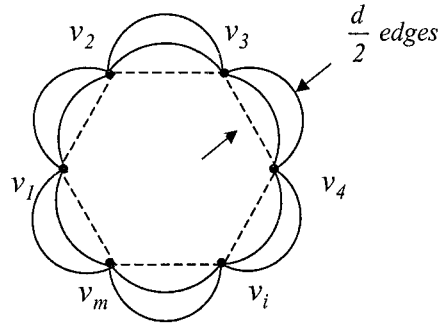
1. Find W by computing the degrees of all vertices of X, write $|W| = 2w$.
2. Compute the distance $d(x, y)$ between x and y in G for every pair $\{x, y\}$ of vertices of W.
3. Form the complete graph K on W.
4. Find a minimum weight perfect matching $M = \{x_i y_i : i = 1,2,\cdots,w\}$ of K with respect to the distance function d computed in step two.
5. Determine a path P_i of length $d(x_i, y_i)$ between x_i and y_i in G, for $i = 1,2,\cdots,w$.
6. Let S be the symmetric difference of the paths $P_1, P_2, \cdots, P_w$.
7. Output $X = C + S$.

Here, it is required finding a spanning tree of G with least weight, which has the same vertices of odd degree as X. The problem of finding such a tree is solved by an application of Chinese Postman Problem. This is the problem of finding the least weighted closed walk covering all the edges in an edge-weighted graph. [16]

2.3. Augmenting Cut-set Codes

Hakimi and Frank consider in [1] constructing graphs with n edges and maximum ranks in which every cut-set contains at least d edges, to obtain optimum cut-set codes.

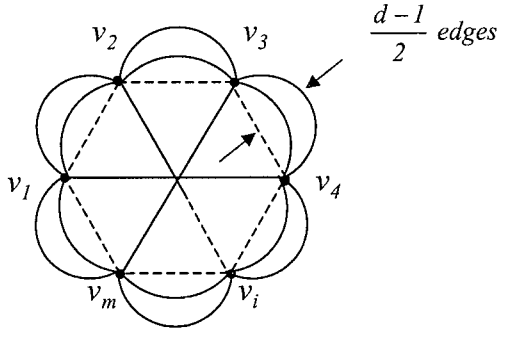
For this purpose, the graphs shown below in Figure 2.2. have been constructed. Here, the ranks of the graphs are $m-1=\left\lceil \frac{2n}{d} \right\rceil -1$ where m is the number of the vertices of the graph, and n' is the minimum number of edges required for keeping d fixed. For $n > n'$, remaining $n-n'$ edges can be added arbitrarily.



d even; $m=\left\lceil \frac{2n}{d} \right\rceil$ even

$n'=\frac{md}{2}$

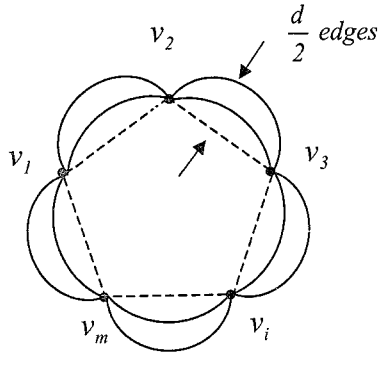
(a)



d odd; $m=\left\lceil \frac{2n}{d} \right\rceil$ even

$n'=\frac{md}{2}$

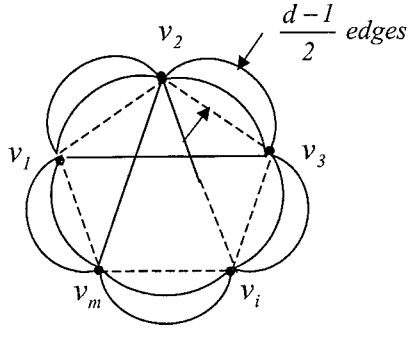
(b)



d even; $m=\left\lceil \frac{2n}{d} \right\rceil$ odd

$n'=\frac{md}{2}$

(c)



d odd; $m=\left\lceil \frac{2n}{d} \right\rceil$ odd

$n'=\frac{md+1}{2}$

(d)

Figure 2.2. Graphs with n edges and maximum ranks, having at least d edges in each cut-set.

Given n and d, these graphs have maximum ranks: Since every cut-set is assumed to have at least d edges and deleting all edges incident to a vertex makes the graph disconnected, $d(v) \geq d$ must hold for every vertex v of the graph G. So we have $m \cdot d \leq \sum d(v) = 2n$, and hence $m \leq \frac{2n}{d}$. Maximum value of m is achieved when $m = \left\lfloor \frac{2n}{d} \right\rfloor$.

Even though these graphs lead to optimal cut-set codes with length n and minimum distance d, the dimension of the resulting code is low when compared with the Plotkin bound [17]. To overcome this situation, it is shown that if $d \leq \left\lfloor \frac{2n}{d} \right\rfloor$, then the rank k of the cut-set space $(k = m - 1)$ can be increased to $k + \left\lfloor \frac{d}{2} \right\rfloor - 1$.

Example 2.3 Consider the graph below constructed by the method mentioned with $n = 18$ and $d = 6$. Then $k = 5$.

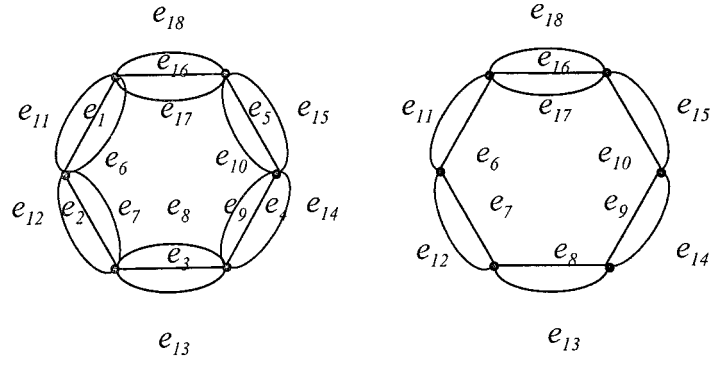


Figure 2.3. Graphs that are used for increasing the dimension of the cut-set code.

Pick a tree $T = \{e_1, e_2, e_3, e_4, e_5\}$. The cut-set matrix corresponding to T is

$$A_c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Then *G-T* has two disjoint cycles that contain every vertex of *G*, namely $h_1 = \{e_6, e_7, e_8, e_9, e_{10}, e_{16}\}$ and $h_2 = \{e_{11}, e_{12}, e_{13}, e_{14}, e_{15}, e_{17}\}$. (Such cycles are called as *Hamiltonian cycles*.) These cycles cannot be cut-sets or union of disjoint cut-sets, because they do not contain any edge of *T*. So they are linearly independent with the rows of A_c. Adding these cycles (considered as vectors) to the rows of A_c, a new matrix A_c^* is obtained. This matrix is shown to be the generating matrix of a code C^* with parameters *(18,7,6)* by proving that the linear combinations of the rows of this new matrix have minimum weight *6*.

2.4. First Method for Augmenting the Cycle Codes

To obtain higher efficiency, Hakimi and Bredson develop some methods for increasing the dimensions of the cycle codes in [3]. They were more interested in the cycle codes rather than the cut-set codes, because they say that for every *n* and *d*, there exists a cycle code, which is at least as efficient as any cut-set code.

The main idea is to find a set of *p* independent vectors of length *n*, written as rows of a matrix A, such that the matrix $\begin{bmatrix} B_f \\ A \end{bmatrix}$ is a generating matrix of a *(n, N (G)+p, d)* code.

2.4.1. Augmenting without Partitioning the Vertex Set of the Graph

Hakimi and Bredson mention a method in [3] developed by Hakimi [18]. Given integers *n* and *d*, this is a procedure for constructing a graph *G* with *n* edges such that every cycle, and hence every even subgraph, has at least *d* edges. The resulting graph turns out to

be bipartite for even values of n, since by construction, every cycle in the graph has even length. Such graphs are referred as *(n, d) graphs*.

Let G be a connected *(n, d)* graph with m vertices. Fix a vertex v^* of G and define a set of paths $P = \{p(v^*, v_i) \mid v_i \in V(G)\} = \{p(v^*, v_1), \cdots, p(v^*, v_m)\}$. Here $p(v^*, v^*)$ is assumed to be an empty subgraph of G. Clearly, $|P| = m$. Assume $m \geq 2d$ and let g_p be the symmetric difference of $2d$ or more paths in P. Let the subgraphs g_1 and g_2 be given and let $d_i(g_j)$ denote the degree of v_i in g_j, $j = 1, 2$. Then it can be shown that if $d_i(g_1)$ and $d_i(g_2)$ have the same parity, then $d_i(g_1 \oplus g_2)$ is even, otherwise $d_i(g_1 \oplus g_2)$ is odd. Then the following is proved:

Lemma 2.3 If E is an even subgraph in G, then $|g_p \oplus E| \geq d$.

Using the above information, the first method of augmentation is given as follows: Construct a graph G_1, with m edges, in which every cycle (if there is any) contains at least $2d$ edges. Let P^* be the matrix whose rows are all paths in P, and $B_f(G_1)$ be the fundamental cycle matrix of G_1. Define $B^*(G_1) = B_f(G_1) \cdot P^*$, which is a $(N(G_1) \times n)$ matrix. Then the matrix

$$A_1(G) = \begin{bmatrix} B_f(G) \\ B^*(G_1) \end{bmatrix}$$

is a basis for an *(n, N (G)+N (G$_1$), d)* binary linear code. If G_1 contains at least $4d$ vertices, this procedure can be carried out again to obtain a higher dimension.

2.4.2. Augmenting with Partitioning the Vertex Set of the Graph

Case 1. If d is even, then G is bipartite because it is assumed to be an *(n, d)* graph. So, we can partition the vertex set V of G into two disjoint sets V_1 and V_2. Fix the vertices $v_1^* \in V_1$

and $v_2^* \in V_2$. Define the following sets of paths: $P_1 = \{ p(v_1^*, v_i) \mid v_i \in V_1 \}$ and $P_2 = \{ p(v_2^*, v_i) \mid v_i \in V_2 \}$. So $|P_1| = |V_1|$ and $|P_2| = |V_2|$. Assume $|V_i| \geq d$, for $i = 1,2$ and let g_{p_i} be the symmetric difference of an even number $r_i \ (\geq d)$ of paths in P_i, $i = 1,2$.

Lemma 2.4 If E is an even subgraph of G, then $|g_{p_1} \oplus E| \geq d$, $|g_{p_2} \oplus E| \geq d$, and $|g_{p_1} \oplus g_{p_2} \oplus E| \geq d$.

The augmentation is done as follows: Construct two graphs G_1 and G_2, using Hakimi's procedure, with $|V_1|$ and $|V_2|$ edges respectively, such that every cycle contains at least d edges. Since d is even, every cycle in these graphs will have even number of edges. Let $B_f(G_i)$ be the fundamental cycle matrix of G_i. Construct P_i^*, whose rows are all the paths in P_i. Let $B^*(G_i) = B_f(G_i) \cdot P_i^*$, $i = 1,2$. Observe that these are $(N(G_i) \times n)$ matrices, for $i = 1,2$. Then the matrix

$$A_1(G) = \begin{bmatrix} B_f(G) \\ B^*(G_1) \\ B^*(G_2) \end{bmatrix}$$

is a basis for a $(n, N(G) + N(G_1) + N(G_2), d)$ binary linear code.

Case 2. If d is odd, we cannot apply the above procedure directly because G would not be bipartite. So a generalization is made to include this case. For this purpose, assume that there are p nonempty disjoint independent sets of vertices $V_1, V_2, \cdots, V_p$ such that $\bigcup_{i=1}^{p} V_i = V$. Without loss of generality, assume $|V_p| \leq |V_i|$ for $j = 1,2,\cdots p-1$. Let $v_p^* \in V_p$ be fixed. Define the set of paths $P_1, P_2, \cdots, P_p$ as $P_j = \{ p(v_p^*, v_i) \mid v_i \in V_j \text{ and } v_i \neq v_p^* \}$, $j = 1,2,\cdots, p$. Define g_{p_j} to be the symmetric difference of d or more paths in P_j. If for some k $|P_k| < d$, then assume g_{p_k} to be empty. Then any *mod 2* linear combination of

vectors $g_{p_1}, g_{p_2}, \cdots, g_{p_p}$ is a vector g_b, which differs in at least d components from a vector of any even subgraph of G, except when g_b is the zero vector.

First construct a graph G_i, for all $1 \le i \le p$, with $|P_i|$ edges in which every cycle contains at least d edges. Let $B_f(G_i)$ be the fundamental cycle matrix of G_i, which is a $(N(G_i) \times |P_i|)$ matrix, and P_i^* be the $(|P_i| \times n)$ matrix whose rows are all paths in P_i. Define $B^*(G_i) = B_f(G_i) \cdot P_i^*$. Observe that these are $(N(G_i) \times n)$ matrices, for $i = 1,2$. Then the matrix

$$A_1(G) = \begin{bmatrix} B_f(G) \\ B^*(G_1) \\ \vdots \\ B^*(G_p) \end{bmatrix}$$

is a basis for a $(n, \mathrm{N(G)} + \sum_{i=1}^{p} \mathrm{N(G_i)}, d)$ binary linear code.

For recursive augmentation, given a (n, d) graph G, construct graphs $G_1, G_2, \cdots G_p$ as suggested above and define for every (n, d) graph G the matrix

$$A_1(G) = \begin{bmatrix} B_f(G) \\ B^*(G_1) \\ \vdots \\ B^*(G_p) \end{bmatrix}$$

where $B^*(G_j) = B_f(G_j) \cdot P_j^*$, $j = 1,2,\cdots,p$.

Then recursively define for every integer $i > 1$,

$$A_i(G) = \begin{bmatrix} B_f(G) \\ A^*_{i-1}(G_1) \\ \vdots \\ A^*_{i-1}(G_p) \end{bmatrix}$$

where $A^*_{i-1}(G_j) = A_{i-1}(G_j) \cdot P^*_j$, $j = 1,2,\cdots,p$.

The recursion terminates when for some integer k the number of rows in $A_k(G)$ is the same as the number of rows in $A_{k-1}(G)$. For $i \geq 1$, $A_i(G)$ forms a basis for a binary code of length n with minimum distance d. Same argument may be applied to the augmentation without partitioning the vertices.

Next, the efficiency (k/n, where k is the dimension and n is the length of the code) of these augmented codes are discussed. Dimensions of augmented cycle codes are compared with the Varsharmov-Gilbert bound [19]. For many values of n and d, especially when $n \leq 100$ and $d \leq 20$, efficiency of graph theoretic codes remain above the bound. The conclusion is that one generally obtains higher efficiencies using recursive augmentation with partitioning the vertices than without partitioning the vertices and the second method seems to give better results for even values of d than odd values of d.

2.5. Second Method for Augmenting Cycle Codes

In [7], Jungnickel and Vanstone show that a graphical code $\boldsymbol{C} = \boldsymbol{C}_E(G)$ with parameters $(n, N(G), d)$ can be extended to a $(n, N(G)+1, d)$ code $\boldsymbol{C}'$, provided that $n \geq 2d$. $\boldsymbol{C}'$ is obtained by adjoining any subgraph S with odd degree pattern of weight $w \geq 2d$ as a row of B_f. Here, one must have a bond on the weight of an arbitrary vector $C + S$ of $\boldsymbol{C}'$, where $C \in \boldsymbol{C}$. Let W be the odd degree pattern of S with weight w. Then the odd degree pattern of $C + S$ is also W. The smallest possible weight of $C + S$ is obtained when $C + S$ consists of $w/2$ independent edges. To guarantee that $\boldsymbol{C}'$ still has minimum weight G, we must have $w \geq 2d$.

In general, to increase the dimension by $k \geq 2$, according to the argument above, k linearly independent subgraphs, $S_1, S_2, \cdots, S_k$ are needed as additional rows of B_f. Each of these subgraphs must have odd degree patterns of weight bigger than or equal to *2d*. Also, the symmetric difference of any of these subgraphs must have odd degree patterns of weight at least *2d*. Hence the odd degree patterns related with these subgraphs should themselves form an even binary code $\boldsymbol{O}$ (i.e., every code has even weight) with minimum distance at least *2d*. This is stated as a theorem:

Theorem 2.2 The dimension can be increased by $k \geq 2$, to get a code $\boldsymbol{C}^*$, provided that there exists an even binary *(n, k, 2d)* code $\boldsymbol{O}$. $\boldsymbol{C}^*$ is obtained by adjoining k linearly independent subgraphs, $S_1, S_2, \cdots, S_k$, to C with odd degree patterns forming a basis for $\boldsymbol{O}$ as additional rows of B_f.

Jungnickel and Vanstone also show in [7] that Hakimi and Bredson's method without partitioning the vertex set is a special case of the method given above. In the same manner, the method with partitioning the vertex set is generalized as follows:

Theorem 2.3 Consider a graphical code $\boldsymbol{C} = \boldsymbol{C}_E(G)$ with parameters *(n, N(G), d)* based on the connected graph G, and assume $V_1, V_2, \cdots, V_p$ is a partition of the vertex set V into independent sets with cardinalities $p_1, p_2, \cdots, p_p$ respectively. Then $\boldsymbol{C}$ may be extended to a graphical code $\mathbb{C}^*$ with parameters *(n, N(G)* + $(k_1 + k_2 + \cdots + k_c)$*, d)* provided there exists even binary (p_i, k_i, d) codes $\boldsymbol{O}_i$, for $i = 1, 2, \cdots, p$. Such a code can be obtained by adjoining $\mathbb{C}$ arbitrary sets of k_i linearly independent subgraphs of G with odd degree patterns contained in V_i and forming a basis for $\boldsymbol{O}_i$ as additional rows of B_f.

2.6. Decoding Augmented Cut-Set Codes

L.S. Bobrow, in [5], suggested a method for decoding augmented cut-set codes that Hakimi and Bredson have obtained, which were mentioned in Section 2.3.

The original cut-set code is a subspace of the augmented code, so the dual of the augmented code is a subspace of the dual of the original code. The dual of the augmented code consists of all the even subgraphs that have even number of edges in common with the Hamiltonian cycles that are added.

The decoding process is explained with an example. Consider the graph in Figure 2.3.(a). According to the augmentation scheme mentioned in section 2.3., the dimension of the corresponding code can be increased by two by adding the vector representations of the Hamiltonian cycles $h_1 = \{e_6, e_7, e_8, e_9, e_{10}, e_{16}\}$ and $h_2 = \{e_{11}, e_{12}, e_{13}, e_{14}, e_{15}, e_{17}\}$ to the generating matrix of the code. Consider the even subgraphs $c_1 = \{e_1, e_2, e_3, e_4, e_5, e_{18}\}$, $c_2 = \{e_1, e_2, e_8, e_9, e_{10}, e_{16}\}$, $c_3 = \{e_1, e_2, e_{13}, e_{14}, e_{15}, e_{17}\}$, $c_4 = \{e_1, e_2, e_6, e_7\}$ $c_5 = \{e_1, e_2, e_{11}, e_{12}\}$. These cycles are in the dual of the augmented code, and orthogonal on $b_1 + b_2$, where $b_1, b_2, \cdots, b_{18}$ are the coordinate positions of the codeword. By the structure of the graph, it can be seen that there are five vectors in the nullspace that are orthogonal on $b_2 + b_3$. The parity checks corresponding to these five vectors will give us an estimate on $b_2 + b_3$. Similarly, estimates on $b_2 + b_8$, $b_2 + b_{13}$, $b_2 + b_6$ can be obtained. These quantities are then combined with the original parity checks corresponding to c_1, c_2, c_3 and c_4 respectively. Including the parity check corresponding to c_5, we get a set of five parity checks orthogonal on b_1. From here b_1 can be estimated. Similarly, estimates for the other coordinate positions are obtained.

2.7. First Method for Decoding Augmented Cycle Codes

The augmented binary codes are majority decodable and decoding schemes for both methods are given in [3].

For the first method, (without partitioning the vertices) let the edges of G_1 be $e_1, e_2, \cdots e_m$ and assume e_i corresponds to the vertex v_i of G. Then a one-to-one correspondence is established between e_i and a path $p(v^*, v_i) \in P$. If $\boldsymbol{g}_t$ is the transmitted vector then we can write $\boldsymbol{g}_t = \boldsymbol{E} + \boldsymbol{g}_p$, where E is an even subgraph of G and $\boldsymbol{g}_p$ is a

linear combination of the rows of the matrix $B^*(G_1)$. $\boldsymbol{g}_p$ can also be considered as the symmetric difference of a subset of paths in P corresponding to the edges of an even subgraph E_{p_1} in G_1. For every subgraph g_r of G, a subgraph g_{r_1} of G_1 is defined as: $g_{r_1} = \{e_i | i = 1,2,\cdots,m \text{ and } d_i(g_r) \text{ is odd}\}$.

Given these definitions, it is proved that if $\boldsymbol{g}_t = \boldsymbol{E} + \boldsymbol{g}_p$ is transmitted and $\boldsymbol{g}_r = \boldsymbol{g}_t + \boldsymbol{g}_e$ is received and the subgraphs g_{r_1} and E_{p_1} are defined as above, then $|E_{p_1} \oplus g_{p_1}| \le 2\left\lfloor \frac{d-1}{2} \right\rfloor$. Here, it is assumed that $w(\boldsymbol{g}_e) \le \left\lfloor \frac{d-1}{2} \right\rfloor$.

Let $\boldsymbol{g}_r$ be received. Then g_{r_1} of G_1 can be determined. g_{r_1} differs from E_{p_1} in at most $2\left\lfloor \frac{d-1}{2} \right\rfloor$ edges by the above statement. Since every cycle in G_1 contains at least $2d$ edges, E_{p_1} can be determined from g_{r_1} by majority decoding of G_1. From E_{p_1}, $\boldsymbol{g}_p$ can be found. Since $\boldsymbol{g}_r + \boldsymbol{g}_p = \boldsymbol{E} + \boldsymbol{g}_e$, by majority decoding on G, E can be found. Finally, $\boldsymbol{g}_t = \boldsymbol{E} + \boldsymbol{g}_p$ is obtained.

The decoding scheme for the second method (with partitioning the vertices) is a generalization of the argument just mentioned.

2.8. Second Method for Decoding Augmented Cycle Codes

Algorithm 2.2 For decoding the augmented *(n, N(G)+k, d)* code $\boldsymbol{C}^*$, which is obtained by using the even binary *(m, k, 2d)* code $\boldsymbol{O}$, where $d \ge 2t + 1$, let R be the word received and assume that at most t errors have occurred. Then $R = C + S + E$ for some $C \in \boldsymbol{C}$, some unknown subgraph S with odd degree pattern $W \ne \boldsymbol{0}$ in $\boldsymbol{O}$, and some graph E consisting of at most t edges. The decoding algorithm is given as follows:

1. Find the odd degree pattern R' of R by computing the degrees of all vertices in R.
2. Using $\boldsymbol{O}$, decode R' into the correct odd degree pattern $W = R' + E' \in \boldsymbol{O}$.

3. Find a subgraph S of G corresponding to the odd degree pattern W.
4. Put $X' = R + S$ and decode X' into a subgraph $C \in \boldsymbol{C}$ using Algorithm 2.2.
5. Output $X = C + S$.

Here, it is stated that this algorithm's efficiency depends on the efficiency of decoding the code $\boldsymbol{O}$.

For decoding augmented codes obtained by the method given by Theorem 2.3, the following algorithm is given:

Algorithm 2.3 Consider the $(n,\ N(G)+(k_1+k_2+\cdots+k_c),\ d)$ code $\boldsymbol{C}^*$, which is obtained by using the binary $(m,\ k_i,\ d)$ codes $\boldsymbol{O}_i\ (i = 1,2,\cdots,p)$. Let R be the word received and assume that at most t errors have occurred. Then $R = C + S_1 + S_2 + \cdots + S_p + E$ for some $C \in \boldsymbol{C}$, some subgraphs S_i with odd degree patterns $W_i \neq 0$ in $\boldsymbol{O}_i$, and some graph E consisting of at most t edges.

1. Find the odd degree patterns R_i' of R restricted to V_i by computing the degrees of all vertices in R.
2. Using $\boldsymbol{O}_i$, decode R_i' into the correct odd degree pattern $W_i = R_i' + E_i' \in \boldsymbol{O}_i$.
3. For all $i = 1,2,\cdots,p$ with $E_i \neq 0$, let S_i be some subgraph of G corresponding to the odd degree pattern W_i.
4. Put $X' = R + S_1 + S_2 + \cdots + S_p$ and decode X' into a subgraph $C \in \mathbf{C}$.
5. $X = C + S_1 + S_2 + \cdots + S_p$

2.9. Some Problems Related to Graphical Codes

2.9.1. A Problem in Graphical Enumeration

Jungnickel and Vanstone present an alternative proof of a result due to Read (1962) which gives the generating function for the number of labeled Eulerian graphs with m

vertices in [10]. Appliying the same method they prove a similar theorem concerning bipartite Eulerian graphs.

Definition 2.5 The *(homogenous) weight enumerator* of a binary code $\boldsymbol{C}$ of length n is defined as the polynomial

$$A_c(x,y)=\sum_{i=1}^{n} A_i x^{n-i} y^i,$$

where A_i denotes the number of vectors with weight i in $\boldsymbol{C}$. Observe that this polynomial is homogenous of degree n, since the powers of x and y add up to n in each term. If $\boldsymbol{C}$ is a linear code of dimension k the weight enumerator of the dual code $\boldsymbol{C}^{\perp}$ of $\boldsymbol{C}$ is given by

$$A_{C^{\perp}}(x,y)=\frac{1}{2^k}A_C(x+y,x-y),$$

see MacWilliams and Sloane [20] for details.

Theorem 2.4 The polynomial $w_m(x)$ which has as the coefficient of x^n the number of labeled Eulerian graphs with m vertices and n edges is given by

$$w_m(x)=\frac{1}{2^m}(1+x)^{\binom{m}{2}}\sum_{i=0}^{m}\binom{m}{i}\left(\frac{1-x}{1+x}\right)^{i(m-i)}.$$

Proof. Since the Eulerian graphs with m vertices are exactly the even subgraphs of the complete graph K_m, $w_m(x)$ can be considered as the (non-homogeneous) weight enumerator of the even graphical code $\boldsymbol{C}=\boldsymbol{C}(K_m)$. Here, the weight enumerator of the dual code is calculated first, which is easier to achieve. Note that the dual code is the cut-set space $\boldsymbol{B(G)}$ of K_m(of dimension m-1). Any i-subset of the vertex set V of K_m determines a unique partition $(X, V\backslash X)$ of V which corresponds to the cut-set consisting

of the $i(m-i)$ edges joining a vertex in X to a vertex in $V\backslash X$. Considering that we count each edge in these cut-sets twice this way, the homogeneous weight enumerator of $\boldsymbol{B}(G)$ is

$$z_m(x,y)=\frac{1}{2}\sum_{i=0}^{m}\binom{m}{i}x^{\binom{m}{2}-i(m-i)}y^{i(m-i)}.$$

From this we get the homogeneous weight enumerator of $\boldsymbol{C}$:

$$w_m(x,y)=\frac{1}{2^{m-1}}z_m(x+y,x-y).$$

To obtain $w_m(x)$ we just need to substitute and set $y=1$.

Using the same approach, the generating function for the number of labeled bipartite Eulerian graphs with parts of p and q vertices is obtained.

Theorem 2.5 The polynomial $w_{p,q}(x)$ which has as the coefficient of x^n the number of labeled bipartite Eulerian graphs with $p+q$ vertices (with the two parts having p and q vertices respectively) and n edges is given by

$$w_{p,q}(x)=\frac{1}{2^{p+q-1}}(1+x)^{pq}\sum_{k=0}^{p}\sum_{h=0}^{q}\binom{p}{k}\binom{q}{h}\left(\frac{1-x}{1+x}\right)^{k(q-h)+h(p-k)}.$$

Proof. Since the Eulerian graphs with $p+q$ vertices are exactly the even subgraphs of the complete bipartite graph $K_{p,q}$, $w_{p,q}(x)$ can be considered as the (non-homogeneous) weight enumerator of the even graphical code $\boldsymbol{C}=\boldsymbol{C}(K_{p,q})$. Again, the weight enumerator of the dual code is calculated first. Note that the dual code is the cut-set space $\boldsymbol{B}$ of $K_{p,q}$ (of dimension $p+q-1$). Let the vertex set V be partitioned into sets U and U' respectively. Any pair (X,Y) consisting of a k-subset X of U and h-subset Y of U' determines a unique partition $(X\cup Y, (U\backslash X)\cup(U'\backslash Y))$ of V which corresponds to the

cut consisting of the $k(q-h)+h(p-k)$ edges either joining a vertex in X to a vertex in $U'\backslash Y$ or joining a vertex in Y to a vertex in $U\backslash X$. Hence, the homogeneous weight enumerator of $\boldsymbol{B}$ is

$$z_{p,q}(x,y)=\sum_{k=0}^{p}\sum_{h=0}^{q}\binom{p}{k}\binom{q}{h}x^{pq-k(q-h)-h(p-k)}y^{k(q-h)+h(p-k)}.$$

From this we get the homogeneous weight enumerator of **C**:

$$w_{p,q}(x,y)=\frac{1}{2^{p+q-1}}z_{p,q}(x+y,x-y).$$

To obtain $w_p(x)$ we just need to substitute and set $y=1$.

2.9.2. An Application of Difference Sets to a Problem Concerning Graphical Codes

Jungnickel and Vanstone, in [11], used difference sets to solve the problem of finding out when the binary code generated by the complete graph K_n is contained in some binary Hamming code.

Example 2.4 Consider the binary *(15, 10, 3)* code $\boldsymbol{C}=\boldsymbol{C}(K_6)$. The Hamming code $\boldsymbol{H}$ of the same length has parameters *(15, 11, 3)* . Indeed, $\boldsymbol{C}$ is included in $\boldsymbol{H}$: If we adjoin all subgraphs of K_6 to $\boldsymbol{C}$, we get a code with parameters (15, 11, 3) [8]. But the only linear code with these parameters is $\boldsymbol{H}$ [20].

Next comes the general problem when the binary code generated by K_n is contained in some Hamming code of length 2^m-1, for some m. We obviously have the condition

$$\binom{n}{2}=2^m-1$$

Definition 2.6 A k-subset D of a group G of order v a (v, k, λ) is called as a *difference set* if the list of differences $d - d' \ (d, d' \in D, d \neq d')$ contains each element of G exactly λ times.

Lemma 2.5 The binary code $\boldsymbol{C}$ generated by K_n is contained in the Hamming code $\boldsymbol{H}$ of length $2^m - 1$ if and only if there exists an elementary abelian difference set with parameters $(2^m, n, 2)$.

Proof. Let $G = K_n$. First assume that there exists such a difference set D in the additive group of $GF(2^m)$. Label the n vertices of G with elements of D. Here, $GF(2^m)$ is considered as the m-dimensional vector space over $GF(2)$. Let x be a nonzero element in $GF(2^m)$. Since $\lambda = 2$, there exists a unique edge $e = dd'$ with $d, d' \in D$. Let M be the parity-check matrix of $\boldsymbol{H}$. Identify e with column x of M. It can be seen that any cycle of G gives rise to a set of columns of M adding up to zero. Since $\boldsymbol{C}$ is generated by the cycles of G, it is contained in $\boldsymbol{H}$.

Now assume that $\boldsymbol{C}$ is contained in $\boldsymbol{H}$. Let M and M' be the parity check matrices of $\boldsymbol{H}$ and $\boldsymbol{C}$ respectively. By hyphothesis, M' can be obtained by adding further rows to M. Corresponding each edge of G with a column of M', we can induce a labelling φ of the edges of G with the nonzero elements of $GF(2^m)$. Now, φ is used to label the n vertices of G: Choose a fixed vertex v_0 and label it with the vector $\boldsymbol{d}_0 = \boldsymbol{0}$. Label the rest of the vertices with $\boldsymbol{d}_v = \varphi(vv_0)$. $D = \{d_v : v \in G\}$ is the difference set with the desired parameters. Indeed, let $\boldsymbol{x}$ be a nonzero element of $GF(2^m)$. If x is one of the vertex labels as defined above, then $\boldsymbol{x} = \boldsymbol{d}_0 + \boldsymbol{d}_v$. Otherwise, $\boldsymbol{x}$ has a unique represntation $\boldsymbol{x} = \varphi(vw)$ with $v, w \neq v_0$. The vertices v, w and v_0 form a cycle of length 3 in G, which gives the condition

$$\varphi(vw) + \varphi(wv_0) + \varphi(v_0v) = 0,$$

and hence

$$x = \varphi(wv_0) + \varphi(v_0 v) = \boldsymbol{d}_w + \boldsymbol{d}_v .$$

Theorem 2.6 The binary code generated by K_n is contained in the Hamming code of length $2^m - 1$ if and only if n is one of the numbers *2,3* and *6*.

Proof. The numbers n for which there exists an elementary abelian difference set with parameters $(2^m, n, 2)$ must be determined. A nontrivial difference set in a *2*-group always has parameters of the form $(2^{2a+2}, 2^{2a+1} \pm 2^a, 2^{2a} \pm 2^a)$ [21].

2.9.3. Codes Based On Complete Graphs

The problem of embedding the binary code $\boldsymbol{C}_n$, generated by the complete graph K_n, into a shortening of the Hamming code $\boldsymbol{H}_m$ of length $2^m - 1$ for some m is considered in [12]. Let *h(n)* be the smallest value of m for which $\boldsymbol{C}_n$ can be embedded into $\boldsymbol{H}_m$. Such an embedding will be called *optimal*. One trivial bound on *h(n)* is

$$n(n-1) \le 2^{h(n)+1} - 2 ,$$

since the length of $\boldsymbol{C}_n$ can be at most $2^m - 1$.

To characterize all embeddings of $\boldsymbol{C}_n$ into shortened Hamming code, the lemma given by Jungnickel and Vanstone in [17] is generalized.

Definition 2.7 A k-subset D of a group G of order v is called as an *incomplete* (v, k, λ) *difference set* if the list of differences $d - d'$ $(d, d' \in D, d \neq d')$ contains each element of G either not at all or exactly λ times.

Lemma 2.6 The code $\boldsymbol{C}_n$ is contained in a shortening of the Hamming code $\boldsymbol{H}_m$ of length $2^m - 1$ if and only if there exists an incomplete elementary Abelian difference set with parameters $(2^m, n, 2)$.

Proof. Let $G = K_n$. First assume that there exists such an incomplete difference set D in the additive group of $GF(2^m)$. Label the n vertices of G with elements of D. Here, $GF(2^m)$ is considered as the m-dimensional vector space over $GF(2)$. Let x be a nonzero element in $GF(2^m)$ and assume that x can be represented as a difference from D. Then $x = d + d', \quad d, d' \in D$. Since $\lambda = 2$, there exists a unique edge $e = dd'$. Let M be the parity-check matrix of $\boldsymbol{H}_m$. We can identify the edges of G with certain columns of M. Let $\boldsymbol{H}$ be the shortening of $\boldsymbol{H}_m$ obtained by keeping only the columns of M corresponding to the edges of G. It can be seen that any cycle of G gives rise to a set of columns of the parity check matrix of $\boldsymbol{H}$ adding up to zero. Since $\boldsymbol{C}$ is generated by the cycles of G, it is contained in $\boldsymbol{H}$.

Now assume that $\boldsymbol{C}_n$ is contained in a code $\boldsymbol{H}$ obtained from $\boldsymbol{H}_m$ by shortening. Let M' be the parity check matrix of $\boldsymbol{C}_n$, which can be obtained by choosing $n(n-1)/2$ columns from M. Since it is assumed that $\boldsymbol{C}_n$ is contained $\boldsymbol{H}$, M' can be obtained by adding further rows to the parity check matrix M'' of $\boldsymbol{H}$. This induces an injective mapping φ from the edges of G into the set of nonzero elements of $GF(2^m)$. Now, φ is used to label the n vertices of G: Choose a fixed vertex v_0 and label it with the vector $d_0 = 0$. Label the rest of the vertices with $d_v = \varphi(vv_0)$. $D = \{d_v : v \in G\}$ is the difference set with the desired parameters. Indeed, let x be a nonzero element of $GF(2^m)$. If x is one of the vertex labels as defined above, then $x = d_0 + d_v$. If x has a representation $x = \varphi(vw)$ with $v, w \neq v_0$, then the vertices v, w and v_0 form a cycle of length 3 in G, which gives the condition

$$\varphi(vw) + \varphi(wv_0) + \varphi(v_0v) = 0,$$

and hence $x = \varphi(wv_0) + \varphi(v_0 v) = d_w + d_v$. Since φ is an injection, D is an incomplete difference set.

Lemma 2.7 The code $\boldsymbol{C}_n$ $(n \geq 6)$ is contained in a shortening of the Hamming code $\boldsymbol{H}_m$ if and only if there exists a binary *(n-1, k, 5)* code which admits a parity check matrix with m rows.

Proof. The existence of an incomplete difference set with parameters $(2^m, n, 2)$ is required. It may be assumed that D contains $\boldsymbol{0}$, so let $D^* = D \backslash \{\boldsymbol{0}\}$. An arbitrarily given *n-1* subset D^* of $GF(2^m)^*$ gives rise to an incomplete difference set D in G if and only if no four elements of D^* are linearly dependent in $GF(2^m)$: Consider any linear dependence of the form

$$d_1 + d_2 + d_3 + d_4 = 0.$$

This gives rise to four difference representations

$$x = d_1 + d_2 = d_3 + d_4$$

from D. Then D is an incomplete difference set if and only if the *n-1* elements of D^* form the columns of a parity check matrix H (with m rows) of a code $\boldsymbol{C}$ of length *n-1* and minimum distance *5*.

Observe that the code $\boldsymbol{C}$ mentioned in the above lemma has dimension $k = n - 1 - \operatorname{rank} H \geq n - 1 - m$. In order to make m as small as possible, we must have $m = n - 1 - k$.

Theorem 2.7 Let $k = k(n)$ denote the largest dimension for which a binary *(n-1, k, 5)* code exists. For $n \geq 6$,

$$h(n) = n - 1 - k .$$

Theorem 2.8 $h(2^k + 1) = 2k$ for all k and $2k \le h(n) \le 2k + 2$ for all n with $2^k + 1 \le n \le 2^{k+1}$. Moreover, one can always obtain an optimal embedding of $\boldsymbol{C}_n$, which is a graphical code based on K_n.

Proof. Let n be given. k is defined by $2^k + 1 \le n \le 2^{k+1}$. From Lemma 2.6, it can be seen that $h(n) \ge 2k$. Now let $n = 2^k + 1$. Then there exists a double error correcting BCH code $\boldsymbol{C}'$ with parameters $(2^k + 1, 2^k - 2k, 6)$ [21]. Puncturing this code gives a binary code $\boldsymbol{C}$ with parameters $(2^k, 2^k - 2k, 5)$. From Theorem 2.7, one has $h(2^k + 1) \le 2k$, hence $h(2^k + 1) = 2k$. Thus the inequality in the theorem is obtained.

Let n be a number satisfying the above equation. The code $\boldsymbol{C}_h$ with $h = 2^{k+1} + 1$ may be embedded into a shortening of the Hamming code $\boldsymbol{H}_{2k+2}$. This means that $\boldsymbol{C}_n$ can be embedded into this Hamming code for all $n \le h$.

Now assume that $\boldsymbol{C}_n$ can be embedded into a code $\boldsymbol{H}$ that is a shortening of the Hamming code $\boldsymbol{H}_m$. By the previous theorem, there exists an optimal binary $(n-1, n-1-m, 5)$ code C. Then the parity-check extension of is a code $\boldsymbol{C}'$ with parameters $(n, n-1-m, 6)$. According to Theorem 2.3., by using $\boldsymbol{C}'$ (corresponding to the code $\boldsymbol{O}$ in the theorem), $\boldsymbol{C}_n$ can be augmented by *n-1-m* dimensions to obtain a code $\boldsymbol{C}^*$ with parameters

$$\left(\frac{n(n-1)}{2}, \frac{n(n-1)}{2} - m, 3 \right)$$

hence, it has a parity check matrix M with m rows. By adding suitable columns, M can be augmented to a parity check matrix of $\boldsymbol{H}_m$. So $\boldsymbol{C}^*$, which is based on K_n, is an optimal embedding of $\boldsymbol{C}_n$ into a shortening of $\boldsymbol{H}_m$.

3. OBTAINING CODES FROM CUBIC BIPARTITE PLANAR GRAPHS

Another method for construction of codes from graphs was given by Haluk Oral in [13]. He considered obtaining self-dual codes by use of cubic planar bipartite graphs.

Definition 3.1 Through this discussion, let G be a connected cubic planar bipartite graph with m vertices. The *face-vertex incidence matrix* $D = [d_{ij}]$ is defined to be the matrix whose columns are indexed by the vertices and rows by the faces of G such that

$$d_{ij} = \begin{cases} 1, & \text{if } j \text{ is incident with } f_i, \\ 0, & \text{otherwise.} \end{cases}$$

For a face f, the corresponding row of D is also denoted by f. So subsets of the faces of G are identified with the corresponding subsets of rows of D. Since a cubic bipartite graph cannot have edge-connectivity one, any edge of G must be incident with two faces. It can be shown that a cubic planar graph is 3-face colorable if and only if it is bipartite. So we can color the faces of G with three colors.

Lemma 3.1 The only minimal dependent subsets of the faces of G are pairwise union of two color classes.

Proof. To show this, the followings are proven: Given a minimal dependent set of faces M, every vertex of the graph is incident with exactly two elements of M and if M contains a face colored by a certain color then it must contain all the faces colored by that color. These two statements imply that M is a union of two color classes.

Theorem 3.1 Let f_1 and f_2 be any two faces of G of different colors in a three coloring of G. If we delete the rows of D corresponding to these faces, the resulting matrix S is a generator matrix for a self-dual code of length m. Moreover, this code is independent of the choices of f_1 and f_2.

Proof. First it is shown that there are two minimal dependent subsets M_1 and M_2 such that

i) $f_1 \in M_1, f_2 \notin M_1$,

ii) $f_1 \notin M_2, f_2 \in M_2$.

Indeed, M_1 can be taken as the union of two color classes not containing the color of f_2 and M_2 can be taken as the union of two color classes not containing the color of f_1. So the rowspace of S is equal to the rowspace of D. Next, it is needed to show that the rows of S are orthogonal and the resulting code has dimension $m / 2$. Since G is bipartite, every row of S has even weight. Since G is cubic, two faces cannot have an odd number of edges in common and they cannot share two adjacent edges. Hence any two faces have even number of vertices in common. This implies that any two rows of S are orthogonal. By Euler's formula

$$m - \frac{3m}{2} + |F| = 2 .$$

Hence,

$$|F| = \frac{m}{2} + 2 ,$$

where $|F|$ is the number of faces of G So S has $m / 2$ rows. It is concluded that S is the generator matrix for a self-dual code of length m, and this code is independent of the faces deleted.

Next, some relations between the graph and the code obtained form that graph are mentioned. It is shown that the code obtained must have minimum distance two or four. It is stated that if $d = 2$, then G is at most 2-connected and if the graph has connectivity two, then it yields a decomposable code. So if the code is indecomposable, it has connectivity three. The converse is proven by showing that if $U \subseteq V(G)$ is a nonempty proper subset

of the support of a face f (i.e., vertices contained in f), then U cannot be the support of a codeword. This implies that the vertices of a face must be in the same component of the code $\boldsymbol{C}$ and since G is connected, all vertices of G belong to the same component of $\boldsymbol{C}$. Hence, $\boldsymbol{C}$ is indecomposable.

The rank of the face-incidence matrix D is greater than or equal to $m / 2$: To see this, the following is proven:

Lemma 3.2 Let G be a planar 2-edge connected graph on m vertices such that G has maximum valency three and has some vertex u of degree two. Let $F(G)$ be the set of faces of G. If f is a face incident with u then $F(G)$-f is an independent subset of $[GF(2)]^m$. Then it is concluded that the rank of the matrix D is at least $m / 2$.

Finally, it is stated that, given any cubic planar graph (not necessarily bipartite), self-orthogonal codes can be constructed. Let G be a graph with t faces of odd degree and D be its face-vertex incidence matrix with the faces of odd degree as its first t rows. A new matrix is defined as:

$$D^* := \begin{pmatrix} & \vdots & I_t \\ D & \vdots & \cdots \\ & \vdots & 0 \end{pmatrix}$$

Then the rowspace of this matrix is a self-orthogonal code, since it can be seen that any two rows of D^* are orthogonal.

4. OBTAINING CODES FROM THE TOWERS OF HANOI GRAPHS

Using the Towers of Hanoi puzzle, Cull and Nelson [14] defined an infinite family of graphs and from these graphs they obtained perfect one error correcting codes.

Definition 4.1 A *perfect one error correcting code* on a graph $G=(V,E)$ is a set of *codewords* $C \subset V$ such that no two codewords are adjacent and every non-codeword is adjacent to exactly one codeword.

4.1. The Tower of Hanoi Puzzle

There are three towers in the puzzle, called *0,1* and *2*, and n disks $1,2,\cdots,n$ where *1* is the smallest and n is the largest disk. The aim is to move the n disks from one tower to another, with the help of a third tower. Only the smallest disk on a tower can be moved and it can be placed on an empty tower or on top of a larger disk. A configuration of the puzzle is specified by an array D, where d_i is the tower *(0,1* or *2)* that contains the i^{th} disk. Note that knowing which disks are on one tower also allows us to know the order of the discs, because of the rules of the puzzle. The Towers of Hanoi will be denoted by T of H throughout the discussion.

4. 2. Constructing the Graphs

The graph is defined in the following manner: The vertices are the configurations and two vertices are adjacent if and only if the corresponding configurations can be reached from one another by one legal move. If a vertex corresponds to $d_1,d_2,\cdots,d_n$ then it is adjacent to $d_1+1,d_2,\cdots,d_n$ and $d_1+2,d_2,\cdots,d_n$ where addition is *mod 3*. (Since the first disk is the smallest, it can be moved to any of the two remaining towers.) If there is a j such that $d_1=d_2=\cdots=d_{j-1}\neq d_j$ then this vertex is also adjacent to $d_1,d_2,\cdots,$ $d_{j-1},\hat{d}_j,\cdots,d_n$ where $\hat{d}_j$ is not equal to either d_1 or d_j. For example, assume the first

$j-1$ disks are on the first tower and the j^{th} disk is on the second. Then we can move the j^{th} disk to the third tower. If $d_1 = d_2 = \cdots = d_n$ then such a vertex is adjacent to only two other vertices. These vertices are called *corner* vertices.

The graph is drawn in levels. The vertex $00\cdots0$ is at level 0 and the two vertices adjacent to it are at level one. Recursively, H_n, the T of H graph for n disks, is constructed by the diagram:

$$H_{n-1}$$
$$/ \quad \backslash$$
$$H_n = H_{n-1} \text{ —— } H_{n-1}$$

Here, three copies of H_{n-1} is connected as follows: add an edge to connect the left corner vertex of the bottom row of the top H_{n-1} to the top vertex of lower left H_{n-1} and add an edge to connect the right corner vertex of the bottom row of the top H_{n-1} to the top vertex of lower right H_{n-1}. Also connect the bottom right corner of the lower left H_{n-1} and bottom left corner of the lower right H_{n-1} with an edge. If H_{n-1} has levels 0 through l, then H_n will have $2l+1$ levels.

To display the labeling, let L_n be the labeled graph. Then

$$RL_n0$$
$$/ \quad \backslash$$
$$L_{n+1} = \uparrow RL_n1 \text{ —— } \downarrow RL_n2$$

Where RL_n means the labeled graph which is the mirror image of L_n. In L_n, the lower left vertex is labeled $11\cdots1$ and the lower right vertex is labeled $22\cdots2$. In RL_n it is vice versa. RL_n0 is RL_n with each vertex has a 0 appended to its label. $\uparrow RL_n1$ is RL_n

rotated *120* degrees clockwise and has a *1* appended to each label. Similarly $\downarrow RL_n 2$ is RL_n rotated *120* degrees counterclockwise and has a *2* appended to each label.

4.3. Constructing the Codes

To construct the codes, a graph G_n that has the same topology as L_n is introduced. This graph has no labels, but instead there is a circle around each codeword so that when it is placed on top of L_n one can read off the codewords. To construct G_n, another sequence of graphs, U_n are used. U_n is like G_n, has the same topology as L_n and the codewords are circled. But they are only used in the construction, they do not correspond to a code.

$$G_n = \begin{array}{c} G_{n-1} \\ / \quad \backslash \\ \downarrow G_{n-1} \text{ —— } \uparrow G_{n-1} \end{array} \qquad n \text{ even}$$

$$G_n = \begin{array}{c} G_{n-1} \\ / \quad \backslash \\ U_{n-1} \text{ —— } U_{n-1} \end{array} \qquad n \text{ odd}$$

$$U_n = \begin{array}{c} U_{n-1} \\ / \quad \backslash \\ G_{n-1} \text{ —— } G_{n-1} \end{array} \qquad n \text{ even}$$

$$U_n = \begin{array}{c} U_{n-1} \\ / \quad \backslash \\ \downarrow U_{n-1} \text{ —— } \uparrow U_{n-1} \end{array} \qquad n \text{ odd}$$

Where $G_0 = O$ (a codeword) and $U_0 = X$ (not a codeword). As in the construction of L_n, the arrows show that graphs are rotated.

Theorem 4.1 For every $n \geq 0$, G_n defines a perfect one error correcting code.

To prove this theorem, the following lemma is given:

Lemma 4.1 For each $n \geq 2$,

i) each uncircled vertex is adjacent to exactly one circled vertex, no circled vertices are adjacent in G_n,

ii) each noncorner uncircled vertex is adjacent to exactly one circled vertex, no circled vertices are adjacent in U_n,

iii) if n is odd, all three corner vertices are not adjacent to circled vertices in U_n,

iv) if n is even, the apex vertex is not adjacent to a circled vertex and the other two corner vertices are adjacent to exactly one circled vertex in U_n.

This lemma implies that each vertex is either a codeword or adjacent to exactly one codeword and no two codewords are adjacent, hence the theorem follows.

4.4. Characterizing the Codewords

First the number of the codewords are calculated. Let g_n and u_n denote the number of codewords in G_n and U_n respectively. Then

$$g_n = \begin{cases} 3g_{n-1}, & \text{if } n \text{ is even,} \\ g_{n-1} + 2u_{n-1}, & \text{if } n \text{ is odd.} \end{cases}$$

$$u_n = \begin{cases} 2g_{n-1} + u_{n-1} & \text{if } n \text{ is even,} \\ 3u_{n-1} & \text{if } n \text{ is odd.} \end{cases}$$

$$g_0 = 1, u_0 = 0.$$

These equations come directly from the construction of the graph G_n. Observing that $g_n - u_n = g_{n-1} - u_{n-1} = 1$ for all n gives the recursive equation $g_n = 3g_{n-1} - 1 + (-1)^n$ whose solution is

$$\frac{3^n + 2 + (-1)^n}{4} = \begin{cases} (3^n + 3)/4 & n \text{ even}, \\ (3^n + 1)/4 & n \text{ odd}. \end{cases}$$

Which gives the number of codewords in the T of H code.

Theorem 4.2 The codewords of the T of H code are those n element strings over *0, 1, 2*, which satisfy

i) if n is even, then $\#_0 \equiv \#_1 \equiv \#_2 \equiv 0 \pmod 2$,

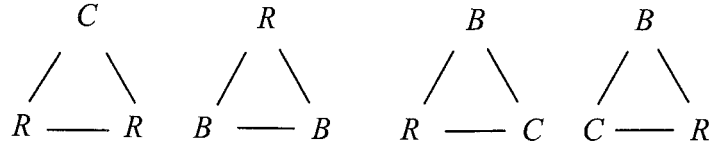
ii) if n is odd, then $\#_0 \equiv 1 \pmod 2$ and $\#_1 \equiv \#_2 \equiv 0 \pmod 2$.

Where $\#_0$ is the number of zeros in the string and $\#_1$ and $\#_2$ are defined similarly.

This theorem is proved in three parts. First, it is shown that the strings satisfying the given conditions, referred as the *satisfying strings*, obey the rules given in the definition of a perfect one error correcting code. Next, the number of these satisfying strings is calculated and it turns out that this number is equal to the number of codewords in the T and H code. Finally, it is shown that the T of H graph has a unique perfect one error correcting code, which has a codeword as the top vertex. The proof of this last comment requires some lemmas.

Definition 4.2 The *predecessor* of a vertex is the unique adjacent vertex in the previous level. The *successors* of a vertex are the adjacent vertices in the next level. A *consistent 3 labeling* for a T of H graph is an assignment of a label from *{C, B, R}* to each vertex so that

i) each triangle in the graph are labeled in one of the following four ways:



ii) the labeling of a successor of a vertex satisfies $succ(C) = R$, $succ(R) = B$, $succ(B) \in \{R, C\}$,

iii) no two vertices labeled C are adjacent,

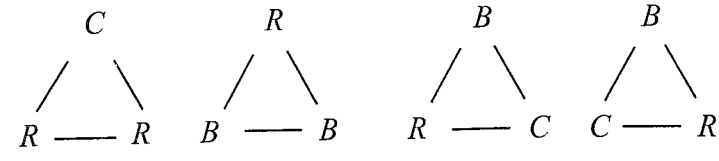
iv) vertices labeled B in the bottom corners are not adjacent to a C, but every other B is adjacent to a C,

v) if the top vertex is labeled R then it is not adjacent to a C, but all other Rs are adjacent to exactly one C.

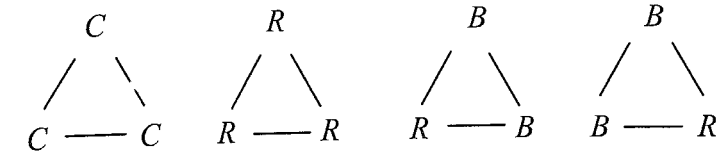
Lemma 4.2 A perfect one error correcting code of H_n induces a consistent *3* labeling of H_n.

Lemma 4.3 Any consistent *3* labeling of H_n gives a consistent *3* labeling for each of the H_{n-1} s used in the construction of H_n.

Lemma 4.4 For each $n \geq 1$, there are exactly four possible consistent *3* labelings of H_n and they have the following forms: for odd n the corner vertices have one of the following four patterns:

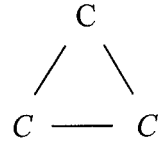


For even n the corner vertices one of the following four patterns:



To prove this, Lemma 4.3 is used.

Finally, we are ready to show the uniqueness. For $n = 0$, H_n is a single vertex and the code in which it is a codeword is a perfect one error correcting code. For $n \geq 1$, if there is a perfect one error correcting code, then there must be a consistent *3* labeling. There are four such labelings for each n. If n is even then ther is only one possible such code, since three of the labelings contain a top R or a bottom B. (Here it is considered that the codewords are the vertices with label C, and the others are noncodewords.) So the pattern is



And it is symmetric with respect to rotation. If n is odd, one of the labelings contain a bottom B, hence cannot be a perfect one error correcting code. There is only one labeling with a codeword with a top C so it is unique.

4.5. Generation of the Codes

To generate the T of H codes, the integers from *0* to $3^n - 1$, represented in base *3*, are used. From the characterization theorem, it is known that a string over *{0, 1, 2}* is a codeword exactly when both the number of *1*s and the number of *2*s are even.

Algorithm 4.1

FOR $I = 0$ TO $3^n - 1$
    IF $\#_1(I) = \#_2(I) = 0 \pmod 2$
    THEN output I in base *3* as a codeword

Another way to generate these codes is to use a recursive algorithm, which is based on the structure of the generating graphs. Here, G_n will denote not only the graph with

codewords circled but the set of strings that correspond to these codewords. In terms of strings, the generation of G_n is represented as:

For n even,

$$G_n = G_{n-1} \circ 0 \cup \mathrm{T}(\mathrm{G}_{n-1}) \circ 1 \cup \mathrm{T}^2(\mathrm{G}_{n-1}) \circ 2$$

where $\circ$ means concatenation (for example $G_{n-1} \circ 0$ means we append 0 to each string in G_{n-1}), $\cup$ is set union and T is the transformation which replaces each character in the string by another character according to the permutation $0 \to 1 \to 2 \to 0$. T^2 means we apply T twice, which will correspond to the permutation $0 \to 2 \to 1 \to 0$.

For n odd,

$$U_n = U_{n-1} \circ 0 \cup T(U_{n-1}) \circ 1 \cup T^2(U_{n-1}) \circ 2,$$

$$G_n = G_{n-1} \circ 0 \cup \Gamma_1(G_{n-1}) \circ 1 \cup \Gamma_2(G_{n-1}) \circ 2,$$

where Γ_1 changes each character by permutation $2 \to 0 \to 2$ and Γ_2 changes each character by permutation $1 \to 0 \to 1$.

For n even,

$$U_n = U_{n-1} \circ 0 \cup \Gamma_1(G_{n-1}) \circ 1 \cup \Gamma_2(G_{n-1}) \circ 2.$$

4.6. Decoding

It is easy to detect an error because we know that there is no error exactly when both the number of *1*s and number of *2*s are even in the string.

If we know $\#_0$, $\#_1$ and $\#_2$, we have to find the first digit i with $\#_i = 1$ and correct it to the next digit j with $\#_j = 1$, $i, j = 0,1,2$.

The following algorithm gives the decoding scheme:

Algorithm 4.2

INPUT: STRING $D = d_1 d_2 \cdots d_n$

$\#_0 = n \ (mod\ 2)$; $\#_1 = \#_2 = 0$

FOR $i = 1$ TO n

$\#_{d_i} = \#_{d_i} + 1 \ (mod\ 2)$

ENDFOR

IF $\#_1 = 1$ and $\#_2 = 1$

THEN scan until first *1* or *2* is found and change it to *2* or *1*

IF $\#_1 = 1$ *and* $\#_2 = 0$

THEN scan until first *0* or *1* is found and change it to *1* or *0*

IF $\#_1 = 0$ *and* $\#_2 = 1$

THEN scan until first *0* or *2* is found and change it to *2* or *0*

Finally, it is shown that the general problem of deciding if a graph has a perfect one error correcting code is NP-complete. For information on NP- completeness, see West [16].

APPENDIX A. SOME BASIC CONCEPTS OF GRAPH THEORY

Definition A.1. A *graph* G is an ordered triple $(V(G), E(G), \Psi_G)$ consisting of a nonempty set $V(G)$ of vertices, a set $E(G)$ of edges, disjoint from $V(G)$, an *incidence* function Ψ_G that associates with each edge of G an unordered pair (not necessarily distinct) vertices of G. If $e \in E(G)$ and $u, v \in V(G)$ then $\Psi_G(e) = uv$ means e joins u to v. Two vertices are called *adjacent* if they are joined by an edge. The *degree* of a vertex v, denoted by $d(v)$, is the number of edges that are incident at v.

Definition A.2. A *subgraph* H of G is a graph with $V(H) \subseteq V(G)$, $E(H) \subseteq E(G)$ and Ψ_H is the restriction of Ψ_G to $E(H)$. A *spanning subgraph* of a graph is a subgraph that contains all the vertices of that graph. A *complete graph* is a graph in which any two vertices are adjacent. A graph, in which the vertex set V can be partitioned into two subsets X and Y, such that each edge has one end in X and one end in Y, is called a *bipartite graph*.

Definition A.3. A *path* between two vertices u and v, denoted by $p(u,v)$, is an alternating sequence of distinct vertices and edges such that each edge is incident at the vertex preceding it and the vertex following it. A graph is said to be *connected* if any pair of vertices can be connected by a path. A *cycle* is a path in which the initial and the final vertices are the same. A cycle that contains all the vertices of the graph is called a *Hamiltonian cycle*. The *length* of each cycle or path is the number of edges in that cycle or path. The *girth* of a graph is the length of the shortest cycle in that graph. The *distance* $d(u, v)$ between vertices u and v is the length of the shortest path between them.

Definition A.4. A spanning subgraph in which every vertex have even degree is called an *even subgraph*. It can be shown that an even subgraph is a cycle or union of edge-disjoint cycles. A *tree* T of a graph G is a connected spanning subraph of a connected graph that contains no cycles. The edges of T are called *branches* and the edges of G-T are called chords.

A *cut-set* is a minimal set of edges of a connected graph such that the removal of these edges makes the graph disconnected. A *seg* is a cut-set or a union of disjoint cut-sets.

Definition A.5. The *edge connectivity* of a graph G, denoted by $\kappa'(G)$, is the minimum number of edges whose deletion disconnects G. A graph is said to be *k-edge connected* if it has edge connectivity at least k.

Definition A.6. A graph is said to be *planar* if it has a drawing such that the edges do not cross except common endpoints. The *faces* of a planar graph are the maximal regions of the plane that are disjoint from the drawing. Two faces are said to be *adjacent* if they share an edge.

Definition A.7. A *k-coloring* of the faces of a graph is a partition of the face set into k independent sets. (an independent set of faces consists of faces that are pairwise non-adjacent). A *color class* is a set of faces receiving the same color.

APPENDIX B. SOME BASIC CONCEPTS OF CODING THEORY

Definition B.1. A linear code $\boldsymbol{C}$ of length n and dimension k is a subspace of $V(n, q)$, the vector space of n-tuples with entries from $GF(q)$. $\boldsymbol{C}$ is called the *q-ary (n, k) code*, (if $q = 2$, then it is called a *binary code*) and the elements of $\boldsymbol{C}$ are called *the codewords*. The $k \times n$ matrix that has a basis of $\boldsymbol{C}$ as its rows is called a *generating matrix* of the code. The *dual* of $\boldsymbol{C}$, is defined as $\boldsymbol{C}^{\perp} = \{ \boldsymbol{u} \in V(N,q) \mid \boldsymbol{u} \cdot \boldsymbol{v} = 0$ (dot product *mod q*), *for all* $\boldsymbol{v} \in \boldsymbol{C} \}$. It can be shown that $\boldsymbol{C}^{\perp}$ is an *(n, n-k)* code. A generating matrix of $\boldsymbol{C}^{\perp}$ is called a *parity check matrix* of **C**. Let $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ be two codes having generating matrices A_1 and A_2 respectively. If there is a code $\boldsymbol{C}$ having a generating matrix of the form

$$A = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix}$$

then $\boldsymbol{C}$ is said to be the *direct sum* of $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$. If a code can be written as a direct sum of two codes, then it is called *decomposable*, otherwise it is called *indecomposable*.

Definition B.2. The *distance d(u, v)* between two codewords $\boldsymbol{u}$ and $\boldsymbol{v}$ is the number of coordinate positions they differ. The *weight* of a codeword u, denoted by $w(\boldsymbol{u})$, is the number of its nonzero coordinate positions. It can easily be seen that $w(\boldsymbol{u} - \boldsymbol{v}) = \mathrm{d}(\boldsymbol{u} - \boldsymbol{v}, \boldsymbol{0}) = d(\boldsymbol{u}, \boldsymbol{v})$. Hence the minimum distance between any two distinct codewords is the same as the minimum weight of the nonzero vectors. If an *(n, k)* code has minimum weight d, we say that it's an *(n, k, d)* code. The *sphere of radius r* is defined to be the set $S_r(u) = \{ v \in V \mid d(u,v) \le r \}$.

A *self-orthogonal code* is a code that is contained in its dual code. If $\boldsymbol{C} = \boldsymbol{C}^{\perp}$, then $\boldsymbol{C}$ is said to be a *self-dual code*. In this case n must be even and $\boldsymbol{C}$ is an *(n, n/2)* code.

When a vector $\boldsymbol{v}$ is received (through a transmission channel etc.) it may be distorted, and we may have to decide which codeword was originally sent. This deciding process is

called *decoding*. The method of decoding a received vector to the closest codeword is called *maximum likelihood* or *majority decoding*.

Theorem B.1. If d is the minimum weight of a code $\boldsymbol{C}$, then $\boldsymbol{C}$ can correct $t = \left[\frac{d-1}{2}\right]$ or fewer errors and conversely.

As seen from this theorem, to correct more errors, we need large minimum weights. In coding theory, given length n, it's important to construct codes with higher dimensions and higher minimum weights. In an *(n, k, d)* code the inequality $n - k \geq d - 1$ must hold, which is called the *Singleton bound*. The inequality $k(k-1)d \leq (nk^2)/2$ is obtained by calculating the average distance between two distinct codewords for a binary code. This is called as the *Plotkin bound*. The *Varshamov-Gilbert* bound assures the existence of a code of length n, minimum distance d or more, and dimension $k \geq n - m$ whenever

$$\binom{n-1}{1} + \binom{n-1}{2} + \cdots + \binom{n-1}{d-2} < 2^m - 1.$$

A code of minimum weight d is called a *perfect code* if all the vectors are contained in the spheres of radius $t = \left[\frac{d-1}{2}\right]$ about the codewords. The *general Hamming codes* over *GF(q)* have the parameters $\left(\frac{q^r - 1}{q-1} = n,\ n - r,\ 3\right)$ and they are single error correcting codes.

Definition B.3. If $\boldsymbol{C}$ is an *(n, k, d)* code, a *shortened code* $\boldsymbol{C}'$ of $\boldsymbol{C}$ is the set of all codewords of $\boldsymbol{C}$ that are *0* in a fixed position with that position deleted. The process of removing a column of a generator matrix of an *(n, k)* code $\boldsymbol{C}$ for even n is called *puncturing* $\boldsymbol{C}$.

REFERENCES

1. Hakimi, S.L., Frank, H., "Cut-Set Matrices and Linear Codes", *IEEE Transactions on Information Theory*, vol. I.T. 11, pp 457-458, July 1965.

2. Bredson, J.G., Hakimi, S.L., "Decoding Graph Theoretic Codes", *IEEE Transactions on Information Theory*, vol. I.T. 13, pp 348-349, April 1967.

3. Hakimi, S.L., Bredson, J.G., "Graph Theoretic Error Correcting Codes", *IEEE Transactions on Information Theory*, vol. I.T. 14, No. 4, pp 584-591, July 1968.

4. Hakimi, S.L., Bredson, J.G., "Ternary Graph Theoretic Error Correcting Codes", *IEEE Transactions on Information Theory*, vol. I.T. 15, pp 435-436, 1969.

5. Bobrow, L.S., "Decoding Augmented Cut-Set Codes", *IEEE Transactions on InformationTheory*, vol. I.T. 17, No. 2, pp 218-220, March 1971.

6. Bobrow, L.S., Hakimi, S.L. "Graph Theoretic Q-ary Codes", *IEEE Transactions on Information Theory*, vol.17, No. 4, pp 215-218, 1971.

7. Jungnickel, D., Vanstone, S.A., "Graphical Codes Revisited", *IEEE Transactions on Information Theory*, vol.43, No. 1, pp. 136-146 , 1997.

8. Jungnickel, D., Vanstone, S.A., "Ternary Graphical Codes", *Journal of Combin. Math. Combin. Comput.*, vol. 29, pp. 17-31, 1999.

9. Jungnickel, D., Vanstone, S.A., "Q-ary Graphical Codes", *Discrete Mathematics* vol. 208/209, pp. 375-386, 1999.

10. Jungnickel, D., Vanstone, S.A., "An Application of Coding Theory to a Problem in Graphical Enumeration", *Archiv. Math.* vol. 65, pp. 461-464, 1995.

11. Jungnickel, D., Vanstone, S.A., "An Application of Differece Sets to a Problem Concerning Graphical Codes", *Journal of Stat. Infer.* vol. 62, pp. 43-46, 1997.

12. Jungnickel, D.,Vanstone, S.A., De Besimini, M.J., "Codes Based on Complete Graphs", *Designs, Codes and Cryptography*, vol. 8. pp. 159-165, 1996.

13. Oral, H., "Constructing Self Dual Codes Using Graphs", *Journal of Comb. Th. Series B.52*, pp. 250-258, 1991.

14. Cull, P., Nelson, I., " Error Correcting Codes on the Tower of Hanoi Graphs", *Discrete Math.* 208-209, pp. 157-175, 1999.

15. Seshu, S., Reed, M.B., *Linear Graphs and Electrical Networks,* Reading, Mass.:Addison Wesley, 1967.

16. West, D.B., *Introduction to Graph Theory*, Prentice Hall, 1996.

17. Van Lint, J.H., *Introduction to Coding Theory*, Springer- Verlag NY 1982.

18. Hakimi, S.L., "Recent Progress and New Problems in Applied Graph Theory", *Proc. 1966 IEEE Region Six Annual Conf.*

19. Pless, V., *An Introduction to Error Correcting Codes*, Wiley-Interscience, NY 1982.

20. MacWilliams, Sloane, *Theory of Errror Correcting Codes,* Amsterdam, The Netherlands: North Holland, 1977.

21. Beth, T., Jungnickel, D., Lenz, H., *Design Theory, vol. 1,* Cambridge University Press, 1999.