Performance of Extremal Codes

Anton Malevich

Otto-von-Guericke University
Magdeburg, Germany

Algebraic Combinatorics and Applications

Thurnau, 16 April, 2010

joint work with Wolfgang Willems

Outline

1. What codes do perform better?

2. What codes are extremal?

3. How to study performance of extremal codes?

4. Concluding remarks

Introduction

- Linear $[n, k, d]$ code C is used for data transmission

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

  A_i is the number of codewords of C of weight i

- Symbol error probability is p
- Bounded distance decoding is used
- Up to $t \leq \frac{d-1}{2}$ errors are corrected

What do we call “performance”?

Probability of erroneous decoding from the transmitter and receiver points of view:

$$\mathrm{P}_{tr}(C,t,p) = \mathrm{P}\left(Y \in \bigcup_{c \neq c' \in C} B_t(c') \;\middle|\; X = c \right),$$

$$\mathrm{P}_{rv}(C,t,p) = \mathrm{P}\left(X \in C \setminus \{c\} \mid Y \in B_t(c) \right),$$

with the random variables

- X – “the sent codeword”,
- Y – “the received vector”.

What codes perform better?

Theorem (Faldum, Lafuente, Ochoa, Willems, '06)

Let C and C' be $[n, k, d]$ codes with weight enumerators $A(x, y)$ and $A'(x, y)$ respectively. If p is small enough, then the following conditions are equivalent:

(a) $\mathrm{P}_{tr}(C, t, p) \leq \mathrm{P}_{tr}(C', t, p)$,

(b) $\mathrm{P}_{rv}(C, t, p) \leq \mathrm{P}_{rv}(C', t, p)$,

(c) $A(1, y) \preceq A'(1, y)$, *where "$\preceq$" means lexicographical ordering.*

Remark

"$\prec$" means $A_d < A'_d$,
or $A_d = A'_d$ and $A_{d+1} < A'_{d+1}$,
or ...

Self-dual codes

- $C^{\perp} = \{u \mid u \cdot v = 0 \text{ for all } v \in C\}$ is the dual code
- If $C = C^{\perp}$ the code is self-dual ($n = 2k$)
- Two types of self-dual codes:
  Type I (singly-even): all weights are even
  Type II (doubly-even): all weights are a multiple of 4

Theorem (Gleason '70)

Weight enumerator $A(x, y)$ of a self-dual code is a polynomial in two invariants f and g, that are

- *for Type I codes:* $f = x^2 + y^2,$
  $g = x^2 y^2 \left(x^2 - y^2\right)^2,$
- *for Type II codes:* $f = x^8 + 14x^4y^4 + y^8,$
  $g = x^4 y^4 \left(x^4 - y^4\right)^4.$

Self-dual codes

- $C^\perp = \{u \mid u \cdot v = 0 \text{ for all } v \in C\}$ is the dual code
- If $C = C^\perp$ the code is self-dual ($n = 2k$)
- Two types of self-dual codes:
  Type I (singly-even): all weights are even
  Type II (doubly-even): all weights are a multiple of 4

Corollary

- *for Type II codes:* $f = x^8 + 14x^4y^4 + y^8,$
  $g = x^4y^4\left(x^4 - y^4\right)^4.$

Length of a Type II code is a multiple of 8

$$n = 24m + 8i, \quad i = 0, 1 \text{ or } 2$$

Extremal doubly-even codes

Corollary (Mallows, Sloane '73)

$$\textit{for Type I codes} \qquad d \leq 2\left\lfloor \frac{n}{8} \right\rfloor + 2,$$

$$\textit{for Type II codes} \qquad d \leq 4\left\lfloor \frac{n}{24} \right\rfloor + 4.$$

- If "=" codes are called extremal
  Weight enumerator is unique
- Zhang '99: no extremal Type II codes for $n > 3952$
- Extremal Type II codes are known only up to $n = 136$
- The bound for Type I codes is NOT tight

Shadows of self-dual codes

- C is a Type I $[n, n/2, d]$-code
  C_0 is a doubly-even subcode; $C_2 := C \setminus C_0$
- Shadow $S = S(C)$ consists of all u, such that:

$$\begin{array}{ll} u \cdot v = 1 & \text{for all } v \in C_0 \\ u \cdot v = 0 & \text{for all } v \in C_2 \end{array}$$

- S is a non-linear code with weight enumerator $S(x, y)$
- $S(x, y) = A\left(\frac{x+y}{\sqrt{2}}, i\frac{x-y}{\sqrt{2}}\right)$
- If $8 \mid n$ then all weights in S are divisible by 4

Extremal singly-even codes

- C is a Type I $[n, n/2, d]$-code
- MALLOWS, SLOANE '73: $d \leq 2\left\lfloor \frac{n}{8} \right\rfloor + 2$ (not tight)

Theorem (RAINS '98)

$$d \leq 4\left\lfloor \frac{n}{24} \right\rfloor + 4, \quad n \not\equiv 22 \bmod 24,$$
$$d \leq 4\left\lfloor \frac{n}{24} \right\rfloor + 6, \quad n \equiv 22 \bmod 24.$$

If $n = 24m$ Type I codes do not reach the bound

- If $n \equiv 8$ or $16 \bmod 24$, both Type I and Type II extremal codes have the same minimal distance

Comparing self-dual and non self-dual codes

- C is a self-dual extremal code of Type II
- C' is a non self-dual code with the same parameters

0	...	d	$d+1$	$d+2$	$d+3$	$d+4$	$d+5$	...	$\sum$
1	$0\dots0$	A_d	0	0	0	$*$	0	...	2^k
1	$0\dots0$	A'_d	$*$	$*$	$*$	$*$	$*$	...	2^k

- $A'(x,y) \prec A(x,y)$ is conjectured, i.e. C' is expected to perform better than C

Counterexample (Cheng, Sloane '89)

- C and C' are $[32,16,8]$-codes
- $A_d = 620 < 681 = A'_d$
- Conjecture is not correct

Comparing self-dual codes for small lengths

$$n = 24m + 8 \text{ or } 24m + 16$$

n	d	A_d for Type II	A_d for Type I
32	8	620	364
40	8	285	$125 + 16\beta$ ($\beta < 10$, $10 \leq \beta \leq 26$) (two known codes with $A_d = 285$)
56	12	8 190	$\leq$ 4 862
64	12	2 976	$1\,312 + 16\beta$ ($\beta < 104$, $104 \leq \beta \leq 284$)
80	16	97 565	$\leq$ 66 845
104	20	1 136 150	$\leq$ 739 046

Type I codes with unique weight enumerator

- s – minimum weight of the shadow S
- Bachoc, Gaborit '04: $2d + s \leq \frac{n}{2} + 4$
  If "=" the code is s-extremal
  A_d is known for s-extremal codes
- If s is smallest possible
  the code is with minimal shadow

If $n = 24m + 8$:

$s = 4m$	for s-extremal codes
$s = 4$	for codes with minimal shadow

Best extremal codes of Type I

C is a code of Type I with shadow S

s – minimum weight of the shadow

$$A^{(s)}(1, y) = 1 + A_d^{(s)} y^d + A_{d+2}^{(s)} y^{d+2} + \cdots + y^n$$

$$A_d^{(4m)} < A_d^{(s)} \quad \text{for all } 4 \leq s < 4m \qquad \text{(Bouyuklieva)}$$

Moreover, we can express $A_d^{(4)}$ through $A_d^{(4m)}$.

Comparing Type I and Type II extremal codes

$$n = 24m + 8$$

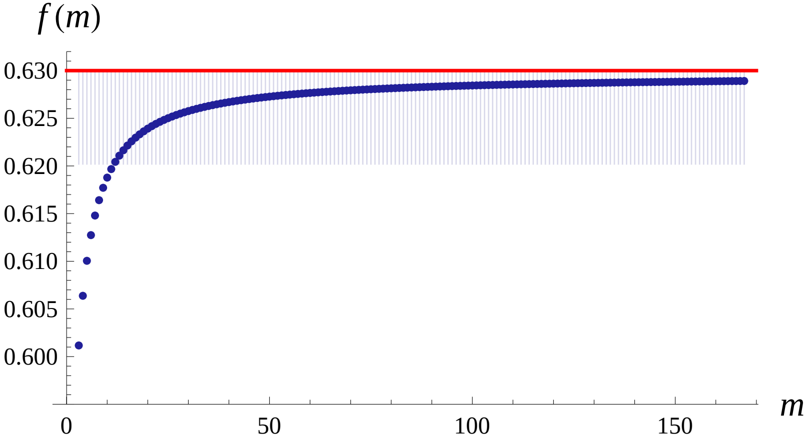
- C – Type II extremal code
- C' – Type I extremal code with min shadow

$$f(m) = \frac{A'_d}{A_d} < 1$$

- C' performs better than C

$\Rightarrow$ s-extremal codes are better than Type II codes

Behaviour of $f(m)$



Concluding remarks

- $n = 24m + 8$
- A lot of different weight enumerators for Type I codes
- $A_d^{(4m)} < \ldots < A_d^{(s_i)} < \ldots < A_d^{(4)} < A_d^{(s_j)} < \ldots < A_d^{(s_k)}$
- For the codes in the tail the problem is not solved

Thank you!