On the Classificaton of the Binary Self-Dual Codes of Length 40

Iliya Bouyukliev

Mariya Dzhumalieva-Stoeva
Bulgarian Academy of Sciences

Venelin Monev
Veliko Tarnovo University

BULGARIA

Outline

The main problem:

Developing a software for classification of combinatorial objects.

In this case: Binary self-dual codes!

1. Definition and history of the problem.
2. The obtained results.
3. Correctness of the results.
4. List of problems which covers our work.
5. What more is possible to be done?

Definitions

- $\mathbb{F}_q$ - finite field with q elements;
- $\mathbb{F}_q^n$ - n-dimensional vector space over $\mathbb{F}_q$;
- Weight of a vector $x \in \mathbb{F}_q^n$: $\mathrm{wt}(x) = |\{i | x_i \neq 0\}|$;
- Linear code of length n and dimension k - k-dimensional subspace of $\mathbb{F}_q^n$;
- Minimum weight of a linear code C:

$$d(C) = \min\{\mathrm{wt}(x) | x \in C,\ x \neq \mathbf{0}\}$$

- C - a linear $[n,k,d]_q$ code.

C - a binary linear $[n,k,d]$ code

- C - self-orthogonal code if $C \subseteq C^{\perp}$
- C - self-dual code if $C = C^{\perp}$
- Any self-dual code has dimension $k = n/2$
- All codewords in a binary self-orthogonal code have even weights
- Doubly-even code - if $4 \mid \mathrm{wt}(v)\ \forall v \in C$
- Singly-even self-dual code - if $\exists v \in C :$ $\mathrm{wt}(v) \equiv 2 \pmod 4$

Equivalent codes, Aut(C)

- Two binary codes C and C' are equivalent if there is a permutation $\pi \in S_n$: $C' = \pi(C)$
- Automorphism of C is a permutation of the coordinates that preserves C
- All automorphisms of C form a group $Aut(C)$
- Extended Golay code: $Aut(g_{24}) = M_{24}$ - 5-transitive and $|M_{24}| = 2^{10}.3^3.5.7.11.23$
- Extended quadratic-residue [48,24,12] code: $Aut(q_{48}) = PSL(2,47)$ - 2-transitive and $|PSL(2,47)| = 2^5.3.23.47$

History

- 1975, Vera Pless – $n \leq 20$
- 1980-90, Conway, Pless, Sloane – $n \leq 30$
- 2006, Bilous, Van Rees, – $n = 32, 34$
- 2008, Melchor, Gaborit – $n = 36$ (Optimal)
- 2011, Harada, Munemasa – $n = 36$
- 2011, Harada, Munemasa;
  C. Aguilar-Melchor, Ph. Gaborit, Jon-Lark Kim,
  L. Sok, P. Sole – $n = 38$ (Optimal)
- 2011, Betsumiya, Harada, Munemasa – $n = 40$
  (Doubly even)

The number of binary SD codes

n	$\sharp_I$	$\sharp_{II}$	$d_{max,I}$	$\sharp_{max,I}$	$d_{max,II}$	$\sharp_{max,II}$
24	46	9	6	1	8	1
26	103		6	1		
28	261		6	3		
30	731		6	13		
32	3 210	85	8	3	8	5
34	24 147		6	938		
36	519 492		8	41		
38	38 682 183*BB		8	2 744		
40	**8 250 058 081**	94 343	8	10 200 655*BBH	8	16 470

*BBH - Bouyuklieva, Bouyukliev, Harada
*BDM - Bouyukliev, Dzhumalieva-Stoeva, Monev

$n = 40$

d	4	6	8
# codes	4 329 329 746	3 871 829 027	10 217 125
# doubly-even codes	77 873	-	16 470
# weight enumerators*	18 460	199	10
# orders of $Aut(C)$	1 112	94	91

d	4	6	8
$\|Aut\|_s$	4	1	1
$\|Aut\|_l$	1275541328062914232320000	14745600	82575360

Correctness

The number of all binary SD codes of even length n is

$$N(n) = \prod_{i=1}^{n/2-1} (2^i + 1) = \sum_{i=1}^{r(n)} \frac{n!}{\mid Aut(C_i) \mid},$$

$U = \{C_1, C_2, \ldots, C_{r(n)}\}$ - the set of the inequivalent binary SD codes of length n

$$\sum_{C \in U} \frac{n!}{|Aut(C)|} |\{x \in C | wt(x) = d\}| = \binom{n}{d} \prod_{i=1}^{n/2-2} (2^i + 1).$$

Main construction

If C is a binary $[n = 2k > 2, k, d]$ SD code (child code), then C is equivalent to a code with a generator matrix

$$G = \left(\begin{array}{cc|cc} x_1 \ldots x_{k-1} & 00\ldots0 & 1 & 0 \\ \hline & & x_1 & x_1 \\ I_{k-1} & A & \vdots & \vdots \\ & & x_{k-1} & x_{k-1} \end{array}\right)$$

where the matrix $(I_{k-1}|A)$ generates a self-dual code (parent code) of length $n-2$.

$d = 2$

There is one-to-one correspondence between the set of all inequivalent self-dual $[n, n/2]$ codes and the set of all inequivalent self-dual $[n+2, n/2+1, 2]$ codes

$$C \mapsto (00|C) \cup (11|C)$$

$r(n,d)$ - the number of the inequivalent binary $[n, n/2, d]$ self-dual codes

$$\Rightarrow r(n+2, 2) = r(n)$$

$d = 4$

If C is a binary $[n = 2k > 2, k, 4]$ SD code, then C is equivalent to a code with a generator matrix

$$G = \left(\begin{array}{c|cc|cc} 11 & 00\cdots0 & 00\cdots0 & 1 & 1 \\ 01 & 00\cdots0 & v & 0 & 1 \\ \hline 00 & I_{k-2} & A & a^T & a^T \end{array}\right)$$

where the matrix $(I_{k-2}|A)$ generates a self-dual code of length $n-4$.

List of problems which cover our work

1. Isomorph free generation.
2. Canonical form $\rho(C)$, canonization and automorphism group $Aut(C)$.
3. Coordinate (column) and codeword invariants.
4. Finding "Proper set of codewords for canonization"
5. Implementation, check for correctness and parallelization.

Isomorph Free Generation (IFG)

We want to construct all inequivalent $[n,k]$ SD codes starting from all inequivalent $[n-2,k-1]$ SD codes without using an equivalence test.

1. How to construct only inequivalent child-codes of one $[n-2,k-1]$ code?
2. How to construct a child $[n,k]$ SD code only from one parent code $[n-2,k-1]$?

IFG is based on the concept for a canonical map.

Canonical map

- G - finite group
- G acts on a set Ω and defines an equivalence relation:

$$g(a) \cong a; \quad g \in G$$

- $\rho: \ \Omega \mapsto \Omega$ - canonical map

$$b \cong a \Rightarrow \rho(b) \equiv \rho(a) \equiv r_a \in \Omega$$

- r_a - canonical representative of the equivalence class
- $\rho(a)$ - canonical form (labeling) of a

The standard case

If C is a binary $[n,k,d]$ code (child code), then C is equivalent to a code with a generator matrix

$$G = \left(\begin{array}{cc|c} & & x_1 \\ I_k & A & \vdots \\ & & x_k \end{array} \right),$$

where the matrix $(I_k|A)$ generates a code (parent code) of length $n-1$.

Canonical map for codes

C - a linear $[n,k]$ code

- the canonical map is a permutation of the coordinates (since $G \cong S_n$);
- $\rho(C) = \{c_\rho = (c_{\rho(1)}, c_{\rho(2)}, \ldots, c_{\rho(n)}),\ c \in C\}$;
- this permutation is unique up to an automorphism of C;

Canonical map and $Aut(C)$

- $Aut(C)$ defines a set of orbits of the coordinates $O = \{O_{i1}, O_{i2} \ldots O_{il}\}$
- The canonical map of C gives an ordering of the orbits $\rho(O) = (O_1, O_2, \ldots, O_l)$
- A **special** orbit – say O_1 or O_l

Orbits and parent codes

- $Aut(C)$ defines a set of orbits of the coordinates $O = \{O_{i1}, O_{i2} \dots O_{il}\}$
- Two coordinates from the same orbit O_j give equivalent parent codes.
- The (child) code C can be obtained from exactly l (number of orbits) inequivalent parent codes.
- One of these parent codes (Special parent code) corresponds to the **Special orbit**.

Key idea for a canonical augmentation

We want to construct the child codes C which come from the **Special parent code**.

Parent test:

- the child code C passes the parent test iff the last added coordinate c_n is in the **Special orbit**.
- we consider only the child codes which pass the parent test.

Computing canonical form of codes

Specific algorithms

- CodeCan by Thomas Feulner
- Kris Coolsaet

Computing canonical form of codes

Reduction to canonical form of graph:

- NAUTY by Brendan McKay
- TRACES by Adolfo Piperno
- BLISS by Tommi Junttila and Petteri Kaski.
- NISHE by Greg Tener

or $\{0, 1\}$ matrix: Q-EXTENTION (my program)

Computing canonical form of codes

New version of Q-EXTENTION written in C /C++ (not in Pascal/Delphi)

- input - $\{0,1\}$ matrix or colored $\{0,1\}$ matrix A;
- output - $\rho(A)$ - the canonical form of A.

The efficiency depends on:

- the size of the matrix;
- coloring - the number of colors;
- regularity.

Coloring and invariants

- A - a matrix which generates the code C
- $Aut(C)$ acts on the columns of A $(Aut(A) = Aut(C))$
- The invariant of a coordinate (column) for the matrix A is a function f: $f(a) \in \mathbb{Z}$
  - if b and c are in the same orbit then $f(b) = f(c)$
  - for any permutation $\sigma \in S_n$ we have $f(a) = f(\sigma(a))$ for $a \in A$ and $\sigma(a) \in \sigma(A)$

Coloring and invariants

- All columns of A with the same value of f define a set of columns which consists of one or more orbits. We call this set a **pseudoorbit**.
- The values of f give an ordering of the pseudoorbits and a coloring of the columns.
- The column a of the matrix A has color $f(a)$.
- We define a **special** color - say the color corresponding to the largest value of f.
- We set the special orbit to be with the special color.

Coloring and parent test

- If the last column have color different from the special color the parent test gives a negative answer.
- If the color of the last coordinate correspond to a pseudoorbit with size 1 then the parent test gives an exact answer in the coloring's step.
- In both cases we skip canonization.
- The number of codes, considered in our case (SD codes with $n = 40$) is:

d=4) all codes - 20 614 314 107, only for 5 226 244 513 of them, a canonical form is computed;

d>4) all codes - 131 822 097 145, only for 6 563 895 920 of them, a canonical form is computed;

Finding Proper set of codewords

We define the following properties for the set $M(C)$ of codewords of the code C

- $M(C)$ generates the code C;
- $M(C)$ is stable with respect to $Aut(C)$;
- $M(C)$ is close to minimal;
- if $C' \cong C'' : \sigma(C') = C''$ then $\sigma(M(C')) \equiv M(C'')$

Finding Proper set of codewords

We chose list (vector) of invariants $F = (f_1, f_2, \ldots, f_s)$
The algorithm:

1. $M(C)$ is empty
2. generate the set D of all codeword with smallest not considered weight
3. find and order pseoudoorbits $\{O_{i1}, O_{i2}, \ldots, O_{il}\}$ of D by size (in the case of the same size by colors) $(O_1, O_2 \ldots O_l)$
4. for r from 1 to l do
   if $rank(M(C) \bigcup O_r) > rank(M(C))$ then
   $M(C) = M(C) \bigcup O_r$
5. if $rank(M(C) < rank(C)$ goto point 2.

What is done and more...

1. The classification of the SD codes of length 38 using the general construction (BB).
2. The classification of the optimal SD codes of length 40 (BBH).
3. The algorithm for $d = 4$.
4. The classification of all SD codes of length 40 using both algorithms.
5. The classification of the optimal SD codes of length 42 using the optimal [40,20,8] codes.

The algorithm

Procedure Augmentation(A: self-dual code; k: dimension);

{ If the dimension of A is equal to k then

{ $U_k := \{U_k \cup A\}$; PRINT $(A, |Aut(A)|)$; };

If the dimension of A is less than k then

{ find the set $Child(A)$ of all inequivalent children of A;

(using already known $Aut(A)$)

For all codes B from the set $Child(A)$ do the following:

if B passes the parent test then Augmentation(B, k); }

}

Procedure Main;

INPUT: U_r – all NBSDC $[2r, r]$; OUTPUT: U_k – all NBSDC $[2k, k]$;

$U_k :=$ (the empty set);

for all codes $A \in U_r$ do the following:

{ find the automorphism group of A; Augmentation(A, k);}

Advantages of the algorithm

- Construction and test for equivalence in one.
- Possibilities for use of invariants in the search of canonical representative and canonical permutation.
- Easy for parallelization.
- Recursive construction (we can start from the trivial code).