An algorithm for construction of extremal and near-extremal \(\mathbb{Z}_4\)-codes

Mravić, Matteo

University of Zagreb

FACULTY OF SCIENCE
DEPARTMENT OF MATHEMATICS

Matteo Mravić

An algorithm for construction of extremal and near-extremal $\mathbb{Z}_4$-codes

DOCTORAL DISSERTATION

Zagreb, 2022.

University of Zagreb

FACULTY OF SCIENCE

DEPARTMENT OF MATHEMATICS

Matteo Mravić

An algorithm for construction of extremal and near-extremal $\mathbb{Z}_4$-codes

DOCTORAL DISSERTATION

Supervisor:

prof. dr. sc. Sanja Rukavina

Zagreb, 2022.

Sveučilište u Zagrebu

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Matteo Mravić

Algoritam konstrukcije ekstremalnih i skoro ekstremalnih $\mathbb{Z}_4$-kodova

DOKTORSKI RAD

Mentor:

prof. dr. sc. Sanja Rukavina

Zagreb, 2022.

IZJAVA O IZVORNOSTI RADA

Ja, Matteo Mravić, student Prirodoslovno-matematičkog fakulteta Sveučilišta u Zagrebu, s prebivalištem na adresi [redacted], [redacted], ovim putem izjavljujem pod materijalnom i kaznenom odgovornošću da je moj doktorski rad pod naslovom: Algoritam konstrukcije ekstremalnih i skoro ekstremalnih $\mathbb{Z}_4$-kodova, isključivo moje autorsko djelo, koje je u potpunosti samostalno napisano uz naznaku izvora drugih autora i dokumenata korištenih u radu.

U Zagrebu, lipanj 2022.

_____

ACKNOWLEDGEMENTS

In the process of writing this dissertation, I received a great deal of support and assistance. Here I would like to express my gratitude to people that helped me on this journey.

First and foremost, I am deeply grateful to my supervisor, Professor Sanja Rukavina, for all the guidance and patience she showed me during this work. Her experience and expertise were invaluable during this process.

Secondly, I would like to express my sincere gratitude to Professor Dean Crnković for his contribution to the construction of Hadamard matrices, and the data that he provided. Without his insight and support, this work would not have been possible.

I would also like to thank the members of the committee for the time and effort invested in reading this paper.

A great thanks goes to my dear colleague Sara Ban for providing me with data and explaining some of her results and methods that I used in this thesis. I would also like to thank all my friends whom I annoyed relentlessly with complaints when I was frustrated, and who always had the compassion for me in such times. They provided me with comfort and strength that enabled me to go on with this work.

Finally, I would like to thank my family for always believing in me. Their support and understanding gave me the strength and courage to finish this work.

SUMMARY

Self-dual $\mathbb{Z}_4$-codes are, depending on their Euclidean weight distribution, divided on Type I and Type II codes. For self-dual $\mathbb{Z}_4$-codes, a theoretical upper bound on the minimum Euclidean weight of a code is known. Codes that meet that upper bound are called extremal $\mathbb{Z}_4$-codes. It is known that Type I extremal $\mathbb{Z}_4$-codes do not exist for lengths 24 and 48. For these lengths, self-dual $\mathbb{Z}_4$-codes that have the best possible minimum weight are called near-extremal $\mathbb{Z}_4$-codes.

The main subject of this thesis are extremal and near-extremal $\mathbb{Z}_4$-codes. It is known that Type II $\mathbb{Z}_4$-codes exist only for lengths divisible by 8. Since we wanted to construct $\mathbb{Z}_4$-codes of Type I and Type II, our work is restricted to such lengths. The known method of construction of a self-dual $\mathbb{Z}_4$-code has a doubly-even binary code of dimension k as a starting point. It consists of choosing one matrix B among the $2^{\frac{k(k+1)}{2}}$ binary $k \times k$ matrices that are suitable for the construction of a self-dual $\mathbb{Z}_4$-code. The usual approach to the construction of extremal or near-extremal $\mathbb{Z}_4$-codes consists of the construction of self-dual $\mathbb{Z}_4$-codes and checking their minimum Euclidean weight. The calculation of the minimum Euclidean weight is necessary to determine extremality or near-extremality of $\mathbb{Z}_4$-codes, but it is also time consuming. Calculation of minimum Euclidean weight of the code gets slower as the length of the examined code gets bigger. This fact, together with the size of the search space, makes this method inefficient. In this thesis, we modified the known method in such way that more than one $\mathbb{Z}_4$-code can be checked for extremality or near-extremality, from one calculation of the minimum Euclidean weight. This increases the efficiency of the existing method. Also, we developed a method to construct a series of Hadamard designs on $4n-1$ points from one skew-symmetric Hadamard matrix of order n. This was motivated by the known fact that incidence matrix of a Hadamard 3-design spans a doubly-even binary code. We used developed algorithms to construct new

extremal $\mathbb{Z}_4$-codes of length 32 and 40. Also, we used the residue codes of the known extremal $\mathbb{Z}_4$-codes of length 40 to construct new extremal $\mathbb{Z}_4$-codes of the same length. Regarding length 48, by our modified method, we obtained already known extremal $\mathbb{Z}_4$-code, and at least two nonequivalent near-extremal $\mathbb{Z}_4$-codes. This near-extremal $\mathbb{Z}_4$-codes are of no great importance since they have the same residue code as the already known extremal $\mathbb{Z}_4$-code of that length. We applied described methods on codes of lengths 56, 64 and 72, but no extremal or near-extremal $\mathbb{Z}_4$-codes were constructed. From obtained codes of length 32 and 40, we constructed strongly regular graphs. All of the obtained graphs were already known. Also, we constructed 1-designs on 32 points from obtained extremal $\mathbb{Z}_4$-codes of length 32. Some of the constructed designs are resolvable.

SAŽETAK

Samodualni $\mathbb{Z}_4$-kodovi, u ovisnosti o njihovoj distribuciji euklidskih težina, podijeljeni su na kodove tipa I i tipa II. Za samodualne $\mathbb{Z}_4$-kodove, teorijske gornje granice minimalnih euklidskih težina su poznate. Kodovi koji dostižu te teorijske granice nazivaju se ekstremalnim $\mathbb{Z}_4$-kodovima. Poznato je da ne postoje ekstremalni $\mathbb{Z}_4$-kodovi tipa I i duljina 24 i 48. Za te duljine, kodovi tipa I koji postižu maksimalnu moguću minimalnu euklidsku težinu nazivaju se skoro ekstremalnim $\mathbb{Z}_4$-kodovima.

Predmet istraživanja ove doktorske disertacije su ekstremalni i skoro ekstremalni $\mathbb{Z}_4$-kodovi. Poznato je da $\mathbb{Z}_4$-kodovi tipa II imaju duljine djeljive s 8. S obzirom da smo htjeli konstruirati $\mathbb{Z}_4$-kodove oba tipa, ograničili smo ovaj rad na duljine kodova djeljive s 8. Poznata metoda konstrukcije samodualnog $\mathbb{Z}_4$-koda polazi od binarnog dvostruko parnog koda dimenzije k, a sastoji se od odabira jedne od $2^{\frac{k(k+1)}{2}}$ binarnih $k \times k$ matrica B koje su pogodne za konstrukciju samodualnog $\mathbb{Z}_4$-koda. Dosadašnji pristup konstrukciji ekstremalnih i skoro ekstremalnih $\mathbb{Z}_4$-kodova sastojao se od slučajnog pretraživanja prostora tih matrica i izračuna minimalne euklidske težine dobivenog samodualnog $\mathbb{Z}_4$-koda. Izračun minimalne euklidske težine samodualnog $\mathbb{Z}_4$-koda nužan je za utvrđivanje ekstremalnosti koda, ali postaje vremenski zahtjevan s porastom duljine koda. Zbog toga, i velikog prostora pretraživanja, ovakav pristup nije efikasan. U ovoj disertaciji modificirali smo opisanu metodu tako da se iz jednog izračuna minimalne euklidske težine $\mathbb{Z}_4$-koda uspješno ispituje ekstremalnost i skoro ekstremalnost većeg broja kodova. Time smo povećali učinkovitost postojeće metode. Također, razvili smo metodu konstrukcije serije Hadamardovih dizajna na $4n-1$ točaka iz koso-simetrične Hadamardove matrice dimenzije n. Motivacija za ovu konstrukciju je poznata činjenica da Hadamardovi 3-dizajni razapinju dvostruko parne binarne kodove, koji su polazna točka konstrukcije samodualnih $\mathbb{Z}_4$-kodova. Pomoću te konstrukcije i modificirane metode konstrukcije ekstremalnih

i skoro ekstremalnih $\mathbb{Z}_4$-kodova, konstruirali smo nove ekstremalne $\mathbb{Z}_4$-kodove duljine 32 i 40. Modificiranu metodu primijenili smo i na rezidualne kodove poznatih ekstremalnih $\mathbb{Z}_4$-kodova duljine 40 te smo, također, konstruirali nove ekstremalne $\mathbb{Z}_4$-kodove. Iz binarnih kodova duljine 48 konstruirali smo jedan, već poznati, ekstremalan $\mathbb{Z}_4$-kod te barem dva neekvivalentna skoro ekstremalna $\mathbb{Z}_4$-koda. Dobiveni skoro ekstremalni kodovi nisu od velikog značaja jer imaju isti rezidualan kod kao i poznat ekstremalan $\mathbb{Z}_4$-kod iste duljine. Konstrukciju smo proveli i za duljine 56, 64 i 72, međutim nismo pronašli ekstremalane niti skoro ekstremalne $\mathbb{Z}_4$-kodove. Iz dobivenih binarnih kodova duljina 32 i 40 konstruirali smo već poznate jako regularne grafove. Također, iz ekstremalnih $\mathbb{Z}_4$-kodova duljine 32 konstruirali smo 1-dizajne sa 32 točke.

KEYWORDS

Extremal $\mathbb{Z}_4$-code, near-extremal $\mathbb{Z}_4$-code, $\mathbb{Z}_4$-code, binary linear code, self-dual code, doubly-even code, Hadamard matrix, Hadamard design, algorithm, Euclidean weight

KLJUČNE RIJEČI

Ekstremalan $\mathbb{Z}_4$-kod, skoro ekstremalan $\mathbb{Z}_4$-kod, $\mathbb{Z}_4$-kod, binaran kod, samodualan kod, dvostruko paran kod, Hadamardova matrica, Hadamardov dizajn, algoritam, euklidska težina

CONTENTS

List of Tables

List of Figures

INTRODUCTION

Coding theory is a branch of mathematics exploring methods for exchanging information in an imperfect communications channel. In an imperfect communication channel errors can be introduced during transmission from source to a receiver. One of the main tasks of coding theory is to construct error-correcting codes which are used to detect and correct these errors. Development of this field began in 1948 when Claude Shannon published the article [47]. In this article, he developed the concept of codes that optimize the ratio of added bits to transmission, and number of errors that can be corrected. In 1950, Richard Hamming published the article [25] in which he introduced the concept of distance between codewords in binary codes, and connected it to the number of errors that can be corrected. Today, this distance is called Hamming distance. In the following years, the rapid development of computer science allowed further development in this field.

A $\mathbb{Z}_4$-code of length n is a submodule of the module $\mathbb{Z}_4^n$. Research on $\mathbb{Z}_4$-codes began in the early 1990s with the publication of the article [26]. In this article, the relationship between $\mathbb{Z}_4$-codes and nonlinear binary Kerdock [37] and Preparata [45] codes was described. Later, it was also shown that $\mathbb{Z}_4$-codes can be used to construct combinatorial objects such as graphs and designs. An example of such a construction is given in the article [24] where $\mathbb{Z}_4$-codes of length 24 are used to construct $5-(24,10,36)$ design.

Three types of codeword weights are defined for $\mathbb{Z}_4$-codes. These are Hamming weight, Euclidean weight, and Lee weight. In our research, we are interested in the Euclidean weight of codewords. Self-dual $\mathbb{Z}_4$-codes are $\mathbb{Z}_4$-codes that are equal to their dual code (with respect to the standard inner product in the $\mathbb{Z}_4$ module $\mathbb{Z}_4^n$). In terms of the Euclidean weight of the codewords, self-dual $\mathbb{Z}_4$-codes are divided into two types. Type II $\mathbb{Z}_4$-codes are self-dual $\mathbb{Z}_4$-codes in which all codewords of the code have the Euclidean weight divisible by 8. Type I $\mathbb{Z}_4$-codes are self-dual $\mathbb{Z}_4$-codes that are not Type II codes.

It is known that Type II $\mathbb{Z}_4$-codes have lengths divisible by 8. Codes of Type I can have arbitrary lengths. In 1993, J. H. Conway and N. J. A. Sloane gave the classification of all self-dual $\mathbb{Z}_4$-codes of lengths up to 8 in [13]. In 1996, P. Gaborit gave the characterization of the self-dual $\mathbb{Z}_4$-codes together with the mass formula in [21]. In 1999, V. Pless, J. S. Leon and J. Fields, in [43], used the mass formula to classify all Type II $\mathbb{Z}_4$-codes of length 16.

In [7], upper bound on the minimum Euclidean weight of $\mathbb{Z}_4$-codes was established. The codes that have minimum Euclidean weight that meets that bound are called extremal $\mathbb{Z}_4$-codes. In [34], all of the residue codes of extremal Type II $\mathbb{Z}_4$-codes of length 24 were classified. In [41], is stated that all extremal Type II $\mathbb{Z}_4$-codes of length 24 were classified by R. A. L. Betty and A. Munemasa. The codes of higher length are not classified. Overview of the known extremal type II $\mathbb{Z}_4$ codes of lengths 32 and 40 can be found in [1] and [2]. New extremal Type II $\mathbb{Z}_4$-codes of length 32, denoted by C_2, C_7, C_{10}, C_{14}, given in [2], are constructed as part of our research. There are exactly two known inequivalent type II extremal $\mathbb{Z}_4$-codes of length 48 [7, 33], and three inequivalent extremal Type II $\mathbb{Z}_4$-codes of length 56 [27, 31]. There is only one known Type II extremal $\mathbb{Z}_4$-code of length 64 given in [27].

Unlike Type II $\mathbb{Z}_4$-codes, Type I codes are less well studied. The upper bound for the minimum Euclidean weight of Type I $\mathbb{Z}_4$-codes of lengths from 25 to 47, except for length 37, is given in [29]. That upper bound is equal to the upper bound given for the Type II codes for Type I $\mathbb{Z}_4$-codes that have length divisible by 8. In [30], the existence of Type I codes of lengths 12, 16, 20, 32, 36, 40 and 44 is proved by applying odd unimodular lattice theory on Type I $\mathbb{Z}_4$-codes. In the same paper, it is proved that Type I $\mathbb{Z}_4$-codes do not exist for lengths 24 and 48. Also, a new upper bound for these two lengths is given. Codes of lengths 24 and 48 whose minimum Euclidean weight is equal to this bound are called near-extremal $\mathbb{Z}_4$-codes. The existence of near-extremal $\mathbb{Z}_4$-codes of length 24 is proved, and one near-extremal $\mathbb{Z}_4$-code of length 48 is given.

Most of the codes of length 32 and greater are constructed by the random search method applied to the characterization of self-dual codes given by P. Gaborit in [21]. Since the calculation of the minimum Euclidean weight of the code is time consuming, we wanted to modify the existing method to increase the number of codes that are checked

for extremality with one calculation of the minimum Euclidean weight. Also, the starting point of the mentioned known construction are doubly-even binary codes. It is known that the incidence matrices of Hadamard 3-designs span doubly-even binary codes [1]. It is well known fact that the Hadamard 3-designs can be obtained from Hadamard 2-designs. Therefore, we developed a method to construct a series of Hadamard 2-designs from a skew-symmetric Hadamard matrix. This construction gives a nice way to generate a series of doubly-even binary codes that can be used to construct $\mathbb{Z}_4$-codes.

This dissertation is divided into four main chapters. In Chapter 1, we give the basic definitions and properties of combinatorial designs, graphs, binary codes and $\mathbb{Z}_4$-codes, needed for better understanding of this thesis. The main scientific contribution of the thesis is in Chapter 2, Chapter 3 and Chapter 4. In Chapter 2, a modification of the existing algorithm is presented. This chapter consists of three subsections. In the first section, we give a theoretical background and overview of the known extremal and near-extremal $\mathbb{Z}_4$-codes. In the second subsection, we give the motivation for the modification of the existing search algorithm. In the third part, the theoretical background of the modification of the existing algorithm is given. Also, the modified algorithm is presented. After that, we compared the two approaches to the traversal of the search space, and analyzed their impact on the performance of the algorithm. In the end of that section, we tested the performance of the modified method on the small example and compared it to the existing method. In Chapter 3, we give the method for construction of series of Hadamard matrices that is used to obtain doubly-even binary codes. Finally, in Chapter 4, we give the overview of the constructed extremal and near-extremal $\mathbb{Z}_4$-codes, and related combinatorial structures.

This thesis has one appendix. It consists of the incidence matrices of the Hadamard 2-designs that were used to construct doubly-even codes of lengths for which our construction, given in Chapter 3, did not provide suitable codes for the construction of the extremal or near-extremal $\mathbb{Z}_4$-codes.

1. PRELIMINARIES

In this chapter we give basic theory of combinatorial structures, Hadamard matrices, binary codes and $\mathbb{Z}_4$-codes. All results given in this chapter are well established and can be found in a variety of sources. For combinatorial structures we used [6, 11, 49]. The subsection on Hadamard matrices is based on [42, 49]. For the general theory on the binary codes and $\mathbb{Z}_4$-codes we used [36, 40]. Other sources in this chapter are referenced as they are needed.

1.1. COMBINATORIAL STRUCTURES

In this section we give some basic theory of combinatorial designs and graphs.

Combinatorial designs

Definition 1.1.1. A t-(v,k,λ) design is a finite incidence structure $\mathscr{D} = (\mathscr{P}, \mathscr{B}, \mathscr{I})$, where $\mathscr{P}$ and $\mathscr{B}$ are disjoint sets and $\mathscr{I} \subseteq \mathscr{P} \times \mathscr{B}$ is the incidence relation, with the following properties:

(i) $|\mathscr{P}| = v$,

(ii) every element of $\mathscr{B}$ is incident with exactly k elements of $\mathscr{P}$,

(iii) every t distinct elements of $\mathscr{P}$ are incident with exactly λ elements of $\mathscr{B}$.

The elements of the set $\mathscr{P}$ are called points and the elements of the set $\mathscr{B}$ are called blocks. A t-(v,k,λ) design is said to be symmetric if $|\mathscr{P}| = |\mathscr{B}|$.

Proposition 1.1.2. Every $t-(v,k,\lambda)$ design is also a $s-(v,k,\lambda_s)$ design with $\lambda_s = \lambda \frac{\binom{v-s}{t-s}}{\binom{k-s}{t-s}}$, and $0 \leq s \leq t$.

Remark 1.1.3. Designs with parameters 2-(v,k,λ) are called balanced incomplete block designs, and are denoted as (v,k,λ)-BIBDs.

Theorem 1.1.4. Let $\mathscr{D}$ be a (v,k,λ)-BIBD. Then the following holds:

(i) Every point of $\mathscr{D}$ occurs in exactly $r = \frac{\lambda(v-1)}{k-1}$ blocks.

(ii) The number of blocks in $\mathscr{D}$ is $b = \frac{vr}{k} = \frac{\lambda(v^2-v)}{k^2-k}$.

Remark 1.1.5. The number r from the previous theorem is called the replication number of a (v,k,λ)-BIBD. When the parameters r and b want to be emphasized for a (v,k,λ)-BIBD, we will denote it as a (v,b,r,k,λ)-BIBD. From Theorem 1.1.4 it follows that $r = k$ for a symmetric BIBD.

Definition 1.1.6. Let $\mathscr{D} = (\mathscr{P},\mathscr{B},\mathscr{I})$ be a $t-(v,k,\lambda)$ design with the set of points $\mathscr{P} = \{P_j | j = 1,2,\ldots,v\}$, and the blocks $\mathscr{B} = \{B_i | i = 1,2,\ldots,b\}$. The incidence matrix of the design $\mathscr{D}$ is a $b \times v$ matrix $M = [m_{ij}]$ such that:

$$m_{ij} = \begin{cases} 1, & (P_j, B_i) \in \mathscr{I}, \\ 0, & (P_j, B_i) \notin \mathscr{I}. \end{cases}$$

The following theorem characterizes binary matrices that are incidence matrices of (v,b,r,k,λ)-BIBDs.

Theorem 1.1.7. Let M be a $b \times v$ binary matrix. The matrix M is an incidence matrix of a (v,b,r,k,λ)-BIBD if and only if both of the following conditions hold:

(i) $M^T M = \lambda J_v + (r-\lambda) I_v$,

(ii) $u_v M^T = k u_b$,

where J_v is the $v \times v$ matrix of all-ones, I_v is the identity matrix of order v, u_v and u_b are all-one vectors of length v and b, respectively.

Remark 1.1.8. Notice that the condition (i) is equivalent to the condition: each point in a (v,b,r,k,λ)-BIBD is incident with r blocks, and every two different points are incident with λ common blocks. The condition (ii) is equivalent to the condition: every block in a (v,b,r,k,λ)-BIBD is incident with k points.

Definition 1.1.9. Let $\mathscr{D}_1 = (\mathscr{P}_1, \mathscr{B}_1, \mathscr{I}_1)$ and $\mathscr{D}_2 = (\mathscr{P}_2, \mathscr{B}_2, \mathscr{I}_2)$ be t-(v,k,λ) designs. The designs $\mathscr{D}_1$ and $\mathscr{D}_2$ are isomorphic if there exists a bijection $f : \mathscr{P}_1 \cup \mathscr{B}_1 \to \mathscr{P}_2 \cup \mathscr{B}_2$ such that f maps $\mathscr{P}_1$ on $\mathscr{P}_2$ and $\mathscr{B}_1$ on $\mathscr{B}_2$, and the following holds:

$$(p,B) \in \mathscr{I}_1 \Leftrightarrow (f(p), f(B)) \in \mathscr{I}_2,\ p \in \mathscr{P}_1,\ B \in \mathscr{B}_1.$$

The function f is called an isomorphism of the designs $\mathscr{D}_1$ and $\mathscr{D}_2$. If $\mathscr{D}_1 = \mathscr{D}_2$ then such mapping is called an automorphism of the design $\mathscr{D}_1$.

Remark 1.1.10. The set of all automorphisms of a design $\mathscr{D}$, together with the composition of the functions, forms a group. This group is called the full automorphism group of the design $\mathscr{D}$, and it is denoted by $\mathrm{Aut}(\mathscr{D})$.

Theorem 1.1.11. Let $\mathscr{D}$ and $\mathscr{D}'$ be $t-(v,k,\lambda)$ designs with incidence matrices $M = [m_{i,j}]$ and $M' = [m'_{i,j}]$. Designs $\mathscr{D}$ and $\mathscr{D}'$ are isomorphic if and only if there exists a permutation σ of the set $\{1,2,\ldots,v\}$ and a permutation τ of the set $\{1,2,\ldots,b\}$ such that:

$$m'_{i,j} = m_{\tau(i),\sigma(j)},$$

for all $i \in \{1,2,\ldots,b\}$ and $j \in \{1,2,\ldots,v\}$.

Theorem 1.1.12. Let $\mathscr{D} = (\mathscr{P}, \mathscr{B}, \mathscr{I})$ be a (v,b,r,k,λ)-BIBD. Let $\overline{\mathscr{I}}$ be a complement relation of the relation $\mathscr{I}$, i.e.:

$$(p,B) \in \overline{\mathscr{I}} \Leftrightarrow (p,B) \notin \mathscr{I}.$$

Then $\overline{\mathscr{D}} = \left(\mathscr{P}, \mathscr{B}, \overline{\mathscr{I}}\right)$ is a $(v,b,b-r,v-k,b-2r+\lambda)$-BIBD.

Definition 1.1.13. Let $\mathscr{D}$ and $\overline{\mathscr{D}}$ be as in the previous theorem. Design $\overline{\mathscr{D}}$ is called the complementary design of the design $\mathscr{D}$.

Remark 1.1.14. If M is an incidence matrix of a BIBD $\mathscr{D}$, then $J - M$ is an incidence matrix of its complementary design, where J is the $b \times v$ matrix of all-ones.

Definition 1.1.15. Let $\mathscr{D}$ be a design with v points and b blocks. Let M be a $b \times v$ incidence matrix of $\mathscr{D}$. The dual design of the design $\mathscr{D}$ is the design with incidence matrix M^T.

Theorem 1.1.16. If B and B' are two different blocks of a symmetric (v,k,λ)-BIBD, then B and B' are incident with λ common points.

Corollary 1.1.17. If M is an incidence matrix of a symmetric (v,k,λ)-BIBD, then M^T is also an incidence matrix of a symmetric (v,k,λ)-BIBD.

Definition 1.1.18. A symmetric (v,k,λ)-BIBD is self-dual if it is isomorphic to its dual design.

Definition 1.1.19. A $t-(v,k,\lambda)$ design $\mathscr{D}$ is quasi-symmetric with the intersection numbers x and y, $x<y$, if any two blocks of $\mathscr{D}$ intersect in either x or y points.

Example 1.1.20. Here we give an example of a quasi-symmetric design. Let $S(3,6,22)$ denote the $3-(22,6,1)$ design with the set of points $\mathscr{P}=\{1,2,\ldots,22\}$, and the following blocks:

$\{1,2,3,5,9,16\}$, $\{1,3,4,6,10,17\}$, $\{1,4,5,7,11,18\}$, $\{1,5,6,8,12,19\}$,
$\{1,2,6,7,13,20\}$, $\{1,3,7,8,14,21\}$, $\{1,2,4,8,15,22\}$, $\{1,2,10,18,19,21\}$,
$\{1,3,11,19,20,22\}$, $\{1,4,12,16,20,21\}$, $\{1,5,13,17,21,22\}$, $\{1,6,14,16,18,22\}$,
$\{1,7,15,16,17,19\}$, $\{1,8,9,17,18,20\}$, $\{1,2,11,12,14,17\}$, $\{1,3,12,13,15,18\}$,
$\{1,4,9,13,14,19\}$, $\{1,5,10,14,15,20\}$, $\{1,6,9,11,15,21\}$, $\{1,7,9,10,12,22\}$,
$\{1,8,10,11,13,16\}$, $\{2,3,4,7,12,19\}$, $\{3,4,5,8,13,20\}$, $\{2,4,5,6,14,21\}$,
$\{3,5,6,7,15,22\}$, $\{4,6,7,8,9,16\}$, $\{2,5,7,8,10,17\}$, $\{2,3,6,8,11,18\}$,
$\{2,3,10,13,14,22\}$, $\{3,4,11,14,15,16\}$, $\{4,5,9,12,15,17\}$, $\{5,6,9,10,13,18\}$,
$\{6,7,10,11,14,19\}$, $\{7,8,11,12,15,20\}$, $\{2,8,9,12,13,21\}$, $\{2,3,15,17,20,21\}$,
$\{3,4,9,18,21,22\}$, $\{4,5,10,16,19,22\}$, $\{5,6,11,16,17,20\}$, $\{6,7,12,17,18,21\}$,
$\{7,8,13,18,19,22\}$, $\{2,8,14,16,19,20\}$, $\{2,4,9,10,11,20\}$, $\{3,5,10,11,12,21\}$,
$\{4,6,11,12,13,22\}$, $\{5,7,12,13,14,16\}$, $\{6,8,13,14,15,17\}$, $\{2,7,9,14,15,18\}$,
$\{3,8,9,10,15,19\}$, $\{2,4,13,16,17,18\}$, $\{3,5,14,17,18,19\}$, $\{4,6,15,18,19,20\}$,
$\{5,7,9,19,20,21\}$, $\{6,8,10,20,21,22\}$, $\{2,7,11,16,21,22\}$, $\{3,8,12,16,17,22\}$,
$\{2,5,11,13,15,19\}$, $\{3,6,9,12,14,20\}$, $\{4,7,10,13,15,21\}$, $\{5,8,9,11,14,22\}$,
$\{2,6,10,12,15,16\}$, $\{3,7,9,11,13,17\}$, $\{4,8,10,12,14,18\}$, $\{2,5,12,18,20,22\}$,
$\{3,6,13,16,19,21\}$, $\{4,7,14,17,20,22\}$, $\{5,8,15,16,18,21\}$, $\{2,6,9,17,19,22\}$,
$\{3,7,10,16,18,20\}$, $\{4,8,11,17,19,21\}$, $\{9,10,14,16,17,21\}$, $\{10,11,15,17,18,22\}$,
$\{9,11,12,16,18,19\}$, $\{10,12,13,17,19,20\}$, $\{11,13,14,18,20,21\}$, $\{12,14,15,19,21,22\}$,
$\{9,13,15,16,20,22\}$

It can be checked that every two blocks from the given list intersect in either 2 points, or are disjoint. Therefore, this design is quasi-symmetric with intersection numbers $x = 0$ and $y = 2$.

Remark 1.1.21. Designs with parameters $t-(v,k,1)$ are called Steiner systems, and are denoted as $S(t,k,v)$.

Definition 1.1.22. A $t-(v,k,\lambda)$ design $\mathscr{D}$ is resolvable if there exists a partition $\mathscr{A}$ of the set of blocks of $\mathscr{D}$ such that every $A \in \mathscr{A}$ is a partition of the set of points of $\mathscr{D}$. The partition $\mathscr{A}$ is called a resolution of $\mathscr{D}$, and its elements are called parallel classes of $\mathscr{D}$.

Graphs

Definition 1.1.23. A graph G is an ordered pair of finite disjoint sets (V,E) such that E is a subset of the set $V^{(2)}$ of unordered pairs of V. The set V is the set of vertices, and E is the set of edges. If G is a graph, then $V = V(G)$ is the vertex set of G, and $E = E(G)$ is the set of edges of G. If $e = \{x,y\} \in E(G)$, then vertices x and y are adjacent vertices of G. Also, x and y are incident with the edge e. Vertices that are adjacent to some vertex x are neighbors of x.

Definition 1.1.24. Let G be a graph, and let $e \in E(G)$. The edge e is a loop if it is incident with exactly one vertex of G. A graph G is a simple graph if none of the edges are loops, and every two vertices of G are incident with at most one edge of G.

Definition 1.1.25. Let $G = (V,E)$ be a simple graph. A complement of G is a simple graph $\overline{G}$ such that $V\left(\overline{G}\right) = V(G)$, and two vertices are adjacent in $\overline{G}$ if they are not adjacent in G.

Definition 1.1.26. Let G be a graph, and let $v \in V(G)$. The degree of the vertex v, denoted by $d_G(v)$, is the number of the edges of G that are incident with v, where the loops are counted twice. If $d_G(v) = k$, for every $v \in V(G)$, then G is a k-regular graph.

Theorem 1.1.27. Let G be a graph with $V(G) = \{v_1, v_2, \ldots, v_n\}$, and let $\varepsilon = |E(G)|$. Then:

$$\sum_{i=1}^{n} d_G(v_i) = 2\varepsilon.$$

Definition 1.1.28. A simple k-regular graph with v vertices is a strongly regular graph with parameters (v,k,λ,μ) if every two adjacent vertices have λ common neighbors, and if every two non adjacent vertices have μ common neighbors.

Theorem 1.1.29. The complement of a strongly regular graph with parameters (v,k,λ,μ) is a strongly regular graph with parameters $(v, v-k-1, v-2-2k+\mu, v-2k+\lambda)$.

Example 1.1.30. Let $J(n,k)$ be a graph with the vertex set:

$$V = \left\{ S \subset \{1,2,\ldots,n\} \,\middle|\, |S| = k \right\}.$$

Let the edges of $J(n,k)$ be defined as follows:

$$\left\{ v, v' \right\} \in E \Leftrightarrow \left| v \cap v' \right| = k-1.$$

The described graph is called the Johnson graph. The special subfamily of the Johnson graphs are triangular graphs. The n-triangular graph is $T(n) = J(n,2)$. Also, a complete graph on n vertices K_n, i.e., a simple graph with n vertices in which every vertex is adjacent to all of the remaining vertices, is $J(n,1)$. In Figure 1.1 some Johnson graphs are presented. As stated, $J(5,1) = K_5$, and $J(4,2) = T(4)$.

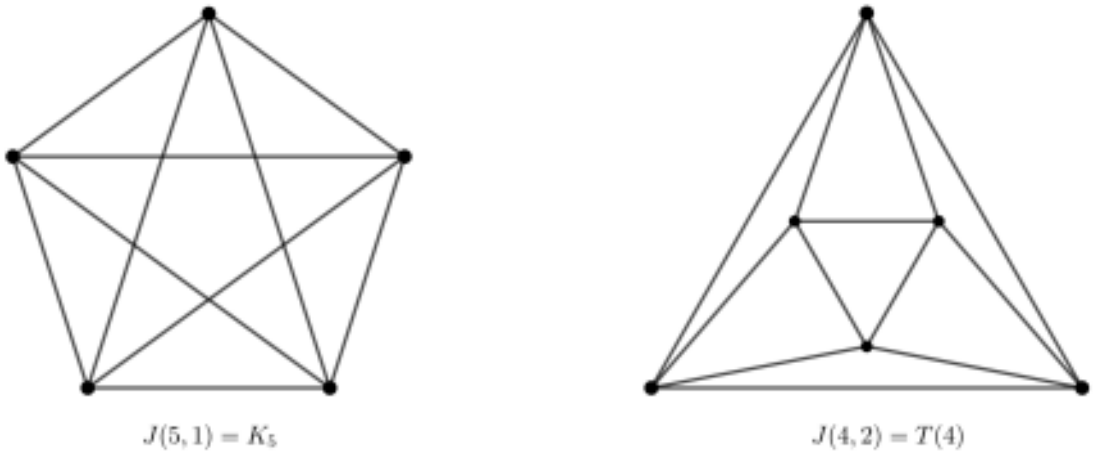

Figure 1.1: Examples of Johnson graphs

It can be shown that the n-triangular graph is a strongly regular graph with parameters $\left(\frac{n(n-1)}{2}, 2(n-2), n-2, 4\right)$.

Definition 1.1.31. A graph G is bipartite if there exists a partition $\{X,Y\}$ of the set of vertices $V(G)$ such that if $\{v_1,v_2\} \in E(G)$ then $v_1 \in X$ and $v_2 \in Y$, or $v_2 \in X$ and $v_1 \in Y$.

Definition 1.1.32. Let V be the set of all binary sequences of length n, and let E be the set of 2-subsets of V such that $\{v_1,v_2\} \in E$ if and only if v_1 and v_2 differ in exactly one coordinate. The graph $Q_n = (V,E)$ is called the n-hypercube graph.

Remark 1.1.33. The n-hypercube graph Q_n is bipartite graph for $n \geq 2$. One block of the partition consists of all sequences that have an even number of ones, and the other one consists of the all sequences that have an odd number of ones.

The following definition and theorem can be found in [48].

Definition 1.1.34. The block graph Γ of a quasi-symmetric $2-(v,k,\lambda)$ design $\mathscr{D} = (\mathscr{P},\mathscr{B},\mathscr{I})$ with intersection numbers x and y is the graph with vertex set $V(\Gamma) = \mathscr{B}$, such that blocks B and B' are adjacent if and only if $|B \cap B'| = y$.

Theorem 1.1.35. Let $\mathscr{D}$ be a quasi-symmetric $2-(v,b,r,k,\lambda)$ design with intersection numbers x and y. Then Γ is a strongly regular graph with parameters (n,a,c,d) where $n = b$, $a = \frac{k(r-1)-x(b-1)}{y-x}$, $c = a+\theta_1\theta_2+\theta_1+\theta_2$, $d = a+\theta_1\theta_2$, $\theta_1 = \frac{r-\lambda-k+x}{y-x}$, $\theta_2 = \frac{x-k}{y-x}$.

Example 1.1.36. Here we will describe the block graph of $S(3,6,22)$ given in Example 1.1.20. By Proposition 1.1.2 we know that $S(3,6,22)$ is also a $2-(22,77,21,6,5)$ design. Also, we know that blocks of $S(3,6,22)$ intersect in either $x = 0$ points or $y = 2$ points. By the previous theorem we get that the block graph of $S(3,6,22)$ is a strongly regular graph with parameters $(77,60,47,45)$. Notice that in this graph blocks B and B' are adjacent if and only if $|B \cap B'| = 2$. By Theorem 1.1.29, the complement of that graph is also strongly regular graph with parameters $(77,16,0,4)$. It is obvious that this graph is the block graph of $S(3,6,22)$ where blocks B and B' are adjacent if and only if $|B \cap B'| = 0$.

1.2. HADAMARD MATRICES AND HADAMARD DESIGNS

1.2.1. Definitions and properties

We start this section by giving the definition of a Hadamard matrix.

Definition 1.2.1. Let $n \in \mathbb{N}$. A Hadamard matrix of order n is a square matrix H of order n such that every entry of H is in $\{-1,1\}$, and $HH^T = H^T H = nI_n$.

From the definition one can immediately conclude that if H is a Hadamard matrix of order n then H^T is also a Hadamard matrix of the same order. This follows from:

$$H^T \left(H^T\right)^T = H^T H = nI_n,$$
$$\left(H^T\right)^T H^T = HH^T = nI_n.$$

There are two Hadamard matrices of order 1:

$$H_1 = \begin{bmatrix} 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix},$$
$$H_2 = \begin{bmatrix} -1 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 \end{bmatrix} \cdot \begin{bmatrix} -1 \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix}$$

The following matrices are examples of Hadamard matrices of order 2 and 4:

$$H_3 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = 2 \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$H_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} = 4 \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Let r_i, $i \in \{1,2,\ldots,n\}$, denote the i-th row of the square matrix H of order n. Then the condition from Definition 1.2.1 is equivalent to:

$$r_i \cdot r_j = \begin{cases} n, & i = j, \\ 0, & i \neq j, \end{cases}$$

where $r_i \cdot r_j$ denotes the usual coordinate-wise scalar product of vectors in $\mathbb{R}^n$. It is obvious from this interpretation, that the matrix obtained from a Hadamard matrix by permuting its rows and columns is a Hadamard matrix of the same order.

Proposition 1.2.2. Let H be a Hadamard matrix of order n, and let H' be the matrix obtained from H by multiplication of any number of rows and columns of H by -1. Then H' is a Hadamard matrix of order n.

Based on the previous proposition, and the discussion before it, we give the following definition.

Definition 1.2.3. Let H be a Hadamard matrix of order n, and let H' be the Hadamard matrix obtained from H by applying a sequence of the following elementary transformations:

- permutation of rows of H,
- permutation of columns of H,
- multiplication of some rows or columns of H by -1.

We say that H and H' are equivalent Hadamard matrices.

Definition 1.2.4. A Hadamard matrix H of order n is normalized (or in standard form) if every entry of the first row and first column of H is equal to 1.

Proposition 1.2.5. Every Hadamard matrix is equivalent to some Hadamard matrix in standard form.

Example 1.2.6. Here we give an example of equivalent Hadamard matrices of order 8. The second Hadamard matrix is normalized. With red color, we marked the row that has been multiplied by -1. and with green color we marked rows that have been permuted.

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ - & 1 & - & - & 1 & - & 1 & 1 \\ 1 & - & - & 1 & 1 & - & 1 & - \\ 1 & 1 & - & - & - & 1 & 1 & - \\ 1 & - & - & - & 1 & 1 & - & 1 \\ 1 & - & 1 & - & - & - & 1 & 1 \\ 1 & 1 & - & 1 & - & - & - & 1 \\ 1 & 1 & 1 & - & 1 & - & - & - \end{bmatrix} \approx \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & - & 1 & 1 & - & 1 & - & - \\ 1 & - & - & 1 & 1 & - & 1 & - \\ 1 & - & - & - & 1 & 1 & - & 1 \\ 1 & 1 & - & - & - & 1 & 1 & - \\ 1 & - & 1 & - & - & - & 1 & 1 \\ 1 & 1 & - & 1 & - & - & - & 1 \\ 1 & 1 & 1 & - & 1 & - & - & - \end{bmatrix}$$

In the beginning of this chapter we presented examples of Hadamard matrices of orders 1, 2, and 4. The following theorem gives a necessary condition for the existence of Hadamard matrix of order n.

Theorem 1.2.7. If there exists a Hadamard matrix of order $n > 2$ then $n \equiv 0 \pmod 4$.

Remark 1.2.8. It is conjectured that the condition from the previous theorem is also sufficient for the existence of Hadamard matrices i.e. that a Hadamard matrix of order $4k$ exists for every $k \in \mathbb{N}$. This conjecture is known as the Hadamard conjecture, and it is still an open problem. The smallest order for which the existence of the Hadamard matrix is not proved is 668.

Now we introduce a special family of Hadamard matrices that will be used in the later construction.

Definition 1.2.9. A Hadamard matrix of order n is a skew-type Hadamard matrix if $H = A + I_n$, where $A^T = -A$ and I_n is the identity matrix of order n.

In [39], a survey of the known non-equivalent skew-type Hadamard matrices is given. There are unique, up to the equivalence, skew-type Hadamard matrices of order 8 and 12. The representatives of these equivalence classes are SH_8 and SH_{12} given in (1.1). For the order 16, there are two skew-type Hadamard matrices, up to the equivalence. These are the matrix SH_{16}, given in (1.2), and its transpose.

$$SH_8 = \begin{bmatrix} +&+&+&+&+&+&+&+ \\ -&+&+&+&-&+&-&- \\ -&-&+&+&+&-&+&- \\ -&-&-&+&+&+&-&+ \\ -&+&-&-&+&+&+&- \\ -&-&+&-&-&+&+&+ \\ -&+&-&+&-&-&+&+ \\ -&+&+&-&+&-&-&+ \end{bmatrix}, SH_{12} = \begin{bmatrix} +&-&-&-&-&-&-&-&-&-&-&- \\ +&+&-&+&-&-&-&+&+&+&-&+ \\ +&+&+&-&+&-&-&-&+&+&+&- \\ +&-&+&+&-&+&-&-&-&+&+&+ \\ +&+&-&+&+&-&+&-&-&-&+&+ \\ +&+&+&-&+&+&-&+&-&-&-&+ \\ +&+&+&+&-&+&+&-&+&-&-&- \\ +&-&+&+&+&-&+&+&-&+&-&- \\ +&-&-&+&+&+&-&+&+&-&+&- \\ +&-&-&-&+&+&+&-&+&+&-&+ \\ +&+&-&-&-&+&+&+&-&+&+&- \\ +&-&+&-&-&-&+&+&+&-&+&+ \end{bmatrix}. \tag{1.1}$$

$$SH_{16} = \begin{bmatrix} +&+&+&+&+&+&+&+&+&+&+&+&+&+&+&+ \\ -&+&+&+&-&+&-&-&+&-&-&+&-&+&+&+ \\ -&-&+&+&+&-&+&-&+&-&+&-&+&+&+&- \\ -&-&-&+&+&+&-&+&+&+&-&+&+&+&-&- \\ -&+&-&-&+&+&+&-&+&-&+&+&+&-&-&+ \\ -&-&+&-&-&+&+&+&+&+&+&-&-&+&-&- \\ -&+&-&+&-&-&+&+&+&+&-&-&+&-&+&- \\ -&+&+&-&+&-&-&+&+&+&-&-&+&-&-&+ \\ -&-&-&-&-&-&-&-&+&+&+&+&+&+&+&+ \\ -&+&+&-&+&-&-&-&+&+&+&-&+&-&-&- \\ -&+&-&+&-&-&-&+&-&-&+&+&+&-&+&- \\ -&-&+&-&-&-&+&+&-&-&-&+&+&+&-&+ \\ -&+&-&-&-&+&+&-&+&-&-&+&+&+&-&- \\ -&-&-&-&+&+&-&+&-&-&+&-&-&+&+&+ \\ -&-&-&+&+&-&+&-&+&-&-&+&-&+&+&- \\ -&-&+&+&-&+&-&-&-&+&+&-&+&-&-&+ \end{bmatrix} \tag{1.2}$$

1.2.2. Hadamard designs

In this section we describe well known connection between Hadamard matrices and combinatorial designs.

Theorem 1.2.10. Let $m > 4$ and $m \equiv 0 \pmod 4$. Hadamard matrix of order m exists if and only if a symmetric $2-(m-1, \frac{m}{2}-1, \frac{m}{4}-1)$ design exists.

A combinatorial design with parameters from the previous theorem is called a Hadamard design. The incidence matrix of a Hadamard design can be obtained from a normalized Hadamard matrix H by deleting the first row and column of H and then replacing every -1 entry with 0 (see the construction of the matrix M in Example 1.2.13). This process is reversible. If M is an incidence matrix of a Hadamard design, then a normalized Hadamard matrix is obtained from M by changing every 0 entry of M to -1 and adding a new first row and column that have all entries equal to 1.

The following theorem gives a method for extending Hadamard designs to 3-designs.

Theorem 1.2.11. Let M be the incidence matrix of a Hadamard $2-(m-1, \frac{m}{2}-1, \frac{m}{4}-1)$ design. Then the matrix:

$$M' = \left[\begin{array}{c|c} 1 & \text{M} \\ \hline 0 & \text{J-M} \end{array}\right],$$

where J is a $(m-1)\times(m-1)$ all-one matrix, is the incidence matrix of a $3-(m, \frac{m}{2}, \frac{m}{4}-1)$ design.

Remark 1.2.12. In the continuation of this thesis, we will refer to combinatorial designs with parameters from Theorem 1.2.11 as Hadamard 3-designs.

In the end of this section we give an example of the construction of a Hadamard $3-(8,4,1)$ design from a Hadamard matrix of order 8.

Example 1.2.13. We start with the Hadamard matrix SH_8 given in (1.1). When this matrix is normalized, the following matrix is obtained:

$$SH_8' = \left[\begin{array}{c|cccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline 1 & - & 1 & 1 & - & 1 & - & - \\ 1 & - & - & 1 & 1 & - & 1 & - \\ 1 & - & - & - & 1 & 1 & - & 1 \\ 1 & 1 & - & - & - & 1 & 1 & - \\ 1 & - & 1 & - & - & - & 1 & 1 \\ 1 & 1 & - & 1 & - & - & - & 1 \\ 1 & 1 & 1 & - & 1 & - & - & - \end{array}\right].$$

The lines inside of matrix SH_8' denote the first row and column which have all entries equal to 1. When first row and column of SH_8' are deleted, and all -1 entries are replaced by 0, an incidence matrix of the Hadamard $2-(7,3,1)$ design is obtained. This matrix is given as follows:

$$M = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Notice that the matrix M is a skew binary matrix. Now one can obtain the incidence

matrix of the Hadamard $3-(8,4,1)$ design from M by Theorem 1.2.11. This matrix is:

$$M' = \left[\begin{array}{cccccccc} 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array}\right].$$

1.3. BINARY CODES

In this section by $\mathbb{F}_2$ we denote the field with the standard modulo 2 addition and multiplication.

Definition 1.3.1. Let $\mathbb{F}_2^n$, $n \in \mathbb{N}$, be the n-dimensional vector space over the field $\mathbb{F}_2$. Let $k \in \mathbb{N}$. A binary linear $[n,k]$ code C of length n and dimension k is a k-dimensional vector subspace of the space $\mathbb{F}_2^n$.

Definition 1.3.2. Let $x \in \mathbb{F}_2^n$. The Hamming weight of x, denoted by $wt_H(x)$, is the number of non-zero coordinates of x.

Definition 1.3.3. For $x, y \in \mathbb{F}_2^n$, the Hamming distance of x and y is $d_H(x,y) = wt_H(x-y)$.

Remark 1.3.4. It can easly be shown that d_H is a metric on $\mathbb{F}_2^n$.

Definition 1.3.5. Let C be a binary linear $[n,k]$ code. The minimum distance of the code C is the number:

$$d(C) = \min\left\{ d_H(x,y) \middle| x,y \in C, x \neq y \right\}.$$

If the code C has minimum distance d, then it is denoted as a $[n,k,d]$ code.

Lemma 1.3.6. Let C be a binary linear $[n,k,d]$ code. Then:

$$d(C) = \min\left\{ wt_H(x) \middle| x \in C, x \neq 0 \right\}.$$

Definition 1.3.7. Let C be a binary linear $[n,k]$ code, and $(b_1, b_2, \ldots, b_k)$ a basis of C. Then the $k \times n$ matrix G:

$$G = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix},$$

is the generator matrix of the code C. The matrix G is in standard form if it is of the form:

$$G = \begin{bmatrix} I_k & A \end{bmatrix},$$

where I_k is the $k \times k$ identity matrix, and A is a $k \times (n-k)$ binary matrix.

Definition 1.3.8. Let C and C' be binary linear $[n,k]$ codes. The codes C and C' are equivalent if there exists a $n \times n$ permutation matrix P such that $C' = \{xP | x \in C\}$.

Remark 1.3.9. Since binary linear codes are vector spaces, it is easy to show that the codes C and C', with generator matrices G and G', are equivalent if and only if there exists a $n \times n$ permutation matrix P such that $G' = GP$.

Theorem 1.3.10. Every binary linear code is equivalent to some binary linear code which has a generator matrix in standard form.

Definition 1.3.11. Let $x, y \in \mathbb{F}_2^n$, $x = (x_1, x_2, \ldots, x_n)$, $y = (y_1, y_2, \ldots, y_n)$. The standard inner product of x and y is defined as the ordinary dot product of x and y modulo 2, i.e.:

$$x \cdot y = x_1 y_1 + x_2 y_2 + \ldots + x_n y_n \,(\text{mod } 2)\,. \tag{1.3}$$

Remark 1.3.12. Notice that addition and multiplication in right side of the equation (1.3) are usual addition and multiplication over $\mathbb{Z}$.

Definition 1.3.13. Let C be a $[n,k]$ binary linear code. The dual code of C is the code:

$$C^{\perp} = \left\{ x \in \mathbb{F}_2^n \middle| x \cdot y = 0, \forall y \in C \right\}.$$

Proposition 1.3.14. If C is a binary linear $[n,k]$ code then $C^{\perp}$ is a binary linear $[n, n-k]$ code and $\left(C^{\perp}\right)^{\perp} = C$.

Definition 1.3.15. Let C be a binary linear code and $C^{\perp}$ its dual code. The code C is self-orthogonal if $C \subset C^{\perp}$, and self-dual if $C = C^{\perp}$.

Definition 1.3.16. A binary linear code C is even if for every $x \in C$ the number $wt_H(x)$ is divisible by 2. It is a doubly-even code if for every $x \in C$ the number $wt_H(x)$ is divisible by 4.

Theorem 1.3.17. Let C be a binary linear code. The following holds:

(i) If C is self-orthogonal and has generator matrix G such that every row of G has weight divisible by 4, then C is doubly-even.

(ii) If C is doubly-even, then C is self-orthogonal.

Definition 1.3.18. Let $m \geq 1$. The Reed-Muller codes of the first order $R(1,m)$ are the binary linear codes defined as follows:

(i) $R(1,1) = \mathbb{F}_2^2$,

(i) For $m > 1$, $R(1,m) = \left\{(u,u), (u,u+\mathbb{1}) \,\middle|\, u \in R(1,m-1)\right\}$, where $\mathbb{1}$ denotes the codeword of all ones.

The following Theorem from [51] gives connection between combinatorial designs and doubly-even codes.

Theorem 1.3.19. Assume that $\mathscr{D}$ is a 2-(v,k,λ) design with block intersection numbers $s_1,\ldots,s_m$. Denote by C the binary code spanned by the incidence matrix of $\mathscr{D}$. If $v \equiv 0 \pmod 8$, $k \equiv 0 \pmod 4$ and $s_1,\ldots,s_m$ are all even, then C_1 is contained in a doubly-even self-dual code of length v.

Let H be a Hadamard matrix of order n. By $C_2(H)$ we denote the binary linear code spanned by the incidence matrix of the Hadamard 3-design defined by H. The following Corollary of previous theorem is proved in [1].

Corollary 1.3.20. Let H be a Hadamard matrix of order $n \equiv 0 \pmod 8$. Then $C_2(H)$ is a doubly-even binary code of length n.

1.4. $\mathbb{Z}_4$-CODES

Definition 1.4.1. Let $\mathbb{Z}_4$ be the ring of integers modulo 4 with usual modulo 4 addition and multiplication. A $\mathbb{Z}_4$-code C of length n, $n \in \mathbb{N}$, is a $\mathbb{Z}_4$ submodule of the $\mathbb{Z}_4$-module $\mathbb{Z}_4^n$. Elements of $\mathbb{Z}_4$-code C are called codewords of C.

The following definition gives the generating set for a $\mathbb{Z}_4$-code.

Definition 1.4.2. Every $\mathbb{Z}_4$-code C contains a set of $k_1 + k_2$ codewords $\{c_1, \ldots, c_{k_1}, c_{k_1+1}, \cdots, c_{k_1+k_2}\}$ such that every codeword in C is uniquely expressible in the form:

$$\sum_{i=1}^{k_1} a_i c_i + \sum_{i=k_1+1}^{k_1+k_2} a_i c_i,$$

where $a_i \in \mathbb{Z}_4$, for $1 \leq i \leq k_1$, and $a_j \in \{0,1\}$, for $k_1 + 1 \leq j \leq k_1 + k_2$. Numbers k_1 and k_2 determine the type of the code C, i.e. C is of type $4^{k_1}2^{k_2}$. The matrix whose rows are $c_i,\ 1 \leq i \leq k_1 + k_2$, is called a generator matrix of the code C.

Example 1.4.3. Let C be a $\mathbb{Z}_4$-code with codewords:

$$1013, 3013, 0220, 2220, 0022, 2022, 1233, 3233,$$
$$0202, 2202, 1031, 3031, 0000, 2000, 1211, 3211.$$

Let $a_1 = 1013$, $a_2 = 0202$, $a_3 = 0022$. It can easily be verified that every codeword of the code C can uniquely be expressed as $c_1a_1 + c_2a_2 + c_3a_3$ for $c_1 \in \mathbb{Z}_4$ and $c_2, c_3 \in \{0,1\}$. Therefore, the code C is of the type $4^1 2^2$, and the generator matrix of the code C is:

$$G = \begin{bmatrix} 1 & 0 & 1 & 3 \\ 0 & 2 & 0 & 2 \\ 0 & 0 & 2 & 2 \end{bmatrix}.$$

Now we define the equivalence of $\mathbb{Z}_4$-codes.

Definition 1.4.4. Let C and C' be $\mathbb{Z}_4$-codes of the same length n. Codes C and C' are equivalent if there exists a monomial[1] $n \times n$ matrix M, with non-zero entries from $\{1,3\}$, such that $C' = \{vM | v \in C\}$. If matrix M is a permutation matrix, then codes C and C' are permutation equivalent.

[1] A monomial matrix is a square matrix with exactly one non-zero element in every row and column.

Remark 1.4.5. From Definition 1.4.4 it follows that equivalent $\mathbb{Z}_4$-codes are of the same type.

Example 1.4.6. Let C and C' be $\mathbb{Z}_4$-codes generated by the matrices G and G':

$$G = \begin{bmatrix} 1 & 0 & 1 & 3 \\ 0 & 2 & 0 & 2 \\ 0 & 0 & 2 & 2 \end{bmatrix}, \quad G' = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 2 & 2 \\ 2 & 0 & 0 & 2 \end{bmatrix}.$$

Notice that G' can be obtained from G by applying the permutation $(1\,3\,2)$ on the columns of the matrix G, and then multiplying the 4th column with 3. In other words for:

$$M = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix},$$

we have $G' = GM$. Therefore, the codes C and C' are equivalent.

Now we define the standard form of a generator matrix for $\mathbb{Z}_4$-codes.

Definition 1.4.7. A generator matrix G of a $\mathbb{Z}_4$-code C of the type $4^{k_1}2^{k_2}$ is in standard form if:

$$G = \begin{bmatrix} I_{k_1} & A & B_1 + 2B_2 \\ O & 2I_{k_2} & 2D \end{bmatrix},$$

where A, B_1, B_2 and D are binary matrices and O is the $k_2 \times k_1$ zero matrix.

Theorem 1.4.8. Every $\mathbb{Z}_4$-code is permutation equivalent to a code that has a generator matrix in standard form.

In the sequel, with $n_i(v)$ we denote the number of coordinates of $v \in \mathbb{Z}_4^n$ equal to i, for $i \in \{0,1,2,3\}$. Now, we introduce the weights for $\mathbb{Z}_4$-codes.

Definition 1.4.9. Let $x \in \mathbb{Z}_4^n$, $n \in \mathbb{N}$. The Hamming weight of x is $wt_H(x) = n_1(x) + n_2(x) + n_3(x)$. The Lee weight of x is $wt_L(x) = n_1(x) + 2n_2(x) + n_3(x)$. The Euclidean weight of x is $wt_E(x) = n_1(x) + 4n_2(x) + n_3(x)$.

Definition 1.4.10. Let C be a $\mathbb{Z}_4$-code. The minimum Hamming weight of the code C is the number:

$$wt_H(C) = \min\{wt_H(v) | v \in C, v \neq 0\}.$$

The minimum Lee weight of the code C is the number:

$$wt_L(C) = \min\{wt_L(v) | v \in C, v \neq 0\}.$$

The minimum Euclidean weight of the code C is the number:

$$wt_E(C) = \min\{wt_E(v) | v \in C, v \neq 0\}.$$

Definition 1.4.11. Let C be a $\mathbb{Z}_4$-code of length n. The Euclidean weight enumerator of the code C is the bivariate polynomial:

$$p(x,y) = \sum_{v \in C} x^{wt_E(v)} y^{4n - wt_E(v)} = \sum_{k \in W} A_k x^k y^{4n-k}, \tag{1.4}$$

where $W = \{wt_E(v) | v \in C\}$, and A_k is the number of codewords from C of Euclidean weight k.

Remark 1.4.12. In this thesis, the Euclidean weight enumerator will only be used to determine the Euclidean weight distribution of $\mathbb{Z}_4$-codes. Therefore, to abbreviate the notation, instead of (1.4), polynomial $p(x,y)$ will be written as univariate polynomial:

$$p(x) = \sum_{k \in W} A_k x^k,$$

with W and A_k as in Definition 1.4.11.

The Euclidean weight distribution of the code is an invariant of equivalent codes. Another invariant of equivalent codes is given in the following definition.

Definition 1.4.13. The symmetric weight enumerator of the $\mathbb{Z}_4$-code C is the polynomial:

$$swe_C(x,y,z) = \sum_{v \in C} x^{n_0(v)} y^{n_1(v)+n_3(v)} z^{n_2(v)}.$$

Example 1.4.14. Let C be the $\mathbb{Z}_4$-code from Example 1.4.3. By the order of the appearance in Example 1.4.3, the Euclidean weights of codewords from C are:

$$3,3,8,12,8,12,7,7,$$

$$8,12,3,3,0,4,7,7.$$

Therefore, $wt_E(C) = 3$. The Euclidean weight distribution of the code C is $p(x) = 1 + 4x^3 + 1x^4 + 4x^7 + 3x^8 + 3x^{12}$. The symmetric weight enumerator of C is $swe_C(x,y,z) = x^4 + x^3z + 3x^2z^2 + 4xy^3 + 3xz^3 + 4y^3z$.

Every $\mathbb{Z}_4$-code C is associated with two binary linear codes of the same length as C, in the following way.

Definition 1.4.15. Let C be a $\mathbb{Z}_4$-code. The residue code and the torsion code of C are binary linear codes defined as:

$$Res(C) = \{c \pmod 2 \mid c \in C\},$$
$$Tor(C) = \{c \in \mathbb{F}_2^n | 2c \in C\}.$$

If a $\mathbb{Z}_4$-code C has the generator matrix G in standard form, then the residue and torsion codes have the following generator matrices:

$$G_{Res} = \begin{bmatrix} I_{k_1} & A & B_1 \end{bmatrix}, \tag{1.5}$$
$$G_{Tor} = \begin{bmatrix} I_{k_1} & A & B_1 \\ O & I_{k_2} & D \end{bmatrix}. \tag{1.6}$$

From generator matrices of $Res(C)$ and $Tor(C)$ it is obvious that $Res(C) \subseteq Tor(C)$.

Definition 1.4.16. For every $x, y \in \mathbb{Z}_4^n$, $x = (x_1, x_2, \ldots, x_n)$, $y = (y_1, y_2, \ldots, y_n)$, the standard inner product is defined as the ordinary dot product of x and y modulo 4, i.e.:

$$x \cdot y = x_1y_1 + x_2y_2 + \ldots + x_ny_n \pmod 4. \tag{1.7}$$

Remark 1.4.17. Notice that addition and multiplication in the right side of (1.7) is usual addition and multiplication over $\mathbb{Z}$.

Definition 1.4.18. The dual code of a $\mathbb{Z}_4$-code C is defined as:

$$C^{\perp} = \left\{x \in \mathbb{Z}_4^n \middle| x \cdot y = 0, \forall y \in C\right\}.$$

A code C is self-orthogonal if $C \subset C^{\perp}$, and self-dual if $C = C^{\perp}$.

The following theorems describe the Euclidean weights of the codewords in self-orthogonal $\mathbb{Z}_4$-codes.

Theorem 1.4.19. If C is a self-orthogonal $\mathbb{Z}_4$-code then $wt_E(v) \equiv 0 \pmod 4$, for every $v \in C$.

Theorem 1.4.20. Let C be a $\mathbb{Z}_4$-code such that $wt_E(v) \equiv 0 \pmod 8$, for every $v \in C$. Then C is self-orthogonal.

The previous two theorems justify the following definition for the self-dual $\mathbb{Z}_4$-codes.

Definition 1.4.21. A self-dual $\mathbb{Z}_4$-code C is a Type II $\mathbb{Z}_4$-code if $wt_E(v) \equiv 0 \pmod 8$, for every $v \in C$. If C is not a Type II $\mathbb{Z}_4$-code, then it is a Type I $\mathbb{Z}_4$-code.

2. AN ALGORITHM FOR THE CONSTRUCTION OF EXTREMAL AND NEAR-EXTREMAL $\mathbb{Z}_4$-CODES

In this chapter, we give an algorithm for the construction of extremal and near-extremal $\mathbb{Z}_4$-codes. It is a modification of an existing algorithm for the search for extremal and near-extremal $\mathbb{Z}_4$-codes described in [22]. This chapter is divided in three sections. The first section introduces extremal and near-extremal $\mathbb{Z}_4$-codes, and their properties. In the second section, we present a motivation for the modification of the existing algorithm. The following section gives the theoretical background for the modification, together with the modified algorithm. At the end of that section, we tested the modified algorithm and compared it to the existing one.

2.1. EXTREMAL AND NEAR-EXTREMAL $\mathbb{Z}_4$-CODES

We ended the previous chapter with the definition of Type I and Type II $\mathbb{Z}_4$-codes. The following theorem, proved in [7, 46], gives an upper bound for the minimum Euclidean weight of a self-dual $\mathbb{Z}_4$-code.

Theorem 2.1.1. Let C be a self-dual $\mathbb{Z}_4$-code of length n. If C is Type II, then the minimum Euclidean weight of C is at most $8\left\lfloor \frac{n}{24} \right\rfloor + 8$. If C is Type I, then the minimum Euclidean weight of C is at most $8\left\lfloor \frac{n}{24} \right\rfloor + 8$ except when $n \equiv 23 (\mathrm{mod}\ 24)$, in which case

the bound is $8\left\lfloor \frac{n}{24} \right\rfloor + 12$. If equality holds in this latter bound, then C is obtained by shortening a Type II code of length $n+1$.

In the following remark, shortening of a $\mathbb{Z}_4$-code is explained.

Remark 2.1.2. The shortening of a $\mathbb{Z}_4$-code C for a given coordinate is done by the following procedure. If all the codewords of C have only 0 or 2 in the given coordinate, then the shortened code of C is obtained from those codewords that have 0 in that coordinate by deleting that coordinate. Otherwise, a shortened code of C is obtained from all codewords of C that have 0 or 2 on the given coordinate by deletion of that coordinate.

Definition 2.1.3. A self-dual $\mathbb{Z}_4$-code C is extremal if its minimum Euclidean weight $wt_E(C)$ meets the bound from Theorem 2.1.1.

In [30], for $k \geq 2$, $\mathbb{Z}_k$ codes (defined as $\mathbb{Z}_k$ submodules of $\mathbb{Z}_k^n$) are studied. The following upper bound for Type I $\mathbb{Z}_4$-codes is deduced from [30, Lemma 2.5, p.7] for the special case of $k = 4$.

Proposition 2.1.4. Let C be a Type I $\mathbb{Z}_4$-code of length n.

(i) If $n = 24$ then $wt_E(C) \leq 12$,

(ii) If $n = 48$ then $wt_E(C) \leq 20$.

Remark 2.1.5. Let C be a self-dual $\mathbb{Z}_4$-code of length n, $n \in \{24, 48\}$. By Theorem 2.1.1, since $n \not\equiv 23 \pmod{24}$, the upper bound on $wt_E(C)$ is either $8\left\lfloor \frac{24}{24} \right\rfloor + 8 = 16$ or $8\left\lfloor \frac{48}{24} \right\rfloor + 8 = 24$. Therefore, an extremal Type I $\mathbb{Z}_4$-code of length 24 should have $wt_E(C) = 16$. Also, an extremal Type I $\mathbb{Z}_4$-code of length 48 should have $wt_E(C) = 24$. By Proposition 2.1.4 we conclude that this is impossible. Therefore, Type I extremal $\mathbb{Z}_4$-codes of lengths 24 and 48 do not exist.

Based on the previous remark we introduce near-extremal $\mathbb{Z}_4$-codes.

Definition 2.1.6. Let C be a self-dual $\mathbb{Z}_4$-code of length n, and let d_E be the upper bound on $wt_E(C)$ from Theorem 2.1.1. The code C is near-extremal $\mathbb{Z}_4$-code if $wt_E(C) = d_E - 4$.

In order to give the construction theorem for self-dual $\mathbb{Z}_4$-codes, an alternative form for the generator matrix of $\mathbb{Z}_4$-code is given in the following theorem from [36].

Theorem 2.1.7. Let C be a self-dual $\mathbb{Z}_4$-code with generator matrix in standard form. Then C has a generator matrix of the form

$$G' = \begin{bmatrix} F & I_k + 2B \\ 2H & O \end{bmatrix}, \tag{2.1}$$

where F, B, H are binary matrices, I_k is $k \times k$ identity matrix and O is zero matrix. If C has a generator matrix of the form (2.1), then the residue and torsion codes have generator matrices of the form:

$$G'_{Res} = \begin{bmatrix} F & I_k \end{bmatrix}, \tag{2.2}$$

$$G'_{Tor} = \begin{bmatrix} F & I_k \\ H & O \end{bmatrix}. \tag{2.3}$$

The following theorem, first proved in [21], gives a characterization of self-dual $\mathbb{Z}_4$-codes.

Theorem 2.1.8. Let C be a $\mathbb{Z}_4$-code with the generator matrix G' of the form (2.1). The code C is self-dual if and only if $Res(C)$ is doubly-even, $Res(C) = Tor(C)^{\perp}$ and B is such that the rows of G' are orthogonal.

Remark 2.1.9. Let C be a self-dual code of length n and type $4^{k_1}2^{k_2}$. By comparison of generator matrices of residue codes given in (1.5) and (2.2), we can conclude that $k = k_1$. Also, since $Tor(C) = (Res(C))^{\perp}$, $\dim(Tor(C)) = n - \dim(Res(C)) = n - k$. From (1.6), we know that $\dim(Tor(C)) = k_1 + k_2 = k + k_2$. Therefore, $k_2 = n - 2k$. Therefore, every self-dual $\mathbb{Z}_4$-code with the residue code of dimension k is of type $4^k 2^{n-2k}$. Since generator matrices of C and $Tor(C)$ are of same dimensions, matrix G' given in (2.1) is a $(n-k) \times n$ matrix.

The following corollary can also be found in [21].

Corollary 2.1.10. Let C be a Type II $\mathbb{Z}_4$-code of length n. Then $Tor(C)$ is an even binary code, $Res(C)$ contains the all-one binary vector, C contains a codeword with all entries from the set $\{1,3\}$, and $n \equiv 0 \pmod 8$.

Remark 2.1.11. Since we are interested in the construction of both Type I and Type II $\mathbb{Z}_4$-codes, we restricted our work to codes of length divisible by 8.

The following remark describes the construction of self-dual $\mathbb{Z}_4$-codes starting from the binary doubly-even code. The discussion in the remark is the summary of various results from [21].

Remark 2.1.12. By Theorem 2.1.8, in order to construct a self-dual $\mathbb{Z}_4$-code C, we have to start with a doubly-even $[n,k]$ binary code $C^{(1)}$ that has a generator matrix of the form (2.2). Since $C^{(1)}$ is self-orthogonal, we can find a generator matrix of its dual code $C^{(2)}$ of the form (2.3). It only remains to determine a binary matrix B such that the rows of the generator matrix (2.1) are orthogonal.

Assume that G is the generator matrix of a self-dual $\mathbb{Z}_4$-code C given in the form (2.1). Let $(f, i+2b)$ be one of the first k rows of G, where f, i and b denote the corresponding rows of matrices F, I_k and B, respectively. Since $C^{(1)} = Res(C)$ is a doubly-even code, the following holds:

$$(f, i+2b) \cdot (f, i+2b) = f \cdot f + i \cdot i + 4b \cdot b (\text{mod } 4) \equiv 0 (\text{mod } 4).$$

Therefore, by Definition 1.4.16, $(f, i+2b) \cdot (f, i+2b) = 0$, i.e. all of the first k rows of G are self-orthogonal.

Further, let $(2h, o)$ and $(2h', o)$ be two (not necessarily different) rows taken from the last $n-2k$ rows of the matrix G, where h, h' and o denotes the corresponding row in matrices H and O respectively. We have that:

$$(2h, o) \cdot (2h', o) = 4hh' (\text{mod } 4) \equiv 0 (\text{mod } 4).$$

Therefore, $(2h, o) \cdot (2h', o) = 0$, i.e. all of the last $n-2k$ rows of G are pairwise orthogonal and are self-orthogonal.

Also, since $C^{(2)} = Tor(C) = (Res(C))^{\perp} = \left(C^{(1)}\right)^{\perp}$, the following holds:

$$(f, i+2b) \cdot (2h, o) = 2fh (\text{mod } 4) \equiv 0 (\text{mod } 4).$$

Therefore, $(f, i+2b) \cdot (2h, o) = 0$. Thus, any of the first k rows of G is orthogonal to any of the last $n-2k$ rows in G.

Therefore, orthogonality of the rows of G entirely depends only on the orthogonality of the different rows among the first k rows in G. Let $B = [b_{u,v}]$, $u,v \in \{1,2,\ldots,k\}$. Let f_m denote the m-th row of F. The inner product of the r-th and the s-th row, $r \neq s$, of the matrix G is given as follows:

$$\begin{aligned} f_r \cdot f_s + \sum_{t=1}^{k} (2b_{rt} + \delta_{rt})(2b_{st} + \delta_{st}) (\mathrm{mod}\ 4) \equiv \\ \equiv f_r \cdot f_s + (2b_{rr}+1)2b_{sr} + 2b_{rs}(2b_{ss}+1) (\mathrm{mod}\ 4) \equiv \\ \equiv f_r \cdot f_s + 2(b_{rs}+b_{sr}) (\mathrm{mod}\ 4), \end{aligned} \tag{2.4}$$

where δ_{uv} denotes the Kronecker delta symbol, and $f_r \cdot f_s$ is the ordinary dot product of f_r and f_s over $\mathbb{Z}$. Since $C^{(1)}$ is self-orthogonal, $f_r \cdot f_s \equiv 0 (\mathrm{mod}\ 2)$. Therefore, $f_r \cdot f_s = 2x$ for some $x \in \mathbb{N}$. So, the (2.4) is congruent to $2x + 2(b_{rs} + b_{sr}) \,(\mathrm{mod}\ 4)$. Now we conclude that the rows of G are mutually orthogonal if and only if the following holds:

$$2x + 2(b_{rs} + b_{sr}) \equiv 0 (\mathrm{mod}\ 4). \tag{2.5}$$

Since $f_r \cdot f_s = 2x$ is congruent to either 0 or 2 modulo 4, one the following holds:

- $2x \equiv 0 (\mathrm{mod}\ 4) \overset{(2.5)}{\Rightarrow} 2(b_{rs} + b_{sr}) \equiv 0 (\mathrm{mod}\ 4) \Rightarrow b_{rs} = b_{sr}$,
- $2x \equiv 2 (\mathrm{mod}\ 4) \overset{(2.5)}{\Rightarrow} 2(b_{rs} + b_{sr}) \equiv 2 (\mathrm{mod}\ 4) \Rightarrow b_{rs} \neq b_{sr}$.

Therefore, it is shown that the lower triangle elements of the matrix B are uniquely determined by the elements in the upper triangle of B by the following condition:

$$b_{rs} = \begin{cases} b_{sr}, f_r \cdot f_s \equiv 0 (\mathrm{mod}\ 4), \\ b_{sr} + 1 (\mathrm{mod}\ 2), f_r \cdot f_s \equiv 2 (\mathrm{mod}\ 4), \end{cases} \tag{2.6}$$

where $f_r \cdot f_s$ is the ordinary dot product of the rows of F. The diagonal elements of B can be chosen freely.

In [21] the following theorem, that gives the number of (not necessary noneqivalent) self-dual $\mathbb{Z}_4$-codes, can be found.

Theorem 2.1.13. Let $N_d(n)$ be the number of distinct self-dual codes over $\mathbb{Z}_4$ of length n, and let $\sigma(n,k)$ be the number of distinct doubly-even binary codes of length n and dimension k. We have:

$$N_d(n) = \sum_{k \leq \frac{n}{2}} \sigma(n,k)\, 2^{\frac{k(k+1)}{2}}.$$

The following necessary condition for the extremality of $\mathbb{Z}_4$-codes can be found in [27].

Lemma 2.1.14. Let C be an extremal Type II $\mathbb{Z}_4$-code of length n. Then the torsion code $Tor(C)$ has minimum weight $d \geq 2\lfloor \frac{n}{24} \rfloor + 2$.

Proof. Let C be an extremal $\mathbb{Z}_4$-code of the length n. From Definition 2.1.3, we know that the minimum Euclidean weight of the code C is $8\left\lfloor \frac{n}{24} \right\rfloor + 8$. Therefore, for every $x \in C$ the following condition holds:

$$wt_E(x) \geq 8\left\lfloor \frac{n}{24} \right\rfloor + 8.$$

By the definition of $Tor(C)$ we have that $2y \in C$ for every $y \in Tor(C)$. Therefore:

$$wt_E(2y) \geq 8\left\lfloor \frac{n}{24} \right\rfloor + 8.$$

Since $y \in \mathbb{F}_2^n$ we have $4wt_H(y) = wt_E(2y)$. Therefore:

$$\begin{aligned} 4wt_H(y) = wt_E(2y) &\geq 8\left\lfloor \frac{n}{24} \right\rfloor + 8, \\ wt_H(y) &\geq 2\left\lfloor \frac{n}{24} \right\rfloor + 2, \end{aligned}$$

which completes the proof. ■

Remark 2.1.15. In the proof of Lemma 2.1.14 only assumption that is used is that the Euclidean extremality upper bound is $8\lfloor n/24 \rfloor + 8$. Therefore, Lemma 2.1.14 is valid even for Type I $\mathbb{Z}_4$-codes that have the same extremality condition as Type II $\mathbb{Z}_4$-codes.

If C is a near-extremal Type I $\mathbb{Z}_4$-code of length 24 then, by Proposition 2.1.4, we know that $wt_E(C) = 12$. In the same way as in the proof of Lemma 2.1.14, for every $y \in Tor(C)$ we have $4wt_H(y) = wt_E(2y) \geq 12$. Therefore, $wt_H(y) \geq 3$. In the same way, and with the same notation, for Type I $\mathbb{Z}_4$-code of length 48 we get $wt_H(y) \geq 5$. Therefore, the minimum weight of torsion code of near-extremal Type I $\mathbb{Z}_4$-code of length 24 or 48 is

at least 3 or 5, respectively. If $Tor(C)$ is even code, then the minimum weight of $Tor(C)$, for Type I $\mathbb{Z}_4$-code C of length 24 or 48, is at least 4 or 6 respectively. This proves the following corollary.

Corollary 2.1.16. If C is Type I near-extremal $\mathbb{Z}_4$-code of length 24 or 48 then the torsion code $Tor(C)$ has the minimum weight at least 3 or 5, respectively. If $Tor(C)$ is even, then its minimum weight is at least 4 for length 24, and at least 6 for length 48.

2.1.1. Known extremal and near-extremal $\mathbb{Z}_4$-codes

In this subsection, we present an overview of the known extremal and near-extremal $\mathbb{Z}_4$-codes. As stated in the Remark 2.1.11 we are interested only in codes of length divisible by 8.

All $\mathbb{Z}_4$-codes of lengths up to 9 were classified by J. H. Conway and N. J. A. Sloane in [13]. Self-dual $\mathbb{Z}_4$-codes of lengths 10 up to 15 are classified in [20] by J. Fields, P. Gaborit, J. S. Leon and V. Pless.

Also, all Type II $\mathbb{Z}_4$-codes of length 16 were classified by V. Pless, J. S. Leon and J. Fields in [43].

The classification of Type I $\mathbb{Z}_4$-codes of length 16 and self-dual $\mathbb{Z}_4$-codes of lengths 17 up to 19 is given by M. Harada and A. Munemasa in [35].

More recently, in 2019, the self-dual $\mathbb{Z}_4$-codes of length 20 were classified by R. A. L. Betty and A. Munemasa in [5]. All of the residue codes of extremal Type II $\mathbb{Z}_4$-codes of length 24 were classified by M. Harada, C. H. Lam and A. Munemasa in [34]. In [41], all extremal Type II $\mathbb{Z}_4$-codes of length 24 were classified by R. A. L. Betty and A. Munemasa. As it was stated in Remark 2.1.5, Type I extremal $\mathbb{Z}_4$-codes of length 24 do not exist. The existence of two near-extremal Type I $\mathbb{Z}_4$-codes of length 24 was proved theoretically by T. A. Gulliver and M. Harada in [23]. Later, in [31], M. Harada constructed 58 nonequivalent near-extremal Type I $\mathbb{Z}_4$-codes of that length that are not the codes from [23]. He concluded that there are at least 60 near-extremal $\mathbb{Z}_4$-codes of length 24.

We are interested only in codes of length larger than 24. The largest known extremal $\mathbb{Z}_4$-code is of length 64 and was constructed by M. Harada in [27]. Since Type I $\mathbb{Z}_4$-codes

are far less researched, we will fist give an overview of the known Type II $\mathbb{Z}_4$-codes for each length. After that, we will give the summary of known facts on Type I $\mathbb{Z}_4$-codes.

Length 32

The following statements can be found in [28].

Proposition 2.1.17. There is an extremal Type II $\mathbb{Z}_4$-code of length 32 whose residue code has dimension k if and only if $k \in \{6,7,\ldots,16\}$.

Proposition 2.1.18. Every binary doubly-even self-dual code of length 32 can be realized as the residue code of some extremal Type II $\mathbb{Z}_4$-code.

Proposition 2.1.19. Up to equivalence, there is a unique extremal Type II $\mathbb{Z}_4$-code of length 32 with residue code of dimension 6.

Remark 2.1.20. In [28] is also proved that the binary residue code from the previous proposition is $\mathrm{RM}(1,5)$.

In [22], 54 inequivalent extremal Type II $\mathbb{Z}_4$-codes with self-dual binary residue codes were constructed (therefore of type 4^{16}).

In [28], one Type II extremal $\mathbb{Z}_4$-code with dimension of the residue code k was constructed for every $k \in \{6,7,\ldots,15\}$. In the same article, 80 nonequivalent (and nonequivalent to codes from [22]) extremal Type II $\mathbb{Z}_4$-codes with residue code of dimension 16 were constructed. All of the binary residue codes of extremal $\mathbb{Z}_4$-codes of length 32 constructed in [28] have minimum weight 4.

In [44] two nonequivalent Type II extremal $\mathbb{Z}_4$-codes of type $4^{11}2^{10}$ are constructed. These codes have binary residue code of minimum weight 12. In [12], the following extremal Type II $\mathbb{Z}_4$-codes were constructed: 219 codes of type $4^{13}2^6$, 205 codes of type $4^{14}2^4$ and 355 codes of type $4^{15}2^2$.

Finally, in [1], the following previously unknown extremal Type II $\mathbb{Z}_4$-codes were constructed: 5 codes of type $4^7 2^{16}$, 25 codes of type $4^8 2^{16}$, 12 codes of type $4^9 2^{14}$, 3 codes of type $4^{10}2^{12}$ and 4942 codes of type $4^{14}2^4$.

Based on the previous discussion, we conclude that there are 5913 known nonequivalent extremal Type II $\mathbb{Z}_4$-codes. The known results are summarized in Table 2.1.

Type	$4^6 2^{20}$	$4^7 2^{18}$	$4^8 2^{16}$	$4^9 2^{14}$	$4^{10} 2^{12}$	$4^{11} 2^{10}$
#	1	6	27	13	4	3
Type	$4^{12} 2^8$	$4^{13} 2^6$	$4^{14} 2^4$	$4^{15} 2^2$	4^{16}	
#	1	220	5148	356	134	

Table 2.1: The type and the number of known extremal Type II $\mathbb{Z}_4$-codes of length 32

Length 40

In [44], 22 extremal $\mathbb{Z}_4$-codes were constructed from self-dual binary codes together with one code that have a binary residue code of dimension 13.

Later, in [4], the complete classification of the self-dual binary codes of length 40 is given. In total, there are 94343 nonequivalent binary self-dual codes of length 40. In [32], the method that gives an extremal $\mathbb{Z}_4$-code from a self-dual binary code of length 40 was developed. With this method, from each of the 94343 binary self-dual codes of length 40, one extremal Type II $\mathbb{Z}_4$-code is constructed (of which 22 were equivalent to codes given in [44]).

Also, in [28], two codes of type $4^7 2^{26}$ were constructed together with one code of type $4^k 2^{40-k}$, for every $k \in \{8, 9, \ldots, 19\}$ (the code of type $4^{13} 2^{14}$ constructed here was nonequivalent to the code from [44]).

In [12], the following new extremal $\mathbb{Z}_4$-codes were constructed: 133 codes of type $4^{17} 2^6$, 501 code of type $4^{18} 2^4$ and 431 code of type $4^{19} 2^2$.

Finaly, in [3], the following new extremal $\mathbb{Z}_4$-codes of Type II were constructed: one code of type $4^7 2^{26}$, 227 codes of type $4^8 2^{24}$, 99 codes of type $4^9 2^{22}$, one code of type $4^{10} 2^{20}$, 2 codes of type $4^{11} 2^{18}$, 19 codes of type $4^{12} 2^{16}$, 13 codes of type $4^{13} 2^{14}$, 4 codes of type $4^{14} 2^{12}$, 2 codes of type $4^{15} 2^4$ and 400 codes of type $4^{18} 2^4$. The summary of known results is given in Table 2.2.

Type	$4^7 2^{26}$	$4^8 2^{24}$	$4^9 2^{22}$	$4^{10} 2^{20}$	$4^{11} 2^{18}$	$4^{12} 2^{16}$	$4^{13} 2^{14}$
#	3	228	100	2	3	20	15
Type	$4^{14} 2^{12}$	$4^{15} 2^{10}$	$4^{16} 2^8$	$4^{17} 2^6$	$4^{18} 2^4$	$4^{19} 2^2$	4^{20}
#	5	3	1	134	902	432	94343

Table 2.2: The type and the number of known extremal Type II $\mathbb{Z}_4$-codes of length 40

Lengths 48, 56, 64

The number of known extremal $\mathbb{Z}_4$-codes of lengths 48, 56 and 64 is small. There are only two known extremal Type II $\mathbb{Z}_4$-codes of length 48, and they are given in [7] and [33]. Both of these codes have self-dual residue codes, so they are of type 4^{24}.

There are three known extremal Type II $\mathbb{Z}_4$-codes of length 56. The extremal Type II $\mathbb{Z}_4$-code of type $4^{14}2^{28}$ is given in [27], and two codes of type 4^{28} in [31].

There is only one known extremal Type II $\mathbb{Z}_4$-code of length 64. This code is constructed in [27] and it is of type $4^{16}2^{32}$.

Type I $\mathbb{Z}_4$-codes

Type I $\mathbb{Z}_4$-codes are far less researched than Type II $\mathbb{Z}_4$-codes. Most of the existing results give theoretical proof of the existence of such codes, for various lengths.

In [29], an existence of extremal Type I $\mathbb{Z}_4$-codes of length 32 is proved for codes with self-dual doubly-even residue codes, without the explicit construction of the codes. To best of our knowledge, there are no classification or enumeration results for extremal Type I $\mathbb{Z}_4$-codes of length 32.

In [9], 16470 nonequivalent extremal Type I $\mathbb{Z}_4$-codes of length 40 and type 4^{20} were constructed. All of these codes have different residue codes that are self-dual binary codes of length 40 and the minimum weight 8 (the classification of binary doubly-even self-dual codes is given in [4]). Also, 16 nonequivalent extremal Type I $\mathbb{Z}_4$-codes were obtained by a random search method. The binary residue codes of these 16 codes have the following parameters: one code is $[40, 10, 16]$, one code is $[40, 10, 12]$, 7 codes are $[40, 8, 16]$, and

finally 7 codes are $[40,8,12]$. Therefore, the constructed 16 extremal Type I $\mathbb{Z}_4$-codes are of types $4^{10}2^{20}$ (2 codes) and $4^8 2^{24}$ (14 codes).

As stated in Remark 2.1.5, extremal Type I $\mathbb{Z}_4$-codes of length 48 do not exist. Therefore, for the length 48, it makes sense to search for near-extremal Type I $\mathbb{Z}_4$-codes. In [30], the existence of Type I near-extremal $\mathbb{Z}_4$-code of length 48 is proved by example. The following generator matrix of Type I near-extremal $\mathbb{Z}_4$-code is given:

$$\begin{bmatrix} & 0 & 1 & \cdots & 1 \\ I_{24} & 1 & & & \\ & \vdots & & R & \\ & 1 & & & \end{bmatrix},$$

where R is circulant 23×23 matrix whith first row:

$$(1,1,3,0,3,3,1,2,0,1,3,2,3,0,0,3,3,2,1,2,1,1,0).$$

This code is of type 4^{24}.

2.2. MOTIVATION FOR THE MODIFICATION

At the beginning of this chapter we will describe the usual approach to the construction of extremal and near-extremal $\mathbb{Z}_4$-codes described in [22].

Let $C^{(1)}$ be a doubly-even binary $[n,k]$ code, and let $C^{(2)} = \left(C^{(1)}\right)^{\perp}$. Assume that the generator matrix of $C^{(2)}$ is in the form (2.3). As stated in Remark 2.1.12, in order to obtain a generator matrix G of a self-dual $\mathbb{Z}_4$-code C of the form (2.1), it is sufficient to construct a $k \times k$ binary matrix B such that the condition (2.6) holds. In the further discussion, we will write $G(B)$ instead of G for a generator matrix of a self-dual $\mathbb{Z}_4$-code, when we want to emphasize the related matrix B. From Theorem 2.1.13 it follows that the number of such matrices B, for a given binary linear code $C^{(1)}$, is $2^{\frac{k(k+1)}{2}}$. So, this is also a number of different (not necessary nonequivalent) self-dual $\mathbb{Z}_4$-codes with the residue code $C^{(1)}$.

The following observation is obvious. Let $B = [b_{i,j}]$ and $B' = [b'_{i,j}]$ be two $k \times k$ matrices that give the generator matrices $G(B)$ and $G(B')$ such that B and B' differ only in diagonal elements in columns i_1, i_2, ..., i_m, $m \leq k$. Therefore, matrices $G(B)$ and $G(B')$ differ only in the diagonal elements of the blocks $I_k + 2B$ and $I_k + 2B'$. This means that the matrix $G(B')$ can be obtained from the matrix $G(B)$ by multiplying the columns $n-k+i_1$, $n-k+i_2$, ..., $n-k+i_m$ of $G(B)$ with 3. Therefore, the matrices $G(B)$ and $G(B')$ define monomially equivalent self-dual $\mathbb{Z}_4$-codes. By this simple observation, we conclude that we can always choose matrices B that have all diagonal elements equal to 0. Therefore, the number of matrices B that generate self-dual $\mathbb{Z}_4$-code with fixed binary residue code $C^{(1)}$ is equal to the number of choices for the elements above the diagonal of B. Since B is a binary $k \times k$ matrix, this number is $2^{\frac{k(k-1)}{2}}$.

The usual search method for extremal or near-extremal self-dual $\mathbb{Z}_4$-codes would consist of (randomly) generating a suitable matrix B, and then checking the minimum Euclidean weight of the code C with generator matrix $G(B)$. Computation of the minimum Euclidean weight of the code is usually done by the software package MAGMA ([8]). In Table 2.3, we give time of computation of the minimum Euclidean weight for codes of various length and dimension. This computation was done on the machine with an Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz processor, and 16GB RAM memory of frequency 2400MHz.

n	k	Time
16	6	< 0.001s
24	6	0.734s
32	6	170.830s
40	7	51206.48s

Table 2.3: Time of computation of minimum Euclidean weight of the self-dual $\mathbb{Z}_4$-codes with $[n,k]$ residue code executed with MAGMA

It can be seen from Table 2.3 that the execution time increases rapidly with the increase of the length of the code. The size of the search space for the codes that have the residue code of the dimension 6 is $2^{15} = 32768$. Therefore, all self-dual $\mathbb{Z}_4$-codes of length 16 can be checked for minimum weight in $32,768$ seconds. For codes of length 24 this execution time would be $24051,712$ seconds, which is approximately 6.6 hours. For a binary code of length 32 and dimension 6 this time would be $5597757,44$ seconds, which is approximately $10,6$ years. For codes of higher dimension and length this numbers would get even larger. From this discussion it is clear that complete classification of the extremal and near-extremal $\mathbb{Z}_4$-codes is out of reach for larger lengths.

The main motivation for our modification is to speed up the process of finding extremal and near-extremal $\mathbb{Z}_4$-codes by reducing the number of long lasting calculations that are executed, i.e., by reducing the number of minimum Euclidean weights that are calculated. We developed a method in which we can predict the minimum Euclidean weights of the codes that slightly differ from the code whose Euclidean weight distribution is known.

2.3. THEORETICAL BACKGROUND AND THE MODIFIED ALGORITHM

Let C be a self-dual $\mathbb{Z}_4$-code with the generator matrix $G(B)$ in form (2.1), i.e.:

$$G(B) = \begin{bmatrix} F & I_k + 2B \\ 2H & O \end{bmatrix}, \tag{2.7}$$

where $B = [b_{s,t}]$ is a $k \times k$ binary matrix for which self-duality condition (2.6) holds. Let (i, j), $1 \leq i < j \leq k$, be an upper diagonal position of B such that $b_{ij} = 0$. Let $B' = [b'_{s,t}]$ be a $k \times k$ binary matrix such that:

$$b'_{st} = \begin{cases} b_{st}, & (s,t) \neq (i,j) \\ 1, & (s,t) = (i,j), \end{cases} \tag{2.8}$$

for all $1 \leq s < t \leq k$, and such that the $\mathbb{Z}_4$-code C' is generated by $G(B')$, i.e.:

$$G\left(B'\right) = \begin{bmatrix} F & I_k + 2B' \\ 2H & O \end{bmatrix}, \tag{2.9}$$

is self-dual (condition (2.6) holds for B'). We say that B' is the (i, j)-neighbor of B, and that the code C' is the (i, j)-neighbor of the code C.

In the Definition 1.4.13 we introduced the symmetric weight enumerator of a $\mathbb{Z}_4$-code. Recall that for $v \in \mathbb{Z}_4^n$, $n_i(v)$ is the number of coordinates of v equal to i, $i \in \{0, 1, 2, 3\}$. For the purpose of the following proposition we introduce the following notation. With $swe(v)$ we denote the monomial $x^{n_0(v)} y^{n_1(v)+n_3(v)} z^{n_2(v)}$.

The following two theorems are the core of our method.

Theorem 2.3.1. Let C be a self-dual $\mathbb{Z}_4$-code of length n with the generator matrix $G(B)$ of the form (2.7). Let C' be the (i, j)-neighbor of C, with the generator matrix $G(B')$ of the form (2.9). Let $v \in C$ be of the form:

$$v = c_i g_i + c_j g_j + \sum_{\substack{m=1 \\ m \neq i,j}}^{k} c_m g_m + \sum_{m=k+1}^{n-k} c_m g_m, \tag{2.10}$$

where g_s, $s \in \{1,2,\ldots,n-k\}$, is the s-th row of the matrix $G(B)$. Let $v' \in C'$ be:

$$v' = c_i g'_i + c_j g'_j + \sum_{\substack{m=1 \\ m\neq i,j}}^{k} c_m g_m + \sum_{m=k+1}^{n-k} c_m g_m. \tag{2.11}$$

If c_i and c_j are both even or both odd then $swe(v') = swe(v)$.

Proof. First notice that, since C' is (i,j)-neighbor of C, the block matrices $I_k + 2B$ and $I_k + 2B'$ differ only in the positions (i,j) and (j,i). Therefore, the coordinates in which v and v' can differ are $n-k+j$ and $n-k+i$.

If $c_i = c_j = 0$ then $v = v'$ and therefore $swe(v') = swe(v)$. Further, if $c_i = c_j = 2$ then $(2g_i)_{n-k+j} = (2g'_i)_{n-k+j} = 0$, and $(2g_j)_{n-k+i} = (2g'_j)_{n-k+i} = 0$. Since the generators g_i and g'_i, and g_j and g'_j are only generators that differ in (2.10) and (2.11), and they only differ in coordinates $n-k+j$ and $n-k+i$, the coordinates $n-k+j$ and $n-k+i$ of codewords v and v' are the same. Thus, $v = v'$, and therefore $swe(v') = swe(v)$. Assume that $c_i, c_j \in \{1,3\}$. The following holds:

$$(v)_{n-k+i} = \underbrace{(c_i g_i)_{n-k+i}}_{\in\{1,3\}} + \underbrace{(c_j g_j)_{n-k+i}}_{\in\{0,2\}} + \underbrace{\left(\sum_{\substack{m=1 \\ m\neq i,j}}^{k} c_m g_m\right)_{n-k+i}}_{\in\{0,2\}} \in \{1,3\},$$

$$(v')_{n-k+i} = \underbrace{(c_i g'_i)_{n-k+i}}_{\in\{1,3\}} + \underbrace{(c_j g'_j)_{n-k+i}}_{\in\{0,2\}} + \underbrace{\left(\sum_{\substack{m=1 \\ m\neq i,j}}^{k} c_m g_m\right)_{n-k+i}}_{\in\{0,2\}} \in \{1,3\}.$$

Therefore, the $(n-k+i)$-th coordinate of both v and v' is odd. In a similar way we can conclude that the $(n-k+j)$-th coordinates of v and v' is also odd. Thus, $n_2(v) = n_2(v')$, and $n_1(v) + n_3(v) = n_1(v') + n_3(v')$. Therefore, $swe(v') = swe(v)$.

Now, we observe the case when $c_i \neq c_j$, and $c_i, c_j \in \{0,2\}$. First, let $c_i = 0$ and $c_j = 2$. Since $c_i = 0$, the $(n-k+j)$-th coordinates in v and v' are the same. Further, since $c_j = 2$, we have that $(2g_j)_{n-k+i} = (2g'_j)_{n-k+i} = 0$. Therefore, $swe(v') = swe(v)$. If $c_i = 2$ and $c_j = 0$, the $(n-k+i)$-th coordinates in v and v' are the same. Then, from $(2g_j)_{n-k+j} = (2g'_j)_{n-k+j} = 0$ we get that $swe(v') = swe(v)$. This completes the proof.

■

Theorem 2.3.2. Let C, C', $G(B)$, $G(B')$, v and v' be as in Theorem 2.3.1. Let c_i and c_j be of different parity, and $s \in \{i,j\}$ such that c_s is even. Let $\sigma\left(c_i,c_j\right)$ be defined as follows:

$$\sigma\left(c_i,c_j\right) = \begin{cases} -1, & 2 \in \left\{c_i,c_j\right\}, \\ 1, & 2 \notin \left\{c_i,c_j\right\}. \end{cases}$$

Let S be the following set:

$$S = \left\{t \middle| t \in \{1,2,\ldots,k\} \setminus \{i,j\}, (c_t g_t)_{n-k+s} = 2\right\}.$$

Then, $swe(v') = swe(v) \cdot z^r$, where r is defined as follows:

$$r = \begin{cases} (-1)^{|S|} \cdot \sigma\left(c_i,c_j\right), & b_{ij} \neq b_{ji} \text{ and } s = j, \\ (-1)^{|S|+b_{ij}+b_{ji}} \cdot \sigma\left(c_i,c_j\right), & \text{otherwise.} \end{cases} \tag{2.12}$$

Proof. Let us assume that $b_{ij} = b_{ji}$. Since B is a binary matrix, we have to prove that the value of r for this case is:

$$r = (-1)^{|S|} \cdot \sigma\left(c_i,c_j\right).$$

We will observe only the case when $s = i$, since the case when $s = j$ can be proved in the same way. In that case, since $c_i \in \{0,2\}$, the $(c_i g_i)_{n-k+j} = (c_i g_i')_{n-k+j} = 0$. Therefore, v and v' can only differ in the $(n-k+i)$-th coordinate. We have two possibilities for c_i:

- If $c_i = 0$ and $c_j \in \{1,3\}$. It holds $\sigma\left(c_i,c_j\right) = 1$. Let $g_{i_1}, g_{i_2}, \ldots, g_{i_\alpha}$ be the generators of v s.t. $(c_{i_t} g_{i_t})_{n-k+i} \neq 0$, for all $t \in \{1,2,\ldots,\alpha\}$, $i_t \neq j$. This is equivalent to the condition $(c_{i_t} g_{i_t})_{n-k+i} = 2$, so $|S| = \alpha$. We have:

$$(v)_{n-k+i} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+i} + \underbrace{(c_j g_j)_{n-k+i}}_{=0} = \begin{cases} 0, & 2|\alpha, \\ 2, & 2\not|\,\alpha. \end{cases}$$

$$\left(v'\right)_{n-k+i} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+i} + \underbrace{(c_j g_j')_{n-k+i}}_{=2} = \begin{cases} 2, & 2|\alpha, \\ 0, & 2\not|\,\alpha. \end{cases}$$

  Therefore:

$$swe(v') = \begin{cases} swe(v) \cdot z, & 2|\alpha, \\ swe(v) \cdot z^{-1}, & 2\not|\,\alpha. \end{cases} = swe(v) \cdot z^{(-1)^\alpha}$$

- If $c_i = 2$ and $c_j \in \{1,3\}$, then $\sigma\left(c_i, c_j\right) = -1$. For $g_{i_1}, g_{i_2}, \ldots, g_{i_\alpha}$ as in previous case, we have:

$$(v)_{n-k+i} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+i} + \underbrace{(2g_i)_{n-k+i}}_{=2} + \underbrace{(c_j g_j)_{n-k+i}}_{=0} = \begin{cases} 2, & 2|\alpha, \\ 0, & 2\nmid\alpha. \end{cases}$$

$$\left(v'\right)_{n-k+i} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+i} + \underbrace{(2g'_i)_{n-k+i}}_{=2} + \underbrace{(c_j g'_j)_{n-k+i}}_{=2} = \begin{cases} 0, & 2|\alpha, \\ 2, & 2\nmid\alpha. \end{cases}$$

  So, we can conclude that:

$$swe(v') = \begin{cases} swe(v)\cdot z^{-1}, & 2|\alpha, \\ swe(v)\cdot z, & 2\nmid\alpha. \end{cases} = swe(v)\cdot z^{(-1)^{\alpha+1}}.$$

Therefore, for $s = i$, it holds $r = (-1)^{|S|}\cdot\sigma\left(c_i, c_j\right)$.

Let $b_{ij} \neq b_{ji}$. First we have the case when $s = i$. In that case, we have to prove that $r = (-1)^{|S|+1}\cdot\sigma\left(c_i, c_j\right)$, since $b_{ij} + b_{ji} = 1$. As before, codewords v and v' can differ only in coordinate $n-k+i$. We have the following possibilities:

- For $c_i = 0$, and $c_j \in \{1,3\}$, it holds $\sigma\left(c_i, c_j\right) = 1$. Let $g_{i_1}, g_{i_2}, \ldots, g_{i_\alpha}$ be the generators of v s.t. $(c_{i_t} g_{i_t})_{n-k+i} = 2$, for all $t \in \{1,2,\ldots,\alpha\}$, $i_t \neq j$. This gives $|S| = \alpha$. Then, we have:

$$(v)_{n-k+i} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+i} + \underbrace{(c_j g_j)_{n-k+i}}_{=2} = \begin{cases} 2, & 2|\alpha, \\ 0, & 2\nmid\alpha. \end{cases}$$

$$\left(v'\right)_{n-k+i} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+i} + \underbrace{(c_j g'_j)_{n-k+i}}_{=0} = \begin{cases} 0, & 2|\alpha, \\ 2, & 2\nmid\alpha. \end{cases}$$

  Therefore:

$$swe(v') = \begin{cases} swe(v)\cdot z^{-1}, & 2|\alpha, \\ swe(v)\cdot z, & 2\nmid\alpha. \end{cases} = swe(v)\cdot z^{(-1)^{\alpha+1}}.$$

- For $c_i = 2$ and $c_j \in \{1,3\}$ we have $\sigma\left(c_i, c_j\right) = -1$. Again, let $g_{i_1}, g_{i_2}, \ldots, g_{i_\alpha}$ be the generators of v s.t. $(c_{i_t} g_{i_t})_{n-k+i} = 2$, for all $t \in \{1,2,\ldots,s\}$, $i_t \neq j$. As before, the $(n-k+i)$-th coordinate of v and v' is given by:

$$(v)_{n-k+i} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+i} + \underbrace{(2g_i)_{n-k+i}}_{=2} + \underbrace{(c_j g_j)_{n-k+i}}_{=2} = \begin{cases} 0, & 2|\alpha, \\ 2, & 2\nmid\alpha. \end{cases}$$

$$\left(v'\right)_{n-k+i} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+i} + \underbrace{(2g'_i)_{n-k+i}}_{=2} + \underbrace{(c_j g'_j)_{n-k+i}}_{=0} = \begin{cases} 2, & 2|\alpha, \\ 0, & 2\nmid\alpha. \end{cases}$$

Therefore:

$$swe(v') = \begin{cases} swe(v)\cdot z, & 2|\alpha, \\ swe(v)\cdot z^{-1}, & 2\nmid\alpha. \end{cases} = swe(v)\cdot z^{(-1)^\alpha}.$$

Therefore, $r = (-1)^{|S|+1} \cdot \sigma\left(c_i, c_j\right)$.

The remaining case is when $s = j$. In that case, we should prove that $r = (-1)^{|S|} \cdot \sigma\left(c_i, c_j\right)$. As before, codewords v and v' can differ only in coordinate $n-k+j$. We have the following possibilities:

- Let $c_j = 0$ and $c_i \in \{1,3\}$. Thus, $\sigma\left(c_i, c_j\right) = 1$. Let $g_{i_1}, g_{i_2}, \ldots, g_{i_\alpha}$ be the generators of v s.t. $(c_{i_t} g_{i_t})_{n-k+j} = 2$, for all $t \in \{1,2,\ldots,s\}$, $i_t \neq i$. The following holds:

$$(v)_{n-k+j} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+j} + \underbrace{(c_i g_i)_{n-k+j}}_{=0} = \begin{cases} 0, & 2|\alpha, \\ 2, & 2\nmid\alpha. \end{cases}$$

$$\left(v'\right)_{n-k+j} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+j} + \underbrace{(c_i g'_i)_{n-k+j}}_{=2} = \begin{cases} 2, & 2|\alpha, \\ 0, & 2\nmid\alpha. \end{cases}$$

$$swe(v') = \begin{cases} swe(v)\cdot z, & 2|\alpha, \\ swe(v)\cdot z^{-1}, & 2\nmid\alpha. \end{cases} = swe(v)\cdot z^{(-1)^\alpha}.$$

- Let $c_j = 2$ and $c_i \in \{1,3\}$. Therefore, $\sigma\left(c_i, c_j\right) = -1$. With same notation of generators, we have:

$$(v)_{n-k+j} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+j} + \underbrace{(2g_j)_{n-k+j}}_{=2} + \underbrace{(c_i g_i)_{n-k+j}}_{=0} = \begin{cases} 2, & 2|\alpha, \\ 0, & 2\nmid\alpha. \end{cases}$$

$$\left(v'\right)_{n-k+j} = \left(\sum_{t=1}^{\alpha} c_{i_t} g_{i_t}\right)_{n-k+j} + \underbrace{(2g'_j)_{n-k+j}}_{=2} + \underbrace{(c_i g'_i)_{n-k+j}}_{=2} = \begin{cases} 0, & 2|\alpha, \\ 2, & 2\nmid\alpha. \end{cases}$$

And this gives the following:

$$swe(v') = \begin{cases} swe(v)\cdot z^{-1}, & 2|\alpha, \\ swe(v)\cdot z, & 2\nmid\alpha. \end{cases} = swe(v)\cdot z^{(-1)^{\alpha+1}}.$$

Therefore, it holds $r = (-1)^{|S|}\cdot\sigma\left(c_i, c_j\right)$.

This completes the proof. ■

Remark 2.3.3. The value of the parameter r given by (2.12) is represented in Table 2.4 for all possible coefficients c_i and c_j. In that table, I and J represent set S from Theorem 2.3.2, for $s = i$ and $s = j$, respectively. The parity of the size of sets marked by $*$ does not impact the value of r. By Theorem 2.3.1, for all coefficients that are not in Table 2.4, holds $r = 0$.

$b_{i,j} = b_{j,i}$				
c_i	c_j	$\lvert I\rvert \pmod 2$	$\lvert J\rvert \pmod 2$	r
0	1,3	0	$*$	1
		1	$*$	-1
1,3	0	$*$	0	1
		$*$	1	-1
2	1,3	0	$*$	-1
		1	$*$	1
1,3	2	$*$	0	-1
		$*$	1	1

$b_{i,j} \neq b_{j,i}$				
c_i	c_j	$\lvert I\rvert \pmod 2$	$\lvert J\rvert \pmod 2$	r
0	1,3	0	$*$	-1
		1	$*$	1
1,3	0	$*$	0	1
		$*$	1	-1
2	1,3	0	$*$	1
		1	$*$	-1
1,3	2	$*$	0	-1
		$*$	1	1

Table 2.4: The parameter r from Remark 2.3.3

Corollary 2.3.4. With the notation as in Theorems 2.3.1 and 2.3.2, the following holds:

(i) $wt_H(v') = wt_H(v) + r$,

(ii) $wt_L(v') = wt_L(v) + 2r$,

(iii) $wt_E(v') = wt_E(v) + 4r$.

Proof. From the Definition 1.4.9 we have that the Hamming, Lee and Euclidean weight of v are:

$$wt_H(v) = n_1(v) + n_2(v) + n_3(v)$$
$$wt_L(v) = n_1(v) + 2n_2(v) + n_3(v)$$
$$wt_E(v) = n_1(v) + 4n_2(v) + n_3(v).$$

From Theorem 2.3.1 and Theorem 2.3.2, we have that $swe(v') = x^{n_0(v)} y^{n_1(v)+n_3(v)} z^{n_2(v)+r}$, where r is given in Table 2.4. Therefore, the weights of v' are:

$$wt_H(v') = n_1(v) + n_2(v) + r + n_3(v) = wt_H(v) + r$$
$$wt_L(v') = n_1(v) + 2(n_2(v) + r) + n_3(v) = wt_L(v) + 2r$$
$$wt_E(v') = n_1(v) + 4(n_2(v) + r) + n_3(v) = wt_E(v) + 4r.$$ ■

Corollary 2.3.5. Let C and C' be the self-dual $\mathbb{Z}_4$-codes such that C' is the (i,j)-neighbor of the code C. Let $v \in C$ and $v' \in C'$ be codewords as in Theorem 2.3.1. Then $wt_E(v') \in \{wt_E(v)-4, wt_E(v), wt_E(v)+4\}$.

Proof. The statement follows immediately from Corollary 2.3.4 from the fact that $r \in \{-1,0,1\}$. ■

In the following theorem, we give an important connection between Euclidean weight enumerators of self-dual $\mathbb{Z}_4$-code C and its (r,s)-neighbor.

Theorem 2.3.6. Let C be a self-dual $\mathbb{Z}_4$-code of length n, and C' its (r,s)-neighbor. Let $p(x) = \sum_{i=0}^{n} A_{4i}x^{4i}$ and $p'(x) = \sum_{i=0}^{n} A'_{4i}x^{4i}$ be the Euclidean weight enumerators of C and C', respectively. For $m \in \{0,4,\ldots,4n\}$, let S_m and S'_m denote the sets of codewords of Euclidean weight m from C and C', respectively. For $i = 1,2,\ldots,n$, we define the following sets and their cardinality:

$$\begin{aligned} S_{4i}^{-} &= \left\{v \in S_{4i} | v' \in S'_{4i-4}\right\}, & |S_{4i}^{-}| &= A_{4i}^{-}, \\ S_{4i}^{0} &= \left\{v \in S_{4i} | v' \in S'_{4i}\right\}, & |S_{4i}^{0}| &= A_{4i}^{0}, \\ S_{4i}^{+} &= \left\{v \in S_{4i} | v' \in S'_{4i+4}\right\}, & |S_{4i}^{+}| &= A_{4i}^{+}, \end{aligned}$$

where for $v \in C$, v' is the codeword from C' with the same coefficients as v. Then the following holds:

(i) $A'_{4i} = A^{+}_{4i-4} + A^{0}_{4i} + A^{-}_{4i+4}$, for $i > 1$, and $A'_4 = A^0_4 + A^-_8$

(ii) $A'_4 + A'_8 + \ldots + A'_{N-4} = A_4 + A_8 + \ldots + A_{N-4} - A^{+}_{N-4} + A^{-}_{N}$,

where $N \in \{8,12,\ldots,4n\}$.

Proof. Notice that $|S_m| = A_m$, and $|S'_m| = A'_m$. From Corollary 2.3.5, we know that the Euclidean weight of $v' \in C'$ belongs to the set $\{wt_E(v)-4, wt_E(v), wt_E(v)+4\}$. Therefore, for $v' \in S'_{4i}$, $wt_E(v) \in \{4i-4, 4i, 4i+4\}$. Depending on the value of the $wt_E(v)$, we have the following:

$$\begin{aligned} wt_E(v) = 4i-4 &\Rightarrow v \in S^{+}_{4i-4}, \\ wt_E(v) = 4i &\Rightarrow v \in S^{0}_{4i}, \\ wt_E(v) = 4i+4 &\Rightarrow v \in S^{-}_{4i+4}. \end{aligned}$$

Therefore, $|S^+_{4i-4} \cup S^0_{4i} \cup S^-_{4i+4}| = |S'_{4i}|$, which gives the (i). The special case of A'_4 follows from the fact that $A^+_0 = 0$. This is valid since the zero-codeword is uniquely expressed as $\sum_{s=1}^{n-k} 0 \cdot g_s$, where g_s are rows of the generator matrix of C, and $k = \dim(Res(C))$. Therefore, by Table 2.4, it never changes weight.

Now we can use statement (i) to prove (ii). From (i) we have:

$$\begin{aligned}
A'_4 &= A^0_4 + A^-_8, \\
A'_8 &= A^+_4 + A^0_8 + A^-_{12}, \\
A'_{12} &= A^+_8 + A^0_{12} + A^-_{16}, \\
&\vdots \qquad\qquad \vdots \\
A'_{N-8} &= A^+_{N-12} + A^0_{N-8} + A^-_{N-4}, \\
A'_{N-4} &= A^+_{N-8} + A^0_{N-4} + A^-_N.
\end{aligned}$$

Also, since $A^+_0 = 0$ and $A_{4i} = A^-_{4i} + A^0_{4i} + A^+_{4i}$, the following holds:

$$\begin{aligned}
A'_4 + A'_8 + \ldots + A'_{N-8} + A'_{N-4} = A^0_4 + A^+_4 + \left(A^-_8 + A^0_8 + A^+_8\right) + \left(A^-_{12} + A^0_{12} + A^+_{12}\right) + \ldots + \\
+ \left(A^-_{N-8} + A^0_{N-8} + A^+_{N-8}\right) + A^-_{N-4} + A^0_{N-4} + A^-_N = \\
= A^0_4 + A^+_4 + A_8 + A_{12} + \ldots + A_{N-8} + A^-_{N-4} + A^0_{N-4} + A^-_N.
\end{aligned}$$

Notice that $A^-_4 = 0$, since zero-codeword is uniquely expressed as $\sum_{s=1}^{n-k} 0 \cdot g_s$. Therefore, $S_4 = S^0_4 \cup S^+_4$, and this union is disjoint. So, $A_4 = A^0_4 + A^+_4$. The previous expression becomes:

$$A'_4 + A'_8 + \ldots + A'_{N-8} + A'_{N-4} = A_4 + A_8 + A_{12} + \ldots + A_{N-8} + A_{N-4} - A^+_{N-4} + A^-_N.$$

This completes the proof. ■

Remark 2.3.7. Notice that the left side of (ii) in Theorem 2.3.6 represent the number of all non-trivial words in C' that have the Euclidean weight smaller than N. Especially, if $N = 8\lfloor \frac{n}{24} \rfloor + 8$, the C' will be extremal if and only if the right side of the equation in the second statement is equal to 0. This enables the determination of the extremality of the code C' from the coefficients in the Euclidean weight enumerator of the code C, and the numbers A^+_{N-4} and A^-_N.

Main goal of our modification is to reduce the number of the minimum Euclidean weight calculations. We achieve this by increasing the number of the codes whose extremality we can check with one calculation of the minimum Euclidean weight.

Let B be a binary matrix that have $t > 0$ upper diagonal elements equal to 0. From the Euclidean weight distribution of the code C determined by $G(B)$, Proposition 2.4, and the second statement in Theorem 2.3.6, we can determine whether any of the t codes that are neighbors of C are extremal.

Algorithm of the construction: Let N be desired minimum Euclidean weight of the self-dual $\mathbb{Z}_4$-code ($N = 8\left\lfloor \frac{n}{24} \right\rfloor + 8$ for the extremal $\mathbb{Z}_4$-codes, and $N = 8\left\lfloor \frac{n}{24} \right\rfloor + 4$ for the near-extremal $\mathbb{Z}_4$-codes). Steps of the algorithm:

1. Start with the arbitrary matrix B.
2. In each iteration of the algorithm do the following.
   - 2.1 Generate a self-dual $\mathbb{Z}_4$-code C with the generator matrix $G(B)$, and evaluate
   $$D = \left|\{v \in C | 0 < wt_E(v) < N\}\right|,$$
   (this is $A_4 + A_8 + \ldots + A_{N-4}$ from Theorem 2.3.6).
   - 2.2 If $D = 0$ then C is extremal.
   - 2.3 Determine the sets S_{N-4} and S_N defined in Theorem 2.3.6.
   - 2.4 For every upper diagonal element (i, j) of the matrix B which is equal to 0 determine the (i, j)-neighbor C'. If extremality of the corresponding neighbor is undetermined, by using Theorem 2.3.1 and Theorem 2.3.2, evaluate numbers A^+_{N-4}, A^-_N and $d = D - A^+_{N-4} + A^-_N$.
   - 2.5 All C' that have $d = 0$ are extremal (see Remark 2.3.7).
   - 2.6 Mark all neighbors of B as checked.
   - 2.7 Repeat the process with the next unchecked matrix B.

Example 2.3.8. In this example we will demonstrate one iteration of the given algorithm. All of the computations given here are executed with software MAGMA. Let $C^{(1)}$ be the doubly-even binary code generated with the matrix:

$$G^{(1)} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Its dual code $C^{(2)}$ has the generator matrix:

$$G^{(2)} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

From the condition (2.6) in Remark 2.1.12 we construct a 6×6 binary matrix B that has all upper diagonal elements equal to 0:

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

The generator matrix of the self-dual $\mathbb{Z}_4$-code C is the matrix:

$$G(B) = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 2 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 2 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 2 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \\ 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 2 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Let $N = 8\left\lfloor \frac{16}{24} \right\rfloor + 8 = 8$ be the target weight i.e., we want to construct an extremal $\mathbb{Z}_4$-code. Notice that, since $N = 8$, it holds $S_{N-4} = S_4$. Also, from the definition of D, it follows that in this case, $D = |S_4|$ (this also means that $D = A_4$ from Theorem 2.3.6). The set of all codewords of Euclidean weight 4 is given as follows:

$$S_4 = \{3000000011003000, 0000103000000330, 1000000033001000, 0000301000000110\}.$$

Let v_1, v_2, v_3, v_4 be the codewords from S_4, given by the order of appearance. In the terms of the generator matrix $G(B)$, these codewords can be written as:

$$v_1 = 2g_1 + 2g_2 + 3g_3 + g_7 + g_8, \qquad v_2 = g_4 + 3g_5,$$
$$v_3 = 2g_1 + 2g_2 + g_3 + g_7 + g_8, \qquad v_4 = 3g_4 + g_5,$$

where g_i stands for the i-th row of the matrix $G(B)$. Before we can determine the number A_4^+, we need to choose (i, j)-neighbor of B that we will examine. From Theorem 2.3.2, we know that only the coefficients next to g_i, for $i \in \{1, 2 \ldots, 6\}$, are important for the change of the weight of codeword, and that coefficients c_i and c_j should be of different parity. Therefore, we should not take the following pairs of (i, j) into the consideration:

$$\begin{aligned} &(1,a), && a = 2,3,4,5,6,\\ &(2,b), && b = 3,4,5,6,\\ &(4,c), && c = 5,6,\\ &(3,6)\,(5,6). \end{aligned}$$

Changes of weights in all codewords of S_4 can happen only for the pairs $(3,4)$, $(3,5)$. Let us consider the $(3,5)$-neighbor of B. Since $b_{3,5} \neq b_{5,3}$, from the second part of Table 2.4, we have the following.

- For v_1: Since $c_3 = 3$, and $c_5 = 0$ we have to check the parity of the $|J|$ (see Remark 2.3.3). $|J|$ is the number of all non-zero elements in the 5-th column in B, in the rows 1 and 2 (since the non-zero generators of v_1 are g_1, g_2 and g_3, and the rows 3 and 5 are not counted). This gives $|J| = 0$. From the table we get $r = 1$ and therefore, by Corollary 2.3.4, $wt_E(v_1') = 4+4 = 8$. In other words, $v_1 \in S_4^+$.
- For v_2: Since $c_3 = 0$, and $c_5 = 3$ we have to look at the parity of the $|I|$ (see Remark 2.3.3). $|I|$ is the number of all non-zero elements in the 3-rd column and the 4-th row in B (since the non-zero generators of v_2 are g_4, g_5, and rows 3 and 5 are not counted). This gives $|I| = 1$. From the table we get $r = 1$, and therefore, by Corollary 2.3.4, $wt_E(v_1') = 4+4 = 8$. In other words, $v_1 \in S_4^+$.
- In the similar way we conclude that $v_3, v_4 \in S_4^+$.

Therefore, $A^+_{N-4} = A^+_4 = |S^+_4| = 4$. It still remains to determine the number A^-_8. Since the code C has 242 codewords of weight 8, we will calculate this by MAGMA. We verified, in the same way as for A^+_4, that $A^-_8 = 0$. Now we can determine d:

$$d = D - A^+_{N-4} + A^-_N = D - A^+_4 + A^-_8 = 4 - 4 + 0 = 0.$$

Therefore, a $(3,5)$-neighbor of C is extremal. The generator matrix of that code is:

$$G(B') = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 2 & 2 & 1 & 0 & 2 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 2 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \\ 0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 2 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

where B' stands for $(3,5)$-neighbor of B.

2.3.1. Testing the algorithm

Before we test the actual implemented algorithm described in the previous section, we will analyze how the order in which matrices B are chosen impacts the performance of the algorithm.

We have seen that every matrix B is uniquely determined by its upper diagonal elements by the condition (2.6). Therefore, we can identify each $k \times k$ matrix B with the binary sequence $n(B)$, of length $\frac{k(k-1)}{2}$, that consist of the upper diagonal elements of B, in the following way:

$$B = \begin{bmatrix} 0 & b_{1,2} & b_{1,3} & \cdots & b_{1,k} \\ \star & 0 & b_{2,3} & \cdots & b_{2,k} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \star & \star & \cdots & 0 & b_{k-1,k} \\ \star & \star & \star & \cdots & 0 \end{bmatrix} \leftrightarrow n(B) = \left(b_{1,2}, b_{1,3}, \cdots, b_{1,k}, b_{2,3}, \cdots, b_{2,k}, \cdots, b_{k-1,k}\right).$$

We also know, from (2.8), that if B' is a neighbor of B, then B and B' differ in exactly one element in the upper diagonal. Therefore, the binary sequences $n(B)$ and $n(B')$ differ in exactly one coordinate. So, for the code $C^{(1)}$ of the dimension k, the corresponding search

space can be presented by the $\frac{k(k-1)}{2}$-hypercube graph whose set of vertices is given by sequences $n(B)$. Also, from the sequence $n(B)$ that have t coordinates equal to 0, we can determine the minimum Euclidean weight of t neighbors of B that have $t-1$ coordinates equal to 0.

First we will discuss the situation when the matrices B are chosen in the lexicographical order. Since vertices in the same partition are not adjacent, we have to check all the vertices in one of the two partitions. This means that the number of calculations of minimum Euclidean weights would be $2^{\frac{k(k-1)}{2}-1}$ (the half of the search space, since each partition of hypercube graph is of the same size). In order to give more detailed analysis of this case, we present the following example.

Example 2.3.9. Let us consider the 4-hypercube graph. This length of binary sequences $n(B)$ cannot occur in described construction, since equation $\frac{k(k-1)}{2} = 4$ have no integer solutions. Still, this example is small enough to demonstrate the main idea behind the lexicographic traverse of the search space. In Figure 2.1 the search space is given. The vertices with the same number of ones are positioned between red lines. In the right side of the figure, the number of ones for that set of vertices is given.

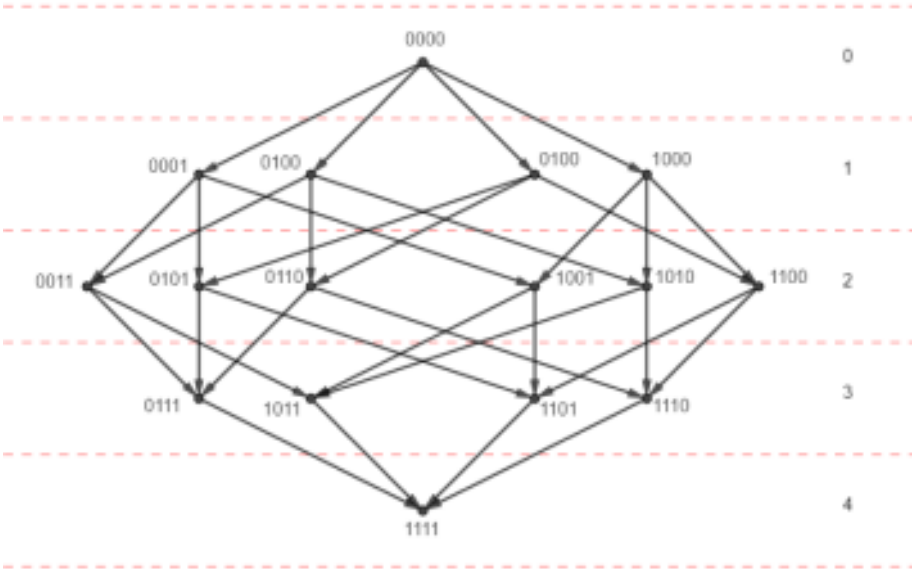

Figure 2.1: The search space for the sequences of length 4

With arrows on the edges of the graph we marked the visibility of each vertex from

other vertices. Precisely, for x and y two vertices in the given figure, with an arrow that starts in x and ends in y we denote that the minimum Euclidean weight of y can be deduced from x. The lexicographical order of the set of vertices is given as follows:

$$0000, 0001, 0010, 0011,$$
$$0100, 0101, 0110, 0111,$$
$$1000, 1001, 1010, 1011,$$
$$1100, 1101, 1110, 1111.$$

In Table 2.5, we give the lexicographic traversal of this graph.

Step	Current vertex	Checked vertices in step	Total number of checked vertices
1.	0000	0000,0001,0010,0100,1000	5
2.	0011	$0011, 0111, 1011$	8
3.	0101	$0101, 1101$	10
4.	0110	$0110, 1110$	12
5.	1001	1001	13
6.	1010	1010	14
7.	1100	1100	15
8.	1111	1111	16

Table 2.5: Steps of the algorithm for the lexicographical traverse of the search space

We can observe that the size of the search space is $2^4 = 16$, and that the number of calculations of minimum Euclidean weights needed to cover the whole search space is 8 (half of the size of the search space). Also, notice how the vertices with odd number of ones are covered by the vertices with even number of ones.

We will now discuss the random order of traverse of the search space. First, we give the following example.

Example 2.3.10. Let the search space be the same as in Example 2.3.9. Assume that the order of traverse of search space is given with the following random order of vertices:

$$0110, 1100, 1000, 0000,$$
$$0011, 1111, 1010, 0001,$$
$$0100, 1001, 0101, 1110,$$
$$0111, 0010, 1101, 1011.$$

For this order, the traverse of search space is given in Table 2.6.

Step	Current vertex	Checked vertices in step	Total number of checked vertices
1.	0110	0110,0111,1110	3
2.	1100	1100, 1101	5
3.	1000	1000, 1001, 1010	8
4.	0000	0000, 0001, 0010, 0100	12
5.	0011	0011, 1011	14
6.	1111	1111	15
7.	0101	0101	16

Table 2.6: Steps of the algorithm for the random order of traverse of search space

We can see that for this order of traverse of the search space, we get that the number of the calculations of the minimum Euclidean weight needed to cover the whole search space is 7. This is better than in Example 2.3.9.

From the previous example we have seen that the random traversal can preform better than the lexicographical one. This is because vertices of the graph can be chosen from

both blocks of partition, while in the lexicographical traversal they are always taken from one block of the partition. This property of the random traversal becomes more noticeable with larger search spaces. With that said, we will now give some statistical analysis of the random traversal of the search space of size $2^{\frac{k(k-1)}{2}}$, and $k \in \{4,5\}$.

For $k = 4$, the size of the search space is $2^6 = 64$. In this case, the binary sequences $n(B)$ are of length 6. The lexicographic traversal of search space needs 32 calculations of minimum Euclidean weights in order to cover the whole search space. We randomly generated 1000 traversals of the search space, and determined the number of calculations of minimum Euclidean weights needed to cover the search space for each of these traversals. We get that the average number of calculations is 24.58. The maximum number of calculations that was needed was 36, and the minimum number was 23. The mean value of the numbers of needed calculations of minimum Euclidean weight was 27. Among the generated 1000 traversals, 15 performed worse than the lexicographic one, which amounts to 1.5% of the generated ones.

For $k = 5$, the size of the search space is $2^{10} = 1024$. In this case, the length of the binary sequences $n(B)$ is 10. The lexicographic traversal needs 512 calculations of minimum Euclidean weights in order to cover the whole search space. As in the case for $k = 4$, we generated 1000 traversals of the search space. The average number of calculations of the minimum Euclidean weight needed to cover the search space was 356.659. In the worst case, that number was 380, and in the best case 335. The mean value was 357. We can see that in this case, all of the generated traversal orders performed significantly better than the lexicographical one.

We will search for extremal and near-extremal $\mathbb{Z}_4$-codes that have dimension of the residue code at least 6. Based on the previous discussion, it is reasonable to use a random choice of matrices B instead of the lexicographical order. In addition to the discussed, another reason why such order makes sense is the uneven distribution of extremal and near-extremal $\mathbb{Z}_4$-codes in the search space. We have seen in Example 2.3.9 that, when the lexicographic order is used, vertices in the search space are covered layer by layer, i.e., first all vertices without ones, then vertices with two ones, etc. This means that a lot of time could pass until extremal and near-extremal $\mathbb{Z}_4$-codes that have generator matrix

$G(B)$ with small number of 0 are reached. With random order approach, we will get the wider sample of codes in the same execution time.

For the end of this discussion we will present the performance of the modified algorithm on a $[16,6,4]$ code given in Example 2.3.8, and compare it to the standard algorithm. The algorithm was tested on the machine with an Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz processor, and 16GB RAM memory of frequency 2400MHz. The standard algorithm finished in 155.844s. The modified method with the lexicographical order of traverse completed in 200.860s. Up until 127.438s of the execution, the modified algorithm performed better. Our method gets slower over time due to the exploit of unchecked neighbors (see the last four steps in Table 2.5). We know that the calculation of the minimum Euclidean weight becomes time consuming even for codes of small lengths (for example length 32), and that the search space grows exponentially with the increase of the dimension of the residue code. Therefore, the impact of the slow down of the method becomes negligible for codes that are yet unclassified.

As discussed before, in order to improve the number of checked codes per time, a version of the modified algorithm with the random choice of matrices B should be used. This modification of the order in which matrices B are generated improves the performance of the algorithm. On the same example, we obtained the 95% of the codes in the 115.047s. In 77s ($\approx 50\%$ of the standard algorithm execution time) the 13321 of the extremal $\mathbb{Z}_4$-codes are constructed, which is 78.2% of the total number of extremal $\mathbb{Z}_4$-codes in this test example. The comparison of modified methods and the standard method is given in Figure 2.2. One can also notice that the total execution time of the modified algorithm with random approach is longer (≈ 400s). This happens due to the exploitation of the available unchecked matrices B. Because the search space is large, we did not give complete order of traverse as in Example 2.3.9 and Example 2.3.10, since that would be to demanding for the memory of the computer. Instead, the new random matrix B is generated in each step. As the most of the search space gets examined, the number of non-checked matrices B gets small, so it takes longer to randomly find yet unchecked matrices B. As we have already discussed, this does not present a problem for the codes of length 32 or greater.

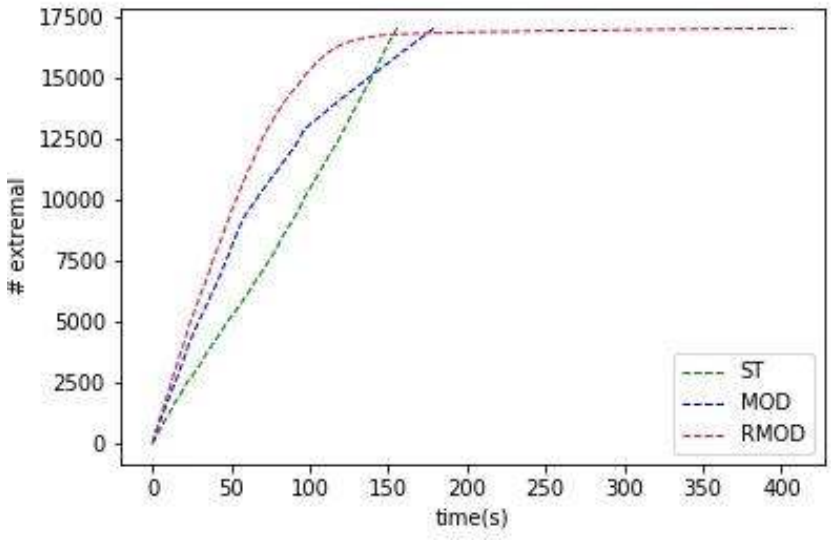

Figure 2.2: Comparison of the standard method (ST), the modified algorithm with lexicographic traversal (MOD), and the modified algorithm with random choice of B (RMOD), on residue code $[16, 6, 4]$

3. CONSTRUCTION OF $2-(4n-1,2n-1,n-1)$ HADAMARD DESIGNS FROM THE SKEW INCIDENCE MATRIX OF A HADAMARD $2-(n-1,\frac{n}{2}-1,\frac{n}{4}-1)$ DESIGN

In the previous chapters we have seen that the starting point of the construction of a self-dual $\mathbb{Z}_4$-code is a doubly-even binary code. In Corollary 1.3.20 it was proved that the incidence matrix of the Hadamard 3-design spans a doubly-even binary code. In this section, we describe a new method of obtaining Hadamard 2-designs on $4n-1$ points, starting from a skew incidence matrix of a Hadamard 2-design on $n-1$ points. This method gives a nice way to construct a series of Hadamard 2-designs from which we can obtain doubly-even binary codes. In whole section we assume that n is a positive integer and $n \equiv 0 \pmod 4$.

Before the main construction, we give the following lemma. This construction as also used in [2].

Lemma 3.1.1. Let B_0 be a skew-symmetric incidence matrix of a 2-$(n-1,\frac{n}{2}-1,\frac{n}{4}-1)$ Hadamard design. Let J_{n-1} be an all-one square matrix of order $n-1$. For matrices

$B_1 = B_0^T$, $B_2 = J_{n-1} - B_1$ and $B_3 = J_{n-1} - B_0$, the following hold:

$$\begin{aligned}
B_0^T B_1 &= \frac{n}{4} J_{n-1} - \frac{n}{4} I_{n-1} - B_0^T, \\
B_0^T B_2 &= \left(\frac{n}{4} - 1\right) J_{n-1} + \frac{n}{4} I_{n-1} + B_0^T, \\
B_1^T B_3 &= \left(\frac{n}{4} - 1\right) J_{n-1} + \frac{n}{4} I_{n-1} + B_0, \\
B_2^T B_3 &= \left(\frac{n}{4} + 1\right) J_{n-1} - \frac{n}{4} I_{n-1} - B_0.
\end{aligned}$$

Proof. Let $\mathscr{D}$ be a 2-$(n-1,\frac{n}{2}-1,\frac{n}{4}-1)$ Hadamard design with incidence matrix B_0. Let $\mathscr{D}'$ be a dual design of $\mathscr{D}$, and let $\overline{\mathscr{D}}$ and $\overline{\mathscr{D}'}$ be the complementary designs of designs $\mathscr{D}$ and $\mathscr{D}'$ respectively. With this notation, it is obvious that B_1, B_2 and B_3 are incidence matrices of designs $\mathscr{D}'$, $\overline{\mathscr{D}'}$ and $\overline{\mathscr{D}}$. Since $\mathscr{D}$ is symmetric, the $\mathscr{D}'$ is also a 2-$(n-1,\frac{n}{2}-1,\frac{n}{4}-1)$ Hadamard design. Since $\overline{\mathscr{D}'}$ and $\overline{\mathscr{D}}$ are complementary designs of designs with parameters 2-$(n-1,\frac{n}{2}-1,\frac{n}{4}-1)$, they are symmetric designs with parameters 2-$(n-1,\frac{n}{2},\frac{n}{4})$. Notice that $r = k$ for all defined designs, since all of the designs are symmetric.

Now from Theorem 1.1.7 we have:

$$B_0^T B_0 = B_1^T B_1 = \left(\frac{n}{4} - 1\right) J_{n-1} + \left(\frac{n}{2} - 1 - \frac{n}{4} + 1\right) I_{n-1} = \left(\frac{n}{4} - 1\right) J_{n-1} + \frac{n}{4} I_{n-1}, \quad (3.1)$$

$$B_2^T B_2 = B_3^T B_3 = \frac{n}{4} J_{n-1} + \left(\frac{n}{2} - \frac{n}{4}\right) I_{n-1} = \frac{n}{4} J_{n-1} + \frac{n}{4} I_{n-1}. \quad (3.2)$$

For the incidence matrices B_0 and B_3 (of complementary designs) we have:

$$\begin{aligned}
B_0^T B_3 = B_0^T (J_{n-1} - B_0) &= \underbrace{B_0^T J_{n-1}}_{\substack{rJ_{n-1} \\ r=\frac{n}{2}-1}} - B_0^T B_0 \overset{(3.1)}{=} \\
&= \left(\frac{n}{4} - 1\right) J_{n-1} - \left(\left(\frac{n}{4} - 1\right) J_{n-1} + \frac{n}{4} I_{n-1}\right) = \frac{n}{4} J_{n-1} - \frac{n}{4} I_{n-1}.
\end{aligned}$$

In the same way we obtain the relation for matrices B_1 and B_2 :

$$B_1^T B_2 = B_1^T (J_{n-1} - B_1) = \frac{n}{4} J_{n-1} + \frac{n}{4} I_{n-1}. \quad (3.3)$$

Now we determine $B_0^T B_1$. Since B_0 is a skew symmetric matrix, and $B_0^T = B_1$, we have that $B_0^T B_1 = \left(B_0^2\right)^T$. Therefore, we have to determine B_0^2. From the definition of the skew symmetric matrix, we know that $B_0 + B_1 = J_{n-1} - I_{n-1}$. Therefore, we have:

$$(B_0 + B_1)^2 = (J_{n-1} - I_{n-1})^2 \Leftrightarrow B_0^2 + B_0 B_1 + B_1 B_0 + B_1^2 = J_{n-1}^2 - 2J_{n-1} + I_{n-1} \quad (3.4)$$

The left side of the previous equation is:

$$\begin{aligned}B_0^2+B_0B_1+B_1B_0+B_1^2 &= B_0^2+B_0B_0^T+B_0^TB_0+B_1^2 = B_0^2+\left(B_0^TB_0\right)^T+B_0^TB_0+B_1^2 \overset{(3.1)}{=}\\ &= B_0^2+\left(\left(\frac{n}{4}-1\right)J_{n-1}+\frac{n}{4}I_{n-1}\right)^T+\left(\left(\frac{n}{4}-1\right)J_{n-1}+\frac{n}{4}I_{n-1}\right)+B_1^2 =\\ &= B_0^2+B_1^2+\left(\frac{n}{2}-2\right)J_{n-1}+\frac{n}{2}I_{n-1}.\end{aligned}$$

Since $J_{n-1}^2=(n-1)J_{n-1}$, the right side of the equation (3.4) is:

$$J_{n-1}^2-2J_{n-1}+I_{n-1}=(n-1)J_{n-1}-2J_{n-1}+I_{n-1}=(n-3)J_{n-1}+I_{n-1}.$$

Therefore, the equation (3.4) is equivalent to the equation:

$$B_0^2+B_1^2+\left(\frac{n}{2}-2\right)J_{n-1}+\frac{n}{2}I_{n-1}=(n-3)J_{n-1}+I_{n-1}$$

From where we get:

$$B_0^2+B_1^2=\left(\frac{n}{2}-1\right)J_{n-1}+\left(1-\frac{n}{2}\right)I_{n-1} \tag{3.5}$$

On the other hand, from $B_1=J_{n-1}-I_{n-1}-B_0$, we have:

$$\begin{aligned}B_1^2 &= \left(J_{n-1}+I_{n-1}-B_0\right)^2 = J_{n-1}^2+B_0^2+I_{n-1}^2-JB_0-B_0J-2J_{n-1}+2B_0 =\\ &= (n-1)J_{n-1}+B_0^2+I_{n-1}-2\left(\frac{n}{2}-1\right)J_{n-1}-2J_{n-1}+2B_0 =\\ &= B_0^2+2B_0+I_{n-1}-J_{n-1}\end{aligned}$$

Now, when we substitute the previous relation into the equation (3.5), we get:

$$\begin{aligned}&B_0^2+B_1^2=\left(\frac{n}{2}-1\right)J_{n-1}+\left(1-\frac{n}{2}\right)I_{n-1},\\ &B_0^2+B_0^2+2B_0+I_{n-1}-J_{n-1}=\left(\frac{n}{2}-1\right)J_{n-1}+\left(1-\frac{n}{2}\right)I_{n-1},\\ &2B_0^2=\frac{n}{2}J_{n-1}-\frac{n}{2}I_{n-1}-2B_0\Big/\cdot\frac{1}{2},\\ &B_0^2=\frac{n}{4}J_{n-1}-\frac{n}{4}I_{n-1}-B_0.\end{aligned}$$

Therefore,

$$B_0^TB_0=\left(B_0^2\right)^T=\left(\frac{n}{4}J_{n-1}-\frac{n}{4}I_{n-1}-B_0\right)^T=\frac{n}{4}J_{n-1}-\frac{n}{4}I_{n-1}-B_0^T.$$

Now we give the proof of the remaining relations. By the definition of the B_2, we have:

$$B_2=J_{n-1}-B_1=J_{n-1}-\left(J_{n-1}-I_{n-1}-B_0\right)=B_0+I_{n-1}.$$

Therefore:

$$B_0^T B_2 = B_0^T (B_0 + I_{n-1}) = B_0^T B_0 + B_0^T = \left(\frac{n}{4}-1\right) J_{n-1} + \frac{n}{4} I_{n-1} + B_0^T.$$

Since $B_1^T = B_0$, and $B_3 = J_{n-1} - B_0$, we have:

$$B_1^T B_3 = B_0 (J_{n-1} - B_0) = \underbrace{B_0 J_{n-1}}_{\substack{kJ_{n-1} \\ k= \\ =\frac{n}{2}-1}} - B_0^2 = \left(\frac{n}{2}-1\right) J_{n-1} - \left(\frac{n}{4} J_{n-1} - \frac{n}{4} I_{n-1} - B_0\right) =$$

$$= \left(\frac{n}{4}-1\right) J_{n-1} + \frac{n}{4} I_{n-1} + B_0.$$

Finaly, since $B_2^T = \left(J_{n-1} - B_0^T\right)^T = J_{n-1} - B_0$ and $B_3 = J_{n-1} - B_0$, we have:

$$B_2^T B_3 = (J_{n-1} - B_0)^2 = J_{n-1}^2 - \underbrace{J_{n-1} B_0}_{\substack{rJ_{n-1} \\ r=\frac{n}{2}-1}} - \underbrace{B_0 J_{n-1}}_{\substack{kJ_{n-1} \\ k=\frac{n}{2}-1}} + B_0^2 =$$

$$= (n-1) J_{n-1} - 2\left(\frac{n}{2}-1\right) J_{n-1} + \frac{n}{4} J_{n-1} - \frac{n}{4} I_{n-1} - B_0 =$$

$$= \left(\frac{n}{4}+1\right) J_{n-1} - \frac{n}{4} I_{n-1} - B_0.$$

■

Now we define matrices that will be used in the construction of Hadamard designs. Let $J_{3,n-1}$ be a $3 \times (n-1)$ all-one matrix. Let A_1, A_2, and A_3, be a $3 \times (n-1)$ matrices given as follows:

$$A_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 1 \end{bmatrix}.$$

Let $\overline{A_1} = J_{3,n-1} - A_1$, $\overline{A_2} = J_{3,n-1} - A_2$, and $\overline{A_3} = J_{3,n-1} - A_3$ Let B_0, B_1, B_2 and B_3 be matrices defined as in Lemma 3.1.1. Let O be an $(n-1) \times 3$ zero matrix. Define the following matrices:

$$H_1 = \begin{bmatrix} I_3 & J_{3,n-1} & A_1 & A_2 & A_3 \\ J_{3,n-1}^T & & & & \\ A_1^T & & & & \\ A_2^T & & D & & \\ A_3^T & & & & \end{bmatrix}, \quad H_2 = \begin{bmatrix} I_3 & J_{3,n-1} & A_1 & A_2 & A_3 \\ \overline{A_3}^T & & & & \\ \overline{A_2}^T & & & & \\ \overline{A_1}^T & & D & & \\ O & & & & \end{bmatrix},$$

where D is a square block matrix of order $4n-4$, of the form:

$$D=\begin{bmatrix} B_{i_1} & B_{i_2} & B_{i_3} & B_{i_4} \\ B_{i_5} & B_{i_6} & B_{i_7} & B_{i_8} \\ B_{i_9} & B_{i_{10}} & B_{i_{11}} & B_{i_{12}} \\ B_{i_{13}} & B_{i_{14}} & B_{i_{15}} & B_{i_{16}} \end{bmatrix},$$

with the possibilities for the combinations of indices $i_1 i_2 \dots i_{16}$ given in Table 3.1.

Table 3.1: The combination of indices for the matrix D

$i_1 i_2 i_3 i_4 i_5 i_6 i_7 i_8 i_9 i_{10} i_{11} i_{12} i_{13} i_{14} i_{15} i_{16}$	
H_1	H_2
0000003303030330	0002003103010332
0000003303031221	0002003112011232
0000003303120321	0002012102110332
0000003312031230	0003003003000333
0000003312121221	0003003003001222
0000012302131221	0003003003110322
0000102312121320	0003003012001233
0000112212121221	0003003012110333
0001003203020331	0003003012111222
0001003212021231	0003012002100333
0001012202120331	0003012002101222
0011002203030330	0003013112110232
0011002203031221	0003103113010333
0011002203120321	0003103113011222
0011002212031230	0012002103110322
0011002212121221	0012002112001233
0111002302120320	0013012103100222
0111012202120221	0112012102110222

The following proposition holds.

Proposition 3.1.2. Let B_0 be an incidence matrix of a Hadamard design with parameters $2-(n-1,\frac{n}{2}-1,\frac{n}{4}-1)$. If B_0 is skew-symmetric, then the matrices H_1 and H_2 are the incidence matrices of $2-(4n-1,2n-1,n-1)$ designs, for any combination of the indices given in Table 3.1.

Proof. In order to prove that each point is incident with $2n-1$ blocks and that any two points are incident with $n-1$ blocks we will use the relations given in Lemma 3.1.1.

Since B_0 and B_1 are incidence matrices of 2-$\left(n-1,\frac{n}{2}-1,\frac{n}{4}-1\right)$ designs, and B_2 and B_3 are incidence matrices of 2-$\left(n-1,\frac{n}{2},\frac{n}{4}\right)$ designs, from Theorem 1.1.7 we have:

$$B_0^TB_0=B_1^TB_1=\left(\frac{n}{4}-1\right)J_{n-1}+\frac{n}{4}I_{n-1},\quad B_2^TB_2=B_3^TB_3=\frac{n}{4}J_{n-1}+\frac{n}{4}I_{n-1}.$$

Let D be the matrix obtained from H_1 and the third row of Table 3.1. The scalar product of each block column of H_1 with itself gives the following identity:

$$I_3^2+J_{3,n-1}J_{3,n-1}^T+A_1A_1^T+A_2A_2^T+A_3A_3^T=(n-1)J_3+nI_3,$$

for the first column, and the following identities for the remaining block columns, respectively:

$$\begin{aligned}
&J_{3,n-1}^TJ_{3,n-1}+4B_0^TB_0=(n-1)J_3+nI_3,\\
&A_1^TA_1+2B_0^TB_0+2B_3^TB_3=(n-1)J_3+nI_3,\\
&A_2^TA_2+B_0^TB_0+B_3^TB_3+B_1^TB_1+B_2^TB_2=(n-1)J_3+nI_3,\\
&A_3^TA_3+B_0^TB_0+B_3^TB_3+B_2^TB_2+B_1^TB_1=(n-1)J_3+nI_3.
\end{aligned}$$

All the diagonal entries are equal to $2n-1$, which proves that the number of points that are incident with each block is $k=2n-1$.

It remains to check the product of the two different block columns of H_1. The product of the first block column with the second, third, fourth and fifth block column, respec-

tively, gives the following:

$$I_3J_{3,n-1}+J_{3,n-1}B_0+A_1B_0+A_2B_0+A_3B_0=(n-1)J_{3,n-1},$$
$$I_3A_1+J_{3,n-1}B_0+A_1B_0+A_2B_3+A_3B_3=(n-1)J_{3,n-1},$$
$$I_3A_2+J_{3,n-1}B_0+A_1B_3+A_2B_1+A_3B_3=(n-1)J_{3,n-1},$$
$$I_3A_3+J_{3,n-1}B_0+A_1B_3+A_2B_2+A_3B_1=(n-1)J_{3,n-1}.$$

The product of the second block column with the third, fourth and fifth, respectively, is determined as follows:

$$J_{3,n-1}^TA_1+2B_0^TB_0+2B_0^TB_3=(n-1)J_{n-1},$$
$$J_{3,n-1}^TA_2+B_0^TB_0+B_0^TB_3+B_0^TB_1+B_0^TB_2=(n-1)J_{n-1},$$
$$J_{3,n-1}^TA_3+B_0^TB_0+B_0^TB_3+B_0^TB_2+B_3^TB_1=(n-1)J_{n-1}.$$

The products of the third block column with the fourth and fifth, respectively, are:

$$A_1^TA_2+B_0^TB_0+B_0^TB_3+B_3^TB_1+B_3^TB_2=(n-1)J_{n-1},$$
$$A_1^TA_3+B_0^TB_0+B_0^TB_3+B_3^TB_2+B_0^TB_1=(n-1)J_{n-1}.$$

Finaly, the product of the fourth and the fifth block column is:

$$A_2^TA_3+B_0^TB_0+B_3^TB_3+B_1^TB_2+B_2^TB_1=(n-1)J_{n-1}.$$

This completes the proof for H_1 with the selected block matrix D, given by the third row of Table 3.1. In a similar way it can be shown that the statement is valid for all other indices in Table 3.1, for both H_1 and H_2. ■

Let us observe the following example.

Example 3.1.3. Let SH_8 be a Hadamard matrix of order 8 from (1.1). In Example 1.2.13 we constructed normalized Hadamard matrix equivalent to the SH_8, and then used

Theorem 1.2.10 to obtain a skew-type incidence matrix of the corresponding Hadamard $2-(7,3,1)$ design. The obtained matrix is:

$$B_0 = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Since B_0 is skew-type matrix, we can use it to construct incidence matrices of $2-(31,15,7)$ designs by Proposition 3.1.2. The matrices B_1, B_2 and B_3 from the proposition are:

$$B_1 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}, B_2 = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix},$$

$$B_3 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}. \tag{3.6}$$

Here we will construct the incidence matrix of designs obtained from the block matrices H_1 and H_2 and the third row of Table 3.1. First we construct the matrix H_1. Since the third row of Table 3.1 is 0000003303120321, the corresponding matrix H_1 is of the form:

$$H_1 = \begin{bmatrix} I_3 & J_{3,n-1} & A_1 & A_2 & A_3 \\ J_{3,n-1}^T & B_0 & B_0 & B_0 & B_0 \\ A_1^T & B_0 & B_0 & B_3 & B_3 \\ A_2^T & B_0 & B_3 & B_1 & B_2 \\ A_3^T & B_0 & B_3 & B_2 & B_1 \end{bmatrix}.$$

Let $\mathscr{D}$ be a design with incidence matrix H_1. If we transpose the matrix H_1 we will get incidence matrix of the dual design $\mathscr{D}'$:

$$H_1^T = \begin{bmatrix} I_3 & J_{3,n-1} & A_1 & A_2 & A_3 \\ J_{3,n-1}^T & B_0^T & B_0^T & B_0^T & B_0^T \\ A_1^T & B_0^T & B_0^T & B_3^T & B_3^T \\ A_2^T & B_0^T & B_3^T & B_1^T & B_2^T \\ A_3^T & B_0^T & B_3^T & B_2^T & B_1^T \end{bmatrix} = \begin{bmatrix} I_3 & J_{3,n-1} & A_1 & A_2 & A_3 \\ J_{3,n-1}^T & B_1 & B_1 & B_1 & B_1 \\ A_1^T & B_1 & B_1 & B_2 & B_2 \\ A_2^T & B_1 & B_2 & B_0 & B_3 \\ A_3^T & B_1 & B_2 & B_3 & B_0 \end{bmatrix}..$$

Since SH_8 is equivalent to SH_8^T, the matrices B_0 and $B_0^T = B_1$ are incidence matrices of isomorphic designs. Therefore, by Theorem 1.1.11, we can obtain B_0 and B_1, from each other, by some permutation of columns σ and rows τ. Since B_2 is incidence matrix of a complementary design of the design with the incidence matrix B_1, and B_3 is incidence matrix of a complementary design of the design with the incidence matrix B_0, we can obtain B_3 and B_2, one from another, with the same permutations σ and τ. Notice that matrices $J_{3,n-1}$, A_1, A_2 and A_3 are invariant to the column permutations. This gives that $J_{3,n-1}^T$, A_1^T, A_2^T and A_3^T are invariant to the row permutations. Therefore, if we apply permutations σ and τ on each of the last four block rows and block columns of H_1^T, we will obtain the matrix H_1. Therefore, design $\mathscr{D}$ is self-dual. For given B_0 and B_1 in (3.6), and $\sigma = \tau = (27)(36)(45)$ we have:

$$B_0 = \begin{bmatrix} 0&1&1&0&1&0&0\\ 0&0&1&1&0&1&0\\ 0&0&0&1&1&0&1\\ 1&0&0&0&1&1&0\\ 0&1&0&0&0&1&1\\ 1&0&1&0&0&0&1\\ 1&1&0&1&0&0&0 \end{bmatrix} \overset{\sigma \text{ on colums}}{\leftrightarrow} \begin{bmatrix} 0&0&0&1&0&1&1\\ 0&0&1&0&1&1&0\\ 0&1&0&1&1&0&0\\ 1&0&1&1&0&0&0\\ 0&1&1&0&0&0&1\\ 1&1&0&0&0&1&0\\ 1&0&0&0&1&0&1 \end{bmatrix}$$

$$\begin{bmatrix} 0&0&0&1&0&1&1\\ 0&0&1&0&1&1&0\\ 0&1&0&1&1&0&0\\ 1&0&1&1&0&0&0\\ 0&1&1&0&0&0&1\\ 1&1&0&0&0&1&0\\ 1&0&0&0&1&0&1 \end{bmatrix} \overset{\sigma \text{ on rows}}{\leftrightarrow} \begin{bmatrix} 0&0&0&1&0&1&1\\ 1&0&0&0&1&0&1\\ 1&1&0&0&0&1&0\\ 0&1&1&0&0&0&1\\ 1&0&1&1&0&0&0\\ 0&1&0&1&1&0&0\\ 0&0&1&0&1&1&0 \end{bmatrix} = B_1.$$

Now we will construct the matrix H_2. The third row of Table 3.1 for the matrix H_2 is 0002012102110332. Therefore, the corresponding block matrix H_2 is of the form:

$$H_2 = \begin{bmatrix} I_3 & J_{3,n-1} & A_1 & A_2 & A_3\\ \bar{A_3}^T & B_0 & B_0 & B_0 & B_2\\ \bar{A_2}^T & B_0 & B_1 & B_2 & B_1\\ \bar{A_1}^T & B_0 & B_2 & B_1 & B_1\\ O & B_0 & B_3 & B_3 & B_2 \end{bmatrix}.$$

The matrix H_2^T is given as follows:

$$H_2^T = \begin{bmatrix} I_3 & J_{3,n-1} & \bar{A_3} & \bar{A_2} & \bar{A_1}\\ A_1^T & B_0^T & B_0^T & B_0^T & B_0^T\\ A_2^T & B_0^T & B_1^T & B_2^T & B_3^T\\ A_3^T & B_0^T & B_2^T & B_1^T & B_3^T\\ O & B_2^T & B_1^T & B_1^T & B_2^T \end{bmatrix} = \begin{bmatrix} I_3 & J_{3,n-1} & \bar{A_3} & \bar{A_2} & \bar{A_1}\\ A_1^T & B_1 & B_1 & B_1 & B_1\\ A_2^T & B_1 & B_0 & B_3 & B_2\\ A_3^T & B_1 & B_3 & B_0 & B_2\\ O & B_3 & B_0 & B_0 & B_3 \end{bmatrix}.$$

We can immediately see, from the first block column, that the matrix H_2^T does not have

the same form as the matrix H_2. Therefore, the corresponding design is not necessarily self-dual.

Based on the discussion in the previous example, we give following remark.

Remark 3.1.4. If the matrix B_0 from Proposition 3.1.2 is an incidence matrix of the self-dual design, then all the designs obtained from H_1 and indices given in Table 3.1 are self-dual. So, in this case, this construction gives at most 54 non-isomorphic designs. If B_0 and B_0^T are incidence matrices of non-isomorphic designs, then the duals of designs obtained from B_0 and H_1 are constructed from B_0^T and H_1. Therefore, in that case, at most 108 non-isomorphic designs can be constructed from B_0 and B_0^T, at most 54 from the matrix B_0, and at most 54 from B_0^T.

4. CONSTRUCTED CODES AND RELATED COMBINATORIAL STRUCTURES

This chapter contains the computational results of our work. The chapter is divided in to three parts. In the first part, we give results about extremal and near-extremal $\mathbb{Z}_4$-codes of smaller lengths: 32, 40 and 48. The second part consists of partial results on codes of lengths 56, 64, and 72. In the last part of this chapter, we give an overview of the constructed combinatorial designs and strongly regular graphs from the codes obtained in the first part of this chapter. For the construction of Hadamard designs and their full automorphism groups we used the software GAP [50]. Rest of the computations were done by MAGMA [8].

4.1. CODES OF LENGTHS 32, 40 AND 48

Codes of length 32

Up to equivalence, there is only one Hadamard design on 7 points with a skew incidence matrix. This design can be constructed from the skew-symmetric Hadamard matrix SH_8 given in (1.1). From that matrix, using Proposition 3.1.2 we constructed 54 Hadamard designs with parameters $2-(31,15,7)$. From the block matrix H_1, we obtained 18 designs $\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_{18}$. Designs $(\mathscr{D}_2, \mathscr{D}_9)$, $(\mathscr{D}_4, \mathscr{D}_{12})$, $(\mathscr{D}_5, \mathscr{D}_{14})$, $(\mathscr{D}_6, \mathscr{D}_{11})$, $(\mathscr{D}_7, \mathscr{D}_{17})$, $(\mathscr{D}_8, \mathscr{D}_{18})$, $(\mathscr{D}_{10}, \mathscr{D}_{13})$ are pairwise dual. From the block matrix H_2 we obtained 18 designs $\mathscr{D}_{19}, \ldots, \mathscr{D}_{36}$, and their duals $\mathscr{D}_{37}, \ldots, \mathscr{D}_{54}$. The structure of the full automorphism

groups of designs $\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_{54}$ is given in Table 4.1.

The size of the group	The structure of the group	Designs
9999360	$PSL(5,2)$	$\mathscr{D}_1$
64512	$(E_{64} : PSL(3,2)) : S_3$	$\mathscr{D}_{22}, \mathscr{D}_{40}$
8064	$(E_{64} : Z_{21}) : Z_6$	$\mathscr{D}_8, \mathscr{D}_{18}, \mathscr{D}_{36}, \mathscr{D}_{54}$
2688	$(\mathbb{Z}_2 : E_{64}) : Frob_{21}$	$\mathscr{D}_{16}$
336	$SL(2,7)$	$\mathscr{D}_3$
336	$E_{16} : Frob_{21}$	$\mathscr{D}_2, \mathscr{D}_4, \mathscr{D}_5, \mathscr{D}_9, \mathscr{D}_{12}, \mathscr{D}_{14}, \mathscr{D}_{19}$ $\mathscr{D}_{25}, \mathscr{D}_{31}, \mathscr{D}_{33}, \mathscr{D}_{37}, \mathscr{D}_{43}, \mathscr{D}_{49}, \mathscr{D}_{51}$
126	$Frob_{21} \times S_3$	$\mathscr{D}_{23}, \mathscr{D}_{41}$
42	$\mathbb{Z}_2 \times Frob_{21}$	$\mathscr{D}_6, \mathscr{D}_7, \mathscr{D}_{11}, \mathscr{D}_{15}, \mathscr{D}_{17}, \mathscr{D}_{21}, \mathscr{D}_{24}, \mathscr{D}_{26}$ $\mathscr{D}_{27}, \mathscr{D}_{28}, \mathscr{D}_{29}, \mathscr{D}_{32}, \mathscr{D}_{34}, \mathscr{D}_{35}, \mathscr{D}_{39},$ $\mathscr{D}_{42}, \mathscr{D}_{44}, \mathscr{D}_{45}, \mathscr{D}_{46}, \mathscr{D}_{47}, \mathscr{D}_{50}, \mathscr{D}_{52}, \mathscr{D}_{53}$
21	$Frob_{21}$	$\mathscr{D}_{10}, \mathscr{D}_{13}, \mathscr{D}_{20}, \mathscr{D}_{30}, \mathscr{D}_{38}, \mathscr{D}_{48}$

Table 4.1: The structure of the full automorphism groups of $2-(31,15,7)$ designs

Let $\mathscr{D}_1^*, \ldots, \mathscr{D}_{54}^*$ denote the corresponding Hadamard $3-(32,16,7)$ designs. From designs $\mathscr{D}_1^*$, ..., $\mathscr{D}_{54}^*$ we obtained 21 inequivalent doubly-even binary codes: $C_{32,1}$, $C_{32,2}$, ..., $C_{32,21}$. The dual codes of all of the constructed codes have minimum weight 4. Therefore, by Lemma 2.1.14, all of these codes are suitable for the construction of extremal and near-extremal $\mathbb{Z}_4$-codes. The classes of the designs that give inequivalent codes are given in Table 4.2. The weight distribution of codes $C_{32,1}, C_{32,2}, \ldots, C_{32,21}$ are given in Table 4.3. The code denoted with $C_{32,1}$ is the $RM(1,5)$ code.

The code	Designs	The code	Designs	The code	Designs
$C_{32,1}$	$\mathscr{D}_1^*$	$C_{32,8}$	$\mathscr{D}_{19}^*$, $\mathscr{D}_{36}^*$	$C_{32,15}$	$\mathscr{D}_{37}^*$ $\mathscr{D}_{51}^*$, $\mathscr{D}_{54}^*$
$C_{32,2}$	$\mathscr{D}_2^*$, $\mathscr{D}_5^*$, $\mathscr{D}_8^*$	$C_{32,9}$	$\mathscr{D}_{20}^*$, $\mathscr{D}_{21}^*$, $\mathscr{D}_{35}^*$	$C_{32,16}$	$\mathscr{D}_{38}^*$, $\mathscr{D}_{39}^*$, $\mathscr{D}_{42}^*$, $\mathscr{D}_{53}^*$
$C_{32,3}$	$\mathscr{D}_3^*$, $\mathscr{D}_4^*$, $\mathscr{D}_{12}^*$, $\mathscr{D}_{16}^*$	$C_{32,10}$	$\mathscr{D}_{22}^*$	$C_{32,17}$	$\mathscr{D}_{40}^*$
$C_{32,4}$	$\mathscr{D}_6^*$, $\mathscr{D}_7^*$, $\mathscr{D}_{13}^*$, $\mathscr{D}_{15}^*$	$C_{32,11}$	$\mathscr{D}_{23}^*$, $\mathscr{D}_{26}^*$, $\mathscr{D}_{27}^*$	$C_{32,18}$	$\mathscr{D}_{41}^*$, $\mathscr{D}_{44}^*$,
$C_{32,5}$	$\mathscr{D}_9^*$, $\mathscr{D}_{18}^*$	$C_{32,12}$	$\mathscr{D}_{24}^*$, $\mathscr{D}_{25}^*$, $\mathscr{D}_{28}^*$, $\mathscr{D}_{31}^*$	$C_{32,19}$	$\mathscr{D}_{42}^*$, $\mathscr{D}_{43}^*$, $\mathscr{D}_{46}^*$, $\mathscr{D}_{49}^*$,
$C_{32,6}$	$\mathscr{D}_{10}^*$, $\mathscr{D}_{11}^*$, $\mathscr{D}_{17}^*$	$C_{32,13}$	$\mathscr{D}_{29}^*$, $\mathscr{D}_{30}^*$, $\mathscr{D}_{32}^*$, $\mathscr{D}_{34}^*$	$C_{32,20}$	$\mathscr{D}_{45}^*$
$C_{32,7}$	$\mathscr{D}_{14}^*$	$C_{32,14}$	$\mathscr{D}_{33}^*$	$C_{32,21}$	$\mathscr{D}_{47}^*$, $\mathscr{D}_{48}^*$, $\mathscr{D}_{50}$

Table 4.2: Classes of designs that give codes $C_{32,1}, C_{32,2}, \ldots, C_{32,21}$

Code	$[n,k,d]$	0	4	8	12	16
$C_{32,1}$	$[32,6,16]$	1				62
$C_{32,2}, C_{32,7}$	$[32,9,8]$	1		28		454
$C_{32,3}$	$[32,12,4]$	1	28	84	420	3030
$C_{32,4}$	$[32,15,4]$	1	56	924	3976	22854
$C_{32,5}$	$[32,9,4]$	1	7		49	398
$C_{32,6}$	$[32,15,4]$	1	42	560	5558	20446
$C_{32,8}$	$[32,10,4]$	1	14	4	98	790
$C_{32,9}, C_{32,13}$	$[32,16,4]$	1	56	1180	11144	40774
$C_{32,10}$	$[32,7,8]$	1		4		118
$C_{32,11}$	$[32,10,8]$	1		32	112	734
$C_{32,12}$	$[32,13,4]$	1	28	228	868	5942
$C_{32,14}$	$[32,10,8]$	1		60		902
$C_{32,15}$	$[32,10,4]$	1	8	28	56	838
$C_{32,16}$	$[32,16,4]$	1	120	1820	8008	45638
$C_{32,17}$	$[32,7,4]$	1	1		7	110
$C_{32,18}$	$[32,10,4]$	1	8	7	140	712
$C_{32,19}$	$[32,13,4]$	1	36	196	924	5878
$C_{32,20}$	$[32,10,4]$	1	1	42	63	810
$C_{32,21}$	$[32,16,4]$	1	50	1120	11438	40318

Table 4.3: Parameters and weight distributions of codes $C_{32,1}, C_{32,2}, \ldots, C_{32,21}$ of length 32

By using the standard search algorithm on codes $C_{32,1}, C_{32,2}, \ldots, C_{32,21}$ we obtained

extremal $\mathbb{Z}_4$-codes given in Table 4.4 (Type I), and Table 4.5 (Type II). All extremal $\mathbb{Z}_4$-codes obtained from one binary code given in Table 4.5 have the same Euclidean weight distribution. In that table, in the column E_{16}, the number of codewords of Euclidean weight 16 for each of the codes is given. Codes in Table 4.5 were used for construction of extremal $\mathbb{Z}_4$-codes in [1].

As stated in Proposition 2.1.19, extremal $\mathbb{Z}_4$-codes obtained from $C_{32,1}$ are equivalent and known. Known extremal $\mathbb{Z}_4$-codes of types $4^7 2^{18}$, $4^9 2^{14}$, and $4^{10} 2^{12}$ have residue codes of minimum weight 4 (constructed in [28]) or 12 (constructed in [1]). Extremal $\mathbb{Z}_4$-codes given in Table 4.5, have residue codes of minimum weight 8. Therefore, these codes are not equivalent to the previously known extremal $\mathbb{Z}_4$-codes. These extremal codes can also be found in [2].

The binary code	The number of obtained extremal $\mathbb{Z}_4$-codes	At least inequivalent	The type	The binary residue code
$C_{32,3}$	13	10	$4^{12}2^8$	[32,12,4]
$C_{32,4}$	6	6	$4^{15}2^2$	[32,15,4]
$C_{32,8}$	35	2	$4^{10}2^{12}$	[32,10,4]
$C_{32,12}$	5	5	$4^{13}2^6$	[32,13,4]
$C_{32,15}$	210	2	$4^{10}2^{12}$	[32,10,4]
$C_{32,16}$	272	240	$4^{16}2^0$	[32,16,4]
$C_{32,18}$	44	1	$4^{10}2^{12}$	[32,10,4]
$C_{32,19}$	188	177	$4^{13}2^6$	[32,13,4]

Table 4.4: Extremal Type I $\mathbb{Z}_4$-codes of length 32 from $C_{32,1}, \ldots, C_{32,21}$

The binary code	The number of obtained extremal $\mathbb{Z}_4$-codes	The type	E_{16}	The binary residue code
$C_{32,1}$	118	$4^6 2^{20}$	128216	[32,6,16]
$C_{32,2}$	114	$4^9 2^{14}$	120152	[32,9,8]
$C_{32,7}$	91	$4^9 2^{14}$	120152	[32,9,8]
$C_{32,10}$	296	$4^7 2^{18}$	123608	[32,7,8]
$C_{32,14}$	304	$4^{10} 2^{12}$	119576	[32,10,8]

Table 4.5: Extremal Type II $\mathbb{Z}_4$-codes of length 32 from $C_{32,1},\ldots,C_{32,21}$

As one can observe from Table 4.4 and Table 4.5, there are binary codes from which no extremal $\mathbb{Z}_4$-codes were found with the standard search algorithm. On these codes we applied the modified search algorithm with the random choice of B. We successfully constructed extremal $\mathbb{Z}_4$-codes from all those codes. The summary of the obtained extremal $\mathbb{Z}_4$-codes is given in Table 4.6. The non-equivalence of the codes with the same residue code was determined form their Euclidean weight distribution. The Type II codes, from this table, of types $4^7 2^{18}$, $4^9 2^{14}$, and $4^{10} 2^{20}$ have residue codes with minimum weight 4, and are not equivalent to the extremal $\mathbb{Z}_4$-codes constructed in [1].

The Type II $\mathbb{Z}_4$-code obtained from the code $C_{32,17}$ is of type $4^7 2^{18}$. The binary code $C_{32,17}$ is equivalent to the residue code of the known extremal $\mathbb{Z}_4$-code of type $4^7 2^{18}$ ($B_{32,7}$ from [28]). Also, Type II extremal $\mathbb{Z}_4$-code constructed from $C_{32,17}$, and the code from [28] both have $E_{16} = 125080$. This means that they may be equivalent codes. The Type II $\mathbb{Z}_4$-code obtained from the code $C_{32,5}$ is of type $4^9 2^{14}$. The known extremal $\mathbb{Z}_4$-code of that type from [28] have the residue code that have 4 codewords of Hamming weight 4. Therefore, known code and extremal Type II $\mathbb{Z}_4$-code constructed from $C_{32,5}$ have nonequivalent residue codes. This means that the Type II $\mathbb{Z}_4$-code constructed from $C_{32,5}$ is new. Type II $\mathbb{Z}_4$-codes of type $4^{10} 2^{20}$ are constructed from $C_{32,11}$ and $C_{32,20}$. The code constructed from $C_{32,11}$ is new since its residue code have minimum weight 8. The known extremal $\mathbb{Z}_4$-code of this type from [28] have the residue code $B_{32,10}$, which have 10 codewords of weight 4. Since $C_{32,20}$ have only one codeword of weight 4, $B_{32,10}$

and $C_{32,20}$ are nonequivalent residue codes. Therefore, obtained Type II $\mathbb{Z}_4$-codes of type $4^{10}2^{20}$ are new. We obtained Type II extremal $\mathbb{Z}_4$-codes of type $4^{15}2^2$ from the code $C_{32,6}$. The known extremal $\mathbb{Z}_4$-code of that type (constructed in [28]) has the residue code with 72 codewords of weight 4. Therefore, extremal Type II $\mathbb{Z}_4$-codes of type $4^{15}2^2$ constructed from $C_{32,6}$ are not equivalent to the extremal $\mathbb{Z}_4$-code from [28]. Also, all 355 extremal Type II $\mathbb{Z}_4$-codes of type $4^{15}2^2$ constructed in [12] have residue codes of minimum weight 8. Therefore, extremal Type II $\mathbb{Z}_4$-codes constructed from $C_{32,6}$ are new.

The 54 known extremal Type II $\mathbb{Z}_4$-codes of type 4^{16} constructed in [22] have residue codes of minimum weight 8. Therefore, extremal Type II $\mathbb{Z}_4$-codes of type 4^{16} constructed from $C_{32,9}$, $C_{32,13}$ and $C_{32,21}$ are not equivalent to those codes. Among the 80 Type II extremal $\mathbb{Z}_4$-codes of type 4^{16} constructed in [28], codes denoted by $D_{32,4}$, $D_{32,5}$, $D_{32,6}$ in the same paper, are equivalent to the codes $C_{32,9}$, $C_{32,13}$ and $C_{32,21}$. The known extremal $\mathbb{Z}_4$-code constructed from $D_{32,4}$ have $E_{16} = 110678$, and extremal Type II codes constructed from $C_{32,9}$ have the following values of E_{16}:110756, 110808, 110844, 110708, 111424, 110664,110800, 110980, 110632, 110772, 110948. Therefore, these codes are new. The known extremal $\mathbb{Z}_4$-code constructed from $D_{32,5}$ have $E_{16} = 110984$, and extremal Type II codes obtained from $C_{32,13}$ have the following values of E_{16}: 110784, 110888, 111032, 110440, 111016, 110784, 110536, 110608, 110480, 111240, 111200. So, this codes are also new. Finally, the known extremal $\mathbb{Z}_4$-code constructed from $D_{32,6}$ have $E_{16} = 110928$, and extremal Type II codes constructed from $C_{32,21}$ have one of the following values of E_{16}:110918, 110546, 110574, 110714, 110626, 110374, 110770, 110346. Thus, these codes are also new.

As stated in Section 2.1.1, to the best of our knowledge, extremal Type I $\mathbb{Z}_4$-codes of length 32 were not yet explicitly constructed. Therefore, the codes given in Table 4.4 and Table 4.6 are the first such codes.

The binary code	The number of obtained extremal $\mathbb{Z}_4$-codes	At least non equivalent Type I	At least non equivalent Type II	The type	The binary residue code
$C_{32,5}$	1664	2	1	$4^9 2^{14}$	$[32,9,4]$
$C_{32,6}$	27	0	27	$4^{15}2^2$	$[32,15,4]$
$C_{32,9}$	11	0	11	4^{16}	$[32,16,4]$
$C_{32,11}$	409	0	1	$4^{10}2^{12}$	$[32,10,8]$
$C_{32,13}$	11	0	11	4^{16}	$[32,16,4]$
$C_{32,17}$	4800	2	1	$4^7 2^{18}$	$[32,7,4]$
$C_{32,20}$	1483	7	5	$4^{10}2^{12}$	$[32,10,4]$
$C_{32,21}$	8	0	8	4^{16}	$[32,16,4]$

Table 4.6: Extremal $\mathbb{Z}_4$-codes of length 32 obtained with the modified search algorithm from $C_{32,1},\ldots,C_{32,21}$

Codes of length 40

On the length 40, the construction of Hadamard designs on 39 points by Proposition 3.1.2 is not possible since the order of the starting skew Hadamard matrix should be:

$$39 = 4n - 1 \Rightarrow n = 10.$$

This contradicts the necessary condition given in Theorem 1.2.7. Therefore we used incidence matrices of Hadamard $2-(39,19,9)$ designs, from [15], given in Table A.1. Up to the isomorphism, we obtained 11 designs denoted by $\mathscr{D}_1,\mathscr{D}_2,\ldots,\mathscr{D}_{11}$. The structure of the full automorphism group of these designs is given in Table 4.7.

The size of the group	The structure of the group	Designs
57	$\mathbb{Z}_{19}:\mathbb{Z}_3$	$\mathscr{D}_1$, $\mathscr{D}_2$, $\mathscr{D}_3$, $\mathscr{D}_4$, $\mathscr{D}_5$, $\mathscr{D}_6$, $\mathscr{D}_7$, $\mathscr{D}_8$
171	$\mathbb{Z}_{19}:\mathbb{Z}_9$	$\mathscr{D}_9$, $\mathscr{D}_{10}$, $\mathscr{D}_{11}$

Table 4.7: The structure of the full automorphism groups of $2-(39,19,9)$ designs $\mathscr{D}_1$,$\mathscr{D}_2$, …, $\mathscr{D}_{11}$.

From these designs we constructed the corresponding $3-(40,20,9)$ Hadamard designs $\mathscr{D}_1^*$,$\mathscr{D}_2^*$, ..., $\mathscr{D}_{11}^*$, from which we constructed 3 nonequivalent doubly-even binary codes $C_{40,1}$, $C_{40,2}$, $C_{40,3}$. The weight distribution of these codes is given in Table 4.8.

The Code	$[n,k,d]$	0	4	8	12	16	20
$C_{40,1}, C_{40,2}$	$[40,20,8]$	1	0	285	21280	239970	525504
$C_{40,3}$	$[40,20,4]$	1	190	4845	38760	125970	709044

Table 4.8: Parameters and the weight distribution of binary codes $C_{40,1}$, $C_{40,2}$, $C_{40,3}$ of length 40.

The codes $C_{40,1}$, $C_{40,2}$, $C_{40,3}$ are self-dual (so, they are given in [4]). All three codes $C_{40,1}$, $C_{40,2}$, $C_{40,3}$ satisfy Lemma 2.1.14. As stated in Section 2.1.1, in [32], from each of 94343 binary self-dual codes of length 40, one extremal Type II $\mathbb{Z}_4$-code is constructed. We reconstructed these known extremal $\mathbb{Z}_4$-codes and identified those codes that have residue code equivalent to $C_{40,1}$, $C_{40,2}$, $C_{40,3}$. In Table 4.9 for these known extremal $\mathbb{Z}_4$-codes, the number of codewords with Euclidean weight 16 is given. In Table 4.10 we give the summary of results on extremal $\mathbb{Z}_4$-codes that we constructed from codes $C_{40,1}$, $C_{40,2}$, $C_{40,3}$. All of the extremal $\mathbb{Z}_4$-codes constructed from $C_{40,1}$ and $C_{40,2}$ are Type II $\mathbb{Z}_4$-codes. From $C_{40,3}$ we constructed 3 codes of Type II, and 4090 codes of Type I. Extremal $\mathbb{Z}_4$-codes constructed from $C_{40,1}$ have one of the following number of codewords of the Euclidean weight 16: $34552, 34540, 34584, 34596, 34794, 34850, 34992$. The extremal $\mathbb{Z}_4$-code constructed from $C_{40,2}$ have 34570 codewords of the Euclidean weight 16. Type II $\mathbb{Z}_4$-codes constructed from $C_{40,3}$ have one of the following number of codewords of minimum weight 16: $24030, 25214, 26014$. Therefore, by the comparison of the number of codewords of Euclidean weight 16 in each constructed code with numbers given in Table 4.9, we conclude that all constructed codes of Type II are new.

Also, as it is stated in section 2.1.1, in [9], Type I extremal $\mathbb{Z}_4$-codes of type 4^{20} were constructed from the self-dual codes of length 40 and minimum weight 8. Since $C_{40,3}$ have the minimum weight 4, all of 4090 Type I codes constructed from $C_{40,3}$ are new.

The binary code	Number of codewords of Euclidean weight 16
$C_{40,1}$	34730
$C_{40,2}$	34694
$C_{40,3}$	25918

Table 4.9: Known extremal $\mathbb{Z}_4$-codes with residue codes $C_{40,1}$, $C_{40,2}$, $C_{40,3}$

The binary code	The number of obtained extremal $\mathbb{Z}_4$-codes	At least non equivalent
$C_{40,1}$	7	7
$C_{40,2}$	1	1
$C_{40,3}$	4194	4093

Table 4.10: New extremal $\mathbb{Z}_4$-codes of length 40 obtained with the modified search algorithm from $C_{40,1}$, $C_{40,2}$, $C_{40,3}$

We also used the residue codes of the known extremal $\mathbb{Z}_4$-codes from [3] and [28]. We were specially interested in finding the new extremal $\mathbb{Z}_4$-codes of types $4^7 2^{26}$, $4^{10}2^{20}$, $4^{11}2^{18}$, $4^{15}2^{10}$, and $4^{16}2^8$, since the small number of the existing extremal codes is known (see Section 2.1.1).

In Table 4.11, the weight distributions of the starting binary residue codes $C'_{40,1}$, $C'_{40,2}$, ..., $C'_{40,11}$ are given together with the reference where the original extremal $\mathbb{Z}_4$-codes were constructed. To find the new extremal $\mathbb{Z}_4$-codes we used the modified search algorithm with the random choice of B. We denote the already known extremal $\mathbb{Z}_4$-codes constructed in [3] and [28] with $\widetilde{C'}_{40,1}, \widetilde{C'}_{40,2}, \ldots, \widetilde{C'}_{40,11}$.

Code	Reference	$[n,k,d]$	0	4	8	12	16	20
$C'_{40,1}$	[28]	$[40,7,12]$	1			1	11	102
$C'_{40,2}$	[28]	$[40,7,16]$	1				15	96
$C'_{40,3}$	[28]	$[40,10,4]$	1	6	1	10	150	688
$C'_{40,4}$	[3]	$[40,10,12]$	1			18	223	540
$C'_{40,5}$	[28]	$[40,11,4]$	1	10	6	22	313	1344
$C'_{40,6}$	[3]	$[40,11,12]$	1			34	479	1020
$C'_{40,7}$	[3]	$[40,11,12]$	1			42	447	1068
$C'_{40,8}$	[28]	$[40,15,4]$	1	37	175	688	5296	20374
$C'_{40,9}$	[3]	$[40,15,8]$	1		10	634	7589	16300
$C'_{40,10}$	[3]	$[40,15,8]$	1		6	658	7529	16380
$C'_{40,11}$	[28]	$[40,16,4]$	1	47	313	1548	10694	40330

Table 4.11: Parameters and weight distributions of residue codes of the known extremal $\mathbb{Z}_4$-codes of length 40

In order to determine whether constructed Type II $\mathbb{Z}_4$-codes are equivalent to the already known Type II $\mathbb{Z}_4$-codes, we compared their Euclidean weight distributions. Euclidean weight distributions of the known Type II $\mathbb{Z}_4$-codes $\widetilde{C'}_{40,2}, \widetilde{C'}_{40,3}, \ldots, \widetilde{C'}_{40,11}$ are given in Table 4.13. The Euclidean weight distribution of the code $\widetilde{C'}_{40,1}$ is omitted since the generator matrix from [28, p.15] (matrix A) does not give the extremal $\mathbb{Z}_4$-code.

The Code	Euclidean weight enumerator
$\widetilde{C}'_{40,2}$	$1+16870x^{16}+39688320x^{24}+2429343405x^{32}+42212779392x^{40}+164801312200x^{48}+345272568960x^{56}+338185000530x^{64}+160981226880x^{72}+38308432548x^{80}+5635777920x^{88}+1176862290x^{96}+366779520x^{104}+87338440x^{112}+13235584x^{120}+1203885x^{128}+59520x^{136}+1510x^{144}+x^{160}$
$\widetilde{C}'_{40,3}$	$1+21758x^{16}+38847568x^{24}+2509071709x^{32}+39073052496x^{40}+171518064728x^{48}+344199907472x^{56}+333515582530x^{64}+161003303056x^{72}+40401241060x^{80}+6130536176x^{88}+914213730x^{96}+174965232x^{104}+29106744x^{112}+3405872x^{120}+290045x^{128}+16944x^{136}+654x^{144}+x^{160}$
$\widetilde{C}'_{40,4}$	$1+32566x^{16}+37785472x^{24}+2568463149x^{32}+37609444864x^{40}+173720402952x^{48}+346388958336x^{56}+329044725458x^{64}+160738711808x^{72}+42377857284x^{80}+6325930112x^{88}+618755026x^{96}+67322880x^{104}+11421960x^{112}+1657728x^{120}+150573x^{128}+7424x^{136}+182x^{144}+x^{160}$
$\widetilde{C}'_{40,5}$	$1+22558x^{16}+38737280x^{24}+2521538909x^{32}+38634022880x^{40}+172365506360x^{48}+344327711552x^{56}+332610202882x^{64}+161003382048x^{72}+40777199172x^{80}+6202617088x^{88}+861277666x^{96}+145043488x^{104}+21787224x^{112}+2369088x^{120}+197053x^{128}+12000x^{136}+526x^{144}+x^{160}$
$\widetilde{C}'_{40,6}$	$1+34574x^{16}+37647488x^{24}+2578608621x^{32}+37262591168x^{40}+174364802408x^{48}+346560136704x^{56}+328290267282x^{64}+160685141312x^{72}+42737367636x^{80}+6374556288x^{88}+569181714x^{96}+44439360x^{104}+5944040x^{112}+830208x^{120}+75117x^{128}+3776x^{136}+78x^{144}+x^{160}$
$\widetilde{C}'_{40,7}$	$1+33406x^{16}+37646112x^{24}+2578273965x^{32}+37294058496x^{40}+174352264264x^{48}+346399629024x^{56}+328468693650x^{64}+160747916352x^{72}+42617396820x^{80}+6375454944x^{88}+586281426x^{96}+47023488x^{104}+6045832x^{112}+830752x^{120}+75501x^{128}+3648x^{136}+94x^{144}+x^{160}$
$\widetilde{C}'_{40,8}$	$1+26402x^{16}+38052240x^{24}+2565874765x^{32}+37587377808x^{40}+174225886616x^{48}+344987924048x^{56}+330164667570x^{64}+161029812048x^{72}+41754580556x^{80}+6358235568x^{88}+717685170x^{96}+74108080x^{104}+6846488x^{112}+510576x^{120}+37197x^{128}+2416x^{136}+226x^{144}+x^{160}$
$\widetilde{C}'_{40,9}$	$1+34657x^{16}+37536172x^{24}+2587364639x^{32}+36984523448x^{40}+174941725730x^{48}+346506811348x^{56}+327834443124x^{64}+160708302096x^{72}+42918028796x^{80}+6419695764x^{88}+545067032x^{96}+26988984x^{104}+1040638x^{112}+60268x^{120}+4819x^{128}+256x^{136}+3x^{144}+x^{160}$
$\widetilde{C}'_{40,10}$	$1+34935x^{16}+37526464x^{24}+2587484969x^{32}+36983363520x^{40}+174945596152x^{48}+346501952032x^{56}+327834384942x^{64}+160714043360x^{72}+42913291758x^{80}+6420260096x^{88}+545790502x^{96}+26829184x^{104}+1006624x^{112}+58144x^{120}+4865x^{128}+224x^{136}+3x^{144}+x^{160}$
$\widetilde{C}'_{40,11}$	$1+27298x^{16}+37891608x^{24}+2572582549x^{32}+37452837944x^{40}+174441135808x^{48}+345136040376x^{56}+329804112474x^{64}+161014813400x^{72}+41911514068x^{80}+6376040904x^{88}+694147082x^{96}+64878888x^{104}+5246992x^{112}+333928x^{120}+22789x^{128}+1480x^{136}+186x^{144}+x^{160}$

Table 4.13: Euclidean weight enumerators of extremal $\mathbb{Z}_4$-codes $\widetilde{C}'_{40,2},\widetilde{C}'_{40,3}\ldots,\widetilde{C}'_{40,11}$

The information about constructed extremal $\mathbb{Z}_4$-codes (of Type I and Type II) is summarized in Table 4.14. The Euclidean weight distribution of all constructed extremal $\mathbb{Z}_4$-codes from $\widetilde{C}'_{40,1},\widetilde{C}'_{40,2},\ldots,\widetilde{C}'_{40,11}$ are given in Table 4.15. By the comparison of Euclidean weight distributions in Table 4.13 and Table 4.15 we can conclude that all extremal Type II $\mathbb{Z}_4$-codes, except one constructed from $C'_{40,6}$, are new.

The binary code	The type	The number of obtained extremal $\mathbb{Z}_4$-codes	At least non equivalent Type I	At least non equivalent Type II
$C'_{40,1}$	$4^7 2^{26}$	488	4	1
$C'_{40,2}$	$4^7 2^{26}$	959	3	0
$C'_{40,3}$	$4^{10} 2^{20}$	17	6	0
$C'_{40,4}$	$4^{10} 2^{20}$	129	3	3
$C'_{40,5}$	$4^{11} 2^{18}$	7	7	0
$C'_{40,6}$	$4^{11} 2^{18}$	31	0	5
$C'_{40,7}$	$4^{11} 2^{18}$	13	0	4
$C'_{40,8}$	$4^{15} 2^{10}$	37	17	0
$C'_{40,9}$	$4^{15} 2^{10}$	7	0	7
$C'_{40,10}$	$4^{15} 2^{10}$	9	0	9
$C'_{40,11}$	$4^{16} 2^{8}$	104	14	1

Table 4.14: Extremal $\mathbb{Z}_4$-codes of length 40 obtained with the modified search algorithm from $C'_{40,1}, \ldots, C'_{40,11}$

The Residue Code	Euclidean weight enumerator
$C'_{40,1}$	$1 + 8454x^{16} + 999424x^{20} + 19376768x^{24} + 199475200x^{28} + 1203405741x^{32} + 5269733376x^{36} + 25377213568x^{40} + 42019012608x^{44} + 816174935576x^{48} + 133349277696x^{52} + 169562209408x^{56} + 190736072704x^{60} + 166235588434x^{64} + 129528987648x^{68} + 80149538432x^{72} + 41936388096x^{76} + 20152183908x^{80} + 6279413760x^{84} + 3837755264x^{88} + 423018496x^{92} + 1128754514x^{96} + 13287424x^{100} + 370247040x^{104} + 147456x^{108} + 87536712x^{112} + 13236608x^{120} + 1204653x^{128} + 59264x^{136} + 1542x^{144} + x^{160}$
	$1 + 10502x^{16} + 1015808x^{20} + 19155584x^{24} + 199393280x^{28} + 1213371309x^{32} + 5206605824x^{36} + 25591000192x^{40} + 41560276992x^{44} + 82241885768x^{48} + 132929224704x^{52} + 169319021696x^{56} + 191734284288x^{60} + 164964771666x^{64} + 130382364672x^{68} + 80040044160x^{72} + 41505193984x^{76} + 20705272932x^{80} + 5879726080x^{84} + 4039221120x^{88} + 348422144x^{92} + 1149205842x^{96} + 9207808x^{100} + 370812288x^{104} + 98304x^{108} + 87538760x^{112} + 13236608x^{120} + 1204653x^{128} + 59264x^{136} + 1542x^{144} + x^{160}$
	$1 + 14598x^{16} + 1048576x^{20} + 18713216x^{24} + 199229440x^{28} + 1233302445x^{32} + 5080350720x^{36} + 26018573440x^{40} + 40642805760x^{44} + 83490670152x^{48} + 132089118720x^{52} + 168832646272x^{56} + 193730707456x^{60} + 162423138130x^{64} + 132089118720x^{68} + 79821055616x^{72} + 40642805760x^{76} + 21811450980x^{80} + 5080350720x^{84} + 4442152832x^{88} + 199229440x^{92} + 1190108498x^{96} + 1048576x^{100} + 371942784x^{104} + 87542856x^{112} + 13236608x^{120} + 1204653x^{128} + 59264x^{136} + 1542x^{144} + x^{160}$
	$1 + 6406x^{16} + 983040x^{20} + 19597952x^{24} + 199557120x^{28} + 1193440173x^{32} + 5332860928x^{36} + 25163426944x^{40} + 42477748224x^{44} + 80993101384x^{48} + 133769330688x^{52} + 169805397120x^{56} + 189737861120x^{60} + 167506405202x^{64} + 128675610624x^{68} + 80259032704x^{72} + 42367582208x^{76} + 19599094884x^{80} + 6679101440x^{84} + 3636289408x^{88} + 497614848x^{92} + 1108303186x^{96} + 17367040x^{100} + 369681792x^{104} + 196608x^{108} + 87534664x^{112} + 13236608x^{120} + 1204653x^{128} + 59264x^{136} + 1542x^{144} + x^{160}$
	$1 + 14598x^{16} + 39684736x^{24} + 2428679085x^{32} + 42275695744x^{40} + 164776281672x^{48} + 344951471232x^{56} + 338541963090x^{64} + 161106667136x^{72} + 38068573284x^{80} + 5637529472x^{88} + 1211080018x^{96} + 371942784x^{104} + 87542856x^{112} + 13236608x^{120} + 1204653x^{128} + 59264x^{136} + 1542x^{144} + x^{160}$

Table 4.15: Euclidean weight enumerators of the obtained extremal $\mathbb{Z}_4$-codes of length 40

The Residue Code	Euclidean weight enumerator
$C'_{40,2}$	$1 + 8678x^{16} + 983040x^{20} + 19601536x^{24} + 199557120x^{28} + 1194104493x^{32} + 5332860928x^{36} + 25100510592x^{40} + 42477748224x^{44} + 81018131912x^{48} + 133769330688x^{52} + 170126494848x^{56} + 189737861120x^{60} + 167149442642x^{64} + 128675610624x^{68} + 80133592448x^{72} + 42367582208x^{76} + 19838954148x^{80} + 6679101440x^{84} + 3634537856x^{88} + 497614848x^{92} + 1074085458x^{96} + 17367040x^{100} + 364518528x^{104} + 196608x^{108} + 87330248x^{112} + 13235584x^{120} + 1203885x^{128} + 59520x^{136} + 1510x^{144} + x^{160}$
	$1 + 6630x^{16} + 966656x^{20} + 19822720x^{24} + 199639040x^{28} + 1184138925x^{32} + 5395988480x^{36} + 24886723968x^{40} + 42936483840x^{44} + 80393739720x^{48} + 134189383680x^{52} + 170369682560x^{56} + 188739649536x^{60} + 168420259410x^{64} + 127822233600x^{68} + 80243086720x^{72} + 42798776320x^{76} + 19285865124x^{80} + 7078789120x^{84} + 3433072000x^{88} + 572211200x^{92} + 1053634130x^{96} + 21446656x^{100} + 363953280x^{104} + 245760x^{108} + 87328200x^{112} + 13235584x^{120} + 1203885x^{128} + 59520x^{136} + 1510x^{144} + x^{160}$
	$1 + 10726x^{16} + 999424x^{20} + 19380352x^{24} + 199475200x^{28} + 1204070061x^{32} + 5269733376x^{36} + 25314297216x^{40} + 42019012608x^{44} + 81642524104x^{48} + 133349277696x^{52} + 169883307136x^{56} + 190736072704x^{60} + 165878625874x^{64} + 129528987648x^{68} + 80024098176x^{72} + 41936388096x^{76} + 20392043172x^{80} + 6279413760x^{84} + 3836003712x^{88} + 423018496x^{92} + 1094536786x^{96} + 13287424x^{100} + 365083776x^{104} + 147456x^{108} + 87332296x^{112} + 13235584x^{120} + 1203885x^{128} + 59520x^{136} + 1510x^{144} + x^{160}$
$C'_{40,3}$	$1 + 11374x^{16} + 950272x^{20} + 19049344x^{24} + 200736768x^{28} + 1250765581x^{32} + 5462327296x^{36} + 21537877824x^{40} + 43379064832x^{44} + 86502960392x^{48} + 134614417408x^{52} + 169502446848x^{56} + 187776499712x^{60} + 165032278834x^{64} + 126931501056x^{68} + 80397955008x^{72} + 43229380608x^{76} + 20811531060x^{80} + 7491846144x^{84} + 3723905920x^{88} + 644284416x^{92} + 772285298x^{96} + 24543232x^{100} + 171923136x^{104} + 262144x^{108} + 29108936x^{112} + 3406848x^{120} + 289869x^{128} + 16960x^{136} + 654x^{144} + x^{160}$
	$1 + 12670x^{16} + 944640x^{20} + 19138000x^{24} + 199587840x^{28} + 1258679389x^{32} + 5429892096x^{36} + 21632753744x^{40} + 43175335936x^{44} + 86801660376x^{48} + 134370268672x^{52} + 169456571664x^{56} + 188234657792x^{60} + 164345046594x^{64} + 127447832576x^{68} + 80301819536x^{72} + 42993479680x^{76} + 21132630884x^{80} + 7262680576x^{84} + 3827208560x^{88} + 615225344x^{92} + 776264290x^{96} + 25498624x^{100} + 171242224x^{104} + 407552x^{108} + 29072568x^{112} + 2560x^{116} + 3405744x^{120} + 290045x^{128} + 16944x^{136} + 654x^{144} + x^{160}$
	$1 + 12542x^{16} + 958976x^{20} + 18943056x^{24} + 200660992x^{28} + 1255802205x^{32} + 5432787968x^{36} + 21637017424x^{40} + 43158910976x^{44} + 86818034776x^{48} + 134378200576x^{52} + 169421346960x^{56} + 188264460288x^{60} + 164351937602x^{64} + 127413860352x^{68} + 80327663760x^{72} + 42993463296x^{76} + 21118080996x^{80} + 7274200576x^{84} + 3824350960x^{88} + 613447680x^{92} + 778275682x^{96} + 24527872x^{100} + 171547120x^{104} + 333824x^{108} + 29087288x^{112} + 512x^{116} + 3405872x^{120} + 290045x^{128} + 16944x^{136} + 654x^{144} + x^{160}$
	$1 + 15614x^{16} + 999424x^{20} + 18539600x^{24} + 199475200x^{28} + 1283798365x^{32} + 5269733376x^{36} + 22174570320x^{40} + 42019012608x^{44} + 88359276632x^{48} + 133349277696x^{52} + 168810645648x^{56} + 190736072704x^{60} + 161209207874x^{64} + 129528987648x^{68} + 80046174352x^{72} + 41936388096x^{76} + 22484851684x^{80} + 6279413760x^{84} + 4330761968x^{88} + 423018496x^{92} + 831888226x^{96} + 13287424x^{100} + 173269488x^{104} + 147456x^{108} + 29100600x^{112} + 3405872x^{120} + 290045x^{128} + 16944x^{136} + 654x^{144} + x^{160}$
	$1 + 12670x^{16} + 977408x^{20} + 18875856x^{24} + 199489536x^{28} + 1268903005x^{32} + 5366420480x^{36} + 21846138960x^{40} + 42717992960x^{44} + 87425563096x^{48} + 133948020224x^{52} + 169215923472x^{56} + 189234016256x^{60} + 163070502466x^{64} + 128302356480x^{68} + 80194864784x^{72} + 42560090112x^{76} + 21685230436x^{80} + 6864385536x^{84} + 4028273008x^{88} + 540284928x^{92} + 796973666x^{96} + 21402624x^{100} + 171766512x^{104} + 374784x^{108} + 29072568x^{112} + 2560x^{116} + 3405744x^{120} + 290045x^{128} + 16944x^{136} + 654x^{144} + x^{160}$
	$1 + 13694x^{16} + 969728x^{20} + 18836944x^{24} + 200511488x^{28} + 1260837981x^{32} + 5403219968x^{36} + 21736159312x^{40} + 42940420096x^{44} + 87133105624x^{48} + 134136837120x^{52} + 169340249360x^{56} + 188758827008x^{60} + 163671598658x^{64} + 127893983232x^{68} + 80257365648x^{72} + 42754777088x^{76} + 21424637796x^{80} + 7059796992x^{84} + 3924793712x^{88} + 581521408x^{92} + 784263778x^{96} + 24483840x^{100} + 171174640x^{104} + 462848x^{108} + 29063352x^{112} + 3072x^{116} + 3405744x^{120} + 290045x^{128} + 16944x^{136} + 654x^{144} + x^{160}$
$C'_{40,4}$	$1 + 32822x^{16} + 37782144x^{24} + 2568483117x^{32} + 37609371648x^{40} + 173720585992x^{48} + 346388628864x^{56} + 329045164754x^{64} + 160738272512x^{72} + 42378186756x^{80} + 6325747072x^{88} + 618828242x^{96} + 67302912x^{104} + 11425288x^{112} + 1657472x^{120} + 150573x^{128} + 7424x^{136} + 182x^{144} + x^{160}$
	$1 + 17782x^{16} + 928768x^{20} + 18469440x^{24} + 199733248x^{28} + 1298451117x^{32} + 5555795968x^{36} + 19740971648x^{40} + 44094271488x^{44} + 87755198664x^{48} + 135207941120x^{52} + 172133776576x^{56} + 186239836160x^{60} + 162412950482x^{64} + 125741666304x^{68} + 80258073600x^{72} + 43854127104x^{76} + 22002971844x^{80} + 8063210496x^{84} + 3619102656x^{88} + 764145664x^{92} + 440217426x^{96} + 33634304x^{100} + 62411904x^{104} + 520192x^{108} + 11385416x^{112} + 3072x^{116} + 1657152x^{120} + 150573x^{128} + 7424x^{136} + 182x^{144} + x^{160}$
	$1 + 17526x^{16} + 928768x^{20} + 18472768x^{24} + 199733248x^{28} + 1298431149x^{32} + 5555795968x^{36} + 19741044864x^{40} + 44094271488x^{44} + 87755015624x^{48} + 135207941120x^{52} + 172134106048x^{56} + 186239836160x^{60} + 162412511186x^{64} + 125741666304x^{68} + 80258512896x^{72} + 43854127104x^{76} + 22002642372x^{80} + 8063210496x^{84} + 3619285696x^{88} + 764145664x^{92} + 440144210x^{96} + 33634304x^{100} + 62431872x^{104} + 520192x^{108} + 11382088x^{112} + 3072x^{116} + 1657408x^{120} + 150573x^{128} + 7424x^{136} + 182x^{144} + x^{160}$
	$1 + 17654x^{16} + 928768x^{20} + 18471104x^{24} + 199733248x^{28} + 1298441133x^{32} + 5555795968x^{36} + 19741008256x^{40} + 44094271488x^{44} + 87755107144x^{48} + 135207941120x^{52} + 172133941312x^{56} + 186239836160x^{60} + 162412730834x^{64} + 125741666304x^{68} + 80258293248x^{72} + 43854127104x^{76} + 22002807108x^{80} + 8063210496x^{84} + 3619194176x^{88} + 764145664x^{92} + 440180818x^{96} + 33634304x^{100} + 62421888x^{104} + 520192x^{108} + 11383752x^{112} + 3072x^{116} + 1657280x^{120} + 150573x^{128} + 7424x^{136} + 182x^{144} + x^{160}$

Table 4.15: Euclidean weight enumerators of the obtained extremal $\mathbb{Z}_4$-codes of length 40

The Residue Code	Euclidean weight enumerator
	$1 + 32694x^{16} + 37783808x^{24} + 2568473133x^{32} + 37609408256x^{40} + 173720494472x^{48} + 346388793600x^{56} + 329044945106x^{64} + 160738492160x^{72} + 42378022020x^{80} + 6325838592x^{88} + 618791634x^{96} + 67312896x^{104} + 11423624x^{112} + 1657600x^{120} + 150573x^{128} + 7424x^{136} + 182x^{144} + x^{160}$
	$1 + 32950x^{16} + 37780480x^{24} + 2568493101x^{32} + 37609335040x^{40} + 173720677512x^{48} + 346388464128x^{56} + 329045384402x^{64} + 160738052864x^{72} + 42378351492x^{80} + 6325655552x^{88} + 618864850x^{96} + 67292928x^{104} + 11426952x^{112} + 1657344x^{120} + 150573x^{128} + 7424x^{136} + 182x^{144} + x^{160}$
$C'_{40,5}$	$1 + 13630x^{16} + 967680x^{20} + 18802464x^{24} + 200077312x^{28} + 1275002365x^{32} + 5401643008x^{36} + 21294200832x^{40} + 42946834432x^{44} + 87972848088x^{48} + 134139493376x^{52} + 169487831648x^{56} + 188734889984x^{60} + 162757459746x^{64} + 127914897408x^{68} + 80249595584x^{72} + 42757840896x^{76} + 21805974436x^{80} + 7049870336x^{84} + 3997996000x^{88} + 583872512x^{92} + 730760642x^{96} + 25004032x^{100} + 141018752x^{104} + 421888x^{108} + 21732664x^{112} + 1024x^{116} + 2367136x^{120} + 197405x^{128} + 11968x^{136} + 526x^{144} + x^{160}$
	$1 + 12590x^{16} + 971776x^{20} + 18823952x^{24} + 199929856x^{28} + 1274113069x^{32} + 5401585664x^{36} + 21298209200x^{40} + 42950160384x^{44} + 87967468296x^{48} + 134129200128x^{52} + 169483871888x^{56} + 188747702272x^{60} + 162776894098x^{64} + 127910424576x^{68} + 80227358384x^{72} + 42752303104x^{76} + 21815942260x^{80} + 7056354304x^{84} + 3998367408x^{88} + 581693440x^{92} + 729086802x^{96} + 24946688x^{100} + 141314192x^{104} + 536576x^{108} + 21771080x^{112} + 5120x^{116} + 2371504x^{120} + 196589x^{128} + 12048x^{136} + 526x^{144} + x^{160}$
	$1 + 12846x^{16} + 958976x^{20} + 18961104x^{24} + 200341504x^{28} + 1263499437x^{32} + 5472363520x^{36} + 21060086512x^{40} + 43430123520x^{44} + 87364782856x^{48} + 134468025856x^{52} + 169832727632x^{56} + 187739992064x^{60} + 163902241810x^{64} + 127203145728x^{68} + 80320292848x^{72} + 43119472640x^{76} + 21337536884x^{80} + 7412563456x^{84} + 3794540528x^{88} + 673323008x^{92} + 698165970x^{96} + 34579456x^{100} + 138718800x^{104} + 915456x^{108} + 21669192x^{112} + 8704x^{116} + 2367728x^{120} + 197229x^{128} + 11984x^{136} + 526x^{144} + x^{160}$
	$1 + 10670x^{16} + 948224x^{20} + 19145040x^{24} + 199983104x^{28} + 1259903405x^{32} + 5500325888x^{36} + 20959203824x^{40} + 43656691712x^{44} + 87040554376x^{48} + 134706898944x^{52} + 169931806672x^{56} + 187228094464x^{60} + 164580717074x^{64} + 126741700608x^{68} + 80375149040x^{72} + 43358453760x^{76} + 21039648244x^{80} + 7620282368x^{84} + 3695437168x^{88} + 706510848x^{92} + 690996178x^{96} + 35115008x^{100} + 138972496x^{104} + 802816x^{108} + 21692104x^{112} + 6144x^{116} + 2367856x^{120} + 197229x^{128} + 11984x^{136} + 526x^{144} + x^{160}$
	$1 + 10542x^{16} + 955392x^{20} + 19045136x^{24} + 200011776x^{28} + 1264147501x^{32} + 5464713216x^{36} + 21084422576x^{40} + 43408896000x^{44} + 87343076104x^{48} + 134549253120x^{52} + 169727059600x^{56} + 187749490688x^{60} + 164047710866x^{64} + 127057047552x^{68} + 80336852656x^{72} + 43183497216x^{76} + 21262853236x^{80} + 7456041984x^{84} + 3796901552x^{88} + 656289792x^{92} + 708635474x^{96} + 29026304x^{100} + 140748944x^{104} + 585728x^{108} + 21769032x^{112} + 5120x^{116} + 2371504x^{120} + 196589x^{128} + 12048x^{136} + 526x^{144} + x^{160}$
	$1 + 11582x^{16} + 951296x^{20} + 19023648x^{24} + 200159232x^{28} + 1265036797x^{32} + 5464770560x^{36} + 21080414208x^{40} + 43405570048x^{44} + 87348455896x^{48} + 134559546368x^{52} + 169731019360x^{56} + 187736678400x^{60} + 164028276514x^{64} + 127061520384x^{68} + 80359089856x^{72} + 43189035008x^{76} + 21252885412x^{80} + 7449558016x^{84} + 3796530144x^{88} + 658468864x^{92} + 710309314x^{96} + 29083648x^{100} + 140453504x^{104} + 471040x^{108} + 21730616x^{112} + 1024x^{116} + 2367136x^{120} + 197405x^{128} + 11968x^{136} + 526x^{144} + x^{160}$
	$1 + 15806x^{16} + 988160x^{20} + 18523104x^{24} + 200355840x^{28} + 1282895485x^{32} + 5340063744x^{36} + 21512550848x^{40} + 42483347456x^{44} + 88602473048x^{48} + 133711637504x^{52} + 169224501792x^{56} + 189763444736x^{60} + 161501047714x^{64} + 128745123840x^{68} + 80140892032x^{72} + 42320814080x^{76} + 22357080356x^{80} + 6663351296x^{84} + 4198273568x^{88} + 505835520x^{92} + 751332162x^{96} + 20375552x^{100} + 141886400x^{104} + 471040x^{108} + 21761720x^{112} + 5120x^{116} + 2370528x^{120} + 196765x^{128} + 12032x^{136} + 526x^{144} + x^{160}$
$C'_{40,6}$	$1 + 34446x^{16} + 37649152x^{24} + 2578598637x^{32} + 37262627776x^{40} + 174364710888x^{48} + 346560301440x^{56} + 328290047634x^{64} + 160685360960x^{72} + 42737202900x^{80} + 6374647808x^{88} + 569145106x^{96} + 44449344x^{104} + 5942376x^{112} + 830336x^{120} + 75117x^{128} + 3776x^{136} + 78x^{144} + x^{160}$
	$1 + 34574x^{16} + 37647488x^{24} + 2578608621x^{32} + 37262591168x^{40} + 174364802408x^{48} + 346560136704x^{56} + 328290267282x^{64} + 160685141312x^{72} + 42737367636x^{80} + 6374556288x^{88} + 569181714x^{96} + 44439360x^{104} + 5944040x^{112} + 830208x^{120} + 75117x^{128} + 3776x^{136} + 78x^{144} + x^{160}$
	$1 + 34702x^{16} + 37645824x^{24} + 2578618605x^{32} + 37262554560x^{40} + 174364893928x^{48} + 346559971968x^{56} + 328290486930x^{64} + 160684921664x^{72} + 42737532372x^{80} + 6374464768x^{88} + 569218322x^{96} + 44429376x^{104} + 5945704x^{112} + 830080x^{120} + 75117x^{128} + 3776x^{136} + 78x^{144} + x^{160}$
	$1 + 34638x^{16} + 37646656x^{24} + 2578613613x^{32} + 37262572864x^{40} + 174364848168x^{48} + 346560054336x^{56} + 328290377106x^{64} + 160685031488x^{72} + 42737450004x^{80} + 6374510528x^{88} + 569200018x^{96} + 44434368x^{104} + 5944872x^{112} + 830144x^{120} + 75117x^{128} + 3776x^{136} + 78x^{144} + x^{160}$
	$1 + 34510x^{16} + 37648320x^{24} + 2578603629x^{32} + 37262609472x^{40} + 174364756648x^{48} + 346560219072x^{56} + 328290157458x^{64} + 160685251136x^{72} + 42737285268x^{80} + 6374602048x^{88} + 569163410x^{96} + 44444352x^{104} + 5943208x^{112} + 830272x^{120} + 75117x^{128} + 3776x^{136} + 78x^{144} + x^{160}$
$C'_{40,7}$	$1 + 33406x^{16} + 37646112x^{24} + 2578273965x^{32} + 37294058496x^{40} + 174352264264x^{48} + 346399629024x^{56} + 328468693650x^{64} + 160747916352x^{72} + 42617396820x^{80} + 6375454944x^{88} + 586281426x^{96} + 47023488x^{104} + 6045832x^{112} + 830752x^{120} + 75501x^{128} + 3648x^{136} + 94x^{144} + x^{160}$

Table 4.15: Euclidean weight enumerators of the obtained extremal $\mathbb{Z}_4$-codes of length 40

The Residue Code	Euclidean weight enumerator
	$1 + 33726x^{16} + 37641952x^{24} + 2578298925x^{32} + 37293966976x^{40} + 174352493064x^{48} + 346399217184x^{56} + 328469242770x^{64} + 160747367232x^{72} + 42617808660x^{80} + 6375226144x^{88} + 586372946x^{96} + 46998528x^{104} + 6049992x^{112} + 830432x^{120} + 75501x^{128} + 3648x^{136} + 94x^{144} + x^{160}$
	$1 + 33214x^{16} + 37648608x^{24} + 2578258989x^{32} + 37294113408x^{40} + 174352126984x^{48} + 346399876128x^{56} + 328468364178x^{64} + 160748245824x^{72} + 42617149716x^{80} + 6375592224x^{88} + 586226514x^{96} + 47038464x^{104} + 6043336x^{112} + 830944x^{120} + 75501x^{128} + 3648x^{136} + 94x^{144} + x^{160}$
	$1 + 33342x^{16} + 37646944x^{24} + 2578268973x^{32} + 37294076800x^{40} + 174352218504x^{48} + 346399711392x^{56} + 328468583826x^{64} + 160748026176x^{72} + 42617314452x^{80} + 6375500704x^{88} + 586263122x^{96} + 47028480x^{104} + 6045000x^{112} + 830816x^{120} + 75501x^{128} + 3648x^{136} + 94x^{144} + x^{160}$
$C'_{40,8}$	$1 + 14626x^{16} + 943888x^{20} + 18755040x^{24} + 199560480x^{28} + 1276268077x^{32} + 5714343984x^{36} + 19184375744x^{40} + 45023930176x^{44} + 87532630360x^{48} + 134967526544x^{52} + 171919287200x^{56} + 184909985504x^{60} + 164334863314x^{64} + 125522403120x^{68} + 80485365760x^{72} + 43990717824x^{76} + 21171075276x^{80} + 8351820464x^{84} + 3362061088x^{88} + 978634976x^{92} + 416639954x^{96} + 88586256x^{100} + 48927424x^{104} + 6930240x^{108} + 5084120x^{112} + 416176x^{116} + 427872x^{120} + 14112x^{124} + 35373x^{128} + 144x^{132} + 2432x^{136} + 226x^{144} + x^{160}$
	$1 + 14114x^{16} + 930736x^{20} + 19128480x^{24} + 197508096x^{28} + 1272911949x^{32} + 5804003824x^{36} + 18834123872x^{40} + 45630720384x^{44} + 87051733528x^{48} + 134924651632x^{52} + 172531081120x^{56} + 184013436032x^{60} + 165085168178x^{64} + 125166285744x^{68} + 80496357984x^{72} + 44191796992x^{76} + 20889178956x^{80} + 8604867856x^{84} + 3178495200x^{88} + 1092547840x^{92} + 356393138x^{96} + 117124048x^{100} + 37148704x^{104} + 11124608x^{108} + 3716760x^{112} + 782032x^{116} + 328416x^{120} + 33408x^{124} + 30925x^{128} + 656x^{132} + 2336x^{136} + 226x^{144} + x^{160}$
	$1 + 15402x^{16} + 923072x^{20} + 19072552x^{24} + 197868928x^{28} + 1273732693x^{32} + 5806209600x^{36} + 18820619240x^{40} + 45625396480x^{44} + 87099746112x^{48} + 134909539264x^{52} + 172468669256x^{56} + 184066266240x^{60} + 165100476890x^{64} + 125116340800x^{68} + 80529191304x^{72} + 44199427584x^{76} + 20866015332x^{80} + 8617012032x^{84} + 3177147256x^{88} + 1088466560x^{92} + 360022986x^{96} + 116140224x^{100} + 37240760x^{104} + 11347200x^{108} + 3543376x^{112} + 838464x^{116} + 290072x^{120} + 36736x^{124} + 28165x^{128} + 704x^{132} + 2264x^{136} + 226x^{144} + x^{160}$
	$1 + 13074x^{16} + 946144x^{20} + 19020480x^{24} + 196997664x^{28} + 1277010813x^{32} + 5804090048x^{36} + 18811494656x^{40} + 45642309056x^{44} + 87103987656x^{48} + 134882801824x^{52} + 172475128000x^{56} + 184077730656x^{60} + 165101581346x^{64} + 125121181696x^{68} + 80515699968x^{72} + 44197547648x^{76} + 20873564444x^{80} + 8614377248x^{84} + 3178391616x^{88} + 1089406688x^{92} + 358928450x^{96} + 116129216x^{100} + 37130496x^{104} + 11384768x^{108} + 3539752x^{112} + 869472x^{116} + 291904x^{120} + 40864x^{124} + 28701x^{128} + 896x^{132} + 2304x^{136} + 226x^{144} + x^{160}$
	$1 + 14874x^{16} + 923376x^{20} + 19132136x^{24} + 197353248x^{28} + 1274519557x^{32} + 5795497360x^{36} + 18858599976x^{40} + 45602679360x^{44} + 87027785968x^{48} + 135029660144x^{52} + 172428272776x^{56} + 183998321120x^{60} + 165218109514x^{64} + 125031565328x^{68} + 80529454920x^{72} + 44251693952x^{76} + 20815887412x^{80} + 8644008528x^{84} + 3171563704x^{88} + 1083355360x^{92} + 367602394x^{96} + 110143536x^{100} + 40404856x^{104} + 9895488x^{108} + 4086240x^{112} + 687312x^{116} + 345560x^{120} + 29216x^{124} + 31445x^{128} + 560x^{132} + 2328x^{136} + 226x^{144} + x^{160}$
	$1 + 13634x^{16} + 946224x^{20} + 18929728x^{24} + 197981888x^{28} + 1274474797x^{32} + 5805391792x^{36} + 18820670272x^{40} + 45624368640x^{44} + 87094160696x^{48} + 134921729136x^{52} + 172478866240x^{56} + 184039879872x^{60} + 165097166610x^{64} + 125140037616x^{68} + 80521651776x^{72} + 44193794560x^{76} + 20873179308x^{80} + 8612471696x^{84} + 3176604864x^{88} + 1090651200x^{92} + 358702674x^{96} + 116570640x^{100} + 37436864x^{104} + 11167744x^{108} + 3627000x^{112} + 789456x^{116} + 298944x^{120} + 32832x^{124} + 28013x^{128} + 592x^{132} + 2240x^{136} + 226x^{144} + x^{160}$
	$1 + 13170x^{16} + 955792x^{20} + 19000016x^{24} + 196801152x^{28} + 1276753885x^{32} + 5804042448x^{36} + 18813634384x^{40} + 45646991232x^{44} + 87097065512x^{48} + 134866482384x^{52} + 172488391696x^{56} + 184102653184x^{60} + 165085063042x^{64} + 125103278352x^{68} + 80528455568x^{72} + 44200677120x^{76} + 20868913980x^{80} + 8617461936x^{84} + 3177684080x^{88} + 1088156800x^{92} + 360087650x^{96} + 115947120x^{100} + 36951024x^{104} + 11417984x^{108} + 3463240x^{112} + 901360x^{116} + 303280x^{120} + 46080x^{124} + 30781x^{128} + 944x^{132} + 2352x^{136} + 226x^{144} + x^{160}$
	$1 + 14010x^{16} + 927232x^{20} + 18766616x^{24} + 199147520x^{28} + 1298356421x^{32} + 5568193536x^{36} + 19658902776x^{40} + 44157136896x^{44} + 88311146832x^{48} + 135023936000x^{52} + 170936086456x^{56} + 186209943552x^{60} + 163283098954x^{64} + 126058862592x^{68} + 80458236568x^{72} + 43720187904x^{76} + 21554453396x^{80} + 7973734912x^{84} + 3649886408x^{88} + 796622848x^{92} + 514264730x^{96} + 45663232x^{100} + 65498472x^{104} + 1437696x^{108} + 6584704x^{112} + 19968x^{116} + 480872x^{120} + 34133x^{128} + 2312x^{136} + 226x^{144} + x^{160}$
	$1 + 13322x^{16} + 944368x^{20} + 18905624x^{24} + 198569952x^{28} + 1272161941x^{32} + 5807010000x^{36} + 18826456888x^{40} + 45614516416x^{44} + 87094899040x^{48} + 134930006896x^{52} + 172469134264x^{56} + 184049565216x^{60} + 165096932314x^{64} + 125125364176x^{68} + 80534425560x^{72} + 44195041920x^{76} + 20866443908x^{80} + 8616320336x^{84} + 3175289288x^{88} + 1090489888x^{92} + 359633674x^{96} + 116060912x^{100} + 37543464x^{104} + 11102400x^{108} + 3637552x^{112} + 788048x^{116} + 305768x^{120} + 32736x^{124} + 28805x^{128} + 624x^{132} + 2248x^{136} + 226x^{144} + x^{160}$
	$1 + 13842x^{16} + 934400x^{20} + 18807920x^{24} + 200079616x^{28} + 1273538813x^{32} + 5730719232x^{36} + 19123108976x^{40} + 45190752256x^{44} + 87172952392x^{48} + 135411651584x^{52} + 171863305776x^{56} + 184301015296x^{60} + 165215789346x^{64} + 124926970880x^{68} + 80574024496x^{72} + 44294166528x^{76} + 20780443420x^{80} + 8612473344x^{84} + 3248773456x^{88} + 1002629888x^{92} + 424460226x^{96} + 79642112x^{100} + 54272848x^{104} + 4603904x^{108} + 5824680x^{112} + 172032x^{116} + 460816x^{120} + 2816x^{124} + 34333x^{128} + 2320x^{136} + 226x^{144} + x^{160}$
	$1 + 12490x^{16} + 954928x^{20} + 18997208x^{24} + 197220608x^{28} + 1276759157x^{32} + 5801660784x^{36} + 18821794904x^{40} + 45629727104x^{44} + 87092174304x^{48} + 134924537072x^{52} + 172448719800x^{56} + 184062534016x^{60} + 165136991994x^{64} + 125084972336x^{68} + 80531040696x^{72} + 44215863040x^{76} + 20847013188x^{80} + 8626802832x^{84} + 3177761416x^{88} + 1085119488x^{92} + 362735402x^{96} + 114395728x^{100} + 37890824x^{104} + 11153280x^{108} + 3594544x^{112} + 834640x^{116} + 296936x^{120} + 37248x^{124} + 28517x^{128} + 784x^{132} + 2280x^{136} + 226x^{144} + x^{160}$

Table 4.15: Euclidean weight enumerators of the obtained extremal $\mathbb{Z}_4$-codes of length 40

The Residue Code	Euclidean weight enumerator
	$1 + 13994x^{16} + 927024x^{20} + 19094280x^{24} + 197561984x^{28} + 1275374101x^{32} + 5804378864x^{36} + 18818900360x^{40} + 45629543552x^{44} + 87090278720x^{48} + 134922451184x^{52} + 172484259368x^{56} + 184024832000x^{60} + 165107161114x^{64} + 125150947760x^{68} + 80502660136x^{72} + 44199551232x^{76} + 20880885988x^{80} + 8604913552x^{84} + 3177949464x^{88} + 1091724928x^{92} + 358052874x^{96} + 116911312x^{100} + 37281688x^{104} + 11219584x^{108} + 3583056x^{112} + 814672x^{116} + 288696x^{120} + 35584x^{124} + 27589x^{128} + 656x^{132} + 2232x^{136} + 226x^{144} + x^{160}$
	$1 + 13970x^{16} + 944784x^{20} + 18902256x^{24} + 198497472x^{28} + 1271107965x^{32} + 5805872272x^{36} + 18836891248x^{40} + 45611685120x^{44} + 87077328584x^{48} + 134939533392x^{52} + 172466283696x^{56} + 184057401280x^{60} + 165116120226x^{64} + 125097615440x^{68} + 80532484912x^{72} + 44205883392x^{76} + 20858959516x^{80} + 8623050416x^{84} + 3175014096x^{88} + 1088608320x^{92} + 360704834x^{96} + 114924720x^{100} + 37933392x^{104} + 10971904x^{108} + 3717928x^{112} + 791152x^{116} + 318352x^{120} + 33600x^{124} + 30365x^{128} + 624x^{132} + 2320x^{136} + 226x^{144} + x^{160}$
	$1 + 20282x^{16} + 999424x^{20} + 17668424x^{24} + 199475200x^{28} + 1343994341x^{32} + 5269733376x^{36} + 20670250696x^{40} + 42019012608x^{44} + 91110536464x^{48} + 133349277696x^{52} + 169548023272x^{56} + 190736072704x^{60} + 157885256746x^{64} + 129528987648x^{68} + 80070994920x^{72} + 41936388096x^{76} + 23835712148x^{80} + 6279413760x^{84} + 4557299928x^{88} + 423018496x^{92} + 636059514x^{96} + 13287424x^{100} + 72702552x^{104} + 147456x^{108} + 6787648x^{112} + 471544x^{120} + 32885x^{128} + 2296x^{136} + 226x^{144} + x^{160}$
	$1 + 13546x^{16} + 952544x^{20} + 18919016x^{24} + 197184576x^{28} + 1278698869x^{32} + 5803874592x^{36} + 18803535304x^{40} + 45638868096x^{44} + 87121217088x^{48} + 134895939808x^{52} + 172456728136x^{56} + 184057971136x^{60} + 165108492730x^{64} + 125133648928x^{68} + 80519769512x^{72} + 44198405888x^{76} + 20869118500x^{80} + 8608954784x^{84} + 3179054776x^{88} + 1091677120x^{92} + 359345834x^{96} + 116211296x^{100} + 37101016x^{104} + 11211904x^{108} + 3496912x^{112} + 867488x^{116} + 291480x^{120} + 44608x^{124} + 28709x^{128} + 1120x^{132} + 2232x^{136} + 226x^{144} + x^{160}$
	$1 + 14706x^{16} + 922736x^{20} + 19052528x^{24} + 198887136x^{28} + 1266800157x^{32} + 5807573840x^{36} + 18853135792x^{40} + 45606772416x^{44} + 87040667048x^{48} + 134949614320x^{52} + 172517055280x^{56} + 184028346144x^{60} + 165092573250x^{64} + 125133577296x^{68} + 80511948016x^{72} + 44197594752x^{76} + 20878438972x^{80} + 8613884880x^{84} + 3176912080x^{88} + 1090348832x^{92} + 357735458x^{96} + 116360816x^{100} + 37357584x^{104} + 11124416x^{108} + 3749320x^{112} + 775376x^{116} + 338960x^{120} + 30432x^{124} + 32125x^{128} + 496x^{132} + 2384x^{136} + 226x^{144} + x^{160}$
	$1 + 14178x^{16} + 940528x^{20} + 19070768x^{24} + 196428608x^{28} + 1279842477x^{32} + 5802513648x^{36} + 18803876496x^{40} + 45651189504x^{44} + 87101746840x^{48} + 134878519344x^{52} + 172499405488x^{56} + 184061230144x^{60} + 165076244818x^{64} + 125140576560x^{68} + 80522335376x^{72} + 44196390912x^{76} + 20875696524x^{80} + 8609213136x^{84} + 3177581584x^{88} + 1089402816x^{92} + 359181138x^{96} + 117107664x^{100} + 37019696x^{104} + 11434240x^{108} + 3487640x^{112} + 831248x^{116} + 281104x^{120} + 35008x^{124} + 27309x^{128} + 528x^{132} + 2224x^{136} + 226x^{144} + x^{160}$
$C'_{40,9}$	$1 + 35015x^{16} + 37529232x^{24} + 2587413065x^{32} + 36984341824x^{40} + 174942149352x^{48} + 346506156112x^{56} + 327835131054x^{64} + 160707815424x^{72} + 42918251294x^{80} + 6419635760x^{88} + 545075910x^{96} + 26986944x^{104} + 1041584x^{112} + 60144x^{120} + 4801x^{128} + 256x^{136} + 3x^{144} + x^{160}$
	$1 + 34733x^{16} + 37536964x^{24} + 2587353827x^{32} + 36984571432x^{40} + 174941611982x^{48} + 346506969916x^{56} + 327834321720x^{64} + 160708323120x^{72} + 42918082688x^{80} + 6419637500x^{88} + 545094788x^{96} + 26982696x^{104} + 1041026x^{112} + 60292x^{120} + 4831x^{128} + 256x^{136} + 3x^{144} + x^{160}$
	$1 + 34779x^{16} + 37538400x^{24} + 2587338397x^{32} + 36984636800x^{40} + 174941458156x^{48} + 346507188704x^{56} + 327834140658x^{64} + 160708381440x^{72} + 42918124442x^{80} + 6419578912x^{88} + 545124354x^{96} + 26975872x^{104} + 1041436x^{112} + 60320x^{120} + 4845x^{128} + 256x^{136} + 3x^{144} + x^{160}$
	$1 + 34633x^{16} + 37537500x^{24} + 2587353655x^{32} + 36984565528x^{40} + 174941634058x^{48} + 346506928676x^{56} + 327834369228x^{64} + 160708286544x^{72} + 42918103796x^{80} + 6419625828x^{88} + 545101728x^{96} + 26979352x^{104} + 1041974x^{112} + 60188x^{120} + 4827x^{128} + 256x^{136} + 3x^{144} + x^{160}$
	$1 + 34899x^{16} + 37535696x^{24} + 2587358405x^{32} + 36984558816x^{40} + 174941646628x^{48} + 346506884464x^{56} + 327834480666x^{64} + 160708113984x^{72} + 42918274370x^{80} + 6419516848x^{88} + 545145274x^{96} + 26969568x^{104} + 1042884x^{112} + 60176x^{120} + 4837x^{128} + 256x^{136} + 3x^{144} + x^{160}$
	$1 + 34869x^{16} + 37534180x^{24} + 2587373547x^{32} + 36984497320x^{40} + 174941783734x^{48} + 346506708444x^{56} + 327834587808x^{64} + 160708146480x^{72} + 42918151832x^{80} + 6419627356x^{88} + 545092124x^{96} + 26983592x^{104} + 1041146x^{112} + 60260x^{120} + 4823x^{128} + 256x^{136} + 3x^{144} + x^{160}$
	$1 + 34685x^{16} + 37540764x^{24} + 2587321211x^{32} + 36984699256x^{40} + 174941325966x^{48} + 346507354372x^{56} + 327834039840x^{64} + 160708353552x^{72} + 42918237200x^{80} + 6419473892x^{88} + 545177580x^{96} + 26960376x^{104} + 1043810x^{112} + 60156x^{120} + 4855x^{128} + 256x^{136} + 3x^{144} + x^{160}$
$C'_{40,10}$	$1 + 34745x^{16} + 37530204x^{24} + 2587458759x^{32} + 36983461880x^{40} + 174945367162x^{48} + 346502304612x^{56} + 327834018012x^{64} + 160714298480x^{72} + 42913179588x^{80} + 6420287076x^{88} + 545788112x^{96} + 26829624x^{104} + 1006214x^{112} + 58204x^{120} + 4875x^{128} + 224x^{136} + 3x^{144} + x^{160}$
	$1 + 34845x^{16} + 37529540x^{24} + 2587460467x^{32} + 36983459336x^{40} + 174945373246x^{48} + 346502282492x^{56} + 327834071880x^{64} + 160714216784x^{72} + 42913259856x^{80} + 6420235388x^{88} + 545809332x^{96} + 26824520x^{104} + 1006802x^{112} + 58180x^{120} + 4879x^{128} + 224x^{136} + 3x^{144} + x^{160}$

Table 4.15: Euclidean weight enumerators of the obtained extremal $\mathbb{Z}_4$-codes of length 40

The Residue Code	Euclidean weight enumerator
	$1 + 34723x^{16} + 37528584x^{24} + 2587474525x^{32} + 36983398096x^{40} + 174945509988x^{48} + 346502118296x^{56} + 327834138882x^{64} + 160714317248x^{72} + 42913067346x^{80} + 6420391864x^{88} + 545737250x^{96} + 26843024x^{104} + 1004580x^{112} + 58280x^{120} + 4861x^{128} + 224x^{136} + 3x^{144} + x^{160}$
	$1 + 34675x^{16} + 37529080x^{24} + 2587472317x^{32} + 36983403376x^{40} + 174945503828x^{48} + 346502116712x^{56} + 327834157890x^{64} + 160714281344x^{72} + 42913106946x^{80} + 6420362824x^{88} + 545751682x^{96} + 26838320x^{104} + 1005492x^{112} + 58200x^{120} + 4861x^{128} + 224x^{136} + 3x^{144} + x^{160}$
	$1 + 34839x^{16} + 37527584x^{24} + 2587479017x^{32} + 36983382528x^{40} + 174945555672x^{48} + 346502012224x^{56} + 327834321582x^{64} + 160714089824x^{72} + 42913269582x^{80} + 6420265376x^{88} + 545791206x^{96} + 26828224x^{104} + 1006912x^{112} + 58112x^{120} + 4865x^{128} + 224x^{136} + 3x^{144} + x^{160}$
	$1 + 34797x^{16} + 37529020x^{24} + 2587467371x^{32} + 36983429400x^{40} + 174945441598x^{48} + 346502194468x^{56} + 327834123600x^{64} + 160714237616x^{72} + 42913193568x^{80} + 6420293444x^{88} + 545782204x^{96} + 26831320x^{104} + 1006066x^{112} + 58204x^{120} + 4871x^{128} + 224x^{136} + 3x^{144} + x^{160}$
	$1 + 34855x^{16} + 37527504x^{24} + 2587478729x^{32} + 36983386400x^{40} + 174945538952x^{48} + 346502054992x^{56} + 327834247662x^{64} + 160714180640x^{72} + 42913188798x^{80} + 6420317296x^{88} + 545767622x^{96} + 26835424x^{104} + 1005584x^{112} + 58224x^{120} + 4865x^{128} + 224x^{136} + 3x^{144} + x^{160}$
	$1 + 34729x^{16} + 37530412x^{24} + 2587457511x^{32} + 36983466456x^{40} + 174945355722x^{48} + 346502325204x^{56} + 327833990556x^{64} + 160714325936x^{72} + 42913158996x^{80} + 6420298516x^{88} + 545783536x^{96} + 26830872x^{104} + 1006006x^{112} + 58220x^{120} + 4875x^{128} + 224x^{136} + 3x^{144} + x^{160}$
	$1 + 34879x^{16} + 37529480x^{24} + 2587459305x^{32} + 36983466864x^{40} + 174945350768x^{48} + 346502323704x^{56} + 327834020670x^{64} + 160714262528x^{72} + 42913228710x^{80} + 6420252664x^{88} + 545801286x^{96} + 26827440x^{104} + 1006120x^{112} + 58248x^{120} + 4881x^{128} + 224x^{136} + 3x^{144} + x^{160}$
$C'_{40,11}$	$1 + 11562x^{16} + 958176x^{20} + 19038640x^{24} + 196291520x^{28} + 1285412861x^{32} + 5787906592x^{36} + 18729647920x^{40} + 45608893824x^{44} + 87214859304x^{48} + 135067054304x^{52} + 172572634480x^{56} + 183957837888x^{60} + 164901808546x^{64} + 125013466912x^{68} + 80504997360x^{72} + 44272954624x^{76} + 20957598700x^{80} + 8652027808x^{84} + 3187230480x^{88} + 1079881792x^{92} + 347440066x^{96} + 108395872x^{100} + 32352656x^{104} + 9485696x^{108} + 2594312x^{112} + 632992x^{116} + 169424x^{120} + 25536x^{124} + 16029x^{128} + 352x^{132} + 1360x^{136} + 186x^{144} + x^{160}$
	$1 + 16026x^{16} + 920064x^{20} + 19102336x^{24} + 196287360x^{28} + 1285933485x^{32} + 5788660352x^{36} + 18724871744x^{40} + 45606559488x^{44} + 87227587544x^{48} + 135070413568x^{52} + 172558087680x^{56} + 183954800768x^{60} + 164905553810x^{64} + 125015594368x^{68} + 80512871616x^{72} + 44272140800x^{76} + 20949441340x^{80} + 8651406336x^{84} + 3189806208x^{88} + 1080845440x^{92} + 347538898x^{96} + 108037504x^{100} + 32245696x^{104} + 9465600x^{108} + 2566552x^{112} + 653056x^{116} + 172800x^{120} + 28544x^{124} + 16621x^{128} + 640x^{132} + 1344x^{136} + 186x^{144} + x^{160}$
	$1 + 12562x^{16} + 927744x^{20} + 18581640x^{24} + 200466432x^{28} + 1294682277x^{32} + 5598597120x^{36} + 19450369544x^{40} + 44348497920x^{44} + 88194674096x^{48} + 135305320448x^{52} + 171122523176x^{56} + 185747226624x^{60} + 163642470250x^{64} + 125494714368x^{68} + 80547894824x^{72} + 43980931072x^{76} + 21366689700x^{80} + 8213321728x^{84} + 3565023896x^{88} + 822042624x^{92} + 489857722x^{96} + 42717184x^{100} + 57576472x^{104} + 1040384x^{108} + 5103328x^{112} + 10240x^{116} + 330424x^{120} + 22325x^{128} + 1464x^{136} + 186x^{144} + x^{160}$
	$1 + 18282x^{16} + 929152x^{20} + 18414512x^{24} + 201730688x^{28} + 1291612733x^{32} + 5548892416x^{36} + 19683331248x^{40} + 43902204672x^{44} + 88764613992x^{48} + 134783713408x^{52} + 171072755312x^{56} + 186763684736x^{60} + 162187224802x^{64} + 126506526208x^{68} + 80383652976x^{72} + 43466682880x^{76} + 22035025580x^{80} + 7767611008x^{84} + 3765065488x^{88} + 767056256x^{92} + 492824194x^{96} + 45295872x^{100} + 55606288x^{104} + 1467136x^{108} + 5238728x^{112} + 19328x^{116} + 398032x^{120} + 128x^{124} + 29789x^{128} + 1744x^{136} + 186x^{144} + x^{160}$
	$1 + 12802x^{16} + 947040x^{20} + 19029512x^{24} + 196311200x^{28} + 1285316149x^{32} + 5788121024x^{36} + 18730359720x^{40} + 45607789248x^{44} + 87212896672x^{48} + 135069480224x^{52} + 172575835624x^{56} + 183954788576x^{60} + 164897931130x^{64} + 125015837696x^{68} + 80508876936x^{72} + 44271780992x^{76} + 20954716148x^{80} + 8652409760x^{84} + 3188438680x^{88} + 1079804000x^{92} + 347343594x^{96} + 108390080x^{100} + 32266360x^{104} + 9497792x^{108} + 2595248x^{112} + 630752x^{116} + 177016x^{120} + 25120x^{124} + 16741x^{128} + 384x^{132} + 1368x^{136} + 186x^{144} + x^{160}$
	$1 + 13354x^{16} + 939104x^{20} + 19086672x^{24} + 196119648x^{28} + 1285969597x^{32} + 5788081792x^{36} + 18725805200x^{40} + 45611074880x^{44} + 87224135592x^{48} + 135061927584x^{52} + 172560574480x^{56} + 183959609376x^{60} + 164911641058x^{64} + 125018484800x^{68} + 80500088144x^{72} + 44268299136x^{76} + 20957926380x^{80} + 8651448736x^{84} + 3188669680x^{88} + 1081820832x^{92} + 346791682x^{96} + 107987456x^{100} + 32265392x^{104} + 9331520x^{108} + 2644872x^{112} + 655200x^{116} + 182704x^{120} + 32992x^{124} + 17501x^{128} + 832x^{132} + 1392x^{136} + 186x^{144} + x^{160}$
	$1 + 13282x^{16} + 948816x^{20} + 18938552x^{24} + 196933440x^{28} + 1283076117x^{32} + 5789438800x^{36} + 18737980696x^{40} + 45597301248x^{44} + 87206400128x^{48} + 135084154640x^{52} + 172575904856x^{56} + 183950746944x^{60} + 164895568730x^{64} + 125012350480x^{68} + 80514544312x^{72} + 44273116672x^{76} + 20954066068x^{80} + 8650823152x^{84} + 3186566312x^{88} + 1082027456x^{92} + 347467146x^{96} + 108108016x^{100} + 32389320x^{104} + 9228288x^{108} + 2683984x^{112} + 609456x^{116} + 194120x^{120} + 26048x^{124} + 18693x^{128} + 432x^{132} + 1384x^{136} + 186x^{144} + x^{160}$

Table 4.15: Euclidean weight enumerators of the obtained extremal $\mathbb{Z}_4$-codes of length 40

The Residue Code	Euclidean weight enumerator
	$1 + 13010x^{16} + 923376x^{20} + 19046632x^{24} + 198695840x^{28} + 1281693349x^{32} + 5683665296x^{36} + 19171852456x^{40} + 44937702208x^{44} + 87552822512x^{48} + 135344686064x^{52} + 171867213960x^{56} + 184757833056x^{60} + 164385414122x^{64} + 125171340816x^{68} + 80507261896x^{72} + 44179689856x^{76} + 21122863076x^{80} + 8461610576x^{84} + 3369805496x^{88} + 947162208x^{92} + 424470458x^{96} + 68880432x^{100} + 48263928x^{104} + 3489600x^{108} + 4695456x^{112} + 130768x^{116} + 367064x^{120} + 3744x^{124} + 28597x^{128} + 48x^{132} + 1688x^{136} + 186x^{144} + x^{160}$
	$1 + 15386x^{16} + 908160x^{20} + 19097264x^{24} + 197888032x^{28} + 1276998861x^{32} + 5790492640x^{36} + 18765441424x^{40} + 45573074624x^{44} + 87170501016x^{48} + 135137743552x^{52} + 172570747440x^{56} + 183917802592x^{60} + 164934874994x^{64} + 125002805664x^{68} + 80494483344x^{72} + 44281956480x^{76} + 20954503996x^{80} + 8654433792x^{84} + 3186107024x^{88} + 1082162912x^{92} + 346708658x^{96} + 107075872x^{100} + 33175856x^{104} + 8895680x^{108} + 2919000x^{112} + 552896x^{116} + 218256x^{120} + 20640x^{124} + 19725x^{128} + 352x^{132} + 1456x^{136} + 186x^{144} + x^{160}$
	$1 + 17690x^{16} + 902912x^{20} + 19049600x^{24} + 197592320x^{28} + 1280295565x^{32} + 5792074240x^{36} + 18742386144x^{40} + 45583299584x^{44} + 87216033688x^{48} + 135098215168x^{52} + 172549351872x^{56} + 183958243584x^{60} + 164909370418x^{64} + 124996093440x^{68} + 80523901344x^{72} + 44274651136x^{76} + 20945268284x^{80} + 8656444672x^{84} + 3186339072x^{88} + 1080837888x^{92} + 348347890x^{96} + 107589120x^{100} + 32585504x^{104} + 9223168x^{108} + 2656344x^{112} + 618752x^{116} + 190656x^{120} + 27392x^{124} + 18253x^{128} + 512x^{132} + 1376x^{136} + 186x^{144} + x^{160}$
	$1 + 12858x^{16} + 913408x^{20} + 18850448x^{24} + 199393280x^{28} + 1294837805x^{32} + 5595701248x^{36} + 19460995248x^{40} + 44364922880x^{44} + 88142898360x^{48} + 135297388544x^{52} + 171201326096x^{56} + 185717424128x^{60} + 163608270546x^{64} + 125528686592x^{68} + 80528825264x^{72} + 43980947456x^{76} + 21380830364x^{80} + 8201801728x^{84} + 3568894256x^{88} + 823820288x^{92} + 487583698x^{96} + 43687936x^{100} + 56972688x^{104} + 1114112x^{108} + 5128120x^{112} + 12288x^{116} + 360496x^{120} + 25901x^{128} + 1552x^{136} + 186x^{144} + x^{160}$
	$1 + 27098x^{16} + 37912992x^{24} + 2571714861x^{32} + 37457942048x^{40} + 174427638424x^{48} + 345155952416x^{56} + 329786402706x^{64} + 161024617376x^{72} + 41908152316x^{80} + 6376490720x^{88} + 694464338x^{96} + 64700512x^{104} + 5237464x^{112} + 348256x^{120} + 24557x^{128} + 1504x^{136} + 186x^{144} + x^{160}$
	$1 + 19002x^{16} + 962560x^{20} + 18186896x^{24} + 199147520x^{28} + 1324734509x^{32} + 5406318592x^{36} + 20102355120x^{40} + 42988716032x^{44} + 90016074936x^{48} + 134037229568x^{52} + 170471762960x^{56} + 188712058880x^{60} + 159795820242x^{64} + 128088817664x^{68} + 80200342448x^{72} + 42687365120x^{76} + 23040097436x^{80} + 7002738688x^{84} + 4173291824x^{88} + 600031232x^{92} + 548937682x^{96} + 31449088x^{100} + 58668432x^{104} + 966656x^{108} + 5134264x^{112} + 12288x^{116} + 360496x^{120} + 25901x^{128} + 1552x^{136} + 186x^{144} + x^{160}$
	$1 + 15058x^{16} + 932640x^{20} + 18971784x^{24} + 197593408x^{28} + 1279434373x^{32} + 5791319264x^{36} + 18749494376x^{40} + 45585079424x^{44} + 87198253104x^{48} + 135098790688x^{52} + 172566561896x^{56} + 183951796928x^{60} + 164910979466x^{64} + 125004080608x^{68} + 80507983176x^{72} + 44271820544x^{76} + 20955194212x^{80} + 8655352416x^{84} + 3186681880x^{88} + 1081539264x^{92} + 346553562x^{96} + 107684768x^{100} + 32690872x^{104} + 9206912x^{108} + 2777824x^{112} + 593760x^{116} + 201976x^{120} + 22848x^{124} + 18709x^{128} + 416x^{132} + 1432x^{136} + 186x^{144} + x^{160}$
	$1 + 14346x^{16} + 920640x^{20} + 19124384x^{24} + 197235328x^{28} + 1279945981x^{32} + 5789743552x^{36} + 18752808384x^{40} + 45589243648x^{44} + 87174961096x^{48} + 135111132736x^{52} + 172606042208x^{56} + 183909537664x^{60} + 164891941858x^{64} + 125042741696x^{68} + 80497835520x^{72} + 44266686976x^{76} + 20965107212x^{80} + 8646552256x^{84} + 3187018592x^{88} + 1083783552x^{92} + 345513026x^{96} + 108524352x^{100} + 32445120x^{104} + 9133824x^{108} + 2824552x^{112} + 557760x^{116} + 210848x^{120} + 19584x^{124} + 19165x^{128} + 320x^{132} + 1408x^{136} + 186x^{144} + x^{160}$

Table 4.15: Euclidean weight enumerators of the obtained extremal $\mathbb{Z}_4$-codes of length 40

In [9], extremal Type I $\mathbb{Z}_4$-codes of type $4^8 2^{24}$ and $4^{10}2^{20}$ were constructed. Therefore, our extremal Type I $\mathbb{Z}_4$-codes of types $4^7 2^{26}$, $4^{11}2^{18}$, $4^{15}2^{10}$, and $4^{16}2^8$ are new. The minimum weight of the residue codes of the known extremal Type I $\mathbb{Z}_4$-codes of type $4^{10}2^{20}$ is 12 or 16. Therefore, codes constructed from $C'_{40,3}$ are new. The binary $[40, 10, 12]$ code, which is the residue code of the known extremal Type I $\mathbb{Z}_4$-code of the type $4^{10}2^{20}$, have 10 codewords of weight 12. Therefore, that code is not equivalent to the code $C'_{40,4}$. So, all Type I extremal $\mathbb{Z}_4$-codes obtained from $C'_{40,4}$ are also new.

Codes of length 48

There is only one, up to equivalence, Hadamard design on 11 points with the skew incidence matrix. This design is constructed from the skew-symmetric Hadamard matrix SH_{12} given in (1.1). From the construction given in Proposition 3.1.2, we obtained 54 pairwise nonisomorphic $2-(47,23,11)$ designs. From the block matrix H_1 we obtained 18 designs $\mathscr{D}_1,\ldots,\mathscr{D}_{18}$. From H_2 we obtained another 18 designs $\mathscr{D}_{19},\ldots,\mathscr{D}_{36}$, and their duals $\mathscr{D}_{37},\ldots,\mathscr{D}_{54}$. In Table 4.16, the structure of the full automorphism groups of the obtained designs are shown. These designs were previuosly constructed in [16]. From the designs $\mathscr{D}_1,\ldots,\mathscr{D}_{54}$ we constructed the corresponding Hadamard $3-(48,24,11)$ designs denoted by $\mathscr{D}_1^*,\ldots,\mathscr{D}_{54}^*$.

The size of the group	The structure of the group	Designs
3960	$PSL(2,11)\times S_3$	$\mathscr{D}_1$, $\mathscr{D}_{22}$, $\mathscr{D}_{40}$
330	$(\mathbb{Z}_{11}:\mathbb{Z}_5)\times S_3$	$\mathscr{D}_8$, $\mathscr{D}_{18}$, $\mathscr{D}_{23}$, $\mathscr{D}_{36}$, $\mathscr{D}_{41}$, $\mathscr{D}_{54}$
110	$\mathbb{Z}_2\times(\mathbb{Z}_{11}:\mathbb{Z}_5)$	$\mathscr{D}_2$, $\mathscr{D}_3$, $\mathscr{D}_4$, $\mathscr{D}_5$, $\mathscr{D}_6$, $\mathscr{D}_7$, $\mathscr{D}_9$, $\mathscr{D}_{11}$, $\mathscr{D}_{12}$, $\mathscr{D}_{14}$, $\mathscr{D}_{15}$, $\mathscr{D}_{16}$, $\mathscr{D}_{17}$, $\mathscr{D}_{19}$, $\mathscr{D}_{21}$, $\mathscr{D}_{24}$, $\mathscr{D}_{25}$, $\mathscr{D}_{26}$, $\mathscr{D}_{27}$, $\mathscr{D}_{28}$, $\mathscr{D}_{29}$, $\mathscr{D}_{31}$, $\mathscr{D}_{32}$, $\mathscr{D}_{33}$, $\mathscr{D}_{34}$, $\mathscr{D}_{35}$, $\mathscr{D}_{37}$, $\mathscr{D}_{39}$, $\mathscr{D}_{42}$, $\mathscr{D}_{43}$, $\mathscr{D}_{44}$, $\mathscr{D}_{45}$, $\mathscr{D}_{46}$, $\mathscr{D}_{47}$, $\mathscr{D}_{49}$, $\mathscr{D}_{50}$, $\mathscr{D}_{51}$, $\mathscr{D}_{52}$, $\mathscr{D}_{53}$
55	$\mathbb{Z}_{11}:\mathbb{Z}_5$	$\mathscr{D}_{10}$, $\mathscr{D}_{13}$, $\mathscr{D}_{20}$, $\mathscr{D}_{30}$, $\mathscr{D}_{38}$, $\mathscr{D}_{48}$

Table 4.16: The structure of the full automorphism groups of $2-(47,23,11)$ designs $\mathscr{D}_1,\ldots,\mathscr{D}_{54}$

From $\mathscr{D}_1^*,\ldots,\mathscr{D}_{54}^*$ we obtained 15 binary linear codes $C_{48,1},C_{48,2},\ldots,C_{48,15}$. The classes of designs that give nonequivalent codes are given in Table 4.17.

The code	The Designs	The code	The Designs	The code	The Designs
$C_{48,1}$	$\mathscr{D}^*_{24},\mathscr{D}^*_{25},\mathscr{D}^*_{28},\mathscr{D}^*_{29},$ $\mathscr{D}^*_{30},\mathscr{D}^*_{31},\mathscr{D}^*_{32},\mathscr{D}^*_{34}$	$C_{48,6}$	$\mathscr{D}^*_{41},\mathscr{D}^*_{44}$	$C_{48,11}$	$\mathscr{D}^*_{14}$
$C_{48,2}$	$\mathscr{D}^*_{19},\mathscr{D}^*_{36}$	$C_{48,7}$	$\mathscr{D}^*_{47},\mathscr{D}^*_{48},\mathscr{D}^*_{50}$	$C_{48,12}$	$\mathscr{D}^*_{3},\mathscr{D}^*_{4},\mathscr{D}^*_{6},\mathscr{D}^*_{7},$ $\mathscr{D}^*_{12},\mathscr{D}^*_{13},\mathscr{D}^*_{15},\mathscr{D}^*_{16}$
$C_{48,3}$	$\mathscr{D}^*_{1},\mathscr{D}^*_{2},\mathscr{D}^*_{5},\mathscr{D}^*_{8}$	$C_{48,8}$	$\mathscr{D}^*_{33}$	$C_{48,13}$	$\mathscr{D}^*_{20},\mathscr{D}^*_{21},\mathscr{D}^*_{35}$
$C_{48,4}$	$\mathscr{D}^*_{10},\mathscr{D}^*_{11},\mathscr{D}^*_{17}$	$C_{48,9}$	$\mathscr{D}^*_{45}$	$C_{48,14}$	$\mathscr{D}^*_{38},\mathscr{D}^*_{39},\mathscr{D}^*_{42},\mathscr{D}^*_{43},$ $\mathscr{D}^*_{46},\mathscr{D}^*_{49},\mathscr{D}^*_{52},\mathscr{D}^*_{53}$
$C_{48,5}$	$\mathscr{D}^*_{9},\mathscr{D}^*_{18}$	$C_{48,10}$	$\mathscr{D}^*_{22},\mathscr{D}^*_{23},\mathscr{D}^*_{26},\mathscr{D}^*_{27}$	$C_{48,15}$	$\mathscr{D}^*_{37},\mathscr{D}^*_{40},\mathscr{D}^*_{51},\mathscr{D}^*_{54}$

Table 4.17: The classes of designs that give codes $C_{48,1}, C_{48,2}, \ldots, C_{48,15}$

Weight distributions of the constructed binary codes $C_{48,1}, C_{48,2}, \ldots, C_{48,15}$ are given in Table 4.18.

The code	$[n,k,d]$	0	4	8	12	16	20	24
$C_{48,1}$	$[48,24,4]$	1	132	5346	71284	638319	3007752	9331548
$C_{48,2}$	$[48,14,12]$	1	0	0	136	495	2904	9312
$C_{48,3}$	$[48,13,8]$	1	0	66	0	495	0	7068
$C_{48,4}$	$[48,23,4]$	1	66	902	29018	283679	1733732	4293812
$C_{48,5}$	$[48,13,12]$	1	0	0	66	275	1342	4824
$C_{48,6}$	$[48,14,4]$	1	1	11	121	451	3014	9186
$C_{48,7}$	$[48,24,4]$	1	78	2046	58102	564927	3474108	8578692
$C_{48,8}$	$[48,14,12]$	1	0	0	4	1023	1980	10368
$C_{48,9}$	$[48,14,12]$	1	1	0	77	1111	946	12112
$C_{48,10}$	$[48,14,8]$	1	0	66	4	759	1980	10764
$C_{48,11}$	$[48,13,16]$	1	0	0	0	759	0	6672
$C_{48,12}$	$[48,23,4]$	1	132	5346	67188	367983	980232	5546844
$C_{48,13}$	$[48,24,4]$	1	66	1254	55642	546975	3581028	8407284
$C_{48,14}$	$[48,24,4]$	1	276	10626	134596	735471	1961256	11092764
$C_{48,15}$	$[48,14,4]$	1	12	66	220	495	792	13212

Table 4.18: Parameters and the weight distribution of binary codes $C_{48,1}, C_{48,2}, \ldots, C_{48,15}$

These codes were constructed in [17]. All of the codes $C_{48,1}, C_{48,2}, \ldots, C_{48,15}$, have the dual codes of minimum weight 4. By Lemma 2.1.14, the codes $C_{48,1}, C_{48,2}, \ldots, C_{48,15}$ could be the residue codes of the extremal or near-extremal $\mathbb{Z}_4$-codes if the minimum weight of their duals is at least 6. Therefore, we conclude that no extremal or near-extremal $\mathbb{Z}_4$-codes can be obtained from these codes.

We also constructed binary linear codes from the 53 Hadamard matrices of order 48 given in [38]. In total 6 nonequivalent binary codes $C'_{48,1}$, $C'_{48,2}$,..., $C'_{48,6}$ were obtained. All of these codes are self-dual. The weight distribution of codes $C'_{48,1}$, $C'_{48,2}$,..., $C'_{48,6}$ is

given in Table 4.19.

The code	$[n,k,d]$	0	4	8	12	16	20	24
$C'_{48,1}$	$[48,24,4]$	1	276	10626	134596	735471	1961256	11092764
$C'_{48,2}$	$[48,24,8]$	1	0	276	15088	542823	3979920	7701000
$C'_{48,3}$	$[48,24,8]$	1	0	138	16192	538959	3987648	7691340
$C'_{48,4}$	$[48,24,8]$	1	0	0	17296	535095	3995376	7681680
$C'_{48,5}$	$[48,24,8]$	1	0	138	16192	538959	3987648	7691340
$C'_{48,6}$	$[48,24,8]$	1	0	1518	5152	577599	3910368	7787940

Table 4.19: Parameters and the weight distribution of binary codes $C'_{48,1}, C_{48,2}, \ldots, C'_{48,6}$

Since the minimum weight of the codes $C'_{48,2}$, $C'_{48,3}$, …, $C'_{48,6}$ is greater than 6, by Lemma 2.1.14, these codes are good for the construction of extremal $\mathbb{Z}_4$-codes. The code $C'_{48,1}$ have the minimum weight 4, so this code is not good for the construction of extremal $\mathbb{Z}_4$-codes by Lemma 2.1.14. The codes $C'_{48,4}$ and $C'_{48,6}$ are the residue codes of already known extremal $\mathbb{Z}_4$-codes given in [7] and [33]. From $C'_{48,6}$, we constructed already known extremal Type II $\mathbb{Z}_4$-code given in [33]. Also, we found 146 near-extremal Type I $\mathbb{Z}_4$-codes with the same residue code. At least two of those 146 codes are nonequivalent. The number of codewords of Euclidean weight 20 in these near-extremal codes are $307520, 308752$. All of the constructed near-extremal $\mathbb{Z}_4$-codes are neighbors of the known extremal $\mathbb{Z}_4$-code from [33].

From other binary codes, no extremal or near-extremal $\mathbb{Z}_4$-codes were found.

4.2. CODES OF LENGTHS 56, 64, AND 72

In this subsection we give the overview of the codes of greater lengths. These lengths proved to be too large for the current computational power that was available to us. On these lengths we even had the problem to determine the equivalence of some binary codes that were constructed. Therefore, we are able only to present partial results that can be of use in some future work.

Codes of length 56

As in the case of the length 40, construction given in Proposition 3.1.2 cannot be used since:

$$4n-1=55 \Rightarrow n=14,$$

which contradicts the necessary condition given in Theorem 1.2.7. Therefore, in order to construct Hadamard $2-(55,27,13)$ designs, we used incidence matrices given in Table A.1 for $n=55$. We obtained 15 designs $\mathscr{D}_1,\mathscr{D}_2,\ldots,\mathscr{D}_{15}$, [15]. The first five matrices in Table A.1 yield designs $\mathscr{D}_1,\mathscr{D}_2,\ldots,\mathscr{D}_5$ that are self-dual or pairwise dual. From the second five matrices we obtained designs $\mathscr{D}_6,\mathscr{D}_7,\ldots,\mathscr{D}_{10}$ and their duals $\mathscr{D}_{11},\mathscr{D}_{12},\ldots,\mathscr{D}_{15}$. The structure of the full automorphism group of designs is given in Table 4.20.

The size of the group	The structure of the group	Designs
78	$Frob_{39}:\mathbb{Z}_2$	$\mathscr{D}_3$, $\mathscr{D}_4$, $\mathscr{D}_5$, $\mathscr{D}_7$, $\mathscr{D}_9$, $\mathscr{D}_{10}$, $\mathscr{D}_{12}$, $\mathscr{D}_{14}$, $\mathscr{D}_{15}$
26	D_{26}	$\mathscr{D}_1$, $\mathscr{D}_2$, $\mathscr{D}_6$, $\mathscr{D}_8$, $\mathscr{D}_{11}$, $\mathscr{D}_{13}$

Table 4.20: The structure of the full automorphism group of $2-(55,27,13)$ designs $\mathscr{D}_1,\mathscr{D}_2,\ldots,\mathscr{D}_{15}$

From $\mathscr{D}_1,\mathscr{D}_2,\ldots,\mathscr{D}_{15}$ we constructed the corresponding $3-(56,28,13)$ designs $\mathscr{D}_1^*$, $\mathscr{D}_2^*$, ..., $\mathscr{D}_{15}^*$. These designs give 15 binary linear codes $C_1,C_2,\ldots,C_{15}$ which are dis-

tributed in 4 equivalence classes:

$$C_6, (C_3, C_9), (C_4, C_5, C_{12}, C_{13}),$$
$$(C_1, C_2, C_7, C_8, C_{10}, C_{11}, C_{14}, C_{15}).$$

The weight distribution of nonequivalent codes is given in Table 4.21. All of the constructed codes are self-dual. By Lemma 2.1.14, the minimum weight of the torsion code of the extremal or near-extremal $\mathbb{Z}_4$-code of length 58 should be at least 6. Since the minimum weight of the codes C_1, C_3, and C_6 is 4, they cannot be used for the construction of extremal or near-extremal $\mathbb{Z}_4$-codes. Only the code C_4 satisfies Lemma 2.1.14 since its minimum weight is 8.

#	$[n,k,d]$	0	4	8	12	16	20	24	28
C_1	$[56,28,4]$	1	14	91	17836	769769	11533522	64090299	115612392
C_3	$[56,28,4]$	1	378	20475	376740	3108105	13123110	30421755	174334328
C_4	$[56,28,8]$	1	0	91	8736	614761	11729536	64847419	114034368
C_6	$[56,28,4]$	1	182	10283	188188	1783929	12524330	48013147	143395336

Table 4.21: Parameters and the weight distribution of binary codes of length 56

Codes of length 64

Up to the equivalence, there are two skew-symmetric Hadamard matrices of order 16. These matrices are SH_{16}, given in (1.2), and its transpose. From the corresponding skew Hadamard designs, by Proposition 3.1.2, we obtained 104 nonisomorphic designs with parameters $2-(63,31,15)$. From the first skew-symmetric Hadamard matrix SH_{16}, we obtained designs $\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_{54}$. The designs $\mathscr{D}_1, \mathscr{D}_2, \ldots \mathscr{D}_{18}$ were obtained from the first block matrix H_1 of Proposition 3.1.2. The designs $\mathscr{D}_{19}, \mathscr{D}_{20}, \ldots \mathscr{D}_{36}$ were obtained from the second block matrix H_2 of Proposition 3.1.2. The designs $\mathscr{D}_{37}, \mathscr{D}_{38}, \ldots \mathscr{D}_{54}$ are duals of the designs $\mathscr{D}_{19}, \mathscr{D}_{20}, \ldots \mathscr{D}_{36}$. From the second skew-symmetric Hadamard matrix SH_{16}^T, we obtained designs $\mathscr{D}_{55}, \ldots, \mathscr{D}_{108}$. Again, the first 18 designs are obtained from H_1, and the folowing 36 designs are obtained from H_2 and their duals. The pairs of

designs $(\mathscr{D}_{32}, \mathscr{D}_{89}), (\mathscr{D}_{35}, \mathscr{D}_{86}), (\mathscr{D}_{50}, \mathscr{D}_{107}), (\mathscr{D}_{53}, \mathscr{D}_{104})$ are isomorphic. The structure of the full automorphism groups of the obtained designs are given in Table 4.22.

The size of the group	The structure of the group	Designs
1008	$(E_8 : Frob_{21}) \times S_3$	$\mathscr{D}_{22}, \mathscr{D}_{40}$
504	$Frob_{21} \times S_4$	$\mathscr{D}_{18}, \mathscr{D}_{62}$
4032	$(E_8 : Frob_{21}) \times S_4$	$\mathscr{D}_1, \mathscr{D}_{55}, \mathscr{D}_{76}, \mathscr{D}_{94}$
126	$Frob_{21} \times S_3$	$\mathscr{D}_8, \mathscr{D}_{23}, \mathscr{D}_{41}, \mathscr{D}_{72}, \mathscr{D}_{77}, \mathscr{D}_{95}$
84	$E_4 \times Frob_{21}$	$\mathscr{D}_2, \mathscr{D}_{12}, \mathscr{D}_{14}, \mathscr{D}_{16}, \mathscr{D}_{33}, \mathscr{D}_{51}, \mathscr{D}_{58}, \mathscr{D}_{59},$ $\mathscr{D}_{63}, \mathscr{D}_{70}, \mathscr{D}_{73}, \mathscr{D}_{79}, \mathscr{D}_{85}, \mathscr{D}_{91}, \mathscr{D}_{97}, \mathscr{D}_{103}$
42	$\mathbb{Z}_2 \times Frob_{21}$	$\mathscr{D}_3, \mathscr{D}_4, \mathscr{D}_5, \mathscr{D}_6, \mathscr{D}_7, \mathscr{D}_9, \mathscr{D}_{11}, \mathscr{D}_{15}, \mathscr{D}_{17},$ $\mathscr{D}_{19}, \mathscr{D}_{21}, \mathscr{D}_{24}, \mathscr{D}_{25}, \mathscr{D}_{26}, \mathscr{D}_{27}, \mathscr{D}_{28}, \mathscr{D}_{29}, \mathscr{D}_{31},$ $\mathscr{D}_{32}, \mathscr{D}_{34}, \mathscr{D}_{35}, \mathscr{D}_{36}, \mathscr{D}_{37}, \mathscr{D}_{39}, \mathscr{D}_{42}, \mathscr{D}_{43}, \mathscr{D}_{44},$ $\mathscr{D}_{45}, \mathscr{D}_{46}, \mathscr{D}_{47}, \mathscr{D}_{49}, \mathscr{D}_{50}, \mathscr{D}_{52}, \mathscr{D}_{53}, \mathscr{D}_{54}, \mathscr{D}_{56},$ $\mathscr{D}_{57}, \mathscr{D}_{60}, \mathscr{D}_{61}, \mathscr{D}_{65}, \mathscr{D}_{66}, \mathscr{D}_{68}, \mathscr{D}_{69}, \mathscr{D}_{71}, \mathscr{D}_{75},$ $\mathscr{D}_{78}, \mathscr{D}_{80}, \mathscr{D}_{81}, \mathscr{D}_{82}, \mathscr{D}_{83}, \mathscr{D}_{87}, \mathscr{D}_{88}, \mathscr{D}_{90}, \mathscr{D}_{93},$ $\mathscr{D}_{96}, \mathscr{D}_{98}, \mathscr{D}_{99}, \mathscr{D}_{100}, \mathscr{D}_{101}, \mathscr{D}_{105}, \mathscr{D}_{106}, \mathscr{D}_{108}$
21	$Frob_{21}$	$\mathscr{D}_{10}, \mathscr{D}_{13}, \mathscr{D}_{20}, \mathscr{D}_{30}, \mathscr{D}_{38}, \mathscr{D}_{48},$ $\mathscr{D}_{64}, \mathscr{D}_{67}, \mathscr{D}_{74}, \mathscr{D}_{84}, \mathscr{D}_{92}, \mathscr{D}_{102}$

Table 4.22: The structure of the full automorphism groups of 104 pairwise nonisomorphic $2-(63,31,15)$ designs

From these designs, we obtained 104 pairwise nonisomorphic $3-(64,32,15)$ designs $\mathscr{D}_1^*, \ldots, \mathscr{D}_{104}^*$, which give binary linear codes that distribute in 21 classes of the same weight distributions. The weight distribution classes are given in Table 4.23. All the obtained codes have dual codes of minimum weight 4. By Lemma 2.1.14, the extremal or near-extremal $\mathbb{Z}_4$-code of length 64 should have the torsion code of minimum weight that is at least 6. Therefore, no extremal or near-extremal $\mathbb{Z}_4$-codes can be obtained from these codes.

#	$[n,k,d]$	0	4	8	12	16	20	24	28	32
1	$[64,18,4]$	1	16	120	560	1820	4368	8008	11440	209478
2	$[64,18,4]$	1	1	70	91	1672	2025	16730	14267	192430
3	$[64,18,4]$	1	28	198	56	788	5712	4522	39004	161526
4	$[64,32,4]$	1	178	9564	260294	4059716	37634946	217909924	865313542	2044590966
5	$[64,17,8]$	1	0	56	0	1180	0	11144	0	106310
6	$[64,17,4]$	1	21	99	28	490	2814	1869	19537	81354
7	$[64,32,4]$	1	496	35960	906192	10518300	64512240	225792840	471435600	2748564038
8	$[64,18,8]$	1	0	56	0	1692	0	25480	0	207686
9	$[64,31,4]$	1	240	18040	452816	5260060	32253936	112900424	235712080	1374288454
10	$[64,24,4]$	1	120	1848	11368	64028	255992	722568	1813288	11038790
11	$[64,32,4]$	1	240	18040	452816	5325596	40118256	232175944	760524368	2217736774
12	$[64,32,4]$	1	176	9720	262672	4019740	37199280	219091400	865083280	2043634758
13	$[64,18,8]$	1	0	120	0	1824	480	15288	32032	162654
14	$[64,31,4]$	1	162	7548	172566	2288516	19060722	107236484	432039254	1025873142
15	$[64,25,4]$	1	136	2520	19992	119388	496264	1535464	3661528	21883846
16	$[64,11,4]$	1	1	0	7	28	21	0	99	1734
17	$[64,11,16]$	1	0	0	0	32	0	112	0	1758
18	$[64,17,8]$	1	0	120	0	1820	0	8008	0	111174
19	$[64,19,4]$	1	120	1848	11368	97052	286712	2105992	3863336	20821574
20	$[64,18,4]$	1	22	120	176	1211	3444	6124	46342	147264
21	$[64,10,16]$	1	0	0	0	32	0	112	0	1758

Table 4.23: Parameters and the weight distribution of classes of codes of length 64 obtained from Proposition 3.1.2

From 46 designs with parameters $2-(64,28,12)$ described in [18], we obtained 46 doubly-even binary codes $C_{64,1}, C_{64,2}, \ldots, C_{64,46}$. The following pairs of codes are mutu-

ally equivalent:

$$\begin{array}{llll}
\left(C_{64,3}, C_{64,30}\right), & \left(C_{64,9}, C_{64,10}\right), & \left(C_{64,22}, C_{64,33}\right), & \left(C_{64,24}, C_{64,35}\right), \\
\left(C_{64,5}, C_{64,26}\right), & \left(C_{64,2}, C_{64,29}\right), & \left(C_{64,6}, C_{64,27}\right), & \left(C_{64,4}, C_{64,28}\right), \\
\left(C_{64,40}, C_{64,41}\right), & \left(C_{64,21}, C_{64,32}\right), & \left(C_{64,23}, C_{64,34}\right), & \left(C_{64,38}, C_{64,44}\right), \\
\left(C_{64,20}, C_{64,31}\right). & & &
\end{array}$$

Therefore, there are 33 binary codes, up to the equivalence. The weight distribution of these codes is given in Table 4.24. The dual codes of the codes $C_{64,11}$ and $C_{64,12}$ have minimum weight 4. The dual codes of the codes $C_{64,7}$ and $C_{64,9}$ have minimum weight 6. The dual codes of the remaining codes have minimum weight 8. Therefore, only codes $C_{64,11}$ and $C_{64,12}$ do not satisfy the necessary condition given in Lemma 2.1.14 (the minimum weight of the duals have to be at least 6). Due to the memory limitations, we were not able to produce any of the $\mathbb{Z}_4$-codes from these binary codes.

The Code	$[n,k,d]$	0	4	8	12	16	20	24	28	32
$C_{64,1}$,$C_{64,36}$	$[64,26,12]$	1	0	0	7	7434	283675	3653608	16267742	26683930
$C_{64,2}$,$C_{64,6}$	$[64,27,8]$	1	0	1	161	13916	570381	7300951	32544850	53357206
$C_{64,3}$	$[64,27,8]$	1	0	1	168	14070	568904	7306383	32533776	53371122
$C_{64,4}$	$[64,27,8]$	1	0	1	168	14238	567560	7311087	32524368	53382882
$C_{64,5}$	$[64,27,8]$	1	0	1	140	14350	567644	7309743	32527896	53378178
$C_{64,7}$	$[64,27,8]$	1	0	1	210	15218	569002	7280735	32616068	53255258
$C_{64,8}$	$[64,27,8]$	1	0	1	217	13692	570213	7303639	32537794	53366614
$C_{64,9}$	$[64,27,8]$	1	0	29	658	14714	564970	7289667	32612484	53252682
$C_{64,11}$	$[64,27,8]$	1	0	1	238	18354	603862	7236831	32427068	53645018
$C_{64,12}$	$[64,27,8]$	1	0	1	217	18060	606949	7225239	32450882	53615030
$C_{64,13}$	$[64,26,12]$	1	0	0	98	6888	284858	3652880	16266468	26686478
$C_{64,14}$,$C_{64,15}$	$[64,26,12]$	1	0	0	77	7154	283465	3656968	16258922	26695690
$C_{64,16}$	$[64,26,12]$	1	0	0	63	7294	282835	3658648	16255982	26699218
$C_{64,17}$	$[64,26,12]$	1	0	0	112	6804	285040	3652768	16266272	26686870
$C_{64,18}$	$[64,26,12]$	1	0	0	98	7224	282170	3662288	16247652	26709998
$C_{64,19}$	$[64,26,12]$	1	0	0	7	7294	284795	3649688	16275582	26674130
$C_{64,20}$	$[64,27,8]$	1	0	1	126	14322	568358	7306719	32534364	53369946
$C_{64,21}$,$C_{64,23}$	$[64,27,8]$	1	0	1	28	14686	568876	7301231	32548280	53351522
$C_{64,22}$,$C_{64,24}$	$[64,27,8]$	1	0	1	35	14392	570983	7294119	32562294	53334078
$C_{64,25}$	$[64,26,12]$	1	0	0	77	7126	283689	3656184	16260490	26693730
$C_{64,37}$	$[64,26,12]$	1	0	0	21	7070	286097	3645656	16283226	26664722
$C_{64,38}$	$[64,27,8]$	1	0	1	112	14238	569520	7302127	32543968	53357794
$C_{64,39}$	$[64,27,8]$	1	0	1	14	14994	566902	7307615	32535932	53366810
$C_{64,40}$	$[64,27,8]$	1	0	1	28	14630	569324	7299663	32551416	53347602
$C_{64,42}$	$[64,27,8]$	1	0	1	42	14266	571746	7291711	32566900	53328394
$C_{64,43}$	$[64,26,12]$	1	0	0	105	6678	286293	3648120	16275778	26674914
$C_{64,46}$	$[64,26,16]$	1	0	0	0	7588	282688	3656800	16261568	26691574
$C_{64,45}$	$[64,27,8]$	1	0	1	210	13482	572138	7296639	32552004	53348778

Table 4.24: Parameters and the weight distribution of doubly-even codes of length 64 constructed from the designs given in [18]

Codes of length 72

Since $4n-1=71$ gives $n=18$, the construction given in Proposition 3.1.2 cannot be used due to the condition from Theorem 1.2.7. We constructed and analyzed 45 Hadamard $2-(71,35,17)$ designs from incidence matrices given in Table A.1 under $n=71$, obtained from [15]. From the first fifteen matrices we obtained designs $\mathscr{D}_1,\mathscr{D}_2,\ldots,\mathscr{D}_{15}$ which are self-dual or pairwise dual. From the second fifteen block matrices we obtained $\mathscr{D}_{16},\mathscr{D}_{17},\ldots,\mathscr{D}_{30}$ and their duals $\mathscr{D}_{31},\mathscr{D}_{17},\ldots,\mathscr{D}_{45}$. The structure of the full automorphism groups of designs $\mathscr{D}_1,\mathscr{D}_2,\ldots,\mathscr{D}_{45}$ are given in Table 4.25. These designs were previously constructed in [19].

The size of the group	The structure of the group	Designs
136	$\mathbb{Z}_{17}:\mathbb{Z}_8$	$\mathscr{D}_{12}$, $\mathscr{D}_{28}$, $\mathscr{D}_{43}$
68	$\mathbb{Z}_{17}:\mathbb{Z}_4$	$\mathscr{D}_{14}$, $\mathscr{D}_{15}$, $\mathscr{D}_{29}$, $\mathscr{D}_{30}$, $\mathscr{D}_{44}$, $\mathscr{D}_{45}$
34	D_{34}	$\mathscr{D}_1$ $\mathscr{D}_2$, $\mathscr{D}_3$, $\mathscr{D}_4$, $\mathscr{D}_5$, $\mathscr{D}_6$, $\mathscr{D}_7$, $\mathscr{D}_8$, $\mathscr{D}_9$, $\mathscr{D}_{10}$, $\mathscr{D}_{11}$, $\mathscr{D}_{13}$, $\mathscr{D}_{16}$, $\mathscr{D}_{17}$, $\mathscr{D}_{18}$, $\mathscr{D}_{19}$, $\mathscr{D}_{20}$, $\mathscr{D}_{21}$, $\mathscr{D}_{22}$, $\mathscr{D}_{23}$, $\mathscr{D}_{24}$, $\mathscr{D}_{25}$, $\mathscr{D}_{26}$, $\mathscr{D}_{27}$, $\mathscr{D}_{31}$, $\mathscr{D}_{32}$, $\mathscr{D}_{33}$, $\mathscr{D}_{34}$, $\mathscr{D}_{35}$, $\mathscr{D}_{36}$, $\mathscr{D}_{37}$, $\mathscr{D}_{38}$, $\mathscr{D}_{39}$, $\mathscr{D}_{40}$, $\mathscr{D}_{41}$, $\mathscr{D}_{42}$

Table 4.25: The structure of the full automorphism groups of the $2-(71,35,17)$ designs $\mathscr{D}_1,\mathscr{D}_2,\ldots,\mathscr{D}_{45}$

The corresponding $3-(72,36,17)$ designs $\mathscr{D}_1^*$, $\mathscr{D}_2^*$, …, $\mathscr{D}_{45}^*$ give 45 self-dual binary linear codes $C_1,C_2,\ldots,C_{45}$. These codes are distributed in 8 classes of the same weight distribution:

$$(C_1,C_2,C_5,C_6),(C_3,C_4),(C_{22},C_{23},C_{24},C_{25},C_{26},C_{27},C_{28},C_{29}),$$
$$(C_7,C_9),C_8,$$
$$(C_{10},C_{11},C_{12},C_{13},C_{14},C_{15},C_{18},C_{19},C_{34},C_{35},C_{36},C_{37},C_{38},C_{39},C_{40},C_{41}),$$
$$(C_{16},C_{17},C_{20},C_{21},C_{42},C_{43},C_{44},C_{45}),(C_{30},C_{31},C_{32},C_{33}).$$

The weight distribution of these codes is given in Table 4.26.

The Code	$[n,k,d]$	0	4	8	12	16	20	24	28	32	36
C_1	$[72,36,4]$	1	18	153	13872	551412	26721144	495128196	4317481296	16486794990	26066094572
C_3	$[72,36,8]$	1	0	153	6528	277236	18074880	461482884	4404774528	16585890030	25778464256
C_7	$[72,36,4]$	1	630	58905	1947792	30260340	254186856	1251677700	3796297200	7307872110	43434873668
C_8	$[72,36,4]$	1	306	29529	973488	15131700	131807736	839757636	4144182480	11996428590	34462853804
C_{10}	$[72,36,4]$	1	18	425	13328	745892	25862712	495169812	4324680048	16466504606	26093523052
C_{16}	$[72,36,4]$	1	18	153	10608	590580	26505720	495846276	4315865616	16489380078	26063078636
C_{22}	$[72,36,8]$	1	0	289	6256	374476	17645664	461503692	4408373904	16575744838	25792178496
C_{30}	$[72,36,8]$	1	0	153	4896	296820	17967168	461841924	4403966688	16587182574	25776956288

Table 4.26: Parameters and the weight distribution classes of binary codes of length 72

By Lemma 2.1.14, the torsion code of the extremal or near-extremal $\mathbb{Z}_4$-code of length 72 should have minimum weight at least 8. Therefore, only codes C_3, C_{22}, and C_{30} satisfy the necessary condition given in Lemma 2.1.14. Due to the memory limitations, we were not able to produce any of $\mathbb{Z}_4$-codes.

4.3. COMBINATORIAL STRUCTURES

In this section we describe combinatorial structures that we obtained from constructed $\mathbb{Z}_4$-codes, and their residue and torsion codes.

Strongly regular graphs

At the beginning of this subsection we give the method that was used for construction of strongly regular graphs from binary codes $C_{32,1}$, $C_{32,2}$, ..., $C_{32,21}$, and $C_{48,1}$, $C_{48,2}$, ..., $C_{48,15}$ described in subsection 4.1. This method is not new, and it can also be found in [17].

Let C be any of the mentioned codes, and let n be length of C. Further, let w be any weight of the code C such that $S_w = \{c \in C | wt_H(c) = w\}$ is a non-empty set. For $c = (c_1, c_2, \ldots, c_n) \in S_w$, a support of c is defined as:

$$\operatorname{supp}(c) = \{j | c_j = 1,\ j \in 1, 2, \ldots, w\}.$$

Let B be the following set:

$$B = \{supp(c) \,|\, c \in S_w\}.$$

A set of intersection numbers of the set B is the set $I = \{|b \cap b'| \,|\, b, b' \in B, b \neq b'\}$. Let $P = \left(P^{(1)}, P^{(2)}\right)$ be a partition of the set I. Our goal is to construct a strongly regular graph $G = (B, E)$ such that:

$$\{b, b'\} \in E \Leftrightarrow |b \cap b'| \in P^{(1)}. \tag{4.1}$$

Notice that, if such strongly regular graph exists, then its parameters would be $(|B|, k, \lambda, \mu)$. Since every strongly regular graph is also regular graph, from Theorem 1.1.27, we have:

$$|B|\, k = 2\,|E| \Leftrightarrow k = \frac{2\,|E|}{|B|}. \tag{4.2}$$

Therefore, it is necessary that $\frac{2|E|}{|B|}$ is an positive integer. Also, if for a given partition P described graph G is a strongly regular graph, then the graph defined by $G' = (B, E')$ for E' defined as:

$$\{b, b'\} \in E' \Leftrightarrow |b \cap b'| \in P^{(2)},$$

is the complement of the graph G. Therefore, it is sufficient to construct graphs with edges defined by (4.1), and from the partitions P such that $\left|P^{(1)}\right| \leq \left|P^{(2)}\right|$.

Based on this discussion, for the described set I, we would determine all non-empty subsets of the set I that have at most $\lfloor \frac{|I|}{2} \rfloor$ elements, and define $P^{(1)}$ at that subset. If $\left|P^{(1)}\right| = \lfloor \frac{|I|}{2} \rfloor$, then we do not consider its complement. There are $2^{|I|-1}-1$ such sets. For the fixed set $P^{(1)}$ we determine $|E|$ (where E is defined by (4.1)). If k, given by (4.2), is not an positive integer we discard the set $P^{(1)}$ as possible set for the incidence of a strongly regular graph. For all feasible sets $P^{(1)}$, we constructed graphs $G = (B,E)$ and used computer software GAP ([50]) to determine if G is a strongly regular graph.

Example 4.3.1. We give an example of the described method on the code $C_{32,3}$, and for the weight 4. Notice, from Table 4.3, that there are 28 codewords of the weight 4 in the code $C_{32,3}$. Therefore, the number of vertices in the graph is $|V| = 28$. These codewords form the following set of the supports, S_4:

$$
\begin{array}{llll}
\{1,2,8,24\}, & \{1,3,8,25\}, & \{1,4,8,26\}, & \{1,5,8,27\}, \\
\{1,6,8,28\}, & \{1,7,8,29\}, & \{1,8,20,21\}, & \{2,3,24,25\}, \\
\{2,4,24,26\}, & \{2,5,24,27\}, & \{2,6,24,28\}, & \{2,7,24,29\}, \\
\{2,20,21,24\}, & \{3,4,25,26\}, & \{3,5,25,27\}, & \{3,6,25,28\}, \\
\{3,7,25,29\}, & \{3,20,21,25\}, & \{4,5,26,27\}, & \{4,6,26,28\}, \\
\{4,7,26,29\}, & \{4,20,21,26\}, & \{5,6,27,28\}, & \{5,7,27,29\}, \\
\{5,20,21,27\}, & \{6,7,28,29\}, & \{6,20,21,28\}, & \{7,20,21,29\}.
\end{array}
$$

It can be checked that every two distinct elements of the set S_4 either intersect in 2 points, or are disjoint. Therefore, $I = \{0,2\}$. The only non-trivial partition of I is $P = (\{0\},\{2\})$. This is also confirmed from $2^{|I|-1}-1 = 2^{2-1}-1 = 1$. We can freely chose whether $P^{(1)} = \{0\}$ or $P^{(1)} = \{2\}$, since they are pair of complementary subsets in I. Let $P^{(1)} = \{2\}$. One can verify that the number of pairs of the distinct blocks from S_4 that intersect in 2 elements is 168. Therefore the number of edges in the graph that we want to construct is $|E| = 168$. Therefore, $k = \frac{2 \cdot 168}{28} = 12$. Since this is a positive integer we can try to construct a strongly regular graph. We verified by GAP that this is indeed

a strongly regular graph with parameters $(28,12,6,4)$. This graph is isomorphic to the triangular graph $T(8)$. It is also given in the second row of Table 4.27.

As stated at the beginning of this subsection, we applied the described method on the binary codes of lengths 32 and 48 from Subsection 4.1 that were obtained from construction given in Proposition 3.1.2. The graphs constructed from the codes of length 32 are presented in Table 4.27. The graphs constructed from the codes of length 48 are presented in Table 4.28. The graphs from codes of the length 48 were previously constructed in [17]. No new strongly regular graphs were obtained. In both tables, we give the binary code from which graphs are obtained, the weight w that was used to construct set S_w, the subset of the intersection set $P^{(1)}$ that defines the edges of the graph, the parameters (v,k,λ,μ) of the obtained strongly regular graph and finally the known strongly regular graphs to which obtained graphs are isomorphic. In Table 4.28, we obtained a strongly regular graph from $C_{48,3}$ and set S_{16} with parameters $(495,238,109,119)$. By the entry $J(12,4)$, $d \in \{2,4\}$, we denote the strongly regular graph obtained as the distance graph of the Johnson graph $J(12,4)$ for distance set $\{2,4\}$. In the same table, we obtained a strongly regular graph from $C_{48,9}$ with parameters $(77,16,0,4)$ which is isomorphic to the graph constructed from blocks of quasi-symmetric Steiner triple system $S(3,6,22)$ constructed in [48]. All of isomorphisms were checked by GAP. The information on the known strongly regular graphs were obtained from [10].

The code	Codewords weight w	Incidence intersection	Parameters of SRG	Known graph
$C_{32,2}$	8	$\{4\}$	$(28,12,6,4)$	$T(8)$
$C_{32,3}$	4	$\{2\}$	$(28,12,6,4)$	$T(8)$
$C_{32,12}$	4	$\{2\}$	$(28,12,6,4)$	$T(8)$
$C_{32,15}$	8	$\{4\}$	$(28,12,6,4)$	$T(8)$
$C_{32,16}$	4	$\{2\}$	$(120,28,14,4)$	$T(16)$
$C_{32,19}$	4	$\{2\}$	$(36,14,7,4)$	$T(8)$

Table 4.27: Strongly regular graphs obtained from binary codes of length 32

The code	Codewords weight	Incidence intersection	Parameters of SRG	Known graph
$C_{48,3}$	8	$\{4\}$	$(66,20,10,4)$	$T(12)$
$C_{48,3}$	16	$\{0,8\}$	$(495,238,109,119)$	$J(12,4)$ $d \in \{2,4\}$
$C_{48,7}$	4	$\{2\}$	$(78,22,11,4)$	$T(13)$
$C_{48,9}$	12	$\{0\}$	$(77,16,0,4)$	$S(3,6,22)$ $\lvert B_i \cap B_j \rvert = 0$
$C_{48,14}$	4	$\{2\}$	$(276,44,22,4)$	$T(14)$

Table 4.28: Strongly regular graphs obtained from binary codes of length 48

Combinatorial designs

In this section we will present the combinatorial designs that were constructed from obtained extremal $\mathbb{Z}_4$-codes. The method that we used was first described in [3]. The following theorem can be found in the same paper.

Theorem 4.3.2. Let C be an extremal Type II $\mathbb{Z}_4$-code of length $n = 24, 32, 40$ such that the minimum weight of $Res(C)$ is at least 8. Let S_1, S_2, S_3, S_4 be the sets defined as follows:

$$S_1 = \left\{ v \in C \middle| n_2(v) = 4, n_1(v) = n_3(v) = 0 \right\},$$
$$S_2 = \left\{ v \in C \middle| n_2(v) = 0, n_1(v) = n_3(v) = 16 \right\},$$
$$S_3 = \left\{ v \in C \middle| n_2(v) = 1, n_1(v) = n_3(v) = 12 \right\},$$
$$S_4 = \left\{ v \in C \middle| n_2(v) = 2, n_1(v) = n_3(v) = 8 \right\}.$$

The following statements hold:

(i) $wt_H(C) \geq 4$, $wt_L C \geq 8$, $wt_E(C) = 16$,

(ii) S_1 is the set of all codewords in C of Hamming weight 4,

(iii) S_1 is the set of all codewords in C of Lee weight 8,

(iv) $S_1 \cup S_2 \cup S_3 \cup S_4$ is the set of all codewords in C of Euclidean weight 16,

(v) The set of supports of all codewords in S_1 is the set of supports of codewords of weight 4 in $Tor(C)$.

The following remark can also be found in [3].

Remark 4.3.3. Let d be the minimum weight of $Res(C)$ in previous theorem. If $d = 12$, then S_4 is empty. If $d = 16$ then sets S_3 and S_4 are empty, and the set of supports of all codewords in S_2 is the set of the supports of the codewords of minimum weight in $Tor(C)$.

For the extremal $\mathbb{Z}_4$-codes of length 32 presented in Section 4.1, we constructed the sets S_1, S_2, S_3 and S_4 and checked whether the supports of codewords in this sets form a design. In Table 4.29 we give the summary of constructed designs. All of the designs were constructed from the supports of codewords from S_1. From other sets no designs were constructed. All of the constructed designs are 1-designs. The summary of obtained designs is given in Table 4.29. All binary codes from Table 4.29 except $C_{32,15}$ and $C_{32,16}$ are residue codes of the extremal Type II $\mathbb{Z}_4$-codes. Since codes $C_{32,1}$, $C_{32,7}$, $C_{32,10}$, $C_{32,11}$ and $C_{32,14}$ have minimum weight at least 8, by the statement (v) of Theorem 4.3.2 they depend only on the code $Tor(C)$. The codes $C_{32,9}$ and $C_{32,13}$ have the minimum weight 4 and therefore they do not satisfy that statement. For each of 11 extremal Type II $\mathbb{Z}_4$-codes constructed from $C_{32,9}$ and $C_{32,13}$ we constructed designs from the set S_1 and checked the isomorphism of the obtained designs by MAGMA. All of constructed designs from each of these codes are isomorphic. Therefore, from each of the codes $C_{32,7}$, $C_{32,7}$, $C_{32,9}$, $C_{32,10}$, $C_{32,11}$, $C_{32,13}$ and $C_{32,14}$ (or precisely from their duals), up to isomorphism, one 1-design is constructed.

Codes $C_{32,15}$ and $C_{32,16}$ are binary residue codes of the Type I extremal $\mathbb{Z}_4$-codes, so they are not covered by Theorem 4.3.2. We tried the same construction as for Type II $\mathbb{Z}_4$-codes and we also obtained, up to isomorphism, only one design from each of the codes. The quasi-symmetric designs obtained from $C_{32,9}$ and $C_{32,16}$ are resolvable.

The code	Set of codewords	Parameters of design	Size of the Automorphism group	Number of blocks	Block intersection numbers
$C_{32,1}$	S_1	$1-(32,4,155)$	$2^{28}\cdot 3^4\cdot 5\cdot 7^3$	1240	$0,1,2$
$C_{32,7}$	S_1	$1-(32,4,43)$	$2^{31}\cdot 3^4\cdot 5^2\cdot 7^2$	344	$0,1,2$
$C_{32,9}$	S_1	$1-(32,4,7)$	$2^{18}\cdot 3^2\cdot 7$	56	$0,2$
$C_{32,10}$	S_1	$1-(32,4,91)$	$2^{24}\cdot 3^3\cdot 5\cdot 7$	728	$0,1,2$
$C_{32,11}$	S_1	$1-(32,4,21)$	$2^{21}\cdot 3^2\cdot 7^2$	168	$0,1,2$
$C_{32,13}$	S_1	$1-(32,4,7)$	$2^{10}\cdot 3^3\cdot 5\cdot 7$	56	$0,1,2$
$C_{32,14}$	S_1	$1-(32,4,35)$	$2^{21}\cdot 3^4\cdot 5^2\cdot 7^2$	280	$0,1,2$
$C_{32,15}$	S_1	$1-(32,4,43)$	$2^{15}\cdot 3^2\cdot 5\cdot 7\cdot 31$	344	$0,1,2$
$C_{32,16}$	S_1	$1-(32,4,15)$	$2^{31}\cdot 3^6\cdot 5^3\cdot 7^2\cdot 11\cdot 13$	120	$0,2$

Table 4.29: Combinatorial designs obtained from extremal $\mathbb{Z}_4$-codes of length 32

CONCLUSION

In Chapter 2, we presented a modification of an existing algorithm for the construction of self-dual $\mathbb{Z}_4$-codes. The main goal of the modification was to increase the number of codes that are checked for extremality or near-extremality with one calculation of the minimum Euclidean weight. We have showed that, if $G(B)$ is the generator matrix of a self-dual code determined by $k \times k$ binary matrix B, then the number of codes that can be checked for extremality (near-extremality) is equal to the number of 0-elements in the upper diagonal of B. We have also showed that, the order in which matrices B are chosen, effects the number of calculations of minimum Euclidean weight needed to cover the search space, since the search space has the structure of the hypercube graph. If matrices B were taken in lexicographical order, then the number of calculations of minimum Euclidean weights needed to cover the whole search space was halved compared to the usual approach without modification. We showed, experimentally, that this number is significantly smaller when matrices B are taken randomly. We tested the modified algorithm on the $[16,6,4]$ binary code, and observed its efficiency. We concluded that the modified algorithm is not good for the classification of the extremal and near-extremal $\mathbb{Z}_4$-codes, but still checks the extremality of significantly more $\mathbb{Z}_4$-codes than the unmodified version.

After the analysis of the modified algorithm, in Chapter 3 we developed a new method for the construction of Hadamard designs from skew-symmetric Hadamard matrices. This construction gives a series of at most 54 Hadamard designs on $4n-1$ points from one skew incidence matrix of Hadamard design on $n-1$ points. Since the incidence matrices of Hadamard 3-designs span doubly-even binary codes, this is a nice way to construct input data for the modified algorithm.

In the end of this work we applied the modified method to construct new extremal and near-extremal $\mathbb{Z}_4$-codes. The results were presented in the Chapter 4. From the only

skew-symmetric Hadamard matrix of order 8 (up to the equivalence), we constructed 21 nonequivalent binary codes. On these codes we applied both the standard, and the modified algorithms in attempt to construct extremal and near-extremal $\mathbb{Z}_4$-codes. With the standard method, we obtained at least 4 new Type II extremal $\mathbb{Z}_4$-codes, one of each of the types $4^7 2^{18}$, $4^{10}2^{12}$ and two of the type $4^9 2^{14}$. Also, we obtained at least 443 nonequivalent Type I $\mathbb{Z}_4$-codes of which: 5 of type $4^{10}2^{12}$, 10 of type $4^{12}2^8$, 182 of type $4^{13}2^6$, 6 of type $4^{15}2^2$, and 240 of type 4^{16}. With the standard method we were unable to construct any extremal $\mathbb{Z}_4$-codes from 8 binary codes of length 32. Only when the modified method was applied, we successfully constructed extremal $\mathbb{Z}_4$-codes with this residue codes. At least 64 new extremal Type II $\mathbb{Z}_4$-codes were constructed: one code of type $4^9 2^{14}$, 6 codes of type $4^{10}2^{12}$, 27 codes of type $4^{15}2^2$, and 30 codes of type 4^{16}. Additionally, 11 Type I extremal $\mathbb{Z}_4$-codes were constructed: 2 codes of type $4^9 2^{14}$, 2 codes of type $4^7 2^{18}$, and 7 codes of type $4^{10}2^{12}$. New numbers of known extremal Type II $\mathbb{Z}_4$-codes of length 32 are given in Table 4.30.

Type	$4^6 2^{20}$	$4^7 2^{18}$	$4^8 2^{16}$	$4^9 2^{14}$	$4^{10}2^{12}$	$4^{11}2^{10}$
#	1	7	27	16	11	3
Type	$4^{12}2^8$	$4^{13}2^6$	$4^{14}2^4$	$4^{15}2^2$	4^{16}	
#	1	220	5148	383	164	

Table 4.30: The type and the new numbers of known extremal Type II $\mathbb{Z}_4$-codes of length 32

For the length 40, we were unable to use developed method for the construction of Hadamard matrices, so we used Hadamard matrices given in Table A.1. From them, we obtained, up to the equivalence, 3 self-dual doubly-even binary codes. With the modified algorithm, we obtained 11 new extremal Type II $\mathbb{Z}_4$-codes of type 4^{20}. Also we obtained 4090 extremal Type I $\mathbb{Z}_4$-codes of the same type. We applied the modified algorithm on the residue codes of the known extremal $\mathbb{Z}_4$-codes of length 40 from [3] and [28] (of types on which a small number of extremal codes is known). From those codes, we obtained

at least 29 new extremal Type II $\mathbb{Z}_4$-codes: one of type $4^7 2^{26}$, 3 of type $4^{10}2^{20}$, 8 of type $4^{11}2^{18}$, 16 of type $4^{15}2^{10}$, and one of type $4^{16}2^8$. On the same residue codes, we also constructed 54 new extremal Type I $\mathbb{Z}_4$-codes: 7 of type $4^7 2^{26}$, 9 of type $4^{10}2^{20}$, 7 of type $4^{11}2^{18}$, 17 of type $4^{15}2^{10}$, and 14 of type $4^{16}2^8$. Therefore, the new numbers of known extremal $\mathbb{Z}_4$-codes of Type II are given in Table 4.31.

Type	$4^7 2^{26}$	$4^8 2^{24}$	$4^9 2^{22}$	$4^{10}2^{20}$	$4^{11}2^{18}$	$4^{12}2^{16}$	$4^{13}2^{14}$
#	4	228	100	5	11	20	15
Type	$4^{14}2^{12}$	$4^{15}2^{10}$	$4^{16}2^8$	$4^{17}2^6$	$4^{18}2^4$	$4^{19}2^2$	4^{20}
#	5	19	2	134	902	432	94354

Table 4.31: The type and the numbers of known extremal Type II $\mathbb{Z}_4$-codes of length 40

For the length 48, binary codes constructed from Hadamard matrices obtained by method developed in this thesis were not good for construction of extremal or near-extremal $\mathbb{Z}_4$-codes. We also tried to construct extremal $\mathbb{Z}_4$-codes from binary codes obtained from Hadamard matrices given in [38]. From these codes we constructed already known extremal Type II $\mathbb{Z}_4$-code given in [33]. Also, we constructed at least two nonequivalent near-extremal $\mathbb{Z}_4$-codes that are neighbors of that extremal Type II $\mathbb{Z}_4$-code.

For the codes of greater lengths, the method was unsuccessful in construction of extremal and near-extremal $\mathbb{Z}_4$-codes.

In the end, we used method given in [17] to construct strongly regular graphs from binary codes of length 32 and 48 obtained from Hadamard matrices constructed by method developed in Chapter 3. Since these binary codes of length 48 were already constructed in [17], results obtained in this thesis were confirmation of the results given in that article. For the length 32, we obtained already known strongly regular graphs $T(8)$ and $T(16)$. We also used the method given in [3] to construct 1-designs from the obtained extremal $\mathbb{Z}_4$-codes of length 32. Up to the isomorphism, 9 designs were obtained.

BIBLIOGRAPHY

[1] Ban, S.: *Konstrukcija ekstremalnih $\mathbb{Z}_4$-kodova tipa II*. PhD thesis, Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet, Zagreb, 2019. https://urn.nsk.hr/urn:nbn:hr:217:036156. ↑ 2, 3, 20, 33, 74, 75.

[2] Ban, S., Crnković, D., Mravić, M., and Rukavina, S.: *New extremal type II $\mathbb{Z}_4$ -codes of length 32 obtained from Hadamard matrices*. Discrete Mathematics, Algorithms and Applications, 11, August 2019. ↑ 2, 59, 74.

[3] Ban, S. and Rukavina, S.: *On some new extremal type II $\mathbb{Z}_4$-codes of length 40*. Math. Commun., 25:253–268, 2020. ↑ 34, 79, 80, 104, 105, 108, 109.

[4] Betsumiya, K., Harada, M., and Munemasa, A.: *A complete classification of doubly even self-dual codes of length 40*, 2012. ↑ 34, 35, 78.

[5] Betty, R. A. L. and Munemasa, A.: *Classification of self-dual codes of length 20 over z4 and length at most 18 over $\mathbb{F}_2 + u\mathbb{F}_2$, in: Cryptography and coding - 17th ima international conference, imacc 2019, proceedings*. pages 64–77, 2019. ↑ 32.

[6] Bollobás, B.: *Modern Graph Theory*. Graduate Texts in Mathematics. Springer New York, 1998, ISBN 9780387984889. ↑ 4.

[7] Bonnecaze, A., Solé, P., Bachoc, C., and Mourrain, B.: *Type II codes over $\mathbb{Z}_4$*. IEEE Transactions on Information Theory, 43:969–976, January 1997. ↑ 2, 26, 35, 92.

[8] Bosma, W., Cannon, J., and Playoust, C.: *The magma algebra system. I. the user language*. Journal of Symbolic Computation, 24(3-4):235–265, 1997. ↑ 37, 70.

[9] Bouyuklieva, S., Bouyukliev, I., and Harada, M.: *Some extremal self-dual codes and unimodular lattices in dimension 40*. Finite Fields and Their Applications, 21:67–83, 2013, ISSN 1071-5797. https://www.sciencedirect.com/science/article/pii/S1071579713000178. ↑ 35, 78, 88.

[10] Brouwer, A. E.: *Parameters of Strongly Regular Graphs,*. https://www.win.tue.nl/~aeb/graphs/srg/srgtab.html, Acessed on: 21/01/2022. ↑ 103.

[11] Cameron, P. J. and Lint, J. H.: *Graphs, Codes and Designs*. London Mathematical Society Lecture Note Series. Cambridge University Press, 1980. ↑ 4.

[12] Chan, K. H.: *Three new methods for construction of extremal Type II $\mathbb{Z}_4$-codes*. PhD thesis, University of Illinois at Chicago, 2012. ↑ 33, 34, 76.

[13] Conway, J.H. and Sloane, N.J.A.: *Self-dual codes over the integers modulo 4*. Journal of Combinatorial Theory, Series A, 62(1):30–45, 1993, ISSN 0097-3165. https://www.sciencedirect.com/science/article/pii/0097316593900700. ↑ 2, 32.

[14] Craigen, R. and Kharaghani, H.: *Hadamard Matrices and Hadamard Designs, in: Handbook of Combinatorial Designs, 2nd ed., (C. J. Colbourn and J. H. Dinitz, Eds.)*. Discrete Mathematics and Its Applications. Taylor & Francis, 2007, ISBN 9781584885061.

[15] Crnković, D.: *private communication*. ↑ 77, 93, 99, 117.

[16] Crnković, D. and Rukavina, S.: *Some symmetric (47,23,11) designs*. Glasnik Matematički, 38 (58)(1):1–9, 2003. ↑ 89.

[17] Crnković, D. and Mikulić, V.: *Self-orthogonal doubly-even codes from Hadamard matrices of order 48*. Advances and Applications in Discrete Mathematics, 1, January 2008. ↑ 91, 101, 103, 109.

[18] Crnković, D. and Pavčević, M.-O.: *Some new symmetric designs with parameters (64,28,12)*. Discrete Mathematics, 237:109–118, June 2001. ↑ xii, 96, 98.

[19] Crnković, D and S Rukavina: *On symmetric (71, 35, 17) designs*. Math. Maced, 2:51–58, 2004. ↑ 99.

[20] Fields, J., Gaborit, P., Leon, J.S., and Pless, V.: *All self-dual $\mathbb{Z}_4$ codes of length 15 or less are known*. IEEE Transactions on Information Theory, 44(1):311–322, 1998. ↑ 32.

[21] Gaborit., P.: *Mass formulas for self-dual codes over $\mathbb{Z}_4$ and $\mathbb{F}_q + u\mathbb{F}_q$ rings*. IEEE Transactions on Information Theory, 42:1222–1228, 1996. ↑ 2, 28, 29, 30.

[22] Gaborit, P. and Harada, M.: *Construction of extremal type II codes over $\mathbb{Z}_4$*. Designs, Codes and Cryptography, 16(3):257–269, 1999. ↑ 26, 33, 37, 76.

[23] Gulliver, T. A. and Harada, M.: *Certain self-dual codes over $\mathbb{Z}_4$ and the odd leech lattice*. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pages 130–137. Springer Berlin Heidelberg, 1997. ↑ 32.

[24] Gulliver, T. A. and Harada, M.: *Extremal double circulant type II codes over $\mathbb{Z}_4$ and construction of 5-(24, 10, 36) designs*. Discrete Math., 194(1–3):129–137, 1999, ISSN 0012-365X. https://doi.org/10.1016/S0012-365X(98)00035-1. ↑ 1.

[25] Hamming, R. W.: *Error detecting and error correcting codes*. The Bell System Technical Journal, 29(2):147–160, 1950. ↑ 1.

[26] Hammons, A. R., Kumar, P. V., Calderbank, A. R., Sloane, N. J. A., and Solé, P.: *The $\mathbb{Z}_4$-linearity of Kerdock, Preparata, Goethals, and related codes*. IEEE Transactions on Information Theory, 40(2):301–319, 1994. ↑ 1.

[27] Harada, M.: *Extremal type II $\mathbb{Z}_4$-codes of lengths 56 and 64*. Journal of Combinatorial Theory, Series A, 117:1285–1288, November 2010. ↑ 2, 31, 32, 35.

[28] Harada, M.: *On the residue codes of extremal type II $\mathbb{Z}_4$-codes of lengths 32 and 40*. Computing Research Repository - CORR, 311, April 2011. ↑ 33, 34, 74, 75, 76, 79, 80, 108.

[29] Harada, M.: *Optimal self-dual $\mathbb{Z}_4$-codes and a unimodular lattice in dimension 41*. Finite Fields Their Appl., 18(3):529–536, 2011. https://doi.org/10.1016/j.ffa.2011.11.004. ↑ 2, 35.

[30] Harada, M.: *Extremal type I $\mathbb{Z}_k$-codes and k-frames of odd unimodular lattices*. IEEE Transactions on Information Theory, 61(1):72–81, 2015. ↑ 2, 27, 36.

[31] Harada, M.: *Note on the residue codes of self-dual $\mathbb{Z}_4$-codes having large minimum Lee weights*. Advances in Mathematics of Communications, 10(4):695–706, 2016. ↑ 2, 32, 35.

[32] Harada, M.: *Extremal type II $\mathbb{Z}_4$-codes constructed from binary doubly even self-dual codes of length* 40, 2017. ↑ 34, 78.

[33] Harada, M., Kitazume, M., Munemasa, A., and Venkov, B.: *On some self-dual codes and unimodular lattices in dimension 48*. European Journal of Combinatorics, 26:543–557, July 2005. ↑ 2, 35, 92, 109.

[34] Harada, M., Lam, C. H., and Munemasa, A.: *Residue codes of extremal Type II $\mathbb{Z}_4$-codes and the moonshine vertex operator algebra*. Mathematische Zeitschrift, 274(1-2):685–700, 2013, ISSN 1432-1823. http://dx.doi.org/10.1007/s00209-012-1091-z. ↑ 2, 32.

[35] Harada, M. and A. Munemasa: *On the classification of self-dual $\mathbb{Z}_k$-codes, In: Cryptography and coding*. pages 78–90, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg, ISBN 978-3-642-10868-6. ↑ 32.

[36] Huffman, W. C. and Pless, V.: *Fundamentals of Error-Correcting Codes*. Cambridge University Press, 2003. ↑ 4, 27.

[37] Kerdock, A.M.: *A class of low-rate nonlinear binary codes*. Information and Control, 20(2):182–187, 1972, ISSN 0019-9958. https://www.sciencedirect.com/science/article/pii/S0019995872903762. ↑ 1.

[38] Kotsireas, I. S., Koukouvinos, C., and Seberry, J.: *Hadamard ideals and Hadamard matrices with two circulant cores*. European Journal of Combinatorics, 27(5):658–

668, 2006, ISSN 0195-6698. https://www.sciencedirect.com/science/article/pii/S0195669805000557. ↑ 91, 109.

[39] Koukouvinos, C. and Stylianou, S.: *On Skew-Hadamard matrices*. Discrete Math., 308(13):2723–2731, 2008. https://doi.org/10.1016/j.disc.2006.06.037. ↑ 13.

[40] MacWilliams, F. J. and Sloane, N. J. A.: *The Theory of Error-Correcting Codes*. North-Holland, Amsterdam, 1977, ISBN 9780444851932. ↑ 4.

[41] Munemasa, A.: *Extremal type II $\mathbb{Z}_4$-codes of length 24 and triply even binary codes of length 48*, 2014. http://www.math.is.tohoku.ac.jp/~munemasa/documents/20140812.pdf. ↑ 2, 32.

[42] Orlik, P., Wallis, W.D., Street, A. P., and Seberry, J.: *Combinatorics: Room Squares, Sum-free Sets, Hadamard Matrices*. Number br. 291-292 in *Combinatorics: Room Squares, Sum-free Sets, Hadamard Matrices*. Springer-Verlag, 1972, ISBN 9780387060149. ↑ 4.

[43] Pless, V., Leon, J. S., and Fields, J.: *All $\mathbb{Z}_4$ codes of type II and length 16 are known*. J. Comb. Theory, Ser. A, 78:32–50, 1997. ↑ 2, 32.

[44] Pless, V., Solé, P., Qian, Z., and Antipolis, S.: *Cyclic self-dual $\mathbb{Z}_4$-codes*. Finite Fields and Their Applications, 1997. ↑ 33, 34.

[45] Preparata, F. P.: *A class of optimum nonlinear double-error-correcting codes*. Information and Control, 13(4):378–400, 1968, ISSN 0019-9958. https://www.sciencedirect.com/science/article/pii/S0019995868908747. ↑ 1.

[46] Rains, E. M. and Sloane, N. J. A: *The shadow theory of modular and unimodular lattices*. Journal of Number Theory, 73(2):359–389, 1998, ISSN 0022-314X. https://www.sciencedirect.com/science/article/pii/S0022314X9892306X. ↑ 26.

[47] Shannon, C. E.: *A mathematical theory of communication*. The Bell System Technical Journal, 27(3):379–423, 1948. ↑ 1.

[48] Shrikhande, M. S.: *Quasi-symmetric designs, in: The Handbook of Combinatorial Designs, Second Edition (C. J. Colbourn and J. H. Dinitz, Eds.)*. CRC Press, 2007. ↑ 10, 103.

[49] Stinson, D.R.: *Combinatorial Designs: Construction and Analysis*. Springer Nature Book Archives Millennium. Springer, 2004, ISBN 9780387954875. ↑ 4.

[50] The GAP Group: *GAP - Groups, Algorithms, and Programming, Version 4.11.1*, 2021. https://www.gap-system.org. ↑ 70, 102.

[51] Tonchev, V. D.: *Codes, in: Handbook of Combinatorial Designs, 2nd ed., (C. J. Colbourn and J. H. Dinitz, Eds.)*. Discrete Mathematics and Its Applications. Taylor & Francis, 2007, ISBN 9781584885061. ↑ 20.

CURRICULUM VITAE

Matteo Mravić was born on the 10th of March, 1993 in Rijeka where he finished primary and secondary school. He graduated from Graduate study of Discrete Mathematics and its Applications on Department of Mathematics, University of Rijeka in June 2016. In the same year, he enrolled in doctoral program in mathematics which is coordinated by the Department of Mathematics of the Faculty of Science, University of Zagreb. He is employed as teaching assistant at the Department of Mathematics, University of Rijeka since 2016. Since then, he is also a member of the Society of Mathematicians and Physicists Rijeka and the Alumni Club, Department of Mathematics, University of Rijeka. He is a member of the Seminar on Finite Mathematics in Rijeka, as part of which he held a series of seminars.

A. THE INPUT HADAMARD MATRICES OF SPORADIC CASES

In this section incidence matrices of designs used in constructions are given [15]. In Table A.1 these matrices are written in the following way. The column n denotes the number of points in corresponding design. In the column Matrix the rows of each matrix are written in the hexadecimal form. For example, the first incidence matrix of the design on 39 points is:

$$\begin{bmatrix} 0\times 3FFFF80000 \\ 0\times 5ED4838CD1 \\ 0\times 4F6A45C668 \\ \vdots \end{bmatrix}.$$

In order to obtain the incidence binary matrix, one needs to convert each row in the binary number:

$$0\times 3FFFF80000_{16} = 274877382656_{10} = 1111111111111111110000000000000000000_2,$$

$$0\times 5ED4838CD1 = 407292316881_{10} = 101111011010100100000111000110011010001_2,$$

$$0\times 4F6A45C668 = 341085374056_{10} = 100111101101010010001011100011001101000_2.$$

Since the binary number 1111111111111111110000000000000000000 have 38 digits, and we have a design on 39 points, a leading zero is added to the beggining of the row. Then the previous matrix converts to:

$$\begin{bmatrix} 0111111111111111111000000000000000000 \\ 1011110110101001000001110001100110100001 \\ 1001111011010100100010111000110011010000 \\ \vdots \end{bmatrix}.$$

Table A.1: Incidence matrices of input designs

n	Matrix
39	0x3FFFF80000, 0x5ED4838CD1, 0x4F6A45C668, 0x47B522E334, 0x43DA91719A, 0x41ED48B8CD, 0x60F6A45C66, 0x507B522E33, 0x483DAD1719, 0x641ED68B8C, 0x520F6B45C6, 0x6907B1A2E3, 0x5483DCD171, 0x6A41EE68B8, 0x7520F3345C, 0x5A90799A2E, 0x6D4838CD17, 0x76A41C668B, 0x7B520E3345, 0x7DA90719A2, 0x11663FDA90, 0x28B31BED48, 0x345989F6A4, 0x3A2CC0FB52, 0x1D16607DA9, 0xE8B343ED4, 0x7459A1F6A, 0x23A2C90FB5, 0x31D16487DA, 0x18E8B243ED, 0xC745D21F6, 0x263A2A90FB, 0x331D15487D, 0x198E8EA43E, 0x2CC743521F, 0x1663A5A90F, 0xB31D6D487, 0x598EF6A43, 0x22CC77B521,
	0x3FFFF80000, 0x5ED4822CC7, 0x4F6A451663, 0x47B5268B31, 0x43DA974598, 0x41ED4BA2CC, 0x60F6A1D166, 0x507B50E8B3, 0x483DAC7459, 0x641ED63A2C, 0x520F6B1D16, 0x6907B18E8B, 0x5483DCC745, 0x6A41EE63A2, 0x7520F331D1, 0x5A907D98E8, 0x6D483ACC74, 0x76A419663A, 0x7B5208B31D, 0x7DA904598E, 0x1C668FDA90, 0x2E3343ED48, 0x1719A1F6A4, 0xB8CD0FB52, 0x5C6687DA9, 0x22E3343ED4, 0x11719A1F6A, 0x28B8C90FB5, 0x345C6487DA, 0x1A2E3243ED, 0xD171D21F6, 0x268B8A90FB, 0x3345C5487D, 0x19A2E6A43E, 0xCD173521F, 0x668BDA90F, 0x23345ED487, 0x319A2F6A43, 0x38CD17B521,
	0x3FFFF80000, 0x5D44B1F216, 0x4EA258F90B, 0x67512C7C85, 0x73A8963E42, 0x59D44B1F21, 0x6CEA258F90, 0x567512C7C8, 0x4B3A8963E4, 0x659D40B1F2, 0x52CEA058F9, 0x4967542C7C, 0x44B3AA163E, 0x6259D10B1F, 0x512CEC858F, 0x68967642C7, 0x544B3F2163, 0x6A259F90B1, 0x7512CFC858, 0x7A8963E42C, 0xD09F65BA8, 0x684FB2DD4, 0x23427996EA, 0x31A138CB75, 0x38D09C65BA, 0x3C684A32DD, 0x3E3425196E, 0x1F1A128CB7, 0xF8D0D465B, 0x27C686A32D, 0x13E3475196, 0x9F1A3A8CB, 0x4F8D5D465, 0x27C6EEA32, 0x213E337519, 0x109F1DBA8C, 0x284F8ADD46, 0x3427C16EA3, 0x1A13E4B751,
	0x3FFFF80000, 0x5D44B1A13E, 0x4EA258D09F, 0x67512C684F, 0x73A8963427, 0x59D44F1A13, 0x6CEA278D09, 0x567517C684, 0x4B3A8BE342, 0x659D41F1A1, 0x52CEA4F8D0, 0x4967527C68, 0x44B3A93E34, 0x6259D09F1A, 0x512CE84F8D, 0x68967427C6, 0x544B3A13E3, 0x6A259D09F1, 0x7512CE84F8, 0x7A8963427C, 0xF90B65BA8, 0x7C85B2DD4, 0x23E42996EA, 0x31F210CB75, 0x18F90C65BA, 0x2C7C8232DD, 0x163E45196E, 0xB1F228CB7, 0x58F95465B, 0x2C7CEA32D, 0x2163E75196, 0x10B1F3A8CB, 0x858FDD465, 0x242C7EEA32, 0x32163B7519, 0x390B1DBA8C, 0x3C858ADD46, 0x3E42C16EA3, 0x1F2164B751,
	0x3FFFF80000, 0x5C668BDA90, 0x6E3341ED48, 0x5719A0F6A4, 0x4B8CD07B52, 0x45C6683DA9, 0x62E3341ED4, 0x51719A0F6A, 0x68B8C907B5, 0x745C6483DA, 0x5A2E3241ED, 0x4D171D20F6, 0x668B8A907B, 0x7345C5483D, 0x59A2E6A41E, 0x4CD173520F, 0x4668BDA907, 0x63345ED483, 0x719A2F6A41, 0x78CD17B520, 0x12B7DD338, 0x2095BAE99C, 0x304AD974CE, 0x382568BA67, 0x3C12B45D33, 0x1E095E2E99, 0x2F04AF174C, 0x3782538BA6, 0x1BC129C5D3, 0x2DE094E2E9, 0x16F04E7174, 0x2B782338BA, 0x15BC119C5D, 0xADE0CCE2E, 0x256F026717, 0x12B785338B, 0x95BC699C5, 0x4ADE74CE2, 0x256F3A671,
	0x3FFFF80000, 0x5C668A5EC1, 0x6E33452F60, 0x5719A297B0, 0x4B8CD14BD8, 0x45C668A5EC, 0x62E33052F6, 0x517198297B, 0x68B8CC14BD, 0x745C660A5E, 0x5A2E33052F, 0x4D171D8297, 0x668B8EC14B, 0x7345C760A5, 0x59A2E7B052, 0x4CD173D829, 0x4668BDEC14, 0x63345AF60A, 0x719A297B05, 0x78CD14BD82, 0x106F4DD338, 0x2837A2E99C, 0x141BD174CE, 0xA0DE8BA67, 0x2506F45D33, 0x12837E2E99, 0x2941BF174C, 0x34A0DB8BA6, 0x3A5069C5D3, 0x3D2834E2E9, 0x1E941E7174, 0x2F4A0B38BA, 0x37A5019C5D, 0x1BD284CE2E, 0xDE9426717, 0x6F4A5338B, 0x37A5699C5, 0x1BD2F4CE2, 0x20DE93A671,
	0x3FFFF80000, 0x5C668A0DE9, 0x6E334506F4, 0x5719A2837A, 0x4B8CD141BD, 0x45C66CA0DE, 0x62E332506F, 0x51719D2837, 0x68B8CE941B, 0x745C674A0D, 0x5A2E37A506, 0x4D171BD283, 0x668B8DE941, 0x7345C6F4A0, 0x59A2E37A50, 0x4CD171BD28, 0x4668B8DE94, 0x6334586F4A, 0x719A2837A5, 0x78CD141BD2, 0x12F60DD338, 0x297B02E99C, 0x14BD8174CE, 0xA5EC0BA67, 0x52F645D33, 0x297B62E99, 0x14BDF174C, 0x20A5EB8BA6, 0x3052F1C5D3, 0x18297CE2E9, 0x2C14BE7174, 0x360A5B38BA, 0x3B05299C5D, 0x3D8294CE2E, 0x1EC14A6717, 0x2F60A5338B, 0x17B05699C5, 0xBD82F4CE2, 0x25EC13A671,
	0x3FFFF80000, 0x5C6688256F, 0x6E334412B7, 0x5719A6095B, 0x4B8CD704AD, 0x45C66F8256, 0x62E333C12B, 0x51719DE095, 0x68B8CEF04A, 0x745C637825, 0x5A2E35BC12, 0x4D171ADE09, 0x668B8D6F04, 0x7345C2B782, 0x59A2E15BC1, 0x4CD174ADE0, 0x4668BA56F0, 0x6334592B78, 0x719A2895BC, 0x78CD104ADE, 0x1ED485D338, 0xF6A42E99C, 0x7B52174CE, 0x3DA90BA67, 0x1ED4C5D33, 0x20F6A62E99, 0x107B57174C, 0x83DAB8BA6, 0x241ED1C5D3, 0x120F6CE2E9, 0x2907B67174, 0x1483DB38BA, 0x2A41E99C5D, 0x3520F4CE2E, 0x1A907A6717, 0x2D483D338B, 0x36A41E99C5, 0x3B520F4CE2, 0x3DA903A671,
	0x3FFFF80000, 0x53D4327A86, 0x49EA193D43, 0x64F50C9EA1, 0x727A864F50, 0x593D4327A8, 0x4C9EA193D4, 0x464F50C9EA, 0x4327A864F5, 0x6193D4327A, 0x50C9EA193D, 0x6864F50C9E, 0x54327A864F, 0x6A193D4327, 0x750C9EA193, 0x7A864F50C9, 0x7D4327A864, 0x5EA193D432, 0x4F50C9EA19, 0x67A864F50C, 0x13D4358579, 0x9EA1EC2BC, 0x24F50B615E, 0x327A81B0AF, 0x193D44D857, 0xC9EA66C2B, 0x64F573615, 0x327AF9B0A, 0x2193D3CD85, 0x10C9EDE6C2, 0x2864F2F361, 0x14327D79B0, 0x2A193ABCD8, 0x350C995E6C, 0x3A8648AF36, 0x3D4320579B, 0x1EA1942BCD, 0xF50CE15E6, 0x27A8630AF3,
	0x3FFFF80000, 0x53D4327A86, 0x49EA193D43, 0x64F50C9EA1, 0x727A864F50, 0x593D4327A8, 0x4C9EA193D4, 0x464F50C9EA, 0x4327A864F5, 0x6193D4327A, 0x50C9EA193D, 0x6864F50C9E, 0x54327A864F, 0x6A193D4327, 0x750C9EA193, 0x7A864F50C9, 0x7D4327A864, 0x5EA193D432, 0x4F50C9EA19, 0x67A864F50C, 0xC2BCE7A86, 0x2615E33D43, 0x130AF59EA1, 0x9857ECF50, 0x24C2BB67A8, 0x326159B3D4, 0x3930A8D9EA, 0x3C98506CF5, 0x1E4C2C367A, 0x2F26121B3D, 0x17930D0D9E, 0x2BC98286CF, 0x15E4C54367, 0xAF266A1B3, 0x5793750D9, 0x2BC9FA86C, 0x215E4BD436, 0x30AF21EA1B, 0x185794F50D,

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix
	0x3FFFF80000, 0x53D4318579, 0x49EA1CC2BC, 0x64F50A615E, 0x727A8130AF, 0x593D449857, 0x4C9EA64C2B, 0x464F572615, 0x4327AF930A, 0x6193D3C985, 0x50C9EDE4C2, 0x6864F2F261, 0x54327D7930, 0x6A193ABC98, 0x750C995E4C, 0x7A8648AF26, 0x7D43205793, 0x5EA1942BC9, 0x4F50CE15E4, 0x67A8630AF2, 0x13D4367A86, 0x9EA1B3D43, 0x24F50D9EA1, 0x327A86CF50, 0x193D4367A8, 0xC9EA1B3D4, 0x64F50D9EA, 0x327A86CF5, 0x2193D4367A, 0x10C9EA1B3D, 0x2864F50D9E, 0x14327A86CF, 0x2A193D4367, 0x350C9EA1B3, 0x3A864F50D9, 0x3D4327A86C, 0x1EA193D436, 0xF50C9EA1B, 0x27A864F50D,
55	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x7685A9949E439C, 0x7B42D4C84F21CE, 0x7DA12A662780E7, 0x76D0D53113D073, 0x7B682A9889F839, 0x75B4154E44FC1C, 0x72DA4AA7226E0E, 0x716D6553912707, 0x70B6B2ABC89383, 0x785B1955E459C1, 0x742DCCA8F23CE0, 0x7A16A656790E70, 0x7D0B53293C8738, 0x42790E725E959A, 0x413C87392F4ACD, 0x489E039C97B566, 0x444F41CE4BCAB3, 0x4227E0E525F559, 0x4913F07292FAAC, 0x4C89B83B496D56, 0x4E449C1FA4A6AB, 0x4F224E0FD25355, 0x47916705E939AA, 0x43C8F382F48CD5, 0x49E439C17A566A, 0x44F21CE0BD2B35, 0x21CE6C354CB2F4, 0x20E7761AA6497A, 0x2073BB0D5324BD, 0x2839DD86A9925E, 0x2C1CEEC154C92F, 0x2E0E3760AA7497, 0x27071BB2553A4B, 0x23838DDB2A9D25, 0x29C186ED955E92, 0x2CE0C374CAAF49, 0x2E7061BA6557A4, 0x273830DD32ABD2, 0x239C586E9945E9, 0x1532CBD38C64F2, 0x1A9925EBC62279, 0x154C92F7E3113C, 0x1AA64979F1889E, 0x155324BCF8C44F, 0x12A9D25C7C7227, 0x1954E92E3E3913, 0x1CAA74971F1C89, 0x16557A498F9E44, 0x132ABD24C7CF22, 0x19955E9063E791, 0x14CAAF4A31F3C8, 0x1A6517A718E9E4,
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x7685A9949E439C, 0x7B42D4C84F21CE, 0x7DA12A662780E7, 0x76D0D53113D073, 0x7B682A9889F839, 0x75B4154E44FC1C, 0x72DA4AA7226E0E, 0x716D6553912707, 0x70B6B2ABC89383, 0x785B1955E459C1, 0x742DCCA8F23CE0, 0x7A16A656790E70, 0x7D0B53293C8738, 0x41CE13CAB352F4, 0x40E709E559A97A, 0x4073C4F2ACC4BD, 0x4839A27956725E, 0x4C1C913EAB292F, 0x4E0E489F559497, 0x4707644DAADA4B, 0x4383F224D57D25, 0x49C1F9126ABE92, 0x4CE0BC8B354F49, 0x4E701E459AB7A4, 0x47384F22CD4BD2, 0x439C279166A5E9, 0x2532CBD0739B0D, 0x2A9925E839DD86, 0x254C92F41CEEC3, 0x2AA6497A0E7761, 0x255324BF073BB0, 0x22A9D25F838DD8, 0x2954E92DC1C6EC, 0x2CAA7494E0E376, 0x26557A4A7061BB, 0x232ABD273830DD, 0x29955E939C186E, 0x24CAAF49CE0C37, 0x2A6517A4E7161B, 0x1279718E5E8A65, 0x113CF8C52F5532, 0x189E7C6097AA99, 0x144F3E324BD54C, 0x12279F1925EAA6, 0x19138F8E92E553, 0x1C89C7C74972A9, 0x1E44E3E3A4B954, 0x1F2231F3D24CAA, 0x179118F9E92655, 0x13C88C7EF4932A, 0x19E4463D7A4995, 0x14F2631CBD34CA,
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x7586B18CB342F4, 0x7AC358C459A17A, 0x7561EC622CC0BD, 0x7AB0B63116705E, 0x7D581B1A8B282F, 0x76AC0D8F459417, 0x735646C5A2DA0B, 0x71AB6360D17D05, 0x70D5B1B268BE82, 0x786A98DB344F41, 0x7C350C6D9A37A0, 0x761AC636CD0BD0, 0x7B0D63196685E8, 0x42CD0BD29E539C, 0x416685E94F29CE, 0x48B302F6A784E7, 0x4459C17953D273, 0x4A2CA0BCA9F939, 0x4D16505E54FC9C, 0x468B682F2A6E4E, 0x4345F417952727, 0x49A2FA0BCA9393, 0x4CD13D05E559C9, 0x4668DE80F2BCE4, 0x4B342F42794E72, 0x459A17A13CA739, 0x217A69958C74F2, 0x20BD74CAC62A79, 0x205EBA6763153C, 0x282F5D31B18A9E, 0x2417AE98D8C54F, 0x2A0B974C6C72A7, 0x2D05CBA6363953, 0x2E82E5D31B1CA9, 0x2F4132E98D9E54, 0x27A09974C6CF2A, 0x2BD04CB8636795, 0x25E8265E31B3CA, 0x22F4532F18C9E5, 0x1631D3CBA1659A, 0x1B18A9E7D0A2CD, 0x1D8C54F3E85166, 0x16C62A79F428B3, 0x1363153EFA1459, 0x11B1CA9D7D1A2C, 0x18D8E54CBE8D16, 0x1C6C72A45F468B, 0x163679502FB345, 0x131B3CAA17D9A2, 0x118D9E550BECD1, 0x18C6CF2A85F668, 0x1C63279742EB34,
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x7586B18CB342F4, 0x7AC358C459A17A, 0x7561EC622CC0BD, 0x7AB0B63116705E, 0x7D581B1A8B282F, 0x76AC0D8F459417, 0x735646C5A2DA0B, 0x71AB6360D17D05, 0x70D5B1B268BE82, 0x786A98DB344F41, 0x7C350C6D9A37A0, 0x761AC636CD0BD0, 0x7B0D63196685E8, 0x417A166A7394F2, 0x40BD0B3539CA79, 0x405EC5989CF53C, 0x482F22CE4E6A9E, 0x4417D16727254F, 0x4A0BE8B39392A7, 0x4D05B459C9D953, 0x4E829A2CE4FCA9, 0x4F414D16727E54, 0x47A0E68B392F2A, 0x4BD033479C8795, 0x45E859A1CE53CA, 0x42F42CD0E729E5, 0x2631D3C85E9A65, 0x2B18A9E42F5D32, 0x2D8C54F017AE99, 0x26C62A7A0BD74C, 0x2363153D05EBA6, 0x21B1CA9E82E5D3, 0x28D8E54F4172E9, 0x2C6C72A7A0B974, 0x26367953D04CBA, 0x231B3CA9E8265D, 0x218D9E56F4132E, 0x28C6CF297A0997, 0x2C632794BD14CB, 0x12CD742E9E4C63, 0x1166FA154F3631, 0x18B37D0AA79B18, 0x1459BE8553CD8C, 0x1A2CDF40A9E6C6, 0x1D162FA254E363, 0x168B17D32A71B1, 0x13458BEB9538D8, 0x19A285F7CA8C6C, 0x1CD142F9E54636, 0x1668A17CF2A31B, 0x1B3450BE79518D, 0x159A685D3CB8C6,
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x7586AC349E44F2, 0x7AC356184F2279, 0x7561AB0E27913C, 0x7AB0D58513C89E, 0x7D586AC089E44F, 0x76AC356244F227, 0x73561AB3227913, 0x71AB0D5B913C89, 0x70D586AFC89E44, 0x786AC355E44F22, 0x7C3561A8F22791, 0x761AB0D67913C8, 0x7B0D58693C89E4, 0x427913CA9E54F2, 0x413C89E54F2A79, 0x489E44F2A7953C, 0x444F227953CA9E, 0x4227913CA9E54F, 0x4913C89E54F2A7, 0x4C89E44F2A7953, 0x4E44F227953CA9, 0x4F227913CA9E54, 0x47913C89E54F2A, 0x43C89E44F2A795, 0x49E44F227953CA, 0x44F227913CA9E5, 0x2586D3C89E5B0D, 0x2AC329E44F3D86, 0x2561D4F2278EC3, 0x2AB0AA7913D761, 0x2D58153C89FBB0, 0x26AC4A9E44EDD8, 0x2356654F2266EC, 0x21AB72A7912376, 0x20D5F953C881BB, 0x286ABCA9E450DD, 0x2C351E54F2386E, 0x261ACF2A790C37, 0x2B0D27953C961B, 0x12796C369E4B0D, 0x113CF6194F3586, 0x189E3B0EA78AC3, 0x144F5D8553D561, 0x1227EEC0A9FAB0, 0x1913B76254ED58, 0x1C899BB32A66AC, 0x1E448DDB952356, 0x1F2206EFCA81AB, 0x17914375E550D5, 0x13C8E1B8F2B86A, 0x19E430DE794C35, 0x14F2586D3CB61A,
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x6685A9949E5C63, 0x6B42D4C84F3E31, 0x6DA12A66279F18, 0x66D0D53113CF8C, 0x6B682A9889E7C6, 0x65B4154E44E3E3, 0x62DA4AA72271F1, 0x616D65539138F8, 0x60B6B2ABC88C7C, 0x685B1955E4463E, 0x642DCCA8F2231F, 0x6A16A65679118F, 0x6D0B53293C98C7, 0x52790E725E8A65, 0x513C87392F5532, 0x589E039C97AA99, 0x544F41CE4BD54C, 0x5227E0E525EAA6, 0x5913F07292E553, 0x5C89B83B4972A9, 0x5E449C1FA4B954, 0x5F224E0FD24CAA, 0x57916705E92655, 0x53C8F382F4932A, 0x59E439C17A4995, 0x54F21CE0BD34CA, 0x31CE6C354CAD0B, 0x30E7761AA65685, 0x3073BB0D533B42, 0x3839DD86A98DA1, 0x3C1CEEC154D6D0, 0x3E0E3760AA6B68, 0x37071BB25525B4, 0x33838DDB2A82DA, 0x39C186ED95416D, 0x3CE0C374CAB0B6, 0x3E7061BA65485B, 0x373830DD32B42D, 0x339C586E995A16, 0x532CBD38C7B0D, 0xA9925EBC63D86, 0x54C92F7E30EC3, 0xAA64979F19761, 0x55324BCF8DBB0, 0x2A9D25C7C6DD8, 0x954E92E3E26EC, 0xCAA74971F0376, 0x6557A498F81BB, 0x32ABD24C7D0DD, 0x9955E9063F86E, 0x4CAAF4A31EC37, 0xA6517A718F61B,

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x6685A9949E5C63, 0x6B42D4C84F3E31, 0x6DA12A66279F18, 0x66D0D53113CF8C, 0x6B682A9889E7C6, 0x65B4154E44E3E3, 0x62DA4AA72271F1, 0x616D65539138F8, 0x60B6B2ABC88C7C, 0x685B1955E4463E, 0x642DCCA8F2231F, 0x6A16A65679118F, 0x6D0B53293C98C7, 0x51CE13CAB34D0B, 0x50E709E559B685, 0x5073C4F2ACDB42, 0x5839A279566DA1, 0x5C1C913EAB36D0, 0x5E0E489F558B68, 0x5707644DAAC5B4, 0x5383F224D562DA, 0x59C1F9126AA16D, 0x5CE0BC8B3550B6, 0x5E701E459AA85B, 0x57384F22CD542D, 0x539C279166BA16, 0x3532CBD07384F2, 0x3A9925E839C279, 0x354C92F41CF13C, 0x3AA6497A0E689E, 0x355324BF07244F, 0x32A9D25F839227, 0x3954E92DC1D913, 0x3CAA7494E0FC89, 0x36557A4A707E44, 0x332ABD27382F22, 0x39955E939C0791, 0x34CAAF49CE13C8, 0x3A6517A4E709E4, 0x279718E5E959A, 0x13CF8C52F4ACD, 0x89E7C6097B566, 0x44F3E324BCAB3, 0x2279F1925F559, 0x9138F8E92FAAC, 0xC89C7C7496D56, 0xE44E3E3A4A6AB, 0xF2231F3D25355, 0x79118F9E939AA, 0x3C88C7EF48CD5, 0x9E4463D7A566A, 0x4F2631CBD2B35,
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x6685A994739B0D, 0x6B42D4C839DD86, 0x6DA12A641CEEC3, 0x66D0D5320E7761, 0x6B682A9B073BB0, 0x65B4154F838DD8, 0x62DA4AA5C1C6EC, 0x616D6550E0E376, 0x60B6B2AA7061BB, 0x685B19573830DD, 0x642DCCAB9C186E, 0x6A16A655CE0C37, 0x6D0B5328E7161B, 0x51CE13CA5E8A65, 0x50E709E52F5532, 0x5073C4F097AA99, 0x5839A27A4BD54C, 0x5C1C913D25EAA6, 0x5E0E489E92E553, 0x5707644F4972A9, 0x5383F227A4B954, 0x59C1F913D24CAA, 0x5CE0BC89E92655, 0x5E701E46F4932A, 0x57384F217A4995, 0x539C2790BD34CA, 0x3279718D4CAD0B, 0x313CF8C6A65685, 0x389E7C61533B42, 0x344F3E32A98DA1, 0x32279F1954D6D0, 0x39138F8CAA6B68, 0x3C89C7C65525B4, 0x3E44E3E32A82DA, 0x3F2231F195416D, 0x379118F8CAB0B6, 0x33C88C7E65485B, 0x39E4463D32B42D, 0x34F2631E995A16, 0x532CBD361BC63, 0xA9925EBB0DE31, 0x54C92F5D87F18, 0xAA6497AEC2F8C, 0x55324BF7607C6, 0x2A9D25DBB03E3, 0x954E92CDD91F1, 0xCAA74946ED8F8, 0x6557A48376C7C, 0x32ABD261BA63E, 0x9955E930DC31F, 0x4CAAF4986F18F, 0xA6517A6C378C7,
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x668593C873959A, 0x6B4289E439CACD, 0x6DA144F01CF566, 0x66D0A27A0E6AB3, 0x6B68113F073559, 0x65B4489F839AAC, 0x62DA644DC1CD56, 0x616D7224E0E6AB, 0x60B6F912707355, 0x685B3C8B3839AA, 0x642D9E479C0CD5, 0x6A16CF21CE166A, 0x6D0B2790E70B35, 0x51CE29965E84F2, 0x50E754C92F4279, 0x5073AA6497B13C, 0x5839D5324BC89E, 0x5C1CAA9925E44F, 0x5E0E154E92F227, 0x57074AA7497913, 0x5383E553A4BC89, 0x59C1B2ABD25E44, 0x5CE09955E92F22, 0x5E704CAAF48791, 0x573826557A53C8, 0x539C5328BD29E4, 0x3532F18C9E4D0B, 0x3A9978C44F3685, 0x354CFC62279B42, 0x3AA63E3113CDA1, 0x35531F1889F6D0, 0x32A98F8E44EB68, 0x3954C7C72265B4, 0x3CAA63E39122DA, 0x365531F3C8816D, 0x332A98F9E450B6, 0x39950C7CF2285B, 0x34CAC63E79142D, 0x3A65631D3C9A16, 0x2794BD2B35C63, 0x13CA5E959BE31, 0x89E12F6ACDF18, 0x44F4979566F8C, 0x227A4BEAB27C6, 0x913D25F5583E3, 0xC89E92DAAD1F1, 0xE44F494D578F8, 0xF227A4A6AAC7C, 0x7913D2735463E, 0x3C8DE919AA31F, 0x9E42F4ACD518F, 0x4F217A566B8C7,
	0x4FFFFFFC000000, 0x2FFF8003FFE000, 0x1FFF8000001FFF, 0x6586AC349E5B0D, 0x6AC356184F3D86, 0x6561AB0E278EC3, 0x6AB0D58513D761, 0x6D586AC089FBB0, 0x66AC356244EDD8, 0x63561AB32266EC, 0x61AB0D5B912376, 0x60D586AFC881BB, 0x686AC355E450DD, 0x6C3561A8F2386E, 0x661AB0D6790C37, 0x6B0D58693C961B, 0x527913CA9E4B0D, 0x513C89E54F3586, 0x589E44F2A78AC3, 0x544F227953D561, 0x5227913CA9FAB0, 0x5913C89E54ED58, 0x5C89E44F2A66AC, 0x5E44F227952356, 0x5F227913CA81AB, 0x57913C89E550D5, 0x53C89E44F2B86A, 0x59E44F22794C35, 0x54F227913CB61A, 0x3586D3C89E44F2, 0x3AC329E44F2279, 0x3561D4F227913C, 0x3AB0AA7913C89E, 0x3D58153C89E44F, 0x36AC4A9E44F227, 0x3356654F227913, 0x31AB72A7913C89, 0x30D5F953C89E44, 0x386ABCA9E44F22, 0x3C351E54F22791, 0x361ACF2A7913C8, 0x3B0D27953C89E4, 0x2796C369E54F2, 0x13CF6194F2A79, 0x89E3B0EA7953C, 0x44F5D8553CA9E, 0x227EEC0A9E54F, 0x913B76254F2A7, 0xC899BB32A7953, 0xE448DDB953CA9, 0xF2206EFCA9E54, 0x7914375E54F2A, 0x3C8E1B8F2A795, 0x9E430DE7953CA, 0x4F2586D3CA9E5,
71	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x77213A36C4A7943A5C, 0x7B909D1B6053CA1D2E, 0x7DC84A8DB229E40E97, 0x7EE42146D914F3074B, 0x777210A36E8A7983A5, 0x73B90C51B5453DC1D2, 0x79DC8628D8A29EE0E9, 0x74EE43146E514F7074, 0x7277258A3728A6B83A, 0x713B96C51B94525C1D, 0x709DCB628FCA292E0E, 0x784EE5B145E5149707, 0x742776D8A0F28B4B83, 0x7213BB6C527945A5C1, 0x7909D9B6293CA3D2E0, 0x7C84E8DB169E50E970, 0x7E42746D894F2874B8, 0x429E50E97237B1724E, 0x414F2874B91BD8B927, 0x48A7903A5C8DED5C93, 0x4453CC1D2C46F7AE49, 0x4A29E60E96237BD724, 0x4514F7074B11BCEB92, 0x428A7B83A588DE75C9, 0x49453DC1D2C46F3AE4, 0x4CA29AE0EB62369D72, 0x4E51497077B11A4EB9, 0x4F28A4B83BD88D275C, 0x4794525C1DEC4693AE, 0x43CA2D2E0EF62249D7, 0x49E51697077B1124EB, 0x44F28F4B81BD899275, 0x4A7943A5C0DEC5C93A, 0x453CA1D2E06F62E49D, 0x21D2E6B0D51B631BD8, 0x20E977586A8DB08DEC, 0x2074BBAC3546D846F6, 0x283A5DD618A36C237B, 0x2C1D2AEB0C51B711BD, 0x2E0E95758628DB88DE, 0x27074EBAC3146CC46F, 0x2B83A35D618A376237, 0x25C1D1AEB2C51BB11B, 0x22E0E8D75B628DD88D, 0x2970706BADB147EC46, 0x24B83C35D6D8A2F623, 0x2A5C1E1AEB6C517B11, 0x2D2E0B0D75B629BD88, 0x2E970586B8DB14DEC4, 0x274B82C35C6D8A6F62, 0x23A5C561AE36C437B1, 0x146D8C6F638B4653CA, 0x1A36C237B3C5A229E5, 0x151B611BDBE2D114F2, 0x128DB08DEDF1688A79, 0x1146DC46F4F8B5453C, 0x18A36E23787C5AA29E, 0x1C51B311BE3E2C514F, 0x1628DD88DD1F1728A7, 0x1B146EC46E8F8B9453, 0x1D8A37623747C5CA29, 0x16C51FB119A3E3E514, 0x1B628BD88ED1F0F28A, 0x1DB145EC4568F87945, 0x16D8A6F620B47D3CA2, 0x136C537B105A3E9E51, 0x11B629BD8A2D1F4F28, 0x18DB10DEC7168EA794,

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x77213A36C4A7943A5C,	0x7B909D1B6053CA1D2E,
	0x7DC84A8DB229E40E97,	0x7EE42146D914F3074B,	0x777210A36E8A7983A5,	0x73B90C51B5453DC1D2,	0x79DC8628D8A29EE0E9,
	0x74EE43146E514F7074,	0x7277258A3728A6B83A,	0x713B96C51B94525C1D,	0x709DCB628FCA292E0E,	0x784EE5B145E5149707,
	0x742776D8A0F28B4B83,	0x7213BB6C527945A5C1,	0x7909D9B6293CA3D2E0,	0x7C84E8DB169E50E970,	0x7E42746D894F2874B8,
	0x41D2E14F2AE49D1BD8,	0x40E970A795724E8DEC,	0x4074BC53CAB92646F6,	0x483A5A29E75C92237B,	0x4C1D2D14F3AE4911BD,
	0x4E0E928A79D72588DE,	0x470749453CEB92C46F,	0x4B83A4A29E75C96237,	0x45C1D6514D3AE5B11B,	0x42E0EF28A49D73D88D,
	0x49707794524EB9EC46,	0x44B83BCA29275CF623,	0x4A5C19E51493AF7B11,	0x4D2E0CF28A49D7BD88,	0x4E9702794724EADEC4,
	0x474B853CA392746F62,	0x43A5C29E51C93A37B1,	0x246D8C6F6074B9AC35,	0x2A36C237B03A5DD61A,	0x251B611BD81D2EEB0D,
	0x228DB08DEE0E977586,	0x2146DC46F7074ABAC3,	0x28A36E237B83A55D61,	0x2C51B311BDC1D3AEB0,	0x2628DD88DEE0E8D758,
	0x2B146EC46D70746BAC,	0x2D8A376234B83A35D6,	0x26C51FB11A5C1C1AEB,	0x2B628BD88D2E0F0D75,	0x2DB145EC46970786BA,
	0x26D8A6F6234B82C35D,	0x236C537B13A5C161AE,	0x21B629BD89D2E0B0D7,	0x28DB10DEC4E971586B,	0x129E57168E37B08DB1,
	0x114F2F8B451BD946D8,	0x18A797C5A08DECA36C,	0x1453CBE2D046F651B6,	0x1A29E1F16A237A28DB,	0x1514F0F8B711BD146D,
	0x128A7C7C5988DF8A36,	0x19453A3E2EC46EC51B,	0x1CA29D1F176237628D,	0x1E514E8F8BB11BB146,	0x1F28A347C7D88CD8A3,
	0x179455A3E1EC476C51,	0x13CA2AD1F2F623B628,	0x19E51168FB7B10DB14,	0x14F288B47DBD886D8A,	0x1A79445A3CDEC436C5,
	0x153CA62D1C6F631B62,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x77123A9694B3341E78,	0x7B891D4B48599A0F3C,
	0x7DC48AA5A62CCC079E,	0x7EE24552D1166603CF,	0x777122A96A8B3301E7,	0x73B89154B7459980F3,	0x71DC4CAA59A2CDC079,
	0x78EE22552CD167E03C,	0x7477152A9668B2F01E,	0x723B8E954B3458780F,	0x791DC34AA59A2D3C07,	0x748EE5A550CD179E03,
	0x724772D2AA668BCF01,	0x7123B969573345E780,	0x7891DCB4A999A2F3C0,	0x7C48EA5A56CCD079E0,	0x7E24752D2966683CF0,
	0x42CCD079E23B715A5A,	0x4166683CF11DB8AD2D,	0x48B3301E788EDD5696,	0x4459980F3C476EAB4B,	0x4A2CCC079E23B755A5,
	0x4D166603CF11DBAAD2,	0x468B3701E788ECD569,	0x43459F80F1C4776AB4,	0x49A2CBC07AE23AB55A,	0x4CD161E03F711C5AAD,
	0x4668B4F01DB88F2D56,	0x43345E780EDC4696AB,	0x499A2F3C076E234B55,	0x4CCD179E03B711A5AA,	0x46668BCF01DB88D2D5,
	0x4B3341E780EDC5696A,	0x4599A0F3C076E2B4B5,	0x20F3C699954B4B1DB8,	0x2079E74CCAA5A48EDC,	0x203CF3A66552D2476E,
	0x201E7DD332A96823B7,	0x280F3AE99954B511DB,	0x2C079974CCAA5B88ED,	0x2E03CCBA66552DC476,	0x2F01E65D312A96E23B,
	0x2780F32E9A954B711D,	0x23C079974F4AA5B88E,	0x29E03CCBA5A552DC47,	0x2CF01E65D2D2A96E23,	0x2E780B32E96955B711,
	0x2F3C019974B4ABDB88,	0x279E04CCBA5A54EDC4,	0x23CF02665D2D2A76E2,	0x21E785332E96943B71,	0x152D2C76E3C30E599A,
	0x1A96923B73E1862CCD,	0x154B491DBBF0C31666,	0x1AA5A08EDFF8608B33,	0x1552D4476DFC314599,	0x12A96E23B4FE19A2CC,
	0x1954B711D87F0CD166,	0x14AA5B88EC3F8668B3,	0x1A552DC4761FC33459,	0x1D2A96E23B0FE19A2C,	0x16954B711D87F0CD16,
	0x1B4AA5B88CC3F8668B,	0x15A556DC4461FD3345,	0x12D2AF6E2030FF99A2,	0x196953B712187ECCD1,	0x14B4A9DB8B0C3F6668,
	0x1A5A50EDC7861EB334,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x77123A9694B3341E78,	0x7B891D4B48599A0F3C,
	0x7DC48AA5A62CCC079E,	0x7EE24552D1166603CF,	0x777122A96A8B3301E7,	0x73B89154B7459980F3,	0x71DC4CAA59A2CDC079,
	0x78EE22552CD167E03C,	0x7477152A9668B2F01E,	0x723B8E954B3458780F,	0x791DC34AA59A2D3C07,	0x748EE5A550CD179E03,
	0x724772D2AA668BCF01,	0x7123B969573345E780,	0x7891DCB4A999A2F3C0,	0x7C48EA5A56CCD079E0,	0x7E24752D2966683CF0,
	0x40F3C1666AB4B51DB8,	0x4079E0B3355A5A8EDC,	0x403CF4599AAD2C476E,	0x401E7A2CCD569623B7,	0x480F3D1666AB4B11DB,
	0x4C079E8B3355A588ED,	0x4E03CB4599AAD3C476,	0x4F01E1A2CED568E23B,	0x4780F4D1656AB5711D,	0x43C07E68B0B55BB88E,
	0x49E03B345A5AACDC47,	0x4CF0199A2D2D576E23,	0x4E780CCD1696ABB711,	0x4F3C06668B4B55DB88,	0x479E033345A5AAEDC4,
	0x43CF0599A2D2D476E2,	0x41E782CCD1696A3B71,	0x252D2C76E03CF1A665,	0x2A96923B701E79D332,	0x254B491DB80F3CE999,
	0x2AA5A08EDC079F74CC,	0x2552D4476E03CEBA66,	0x22A96E23B701E65D33,	0x2954B711DB80F32E99,	0x24AA5B88EFC079974C,
	0x2A552DC475E03CCBA6,	0x2D2A96E238F01E65D3,	0x26954B711E780F32E9,	0x2B4AA5B88F3C079974,	0x25A556DC479E02CCBA,
	0x22D2AF6E23CF00665D,	0x296953B711E781332E,	0x24B4A9DB88F3C09997,	0x2A5A50EDC479E14CCB,	0x12CCD7861E3B70A5A5,
	0x11666FC30D1DB952D2,	0x18B337E1848EDCA969,	0x14599FF0C0476F54B4,	0x1A2CCBF86223B6AA5A,	0x1D1661FC3311DA552D,
	0x168B30FE1B88ED2A96,	0x1345987F0DC476954B,	0x19A2CC3F86E23B4AA5,	0x1CD1661FC3711DA552,	0x1668B30FE1B88ED2A9,
	0x13345987F2DC476954,	0x199A28C3FB6E22B4AA,	0x1CCD1061FFB7105A55,	0x16668C30FDDB892D2A,	0x1B3346187CEDC49695,
	0x1599A70C3C76E34B4A,				

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x770C3A9994B4B41DB8,	0x7B861D4CC85A5A0EDC,
	0x7DC30AA6662D2C076E,	0x7EE1855331169603B7,	0x7770C2A99A8B4B01DB,	0x73B86154CF45A580ED,	0x71DC34AA65A2D3C076,
	0x70EE1E5532D168E03B,	0x78770B2A9968B5701D,	0x7C3B81954CB45BB80E,	0x761DC4CAA65A2CDC07,	0x730EE665512D176E03,
	0x71877332AA968BB701,	0x70C3B999574B45DB80,	0x7861DCCCA9A5A2EDC0,	0x7C30EA6656D2D076E0,	0x7E1875332969683B70,
	0x42D2D076E23CF1599A,	0x4169683B711E78ACCD,	0x48B4B01DB88F3D5666,	0x445A580EDC479EAB33,	0x4A2D2C076E23CF5599,
	0x4D169603B711E7AACC,	0x468B4F01DB88F2D566,	0x4B45A380EFC4786AB3,	0x45A2D5C075E23D3559,	0x42D16EE038F11F9AAC,
	0x4968B3701E788ECD56,	0x44B45DB80F3C4666AB,	0x4A5A2EDC079E233355,	0x4D2D176E03CF1199AA,	0x46968BB701E788CCD5,
	0x4B4B41DB80F3C5666A,	0x45A5A0EDC079E2B335,	0x20EDC696954CCB1E78,	0x2076E74B4AA6648F3C,	0x203B73A5A55332479E,
	0x201DBDD2D2A99823CF,	0x280EDAE96954CD11E7,	0x2C076974B4AA6788F3,	0x2E03B4BA5A5533C479,	0x2701DA5D2F2A99E23C,
	0x2B80ED2E95954CF11E,	0x2DC0769748CAA6788F,	0x26E03B4BA665533C47,	0x2B701DA5D332A99E23,	0x2DB80AD2E99955CF11,
	0x2EDC016974CCABE788,	0x276E04B4BA6654F3C4,	0x23B7025A5D332A79E2,	0x21DB852D2E99943CF1,	0x15332C79E3C48E5A5A,
	0x1A99923CF3E2462D2D,	0x154CC91E7BF1231696,	0x1AA6608F3FF8908B4B,	0x155334479DFC4945A5,	0x12A99E23CCFE25A2D2,
	0x1954CF11E47F12D169,	0x1CAA6788F23F8968B4,	0x165533C4791FC4B45A,	0x132A99E23C8FE25A2D,	0x19954CF11E47F12D16,
	0x1CCAA6788D23F8968B,	0x1665573C4491FD4B45,	0x1332AF9E2048FFA5A2,	0x199953CF12247ED2D1,	0x14CCA9E78B123F6968,
	0x1A6650F3C7891EB4B4,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x770C3A9994B4B41DB8,	0x7B861D4CC85A5A0EDC,
	0x7DC30AA6662D2C076E,	0x7EE1855331169603B7,	0x7770C2A99A8B4B01DB,	0x73B86154CF45A580ED,	0x71DC34AA65A2D3C076,
	0x70EE1E5532D168E03B,	0x78770B2A9968B5701D,	0x7C3B81954CB45BB80E,	0x761DC4CAA65A2CDC07,	0x730EE665512D176E03,
	0x71877332AA968BB701,	0x70C3B999574B45DB80,	0x7861DCCCA9A5A2EDC0,	0x7C30EA6656D2D076E0,	0x7E1875332969683B70,
	0x40EDC1696AB3351E78,	0x4076E0B4B5599A8F3C,	0x403B745A5AACCC479E,	0x401DBA2D2D566623CF,	0x480EDD1696AB3311E7,
	0x4C076E8B4B559988F3,	0x4E03B345A5AACDC479,	0x4701DDA2D0D567E23C,	0x4B80EAD16A6AB2F11E,	0x4DC07168B73558788F,
	0x46E03CB4599AAD3C47,	0x4B701A5A2CCD579E23,	0x4DB80D2D1666ABCF11,	0x4EDC06968B3355E788,	0x476E034B4599AAF3C4,
	0x43B705A5A2CCD479E2,	0x41DB82D2D1666A3CF1,	0x25332C79E03B71A5A5,	0x2A99923CF01DB9D2D2,	0x254CC91E780EDCE969,
	0x2AA6608F3C076F74B4,	0x255334479E03B6BA5A,	0x22A99E23CF01DA5D2D,	0x2954CF11E780ED2E96,	0x2CAA6788F1C076974B,
	0x265533C47AE03B4BA5,	0x232A99E23F701DA5D2,	0x29954CF11DB80ED2E9,	0x2CCAA6788EDC076974,	0x2665573C476E02B4BA,
	0x2332AF9E23B7005A5D,	0x299953CF11DB812D2E,	0x24CCA9E788EDC09697,	0x2A6650F3C476E14B4B,	0x12D2D7891E3CF0A665,
	0x11696FC48D1E795332,	0x18B4B7E2448F3CA999,	0x145A5FF120479F54CC,	0x1A2D2BF89223CEAA66,	0x1D1691FC4B11E65533,
	0x168B48FE2788F32A99,	0x1B45A47F13C479954C,	0x15A2D23F89E23CCAA6,	0x12D1691FC4F11E6553,	0x1968B48FE2788F32A9,
	0x14B45A47F33C479954,	0x1A5A2923FB9E22CCAA,	0x1D2D1091FFCF106655,	0x16968C48FDE789332A,	0x1B4B46247CF3C49995,
	0x15A5A7123C79E34CCA,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x76C0DB462D703AB18D,	0x7B606DA316B81D58C6,
	0x7DB036D1895C0EAC63,	0x76D81B68C6AE075631,	0x7B6C0DB4635703AB18,	0x7DB602DA33AB80D58C,	0x76DB016D19D5C06AC6,
	0x736D80B68CEAE03563,	0x71B6C45B4475711AB1,	0x70DB662DA03AB98D58,	0x706DB316D01D5CC6AC,	0x7036D98B680EAE6356,
	0x781B68C5B6075631AB,	0x7C0DB462DB03AB18D5,	0x7606DA316F81D58C6A,	0x7B036D18B5C0EAC635,	0x7D81B68C5AE075631A,
	0x45C0EAC6364FC92E74,	0x4AE075631927E4973A,	0x45703AB18C93F24B9D,	0x4AB81D58C649F925CE,	0x4D5C0EAC6124FC92E7,
	0x4EAE035630927F4973,	0x475701AB1A493FA4B9,	0x43AB80D58F249FD25C,	0x41D5C46AC7924EE92E,	0x40EAE63563C9267497,
	0x4075731AB3E4933A4B,	0x403AB98D5BF2499D25,	0x481D58C6ADF925CE92,	0x4C0EAC6354FC92E749,	0x4E075631AA7E4973A4,
	0x4703AB18D53F24B9D2,	0x4B81D58C689F925CE9,	0x258C6D1F89A31727E4,	0x2AC6328FC6D18A93F2,	0x25631D47E368C449F9,
	0x2AB18AA3F1B46324FC,	0x2D58C151FADA30927E,	0x26AC60A8FD6D18493F,	0x235634547CB68D249F,	0x21AB1E2A3C5B47924F,
	0x28D58F151E2DA3C927,	0x2C6AC78A8F16D1E493,	0x263567C5458B69F249,	0x231AB7E2A0C5B5F924,	0x218D5BF15062DAFC92,
	0x28C6A9F8AA316C7E49,	0x2C6350FC5518B73F24,	0x2631AC7E2A8C5A9F92,	0x2B18D23F17462C4FC9,	0x168C5C9F929CE4B81D,
	0x1B462A4FC94E735C0E,	0x1DA31127E6A738AE07,	0x16D18C93F1539D5703,	0x1B68C249F8A9CFAB81,	0x15B46124FE54E7D5C0,
	0x12DA34927F2A72EAE0,	0x116D1E493F95387570,	0x18B68F249DCA9C3AB8,	0x1C5B47924CE54E1D5C,	0x162DA7C92672A60EAE,
	0x1316D7E49339520757,	0x118B6BF24B9CA903AB,	0x18C5B1F925CE5581D5,	0x1462DCFC90E72BC0EA,	0x1A316A7E4A7394E075,
	0x1D18B13F2539CB703A,				

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x76C0DB462D703AB18D,	0x7B606DA316B81D58C6,
	0x7DB036D1895C0EAC63,	0x76D81B68C6AE075631,	0x7B6C0DB4635703AB18,	0x7DB602DA33AB80D58C,	0x76DB016D19D5C06AC6,
	0x736D80B68CEAE03563,	0x71B6C45B4475711AB1,	0x70DB662DA03AB98D58,	0x706DB316D01D5CC6AC,	0x7036D98B680EAE6356,
	0x781B68C5B6075631AB,	0x7C0DB462DB03AB18D5,	0x7606DA316F81D58C6A,	0x7B036D18B5C0EAC635,	0x7D81B68C5AE075631A,
	0x458C6AE0765CE927E4,	0x4AC63570392E7493F2,	0x45631AB81C973A49F9,	0x4AB18D5C0E4B9D24FC,	0x4D58C6AE0525CE927E,
	0x46AC67570292E6493F,	0x435633AB834973249F,	0x41AB19D5C3A4B9924F,	0x48D588EAE1D25DC927,	0x4C6AC07570E92FE493,
	0x4635603ABA7497F249,	0x431AB01D5F3A4BF924,	0x418D5C0EAF9D24FC92,	0x48C6AE0755CE927E49,	0x4C635703AAE7493F24,
	0x4631AB81D573A49F92,	0x4B18D5C0E8B9D24FC9,	0x268C5C9F91631B47E2,	0x2B462A4FCAB18CA3F1,	0x2DA31127E558C751F8,
	0x26D18C93F2AC62A8FC,	0x2B68C249FB5630547E,	0x25B46124FDAB182A3F,	0x22DA34927CD58D151F,	0x216D1E493C6AC78A8F,
	0x28B68F249E3563C547,	0x2C5B47924F1AB1E2A3,	0x262DA7C9258D59F151,	0x2316D7E490C6ADF8A8,	0x218B6BF2486356FC54,
	0x28C5B1F92631AA7E2A,	0x2462DCFC9318D43F15,	0x2A316A7E498C6B1F8A,	0x2D18B13F26C6348FC5,	0x15C0ED39CA4FC8D18B,
	0x1AE0729CE527E568C5,	0x15703D4E7093F3B462,	0x1AB81AA73A49F8DA31,	0x1D5C09539D24FD6D18,	0x1EAE04A9CC927EB68C,
	0x17570654E6493E5B46,	0x13AB872A73249E2DA3,	0x11D5C3953B924F16D1,	0x10EAE1CA9FC9278B68,	0x107574E54FE492C5B4,
	0x103ABE72A7F24862DA,	0x181D5F3951F924316D,	0x1C0EAB9CA8FC9318B6,	0x1E0751CE567E488C5B,	0x1703ACE7293F25462D,
	0x1B81D273949F93A316,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x76C0DA8F149CE435AC,	0x7B606D47884E721AD6,
	0x7DB032A3C627380D6B,	0x76D81D51E1139D06B5,	0x7B6C0AA8F089CF835A,	0x7DB601547A44E6C1AD,	0x76DB00AA3F227360D6,
	0x736D84551F9138B06B,	0x71B6C62A8DC89D5835,	0x70DB671544E44FAC1A,	0x706DB78AA27226D60D,	0x7036DBC55339136B06,
	0x781B69E2AB9C88B583,	0x7C0DB0F155CE455AC1,	0x7606DC78A8E723AD60,	0x7B036A3C567390D6B0,	0x7D81B51E2939C86B58,
	0x427390D6B24FC95C3A,	0x4139C86B5927E4AE1D,	0x489CE035AC93F3570E,	0x444E741AD649F8AB87,	0x42273E0D6924FD55C3,
	0x49139B06B4927FAAE1,	0x4C89CD835A493FD570,	0x4E44E2C1AF249EEAB8,	0x47227560D7924E755C,	0x43913EB06BC9263AAE,
	0x49C89B5837E4921D57,	0x4CE44DAC1BF2490EAB,	0x4E7222D60DF9258755,	0x4739156B04FC93C3AA,	0x439C8EB5827E48E1D5,
	0x49CE435AC13F2570EA,	0x44E721AD609F92B875,	0x21AD66C635478B27E4,	0x20D6B7631AA3C493F2,	0x206B5BB18D51E249F9,
	0x2835ADD8C6A8F124FC,	0x2C1AD6EC615478927E,	0x260D6B7630AA3C493F,	0x2B06B1BB18551F249F,	0x258358DD8E2A8F924F,
	0x2AC1AC6EC71547C927,	0x2D60D637638AA3E493,	0x26B06B1BB3C551F249,	0x2B58318DD9E2A9F924,	0x25AC18C6ECF154FC92,
	0x2AD60C637478AA7E49,	0x2D6B0631BA3C553F24,	0x26B58318DD1E2A9F92,	0x235AC58C6E8F144FC9,	0x151E2C9F9394A64E72,
	0x1A8F124FCBCA522739,	0x15478927E7E529139C,	0x1AA3C493F1F29489CE,	0x1551E249F8F94A44E7,	0x12A8F124FE7CA52273,
	0x11547C927D3E539139,	0x18AA3E493E9F29C89C,	0x1C551F249D4F94E44E,	0x1E2A8F924CA7CA7227,	0x1F1547C92653E53913,
	0x178AA7E49129F39C89,	0x13C553F24A94F9CE44,	0x11E2A9F9254A7CE722,	0x18F154FC90A53E7391,	0x1478AA7E4A529F39C8,
	0x1A3C513F27294E9CE4,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x76C0DA8F149CE435AC,	0x7B606D47884E721AD6,
	0x7DB032A3C627380D6B,	0x76D81D51E1139D06B5,	0x7B6C0AA8F089CF835A,	0x7DB601547A44E6C1AD,	0x76DB00AA3F227360D6,
	0x736D84551F9138B06B,	0x71B6C62A8DC89D5835,	0x70DB671544E44FAC1A,	0x706DB78AA27226D60D,	0x7036DBC55339136B06,
	0x781B69E2AB9C88B583,	0x7C0DB0F155CE455AC1,	0x7606DC78A8E723AD60,	0x7B036A3C567390D6B0,	0x7D81B51E2939C86B58,
	0x41AD6139CAB87527E4,	0x40D6B09CE55C3A93F2,	0x406B5C4E72AE1C49F9,	0x4835AA2739570F24FC,	0x4C1AD1139EAB86927E,
	0x460D6C89CF55C2493F,	0x4B06B644E7AAE1249F,	0x45835F2271D571924F,	0x4AC1AB9138EAB9C927,	0x4D60D1C89C755DE493,
	0x46B06CE44C3AAFF249,	0x4B583672261D57F924,	0x45AC1F39130EAAFC92,	0x4AD60B9C8B87547E49,	0x4D6B01CE45C3AB3F24,
	0x46B584E722E1D49F92,	0x435AC2739170EA4FC9,	0x251E2C9F906B59B18D,	0x2A8F124FC835ADD8C6,	0x25478927E41AD6EC63,
	0x2AA3C493F20D6B7631,	0x2551E249FB06B5BB18,	0x22A8F124FD835ADD8C,	0x21547C927EC1AC6EC6,	0x28AA3E493D60D63763,
	0x2C551F249EB06B1BB1,	0x2E2A8F924F58358DD8,	0x2F1547C925AC1AC6EC,	0x278AA7E492D60C6376,	0x23C553F2496B0631BB,
	0x21E2A9F926B58318DD,	0x28F154FC935AC18C6E,	0x2478AA7E49AD60C637,	0x2A3C513F24D6B1631B,	0x127397294E4FC8A3C5,
	0x1139CF94A527E551E2,	0x189CE7CA5093F2A8F1,	0x144E73E52A49F95478,	0x122739F29524FCAA3C,	0x19139CF948927E551E,
	0x1C89CA7CA6493E2A8F,	0x1E44E53E53249F1547,	0x1722729F2B924F8AA3,	0x1391394F97C927C551,	0x19C89CA7CBE493E2A8,
	0x1CE44A53E7F248F154,	0x1E722529F1F92478AA,	0x17391294F8FC923C55,	0x139C894A7E7E491E2A,	0x19CE44A53D3F248F15,
	0x14E726529C9F93478A,				

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF8000000001FFFF,	0x768C5B606D703AB18D,	0x7B462DB036B81D58C6,
	0x7DA316D8195C0EAC63,	0x76D18B6C0EAE075631,	0x7B68C5B6075703AB18,	0x75B466DB03AB80D58C,	0x72DA336D81D5C06AC6,
	0x716D19B6C0EAE03563,	0x78B688DB6075711AB1,	0x7C5B406DB03AB98D58,	0x762DA036D81D5CC6AC,	0x7316D01B6C0EAE6356,
	0x718B6C0DB6075631AB,	0x78C5B606DB03AB18D5,	0x7462DB036F81D58C6A,	0x7A316D81B5C0EAC635,	0x7D18B6C0DAE075631A,
	0x45C0EAC6365CE927E4,	0x4AE07563192E7493F2,	0x45703AB18C973A49F9,	0x4AB81D58C64B9D24FC,	0x4D5C0EAC6125CE927E,
	0x4EAE03563292E6493F,	0x475701AB1B4973249F,	0x43AB80D58FA4B9924F,	0x41D5C46AC5D25DC927,	0x40EAE63560E92FE493,
	0x4075731AB27497F249,	0x403AB98D5B3A4BF924,	0x481D58C6AF9D24FC92,	0x4C0EAC6355CE927E49,	0x4E075631AAE7493F24,
	0x4703AB18D573A49F92,	0x4B81D58C68B9D24FC9,	0x258C6D1F89B0372E74,	0x2AC6328FC6D81A973A,	0x25631D47E36C0C4B9D,
	0x2AB18AA3F1B60725CE,	0x2D58C151FADB0292E7,	0x26AC60A8FF6D814973,	0x235634547DB6C1A4B9,	0x21AB1E2A3CDB61D25C,
	0x28D58F151C6DB0E92E,	0x2C6AC78A8C36D87497,	0x263567C5441B6D3A4B,	0x231AB7E2A00DB79D25,	0x218D5BF15206DBCE92,
	0x28C6A9F8AB036CE749,	0x2C6350FC5581B773A4,	0x2631AC7E2AC0DAB9D2,	0x2B18D23F17606C5CE9,	0x16C0DCB9D29CE4B81D,
	0x1B606A5CE94E735C0E,	0x1DB0312E76A738AE07,	0x16D81C9739539D5703,	0x1B6C0A4B9CA9CFAB81,	0x1DB60525CE54E7D5C0,
	0x16DB0692E72A72EAE0,	0x136D87497395387570,	0x11B6C3A4B9CA9C3AB8,	0x10DB61D25CE54E1D5C,	0x106DB4E92E72A60EAE,
	0x1036DE749739520757,	0x181B6F3A4B9CA903AB,	0x1C0DB39D25CE5581D5,	0x1606DDCE90E72BC0EA,	0x1B036AE74A7394E075,
	0x1D81B173A539CB703A,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF8000000001FFFF,	0x768C5B462C5CE82E74,	0x7B462DA3142E74173A,
	0x7DA316D188173A0B9D,	0x76D18B68C60B9D05CE,	0x7B68C5B46105CE82E7,	0x75B462DA3282E74173,	0x72DA316D1B4173A0B9,
	0x716D18B68FA0B9D05C,	0x78B68C5B45D05CE82E,	0x7C5B462DA0E82E7417,	0x762DA316D274173A0B,	0x7316D18B6B3A0B9D05,
	0x718B68C5B79D05CE82,	0x78C5B462D9CE82E741,	0x7462DA316EE74173A0,	0x7A316D18B573A0B9D0,	0x7D18B68C58B9D05CE8,
	0x4173A0B9D25CE92E74,	0x40B9D05CE92E74973A,	0x405CE82E74973A4B9D,	0x482E74173A4B9D25CE,	0x44173A0B9D25CE92E7,
	0x4A0B9D05CE92E74973,	0x4D05CE82E74973A4B9,	0x4E82E74173A4B9D25C,	0x474173A0B9D25CE92E,	0x43A0B9D05CE92E7497,
	0x49D05CE82E74973A4B,	0x4CE82E74173A4B9D25,	0x4E74173A0B9D25CE92,	0x473A0B9D05CE92E749,	0x4B9D05CE82E74973A4,
	0x45CE82E74173A4B9D2,	0x42E74173A0B9D25CE9,	0x268C5CB9D05CE9D18B,	0x2B462A5CE82E75E8C5,	0x2DA3112E74173BF462,
	0x26D18C973A0B9CFA31,	0x2B68C24B9D05CF7D18,	0x25B46525CE82E6BE8C,	0x22DA3692E741725F46,	0x216D1F4973A0B82FA3,
	0x28B68BA4B9D05D17D1,	0x2C5B41D25CE82F8BE8,	0x262DA4E92E7416C5F4,	0x2316D674973A0A62FA,	0x218B6F3A4B9D04317D,
	0x28C5B39D25CE8318BE,	0x2462DDCE92E7408C5F,	0x2A316AE74973A1462F,	0x2D18B173A4B9D1A317,	0x1173A7462E5CE8D18B,
	0x10B9D7A3152E7568C5,	0x105CEFD188973BB462,	0x182E73E8C64B9CDA31,	0x14173DF46125CF6D18,	0x1A0B9AFA3292E6B68C,
	0x1D05C97D1B49725B46,	0x1E82E0BE8FA4B82DA3,	0x1741745F45D25D16D1,	0x13A0BE2FA0E92F8B68,	0x19D05B17D27496C5B4,
	0x1CE8298BEB3A4A62DA,	0x1E7410C5F79D24316D,	0x173A0C62F9CE9318B6,	0x1B9D02317EE7488C5B,	0x15CE8518BD73A5462D,
	0x12E7468C5CB9D3A316,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF8000000001FFFF,	0x75C0EB606DA316B18D,	0x7AE075B036D18B58C6,
	0x75703ED81B68C4AC63,	0x7AB81B6C0DB4635631,	0x7D5C0DB606DA31AB18,	0x7EAE06DB016D18D58C,	0x7757036D80B68C6AC6,
	0x73AB81B6C05B463563,	0x71D5C0DB622DA31AB1,	0x70EAE06DB316D18D58,	0x70757036D98B68C6AC,	0x703AB81B6CC5B46356,
	0x781D5C0DB462DA31AB,	0x7C0EAE06DA316D18D5,	0x7E0753036D18B78C6A,	0x7703AD81B68C5AC635,	0x7B81D6C0DB462D631A,
	0x468C5AC6368FC527E4,	0x4B462D631947E293F2,	0x4DA312B18EA3F049F9,	0x46D18D58C551F924FC,	0x4B68C6AC60A8FC927E,
	0x45B4635630547E493F,	0x42DA31AB1A2A3F249F,	0x416D18D58F151F924F,	0x48B68C6AC78A8FC927,	0x4C5B463563C547E493,
	0x462DA31AB3E2A3F249,	0x4316D18D5BF151F924,	0x418B68C6ADF8A8FC92,	0x48C5B46354FC547E49,	0x4462DE31A87E2B3F24,
	0x4A316B18D63F149F92,	0x4D18B58C691F8A4FC9,	0x258C6CB9D1B03747E2,	0x2AC6325CEAD81AA3F1,	0x2563192E776C0D51F8,
	0x2AB18C9739B606A8FC,	0x2D58C24B9EDB02547E,	0x26AC6525CF6D802A3F,	0x23563692E5B6C1151F,	0x21AB1F4970DB618A8F,
	0x28D58BA4B86DB1C547,	0x2C6AC1D25C36D9E2A3,	0x263564E92C1B6DF151,	0x231AB674940DB7F8A8,	0x218D5F3A4A06DAFC54,
	0x28C6AB9D27036C7E2A,	0x2C6355CE9181B63F15,	0x2631AAE74AC0DB1F8A,	0x2B18D173A7606C8FC5,	0x16C0DD1F8A9CE4D18B,
	0x1B606A8FC54E7368C5,	0x1DB03547E2A739B462,	0x16D81AA3F1539CDA31,	0x1B6C0951F8A9CF6D18,	0x1DB600A8FE54E6B68C,
	0x16DB04547F2A725B46,	0x136D862A3F95382DA3,	0x11B6C7151DCA9D16D1,	0x10DB678A8CE54F8B68,	0x106DB7C54672A6C5B4,
	0x1036DFE2A3395262DA,	0x181B6BF1539CA8316D,	0x1C0DB1F8A9CE5518B6,	0x1606D8FC54E72A8C5B,	0x1B036C7E2A7395462D,
	0x1D81B23F1539CBA316,				

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x75C0EA59A4D32C47E2,	0x7AE0752CD0699623F1,
	0x75703A966A34CB11F8,	0x7AB8194B371A6488FC,	0x7D5C0CA5998D32447E,	0x7EAE0252CEC698223F,	0x7757052965634D111F,
	0x73AB8694B0B1A7888F,	0x71D5C34A5A58D3C447,	0x70EAE1A52F2C69E223,	0x707574D2959635F111,	0x703ABE6948CB1BF888,
	0x781D5B34A6658CFC44,	0x7C0EAD9A5132C67E22,	0x7E0752CD2A99623F11,	0x7703A966974CB11F88,	0x7B81D4B349A6588FC4,
	0x434CB11F8A8FC56996,	0x41A6588FC547E2B4CB,	0x48D32C47E2A3F15A65,	0x4C699223F151F9AD32,	0x4634C911F8A8FCD699,
	0x4B1A6088FC547F6B4C,	0x458D34447E2A3EB5A6,	0x42C69E223F151E5AD3,	0x49634F111F8A8F2D69,	0x4CB1A7888FC54796B4,
	0x4658D7C447E2A2CB5A,	0x432C6FE223F15065AD,	0x499633F111F8A932D6,	0x44CB19F888FC54996B,	0x4A6588FC447E2B4CB5,
	0x4D32C47E223F15A65A,	0x4699623F111F8AD32D,	0x223F1659A52CD347E2,	0x211F8F2CD29668A3F1,	0x288FC396694B3551F8,
	0x2447E1CB34A59AA8FC,	0x2223F4E59A52CC547E,	0x2111FA72CD29662A3F,	0x2888FD396694B3151F,	0x2C447E9CB34A598A8F,
	0x2E223B4E59A52DC547,	0x2F1119A72CD297E2A3,	0x2F888CD396694BF151,	0x2FC44669CB34A5F8A8,	0x27E22334E59A52FC54,
	0x23F1159A72CD287E2A,	0x21F88ACD3966943F15,	0x28FC41669CB34B1F8A,	0x247E24B34E59A48FC5,	0x14B34D1F8B703A6996,
	0x1A59A28FC7B81C34CB,	0x152CD547E1DC0F1A65,	0x12966AA3F2EE078D32,	0x194B3151FB7702C699,	0x14A598A8FFBB81634C,
	0x1A52CC547DDDC0B1A6,	0x1D29662A3CEEE058D3,	0x1694B7151C77712C69,	0x134A5F8A8C3BB99634,	0x19A52FC5441DDCCB1A,
	0x1CD297E2A00EEE658D,	0x16694BF152077732C6,	0x1B34A1F8AB03BA9963,	0x159A50FC5781DD4CB1,	0x12CD2C7E29C0EFA658,
	0x1966923F16E076D32C,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x75C0EA59A4D32C47E2,	0x7AE0752CD0699623F1,
	0x75703A966A34CB11F8,	0x7AB8194B371A6488FC,	0x7D5C0CA5998D32447E,	0x7EAE0252CEC698223F,	0x7757052965634D111F,
	0x73AB8694B0B1A7888F,	0x71D5C34A5A58D3C447,	0x70EAE1A52F2C69E223,	0x707574D2959635F111,	0x703ABE6948CB1BF888,
	0x781D5B34A6658CFC44,	0x7C0EAD9A5132C67E22,	0x7E0752CD2A99623F11,	0x7703A966974CB11F88,	0x7B81D4B349A6588FC4,
	0x423F11A65AD32D47E2,	0x411F88D32D6996A3F1,	0x488FC46996B4CB51F8,	0x4447E634CB5A64A8FC,	0x4223F31A65AD32547E,
	0x4111FD8D32D6982A3F,	0x4888FAC6996B4D151F,	0x4C4479634CB5A78A8F,	0x4E223CB1A65AD3C547,	0x4F111E58D32D69E2A3,
	0x4F888B2C6996B5F151,	0x4FC4419634CB5BF8A8,	0x47E224CB1A65ACFC54,	0x43F112658D32D67E2A,	0x41F88D32C6996A3F15,
	0x48FC4699634CB51F8A,	0x447E234CB1A65A8FC5,	0x24B34D1F888FC59669,	0x2A59A28FC447E3CB34,	0x252CD547E223F0E59A,
	0x22966AA3F111F872CD,	0x294B3151F888FD3966,	0x24A598A8FC447E9CB3,	0x2A52CC547E223F4E59,	0x2D29662A3F111FA72C,
	0x2694B7151F888ED396,	0x234A5F8A8FC44669CB,	0x29A52FC547E22334E5,	0x2CD297E2A3F1119A72,	0x26694BF151F888CD39,
	0x2B34A1F8A8FC45669C,	0x259A50FC547E22B34E,	0x22CD2C7E2A3F1059A7,	0x2966923F151F892CD3,	0x134CB6E0768FC49669,
	0x11A65F703947E34B34,	0x18D32BB81EA3F0A59A,	0x1C6995DC0D51F852CD,	0x1634CEEE04A8FD2966,	0x1B1A677700547E94B3,
	0x158D33BB822A3F4A59,	0x12C699DDC3151FA52C,	0x196348EEE38A8ED296,	0x1CB1A07773C546694B,	0x1658D03BBBE2A334A5,
	0x132C681DDFF1519A52,	0x1996340EEDF8A8CD29,	0x14CB1E0774FC556694,	0x1A658F03B87E2AB34A,	0x1D32C381DE3F1459A5,
	0x169965C0ED1F8B2CD2,				
	0x4FFFFFFFFC00000000,	0x2FFFF80003FFFE0000,	0x1FFFF800000001FFFF,	0x67213A36C4A795C5A3,	0x6B909D1B6053CBE2D1,
	0x6DC84A8DB229E5F168,	0x6EE42146D914F2F8B4,	0x677210A36E8A787C5A,	0x63B90C51B5453C3E2D,	0x69DC8628D8A29F1F16,
	0x64EE43146E514E8F8B,	0x6277258A3728A747C5,	0x613B96C51B9453A3E2,	0x609DCB628FCA28D1F1,	0x684EE5B145E51568F8,
	0x642776D8A0F28AB47C,	0x6213BB6C5279445A3E,	0x6909D9B6293CA22D1F,	0x6C84E8DB169E51168F,	0x6E42746D894F298B47,
	0x529E50E97237B08DB1,	0x514F2874B91BD946D8,	0x58A7903A5C8DECA36C,	0x5453CC1D2C46F651B6,	0x5A29E60E96237A28DB,
	0x5514F7074B11BD146D,	0x528A7B83A588DF8A36,	0x59453DC1D2C46EC51B,	0x5CA29AE0EB6237628D,	0x5E51497077B11BB146,
	0x5F28A4B83BD88CD8A3,	0x5794525C1DEC476C51,	0x53CA2D2E0EF623B628,	0x59E51697077B10DB14,	0x54F28F4B81BD886D8A,
	0x5A7943A5C0DEC436C5,	0x553CA1D2E06F631B62,	0x31D2E6B0D51B62E427,	0x30E977586A8DB17213,	0x3074BBAC3546D9B909,
	0x383A5DD618A36DDC84,	0x3C1D2AEB0C51B6EE42,	0x3E0E95758628DA7721,	0x37074EBAC3146D3B90,	0x3B83A35D618A369DC8,
	0x35C1D1AEB2C51A4EE4,	0x32E0E8D75B628C2772,	0x3970706BADB14613B9,	0x34B83C35D6D8A309DC,	0x3A5C1E1AEB6C5084EE,
	0x3D2E0B0D75B6284277,	0x3E970586B8DB15213B,	0x374B82C35C6D8B909D,	0x33A5C561AE36C5C84E,	0x46D8C6F638B47AC35,
	0xA36C237B3C5A3D61A,	0x51B611BDBE2D0EB0D,	0x28DB08DEDF1697586,	0x146DC46F4F8B4BAC3,	0x8A36E23787C5B5D61,
	0xC51B311BE3E2DAEB0,	0x628DD88DD1F16D758,	0xB146EC46E8F8A6BAC,	0xD8A37623747C435D6,	0x6C51FB119A3E21AEB,
	0xB628BD88ED1F10D75,	0xDB145EC4568F986BA,	0x6D8A6F620B47CC35D,	0x36C537B105A3F61AE,	0x1B629BD8A2D1EB0D7,
	0x8DB10DEC7168F586B,				

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x67213A36C4A795C5A3, 0x6B909D1B6053CBE2D1, 0x6DC84A8DB229E5F168, 0x6EE42146D914F2F8B4, 0x677210A36E8A787C5A, 0x63B90C51B5453C3E2D, 0x69DC8628D8A29F1F16, 0x64EE43146E514E8F8B, 0x6277258A3728A747C5, 0x613B96C51B9453A3E2, 0x609DCB628FCA28D1F1, 0x684EE5B145E51568F8, 0x642776D8A0F28AB47C, 0x6213BB6C5279445A3E, 0x6909D9B6293CA22D1F, 0x6C84E8DB169E51168F, 0x6E42746D894F298B47, 0x51D2E14F2AE49CE427, 0x50E970A795724F7213, 0x5074BC53CAB927B909, 0x583A5A29E75C93DC84, 0x5C1D2D14F3AE48EE42, 0x5E0E928A79D7247721, 0x570749453CEB933B90, 0x5B83A4A29E75C89DC8, 0x55C1D6514D3AE44EE4, 0x52E0EF28A49D722772, 0x59707794524EB813B9, 0x54B83BCA29275D09DC, 0x5A5C19E51493AE84EE, 0x5D2E0CF28A49D64277, 0x5E9702794724EB213B, 0x574B853CA39275909D, 0x53A5C29E51C93BC84E, 0x346D8C6F6074B853CA, 0x3A36C237B03A5C29E5, 0x351B611BD81D2F14F2, 0x328DB08DEE0E968A79, 0x3146DC46F7074B453C, 0x38A36E237B83A4A29E, 0x3C51B311BDC1D2514F, 0x3628DD88DEE0E928A7, 0x3B146EC46D70759453, 0x3D8A376234B83BCA29, 0x36C51FB11A5C1DE514, 0x3B628BD88D2E0EF28A, 0x3DB145EC4697067945, 0x36D8A6F6234B833CA2, 0x336C537B13A5C09E51, 0x31B629BD89D2E14F28, 0x38DB10DEC4E970A794, 0x29E57168E37B1724E, 0x14F2F8B451BD8B927, 0x8A797C5A08DED5C93, 0x453CBE2D046F7AE49, 0xA29E1F16A237BD724, 0x514F0F8B711BCEB92, 0x28A7C7C5988DE75C9, 0x9453A3E2EC46F3AE4, 0xCA29D1F1762369D72, 0xE514E8F8BB11A4EB9, 0xF28A347C7D88D275C, 0x79455A3E1EC4693AE, 0x3CA2AD1F2F62249D7, 0x9E51168FB7B1124EB, 0x4F288B47DBD899275, 0xA79445A3CDEC5C93A, 0x53CA62D1C6F62E49D,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x67213A36C474B9AC35, 0x6B909D1B603A5DD61A, 0x6DC84A8DB01D2EEB0D, 0x6EE42146DA0E977586, 0x677210A36F074ABAC3, 0x63B90C51B783A55D61, 0x69DC8628D9C1D3AEB0, 0x64EE43146EE0E8D758, 0x6277258A3570746BAC, 0x613B96C518B83A35D6, 0x609DCB628E5C1C1AEB, 0x684EE5B1452E0F0D75, 0x642776D8A2970786BA, 0x6213BB6C534B82C35D, 0x6909D9B62BA5C161AE, 0x6C84E8DB15D2E0B0D7, 0x6E42746D88E971586B, 0x529E50E972E49CE427, 0x514F2874B9724F7213, 0x58A7903A5EB927B909, 0x5453CC1D2F5C93DC84, 0x5A29E60E97AE48EE42, 0x5514F70749D7247721, 0x528A7B83A4EB933B90, 0x59453DC1D275C89DC8, 0x5CA29AE0E93AE44EE4, 0x5E514970749D722772, 0x5F28A4B83A4EB813B9, 0x5794525C1D275D09DC, 0x53CA2D2E0C93AE84EE, 0x59E516970649D64277, 0x54F28F4B8324EB213B, 0x5A7943A5C39275909D, 0x553CA1D2E1C93BC84E, 0x346D8C6F60A7943A5C, 0x3A36C237B053CA1D2E, 0x351B611BDA29E40E97, 0x328DB08DED14F3074B, 0x3146DC46F68A7983A5, 0x38A36E2379453DC1D2, 0x3C51B311BCA29EE0E9, 0x3628DD88DE514F7074, 0x3B146EC46F28A6B83A, 0x3D8A37623794525C1D, 0x36C51FB11BCA292E0E, 0x3B628BD88DE5149707, 0x3DB145EC44F28B4B83, 0x36D8A6F6227945A5C1, 0x336C537B113CA3D2E0, 0x31B629BD8A9E50E970, 0x38DB10DEC54F2874B8, 0x1D2E6B0D637B1724E, 0xE97758691BD8B927, 0x74BBAC348DED5C93, 0x83A5DD61846F7AE49, 0xC1D2AEB0E237BD724, 0xE0E95758711BCEB92, 0x7074EBAC188DE75C9, 0xB83A35D62C46F3AE4, 0x5C1D1AEB362369D72, 0x2E0E8D75BB11A4EB9, 0x970706BAFD88D275C, 0x4B83C35D5EC4693AE, 0xA5C1E1AEAF62249D7, 0xD2E0B0D777B1124EB, 0xE970586B9BD899275, 0x74B82C35CDEC5C93A, 0x3A5C561AC6F62E49D,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x67213A36C474B9AC35, 0x6B909D1B603A5DD61A, 0x6DC84A8DB01D2EEB0D, 0x6EE42146DA0E977586, 0x677210A36F074ABAC3, 0x63B90C51B783A55D61, 0x69DC8628D9C1D3AEB0, 0x64EE43146EE0E8D758, 0x6277258A3570746BAC, 0x613B96C518B83A35D6, 0x609DCB628E5C1C1AEB, 0x684EE5B1452E0F0D75, 0x642776D8A2970786BA, 0x6213BB6C534B82C35D, 0x6909D9B62BA5C161AE, 0x6C84E8DB15D2E0B0D7, 0x6E42746D88E971586B, 0x51D2E14F2A37B08DB1, 0x50E970A7951BD946D8, 0x5074BC53C88DECA36C, 0x583A5A29E446F651B6, 0x5C1D2D14F2237A28DB, 0x5E0E928A7B11BD146D, 0x570749453D88DF8A36, 0x5B83A4A29EC46EC51B, 0x55C1D6514F6237628D, 0x52E0EF28A7B11BB146, 0x5970779453D88CD8A3, 0x54B83BCA29EC476C51, 0x5A5C19E516F623B628, 0x5D2E0CF28B7B10DB14, 0x5E97027945BD886D8A, 0x574B853CA0DEC436C5, 0x53A5C29E506F631B62, 0x329E57168D1B62E427, 0x314F2F8B468DB17213, 0x38A797C5A146D9B909, 0x3453CBE2D0A36DDC84, 0x3A29E1F16851B6EE42, 0x3514F0F8B628DA7721, 0x328A7C7C5B146D3B90, 0x39453A3E2D8A369DC8, 0x3CA29D1F16C51A4EE4, 0x3E514E8F8B628C2772, 0x3F28A347C5B14613B9, 0x379455A3E2D8A309DC, 0x33CA2AD1F36C5084EE, 0x39E51168F9B6284277, 0x34F288B47CDB15213B, 0x3A79445A3C6D8B909D, 0x353CA62D1E36C5C84E, 0x46D8C6F63586BC5A3, 0xA36C237B3AC35E2D1, 0x51B611BD9D61BF168, 0x28DB08DEEEB0CF8B4, 0x146DC46F575867C5A, 0x8A36E237ABAC23E2D, 0xC51B311BF5D611F16, 0x628DD88DDAEB08F8B, 0xB146EC46CD75947C5, 0xD8A3762346BADA3E2, 0x6C51FB11835D6D1F1, 0xB628BD88E1AEB68F8, 0xDB145EC470D74B47C, 0x6D8A6F62186BA5A3E, 0x36C537B12C35C2D1F, 0x1B629BD8961AF168F, 0x8DB10DEC6B0D78B47,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x6721394F2874B9724E, 0x6B9098A7943A5CB927, 0x6DC84C53C81D2F5C93, 0x6EE42229E60E97AE49, 0x67721514F3074BD724, 0x63B90A8A7B83A4EB92, 0x69DC81453DC1D275C9, 0x64EE44A29EE0E93AE4, 0x627726514D70749D72, 0x613B9728A4B83A4EB9, 0x609DCF94525C1D275C, 0x684EE3CA292E0E93AE, 0x642771E516970649D7, 0x6213BCF28B4B8324EB, 0x6909DA7947A5C19275, 0x6C84ED3CA1D2E1C93A, 0x6E42729E50E970E49D, 0x546D88E973586AE427, 0x5A36C074BBAC357213, 0x551B603A5DD61BB909, 0x528DB41D2EEB0DDC84, 0x5146DE0E957586EE42, 0x58A36F074ABAC27721, 0x5C51B383A75D613B90, 0x5628DDC1D1AEB09DC8, 0x5B146AE0E8D7584EE4, 0x5D8A3170746BAC2772, 0x56C51CB83835D613B9, 0x5B628A5C1E1AEB09DC, 0x5DB1452E0F0D7484EE, 0x56D8A6970586BA4277, 0x536C574B82C35D213B, 0x51B62BA5C161AF909D, 0x58DB11D2E2B0D7C84E, 0x329E546F611B623A5C, 0x314F2A37B28DB01D2E, 0x38A7911BD946D80E97, 0x3453C88DECA36D074B, 0x3A29E446F451B783A5, 0x3514F6237A28DBC1D2, 0x328A7B11BF146CE0E9, 0x39453D88DD8A377074, 0x3CA29EC46EC51AB83A, 0x3E514F6237628C5C1D, 0x3F28A7B119B1472E0E, 0x379453D88ED8A29707, 0x33CA2DEC476C514B83, 0x39E516F621B629A5C1, 0x34F28B7B10DB15D2E0, 0x3A7941BD886D8AE970, 0x353CA0DEC636C474B8, 0x1D2E5C93A37B1AC35, 0xE972E49D1BD9D61A, 0x74BD724C8DECEB0D, 0x83A5EB92446F77586, 0xC1D2F5C92237ABAC3, 0xE0E93AE4B11BD5D61, 0x70749D72588DFAEB0, 0xB83A4EB92C46ED758, 0x5C1D275CB62366BAC, 0x2E0E93AE7B11A35D6, 0x970749D73D88C1AEB, 0x4B83A4EB9EC470D75, 0xA5C19275EF62386BA, 0xD2E0C93AF7B10C35D, 0xE970649D5BD8961AE, 0x74B8724E8DEC4B0D7, 0x3A5C392746F63586B,

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x6721394F2874B9724E, 0x6B9098A7943A5CB927, 0x6DC84C53C81D2F5C93, 0x6EE42229E60E97AE49, 0x67721514F3074BD724, 0x63B90A8A7B83A4EB92, 0x69DC81453DC1D275C9, 0x64EE44A29EE0E93AE4, 0x627726514D70749D72, 0x613B9728A4B83A4EB9, 0x609DCF94525C1D275C, 0x684EE3CA292E0E93AE, 0x642771E516970649D7, 0x6213BCF28B4B8324EB, 0x6909DA7947A5C19275, 0x6C84ED3CA1D2E1C93A, 0x6E42729E50E970E49D, 0x51D2E236C637B053CA, 0x50E9751B611BD829E5, 0x5074BA8DB08DED14F2, 0x583A5946D846F68A79, 0x5C1D28A36E237B453C, 0x5E0E9451B711BCA29E, 0x57074E28D988DE514F, 0x5B83A3146EC46F28A7, 0x55C1D58A3762379453, 0x52E0EEC51BB11BCA29, 0x597073628FD88DE514, 0x54B83DB145EC46F28A, 0x5A5C1ED8A2F6227945, 0x5D2E0B6C537B113CA2, 0x5E9701B629BD889E51, 0x574B80DB14DEC54F28, 0x53A5C46D886F62A794, 0x346D8F168CA794E427, 0x3A36C78B4453CB7213, 0x351B67C5A229E5B909, 0x328DB3E2D114F3DC84, 0x3146D9F16A8A78EE42, 0x38A368F8B5453C7721, 0x3C51B47C58A29F3B90, 0x3628DA3E2E514E9DC8, 0x3B146D1F1728A64EE4, 0x3D8A368F8B94522772, 0x36C51B47C7CA2813B9, 0x3B628DA3E1E51509DC, 0x3DB142D1F0F28A84EE, 0x36D8A168FA79444277, 0x336C50B47D3CA3213B, 0x31B62C5A3E9E51909D, 0x38DB162D1D4F29C84E, 0x29E546F62E49DC5A3, 0x14F2A37B1724FE2D1, 0x8A7911BDAB927F168, 0x453C88DEF5C92F8B4, 0xA29E446F7AE487C5A, 0x514F62379D7243E2D, 0x28A7B11BCEB931F16, 0x9453D88DE75C88F8B, 0xCA29EC46D3AE547C5, 0xE514F62349D73A3E2, 0xF28A7B11A4EB8D1F1, 0x79453D88D275D68F8, 0x3CA2DEC4493AEB47C, 0x9E516F62249D65A3E, 0x4F28B7B1324EA2D1F, 0xA7941BD8B9275168F, 0x53CA0DEC5C93B8B47,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x67123A9694B335E187, 0x6B891D4B48599BF0C3, 0x6DC48AA5A62CCDF861, 0x6EE24552D11667FC30, 0x677122A96A8B32FE18, 0x63B89154B745987F0C, 0x61DC4CAA59A2CC3F86, 0x68EE22552CD1661FC3, 0x6477152A9668B30FE1, 0x623B8E954B345987F0, 0x691DC34AA59A2CC3F8, 0x648EE5A550CD1661FC, 0x624772D2AA668A30FE, 0x6123B969573344187F, 0x6891DCB4A999A30C3F, 0x6C48EA5A56CCD1861F, 0x6E24752D296669C30F, 0x52CCD079E23B70A5A5, 0x5166683CF11DB952D2, 0x58B3301E788EDCA969, 0x5459980F3C476F54B4, 0x5A2CCC079E23B6AA5A, 0x5D166603CF11DA552D, 0x568B3701E788ED2A96, 0x53459F80F1C476954B, 0x59A2CBC07AE23B4AA5, 0x5CD161E03F711DA552, 0x5668B4F01DB88ED2A9, 0x53345E780EDC476954, 0x599A2F3C076E22B4AA, 0x5CCD179E03B7105A55, 0x56668BCF01DB892D2A, 0x5B3341E780EDC49695, 0x5599A0F3C076E34B4A, 0x30F3C699954B4AE247, 0x3079E74CCAA5A57123, 0x303CF3A66552D3B891, 0x301E7DD332A969DC48, 0x380F3AE99954B4EE24, 0x3C079974CCAA5A7712, 0x3E03CCBA66552C3B89, 0x3F01E65D312A971DC4, 0x3780F32E9A954A8EE2, 0x33C079974F4AA44771, 0x39E03CCBA5A55323B8, 0x3CF01E65D2D2A891DC, 0x3E780B32E9695448EE, 0x3F3C019974B4AA2477, 0x379E04CCBA5A55123B, 0x33CF02665D2D2B891D, 0x31E785332E9695C48E, 0x52D2C76E3C30FA665, 0xA96923B73E187D332, 0x54B491DBBF0C2E999, 0xAA5A08EDFF86174CC, 0x552D4476DFC30BA66, 0x2A96E23B4FE185D33, 0x954B711D87F0D2E99, 0x4AA5B88EC3F87974C, 0xA552DC4761FC2CBA6, 0xD2A96E23B0FE065D3, 0x6954B711D87F132E9, 0xB4AA5B88CC3F99974, 0x5A556DC4461FCCCBA, 0x2D2AF6E2030FE665D, 0x96953B712187F332E, 0x4B4A9DB8B0C3E9997, 0xA5A50EDC7861F4CCB,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x67123A96943CF1A665, 0x6B891D4B481E79D332, 0x6DC48AA5A40F3CE999, 0x6EE24552D0079F74CC, 0x677122A96A03CEBA66, 0x63B89154B701E65D33, 0x61DC4CAA5B80F32E99, 0x68EE22552FC079974C, 0x6477152A95E03CCBA6, 0x623B8E9548F01E65D3, 0x691DC34AA6780F32E9, 0x648EE5A5533C079974, 0x624772D2AB9E02CCBA, 0x6123B96957CF00665D, 0x6891DCB4A9E781332E, 0x6C48EA5A54F3C09997, 0x6E24752D2879E14CCB, 0x50F3C1666A3B70A5A5, 0x5079E0B3351DB952D2, 0x503CF459988EDCA969, 0x501E7A2CCC476F54B4, 0x580F3D166623B6AA5A, 0x5C079E8B3311DA552D, 0x5E03CB459B88ED2A96, 0x5F01E1A2CDC476954B, 0x5780F4D166E23B4AA5, 0x53C07E68B3711DA552, 0x59E03B3459B88ED2A9, 0x5CF0199A2EDC476954, 0x5E780CCD176E22B4AA, 0x5F3C06668BB7105A55, 0x579E033345DB892D2A, 0x53CF0599A0EDC49695, 0x51E782CCD076E34B4A, 0x32CCD7861D4B4AE247, 0x31666FC30EA5A57123, 0x38B337E18552D3B891, 0x34599FF0C2A969DC48, 0x3A2CCBF86154B4EE24, 0x3D1661FC30AA5A7712, 0x368B30FE1A552C3B89, 0x3345987F0D2A971DC4, 0x39A2CC3F86954A8EE2, 0x3CD1661FC34AA44771, 0x3668B30FE1A55323B8, 0x33345987F2D2A891DC, 0x399A28C3F9695448EE, 0x3CCD1061FCB4AA2477, 0x36668C30FE5A55123B, 0x3B3346187D2D2B891D, 0x3599A70C3E9695C48E, 0x52D2C76E34CCBE187, 0xA96923B73A665F0C3, 0x54B491DB9D333F861, 0xAA5A08EDEE999FC30, 0x552D4476D74CCFE18, 0x2A96E23B4BA667F0C, 0x954B711DA5D323F86, 0x4AA5B88EF2E981FC3, 0xA552DC475974D0FE1, 0xD2A96E238CBA787F0, 0x6954B711E65D2C3F8, 0xB4AA5B88F32E861FC, 0x5A556DC45997430FE, 0x2D2AF6E20CCBA187F, 0x96953B712665D0C3F, 0x4B4A9DB89332F861F, 0xA5A50EDC69997C30F,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x66C0DB462D703B4E72, 0x6B606DA316B81CA739, 0x6DB036D1895C0F539C, 0x66D81B68C6AE06A9CE, 0x6B6C0DB463570254E7, 0x6DB602DA33AB812A73, 0x66DB016D19D5C19539, 0x636D80B68CEAE1CA9C, 0x61B6C45B447570E54E, 0x60DB662DA03AB872A7, 0x606DB316D01D5D3953, 0x6036D98B680EAF9CA9, 0x681B68C5B60757CE54, 0x6C0DB462DB03AAE72A, 0x6606DA316F81D47395, 0x6B036D18B5C0EB39CA, 0x6D81B68C5AE0749CE5, 0x55C0EAC6364FC8D18B, 0x5AE075631927E568C5, 0x55703AB18C93F3B462, 0x5AB81D58C649F8DA31, 0x5D5C0EAC6124FD6D18, 0x5EAE035630927EB68C, 0x575701AB1A493E5B46, 0x53AB80D58F249E2DA3, 0x51D5C46AC7924F16D1, 0x50EAE63563C9278B68, 0x5075731AB3E492C5B4, 0x503AB98D5BF24862DA, 0x581D58C6ADF924316D, 0x5C0EAC6354FC9318B6, 0x5E075631AA7E488C5B, 0x5703AB18D53F25462D, 0x5B81D58C689F93A316, 0x358C6D1F89A316D81B, 0x3AC6328FC6D18B6C0D, 0x35631D47E368C5B606, 0x3AB18AA3F1B462DB03, 0x3D58C151FADA316D81, 0x36AC60A8FD6D19B6C0, 0x335634547CB68CDB60, 0x31AB1E2A3C5B466DB0, 0x38D58F151E2DA236D8, 0x3C6AC78A8F16D01B6C, 0x363567C5458B680DB6, 0x331AB7E2A0C5B406DB, 0x318D5BF15062DB036D, 0x38C6A9F8AA316D81B6, 0x3C6350FC5518B6C0DB, 0x3631AC7E2A8C5B606D, 0x3B18D23F17462DB036, 0x68C5C9F929CE547E2, 0xB462A4FC94E72A3F1, 0xDA31127E6A73951F8, 0x6D18C93F1539CA8FC, 0xB68C249F8A9CE547E, 0x5B46124FE54E62A3F, 0x2DA34927F2A73151F, 0x16D1E493F95398A8F, 0x8B68F249DCA9DC547, 0xC5B47924CE54FE2A3, 0x62DA7C92672A7F151, 0x316D7E4933953F8A8, 0x18B6BF24B9CA8FC54, 0x8C5B1F925CE547E2A, 0x462DCFC90E72A3F15, 0xA316A7E4A73951F8A, 0xD18B13F2539CA8FC5,

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x66C0DB462D703B4E72, 0x6B606DA316B81CA739, 0x6DB036D1895C0F539C, 0x66D81B68C6AE06A9CE, 0x6B6C0DB463570254E7, 0x6DB602DA33AB812A73, 0x66DB016D19D5C19539, 0x636D80B68CEAE1CA9C, 0x61B6C45B447570E54E, 0x60DB662DA03AB872A7, 0x606DB316D01D5D3953, 0x6036D98B680EAF9CA9, 0x681B68C5B60757CE54, 0x6C0DB462DB03AAE72A, 0x6606DA316F81D47395, 0x6B036D18B5C0EB39CA, 0x6D81B68C5AE0749CE5, 0x558C6AE0765CE8D81B, 0x5AC63570392E756C0D, 0x55631AB81C973BB606, 0x5AB18D5C0E4B9CDB03, 0x5D58C6AE0525CF6D81, 0x56AC67570292E7B6C0, 0x535633AB834972DB60, 0x51AB19D5C3A4B86DB0, 0x58D588EAE1D25C36D8, 0x5C6AC07570E92E1B6C, 0x5635603ABA74960DB6, 0x531AB01D5F3A4A06DB, 0x518D5C0EAF9D25036D, 0x58C6AE0755CE9381B6, 0x5C635703AAE748C0DB, 0x5631AB81D573A5606D, 0x5B18D5C0E8B9D3B036, 0x368C5C9F91631AB81D, 0x3B462A4FCAB18D5C0E, 0x3DA31127E558C6AE07, 0x36D18C93F2AC635703, 0x3B68C249FB5631AB81, 0x35B46124FDAB19D5C0, 0x32DA34927CD58CEAE0, 0x316D1E493C6AC67570, 0x38B68F249E35623AB8, 0x3C5B47924F1AB01D5C, 0x362DA7C9258D580EAE, 0x3316D7E490C6AC0757, 0x318B6BF248635703AB, 0x38C5B1F92631AB81D5, 0x3462DCFC9318D5C0EA, 0x3A316A7E498C6AE075, 0x3D18B13F26C635703A, 0x5C0ED39CA4FC92E74, 0xAE0729CE527E4973A, 0x5703D4E7093F24B9D, 0xAB81AA73A49F925CE, 0xD5C09539D24FC92E7, 0xEAE04A9CC927F4973, 0x7570654E6493FA4B9, 0x3AB872A73249FD25C, 0x1D5C3953B924EE92E, 0xEAE1CA9FC9267497, 0x7574E54FE4933A4B, 0x3ABE72A7F2499D25, 0x81D5F3951F925CE92, 0xC0EAB9CA8FC92E749, 0xE0751CE567E4973A4, 0x703ACE7293F24B9D2, 0xB81D273949F925CE9,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x66C0DB462D631B47E2, 0x6B606DA316B18CA3F1, 0x6DB036D18958C751F8, 0x66D81B68C6AC62A8FC, 0x6B6C0DB4635630547E, 0x6DB602DA31AB182A3F, 0x66DB016D18D58D151F, 0x636D80B68C6AC78A8F, 0x61B6C45B463563C547, 0x60DB662DA31AB1E2A3, 0x606DB316D18D59F151, 0x6036D98B68C6ADF8A8, 0x681B68C5B46356FC54, 0x6C0DB462DA31AA7E2A, 0x6606DA316F18D43F15, 0x6B036D18B58C6B1F8A, 0x6D81B68C5AC6348FC5, 0x55C0EAC6365CE8D81B, 0x5AE07563192E756C0D, 0x55703AB18C973BB606, 0x5AB81D58C64B9CDB03, 0x5D5C0EAC6125CF6D81, 0x5EAE03563292E7B6C0, 0x575701AB1B4972DB60, 0x53AB80D58FA4B86DB0, 0x51D5C46AC5D25C36D8, 0x50EAE63560E92E1B6C, 0x5075731AB274960DB6, 0x503AB98D5B3A4A06DB, 0x581D58C6AF9D25036D, 0x5C0EAC6355CE9381B6, 0x5E075631AAE748C0DB, 0x5703AB18D573A5606D, 0x5B81D58C68B9D3B036, 0x368C5C9F91703AB18D, 0x3B462A4FCAB81D58C6, 0x3DA31127E55C0EAC63, 0x36D18C93F2AE075631, 0x3B68C249FB5703AB18, 0x35B46124FFAB80D58C, 0x32DA34927DD5C06AC6, 0x316D1E493CEAE03563, 0x38B68F249C75711AB1, 0x3C5B47924C3AB98D58, 0x362DA7C9241D5CC6AC, 0x3316D7E4900EAE6356, 0x318B6BF24A075631AB, 0x38C5B1F92703AB18D5, 0x3462DCFC9381D58C6A, 0x3A316A7E49C0EAC635, 0x3D18B13F26E075631A, 0x58C6D1F8A4FC92E74, 0xAC6328FC527E4973A, 0x5631D47E093F24B9D, 0xAB18AA3F249F925CE, 0xD58C151F924FC92E7, 0x6AC60A8FC927F4973, 0x35634547E493FA4B9, 0x1AB1E2A3F249FD25C, 0x8D58F151F924EE92E, 0xC6AC78A8FC9267497, 0x63567C547E4933A4B, 0x31AB7E2A3F2499D25, 0x18D5BF151F925CE92, 0x8C6A9F8A8FC92E749, 0xC6350FC567E4973A4, 0x631AC7E293F24B9D2, 0xB18D23F149F925CE9,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x66C0DAE075631B2E74, 0x6B606D703AB18C973A, 0x6DB032B81D58C64B9D, 0x66D81D5C0EAC6325CE, 0x6B6C0EAE07563092E7, 0x6DB6075701AB194973, 0x66DB03AB80D58DA4B9, 0x636D81D5C06AC7D25C, 0x61B6C0EAE23562E92E, 0x60DB6075731AB07497, 0x606DB03AB98D593A4B, 0x6036D81D5CC6AD9D25, 0x681B6C0EAC6357CE92, 0x6C0DB6075631AAE749, 0x6606DF03AB18D573A4, 0x6B036B81D58C6AB9D2, 0x6D81B5C0EAC6345CE9, 0x568C5AC6368FC4D81B, 0x5B462D631947E36C0D, 0x5DA312B18EA3F1B606, 0x56D18D58C551F8DB03, 0x5B68C6AC60A8FD6D81, 0x55B4635630547FB6C0, 0x52DA31AB1A2A3EDB60, 0x516D18D58F151E6DB0, 0x58B68C6AC78A8E36D8, 0x5C5B463563C5461B6C, 0x562DA31AB3E2A20DB6, 0x5316D18D5BF15006DB, 0x518B68C6ADF8A9036D, 0x58C5B46354FC5581B6, 0x5462DE31A87E2AC0DB, 0x5A316B18D63F15606D, 0x5D18B58C691F8BB036, 0x35C0EC9F91A316B18D, 0x3AE0724FCAD18B58C6, 0x35703927E768C4AC63, 0x3AB81C93F1B4635631, 0x3D5C0A49FADA31AB18, 0x3EAE0124FD6D18D58C, 0x375704927CB68C6AC6, 0x33AB86493C5B463563, 0x31D5C7249E2DA31AB1, 0x30EAE7924F16D18D58, 0x307577C9258B68C6AC, 0x303ABFE490C5B46356, 0x381D5BF24862DA31AB, 0x3C0EA9F926316D18D5, 0x3E0754FC9118B78C6A, 0x3703AA7E4A8C5AC635, 0x3B81D13F27462D631A, 0x58C6CB9D24FC947E2, 0xAC6325CE927E4A3F1, 0x563192E7493F351F8, 0xAB18C973A49F8A8FC, 0xD58C24B9D24FC547E, 0x6AC6525CC927E2A3F, 0x3563692E6493F151F, 0x1AB1F4973249F8A8F, 0x8D58BA4BB924FC547, 0xC6AC1D25FC927E2A3, 0x63564E92FE493F151, 0x31AB67497F249F8A8, 0x18D5F3A49F924FC54, 0x8C6AB9D24FC927E2A, 0xC6355CE927E483F15, 0x631AAE7493F251F8A, 0xB18D173A49F928FC5,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x668C5B462C5CE9D18B, 0x6B462DA3142E75E8C5, 0x6DA316D188173BF462, 0x66D18B68C60B9CFA31, 0x6B68C5B46105CF7D18, 0x65B462DA3282E6BE8C, 0x62DA316D1B41725F46, 0x616D18B68FA0B82FA3, 0x68B68C5B45D05D17D1, 0x6C5B462DA0E82F8BE8, 0x662DA316D27416C5F4, 0x6316D18B6B3A0A62FA, 0x618B68C5B79D04317D, 0x68C5B462D9CE8318BE, 0x6462DA316EE7408C5F, 0x6A316D18B573A1462F, 0x6D18B68C58B9D1A317, 0x5173A0B9D25CE8D18B, 0x50B9D05CE92E7568C5, 0x505CE82E74973BB462, 0x582E74173A4B9CDA31, 0x54173A0B9D25CF6D18, 0x5A0B9D05CE92E6B68C, 0x5D05CE82E749725B46, 0x5E82E74173A4B82DA3, 0x574173A0B9D25D16D1, 0x53A0B9D05CE92F8B68, 0x59D05CE82E7496C5B4, 0x5CE82E74173A4A62DA, 0x5E74173A0B9D24316D, 0x573A0B9D05CE9318B6, 0x5B9D05CE82E7488C5B, 0x55CE82E74173A5462D, 0x52E74173A0B9D3A316, 0x368C5CB9D05CE82E74, 0x3B462A5CE82E74173A, 0x3DA3112E74173A0B9D, 0x36D18C973A0B9D05CE, 0x3B68C24B9D05CE82E7, 0x35B46525CE82E74173, 0x32DA3692E74173A0B9, 0x316D1F4973A0B9D05C, 0x38B68BA4B9D05CE82E, 0x3C5B41D25CE82E7417, 0x362DA4E92E74173A0B, 0x3316D674973A0B9D05, 0x318B6F3A4B9D05CE82, 0x38C5B39D25CE82E741, 0x3462DDCE92E74173A0, 0x3A316AE74973A0B9D0, 0x3D18B173A4B9D05CE8, 0x173A7462E5CE92E74, 0xB9D7A3152E74973A, 0x5CEFD188973A4B9D, 0x82E73E8C64B9D25CE, 0x4173DF46125CE92E7, 0xA0B9AFA3292E74973, 0xD05C97D1B4973A4B9, 0xE82E0BE8FA4B9D25C, 0x741745F45D25CE92E, 0x3A0BE2FA0E92E7497, 0x9D05B17D274973A4B, 0xCE8298BEB3A4B9D25, 0xE7410C5F79D25CE92, 0x73A0C62F9CE92E749, 0xB9D02317EE74973A4, 0x5CE8518BD73A4B9D2, 0x2E7468C5CB9D25CE9,

Table A.1: Incidence matrices of input designs(cont.)

n	Matrix
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x65C0EA59A4D32DB81D, 0x6AE0752CD06997DC0E, 0x65703A966A34CAEE07, 0x6AB8194B371A657703, 0x6D5C0CA5998D33BB81, 0x6EAE0252CEC699DDC0, 0x6757052965634CEEE0, 0x63AB8694B0B1A67770, 0x61D5C34A5A58D23BB8, 0x60EAE1A52F2C681DDC, 0x607574D29596340EEE, 0x603ABE6948CB1A0777, 0x681D5B34A6658D03BB, 0x6C0EAD9A5132C781DD, 0x6E0752CD2A9963C0EE, 0x6703A966974CB0E077, 0x6B81D4B349A659703B, 0x534CB11F8A8FC49669, 0x51A6588FC547E34B34, 0x58D32C47E2A3F0A59A, 0x5C699223F151F852CD, 0x5634C911F8A8FD2966, 0x5B1A6088FC547E94B3, 0x558D34447E2A3F4A59, 0x52C69E223F151FA52C, 0x59634F111F8A8ED296, 0x5CB1A7888FC546694B, 0x5658D7C447E2A334A5, 0x532C6FE223F1519A52, 0x599633F111F8A8CD29, 0x54CB19F888FC556694, 0x5A6588FC447E2AB34A, 0x5D32C47E223F1459A5, 0x5699623F111F8B2CD2, 0x323F1659A52CD2B81D, 0x311F8F2CD296695C0E, 0x388FC396694B34AE07, 0x3447E1CB34A59B5703, 0x3223F4E59A52CDAB81, 0x3111FA72CD2967D5C0, 0x3888FD396694B2EAE0, 0x3C447E9CB34A587570, 0x3E223B4E59A52C3AB8, 0x3F1119A72CD2961D5C, 0x3F888CD396694A0EAE, 0x3FC44669CB34A40757, 0x37E22334E59A5303AB, 0x33F1159A72CD2981D5, 0x31F88ACD396695C0EA, 0x38FC41669CB34AE075, 0x347E24B34E59A5703A, 0x4B34D1F8B703B9669, 0xA59A28FC7B81DCB34, 0x52CD547E1DC0EE59A, 0x2966AA3F2EE0672CD, 0x94B3151FB77033966, 0x4A598A8FFBB809CB3, 0xA52CC547DDDC14E59, 0xD29662A3CEEE1A72C, 0x694B7151C7770D396, 0x34A5F8A8C3BB869CB, 0x9A52FC5441DDD34E5, 0xCD297E2A00EEF9A72, 0x6694BF1520776CD39, 0xB34A1F8AB03BB669C, 0x59A50FC5781DCB34E, 0x2CD2C7E29C0EE59A7, 0x966923F16E0772CD3,
	0x4FFFFFFFFC00000000, 0x2FFFF80003FFFE0000, 0x1FFFF800000001FFFF, 0x65C0EA59A4D32DB81D, 0x6AE0752CD06997DC0E, 0x65703A966A34CAEE07, 0x6AB8194B371A657703, 0x6D5C0CA5998D33BB81, 0x6EAE0252CEC699DDC0, 0x6757052965634CEEE0, 0x63AB8694B0B1A67770, 0x61D5C34A5A58D23BB8, 0x60EAE1A52F2C681DDC, 0x607574D29596340EEE, 0x603ABE6948CB1A0777, 0x681D5B34A6658D03BB, 0x6C0EAD9A5132C781DD, 0x6E0752CD2A9963C0EE, 0x6703A966974CB0E077, 0x6B81D4B349A659703B, 0x523F11A65AD32CB81D, 0x511F88D32D69975C0E, 0x588FC46996B4CAAE07, 0x5447E634CB5A655703, 0x5223F31A65AD33AB81, 0x5111FD8D32D699D5C0, 0x5888FAC6996B4CEAE0, 0x5C4479634CB5A67570, 0x5E223CB1A65AD23AB8, 0x5F111E58D32D681D5C, 0x5F888B2C6996B40EAE, 0x5FC4419634CB5A0757, 0x57E224CB1A65AD03AB, 0x53F112658D32D781D5, 0x51F88D32C6996BC0EA, 0x58FC4699634CB4E075, 0x547E234CB1A65B703A, 0x34B34D1F888FC46996, 0x3A59A28FC447E234CB, 0x352CD547E223F11A65, 0x32966AA3F111F98D32, 0x394B3151F888FCC699, 0x34A598A8FC447F634C, 0x3A52CC547E223EB1A6, 0x3D29662A3F111E58D3, 0x3694B7151F888F2C69, 0x334A5F8A8FC4479634, 0x39A52FC547E222CB1A, 0x3CD297E2A3F110658D, 0x36694BF151F88932C6, 0x3B34A1F8A8FC449963, 0x359A50FC547E234CB1, 0x32CD2C7E2A3F11A658, 0x3966923F151F88D32C, 0x34CB6E0768FC56996, 0x1A65F703947E2B4CB, 0x8D32BB81EA3F15A65, 0xC6995DC0D51F9AD32, 0x634CEEE04A8FCD699, 0xB1A677700547F6B4C, 0x58D33BB822A3EB5A6, 0x2C699DDC3151E5AD3, 0x96348EEE38A8F2D69, 0xCB1A07773C54796B4, 0x658D03BBBE2A2CB5A, 0x32C681DDFF15065AD, 0x996340EEDF8A932D6, 0x4CB1E0774FC54996B, 0xA658F03B87E2B4CB5, 0xD32C381DE3F15A65A, 0x69965C0ED1F8AD32D,