Relations between the Local Weight Distributions of a Linear Block Code, Its Extended Code, and Its Even Weight Subcode

Kenji YASUNAGA and Toru FUJIWARA

Osaka University, Osaka, Japan

ISIT2005, Adelaide, Australia

Local Weight Distribution (LWD) of a Code

- LWD is the weight distribution of codewords that are neighbor to the zero codeword (called *zero neighbors*) in the code.

- LWD (as well as *Weight Distribution*) is useful for an error performance analysis of the code.
  - LWD could give a tighter bound than usual union bound using Weight Distribution.

Motivation

- LWD of (128,50) extended BCH is obtained, but LWD of (127,50) BCH is not obtained.
  - An algorithm for computing LWD using the automorphism group of a code [Yasunaga, Fujiwara, ISITA 04].
  - Ext. BCH have larger automorphism group than BCH.

Our Goal: To devise a method for obtaining LWDs of BCH from LWDs of ext. BCH

Main Result:
Relations between LWDs of C and C_{ex}

If (1) all the weights of codewords in C_{ex} are multiples of fours

(2) C_{ex} is a *transitive invariant code* (ext. BCH, Reed-Muller),

⇓

LWD of C is obtained from LWD of C_{ex}.

– (128,50) ext. BCH code satisfies above two conditions.

Other Result

- Relations between LWDs of C and C_{even}.
  - Similar approach to relations between C and C_{ex}
  - LWD of even weight subcode of (127,50) BCH code

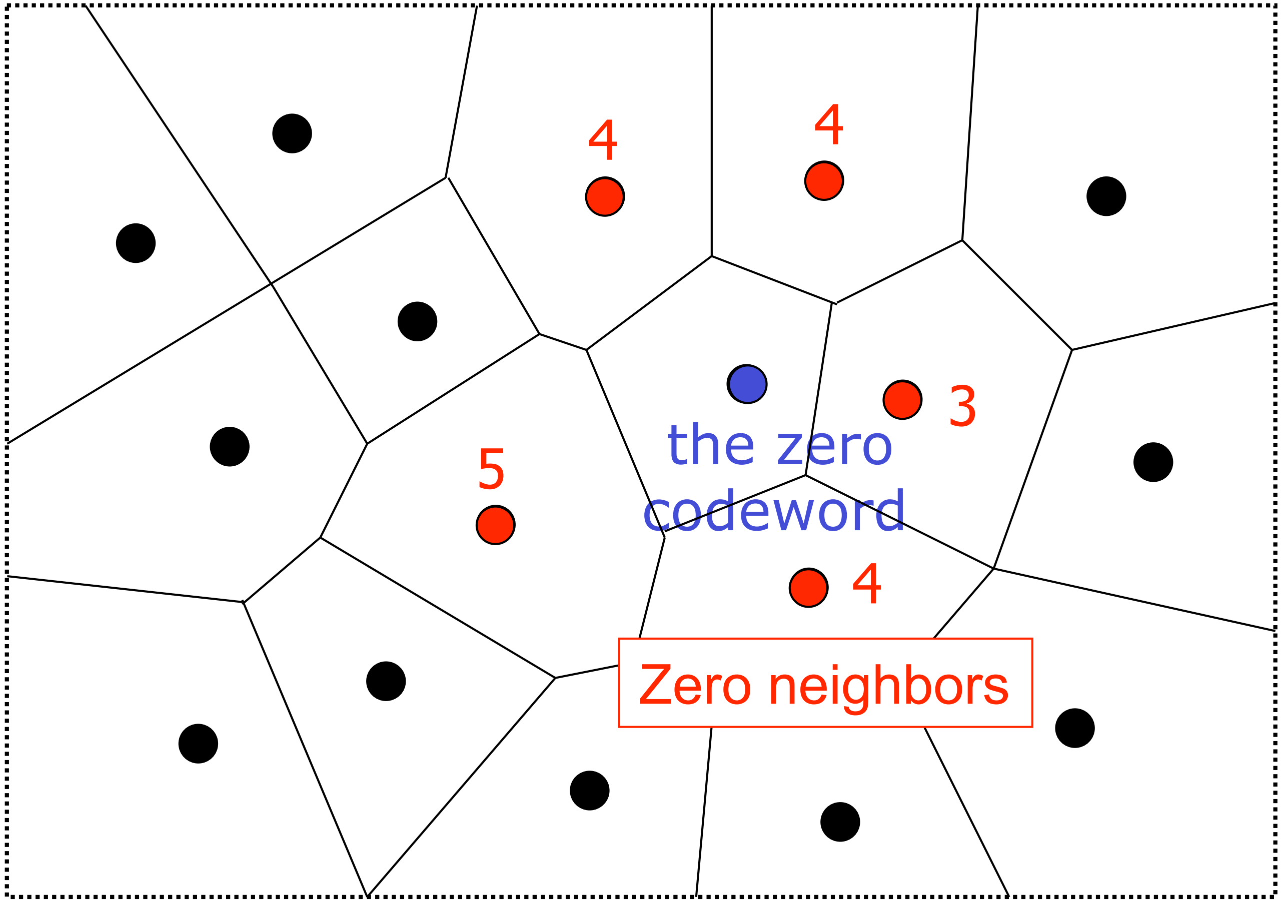
Contents

- Local Weight Distribution
  - Zero neighbor and LWD, Known results
- Details of Our Results

Zero neighbor and LWD

- LWD of C is the weight distribution of *zero neighbors* in C.

Codewords of C on $\boldsymbol{R}^n$



The LWD of C

weight	*the number of zero neighbors*
3	1
4	3
5	1

Known Results for LWD

- Hamming codes, 2nd-order Reed-Muller codes
  - The formulas for the LWDs are derived [Ashikhmin, Barg, IEEE Trans. IT 98].
- Primitive BCH codes
  - (63, k) codes for all k [Mohri, Honda, Morii, IEICE 03]
- Extended primitive BCH codes
  - (128, k) codes for $k \leqq 50$ [Yasunaga, Fujiwara, ISITA 04]
- Reed-Muller codes
  - 3rd-order RM code of length 128 [Yasunaga, Fujiwara, IEICE repo. 04]

Numerical Computation

Contents

- Local Weight Distribution
  - Zero neighbor and LWD, Known results
- Details of Our Results
  - Condition for zero neighbor
  - Zero neighborship between C and C_{ex}
  - Relations between LWDs of C and C_{ex}

Condition for zero neighbor

$\boldsymbol{v}$ is a zero neighbor in C

$\Leftrightarrow$ C does not contain $\boldsymbol{v}_1, \boldsymbol{v}_2 \in C$ such that $\boldsymbol{v} = \boldsymbol{v}_1 + \boldsymbol{v}_2$, $\mathrm{Supp}(\boldsymbol{v}_1) \cap \mathrm{Supp}(\boldsymbol{v}_2) = \phi$.

$$\mathrm{Supp}(\boldsymbol{v}) := \{\, i : v_i \neq 0 \ \text{ for } \ \boldsymbol{v} = (v_1, v_2, \ldots, v_n) \,\}$$

$$\begin{array}{cc}
\boldsymbol{v} & \boxed{1\,|\,1\,|\,0\,|\,1\,|\,0\,|\,1\,|\,1\,|\,1\,|\,0} \\
 & \| \\
\boldsymbol{v}_1 & \boxed{1\,|\,1\,|\,0\,|\,0\,|\,0\,|\,1\,|\,0\,|\,1\,|\,0} \\
 & + \\
\boldsymbol{v}_2 & \boxed{0\,|\,0\,|\,0\,|\,1\,|\,0\,|\,0\,|\,1\,|\,0\,|\,0}
\end{array}$$

If C contains such $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, then $\boldsymbol{v}$ is called *decomposable*.
(i.e. $\boldsymbol{v}$ is decomposable $\Leftrightarrow$ $\boldsymbol{v}$ is not a zero neighbor)

Zero neighborship between C and C_{ex}

$\boldsymbol{v}^{(\mathrm{ex})}$ is a zero neighbor or not in C_{ex}?

$$\Downarrow$$

	(a) $\mathrm{weight}(\boldsymbol{v})$ is odd	(b) $\mathrm{weight}(\boldsymbol{v})$ is even
(1) $\boldsymbol{v}$ is a zero neighbor in C	Zero neighbor	Zero neighbor
(2) $\boldsymbol{v}$ is not a zero neighbor in C	Not zero neighbor	Both cases can occur

(2) $\boldsymbol{v}$ is not a zero neighbor in C
(b) weight($\boldsymbol{v}$) is even

$\boldsymbol{v}$	1	1	0	1	0	1	1	1	0
					=				
$\boldsymbol{v}_1$	1	1	0	0	0	1	0	1	0
					+				
$\boldsymbol{v}_2$	0	0	0	1	0	0	1	0	0

$\boldsymbol{v}$ is decomposable into $\boldsymbol{v}_1+\boldsymbol{v}_2,\ \ \boldsymbol{v}_1,\boldsymbol{v}_2 \in C$

(2) $\boldsymbol{v}$ is not a zero neighbor in C
(b) weight($\boldsymbol{v}$) is even

(i) Both $\boldsymbol{v}_1, \boldsymbol{v}_2$ are even weight

$\boldsymbol{v}^{(ex)}$	1	1	0	1	0	1	1	1	0	0
||										
$\boldsymbol{v}_1^{(ex)}$	1	1	0	0	0	1	0	1	0	0
+										
$\boldsymbol{v}_2^{(ex)}$	0	0	0	1	0	0	1	0	0	0

parity bit

$\boldsymbol{v}^{(ex)}$ is decomposable into $\boldsymbol{v}_1^{(ex)}+\boldsymbol{v}_2^{(ex)}$.
(not a zero neighbor in C_{ex}).

(ii) Both $\boldsymbol{v}_1, \boldsymbol{v}_2$ are odd weight

$\boldsymbol{v}^{(ex)}$	1	1	0	1	0	1	1	1	0	0
||										
$\boldsymbol{v}_1^{(ex)}$	0	1	0	0	0	1	0	1	0	1
+										
$\boldsymbol{v}_2^{(ex)}$	1	0	0	1	0	0	1	0	0	1

parity bit

$\boldsymbol{v}^{(ex)}$ is not decomposable into
$\boldsymbol{v}_1^{(ex)}+\boldsymbol{v}_2^{(ex)}$.

If (ii) occurs for all $\boldsymbol{v}$'s decompositions, $\boldsymbol{v}$ is called *only-odd decomposable*, and $\boldsymbol{v}^{(ex)}$ is a zero neighbor in C_{ex}.

$\boldsymbol{v}$ is an only-odd decomposable codeword.

$\Leftrightarrow$ Zero neighborship between $\boldsymbol{v}$ and $\boldsymbol{v}^{(\mathrm{ex})}$ differs.
($\boldsymbol{v}$ is not zero neighbor, $\boldsymbol{v}^{(\mathrm{ex})}$ is zero neighbor)

$\rightarrow$ Condition for C containing no only-odd decomposable codewords.

Theorem 3 :

If all the weights of codewords in C_{ex} are multiples of four, there is no only-odd decomposable codewords in C.

Examples of above C_{ex}:

- (128, k) extended primitive BCH codes with $k \leqq 57$
- 3rd-order Reed-Muller codes of length $n \geqq 128$

- In the case that C contains no only-odd decomposable codewords,
  - LWD of C_{ex} is obtained from LWD of C
  - To obtain LWD of C from LWD of C_{ex}, we have to know #(zero neighbors of parity bit one).

Lemma 3 :

If C_{ex} is a transitive invariant code of length $n+1$,

$$\#\left(\begin{array}{l}\text{zero neighbors of parity} \\ \text{bit one in } C_{\text{ex}} \text{ with weight } w\end{array}\right) = \frac{w L_w(C_{\text{ex}})}{n+1}$$

Transitive invariant codes:
extended primitive BCH codes, Reed-Muller codes

Proof of Theorem 8.15,
W. W. Peterson and E. J. Weldon, Jr., *Error correcting codes, 2nd Edition*, 1972

Theorem 6 :

If $C_{\rm ex}$ is a transitive invariant code of length $n+1$,

$$L_w(C) = \frac{w+1}{n+1} L_{w+1}(C_{\rm ex}), \qquad \text{for odd } w,$$

$$L_w(C) = \frac{n+1-w}{n+1} L_w(C_{\rm ex}) - N_w(C), \quad \text{for even } w.$$

$N_w(C)$:= #(only-odd decomposable codewords in C with weight w)

If (1) all the weights of codewords in $C_{\rm ex}$ are multiples of four and (2) $C_{\rm ex}$ is a transitive invariant code, LWD of C is determined from LWD of $C_{\rm ex}$.

LWD of (127,50) primitive BCH code

weight	*the number of zero neighbors*
27	40894
28	146050
31	4853051
32	14559153
35	310454802
36	793384494
39	10538703840
40	23185148448
43	199123183160
44	380144258760
47	2154195406104
48	3590325676840
51	13633106229288

weight	*the number of zero neighbors*
52	19925309104344
55	51285782220204
56	65938862854548
59	115927157830260
60	131384112207628
63	158486906385472
64	158486906385472
67	131258388369668
68	115816225032060
71	64917266933304
72	50491207614792
75	15345182164032
76	10499335164864

Conclusion

- LWD is useful for an error performance analysis.

- Relation between LWDs of C and C_{ex}.
  - If (1) all the weights of codewords in C_{ex} are multiples of four and (2) C_{ex} is a transitive invariant code, LWD of C is obtained from LWD of C_{ex}.

- LWDs of (127,50) BCH codes.
  - from LWDs of (128,50) extended BCH codes

Applications of LWD

- Error performance analysis
  - P_e: Error probability of soft decision decoding on AWGN

$$\underbrace{\sum_{i=1}^{n} A_i(C)\, Q\left(\sqrt{i\frac{2E_b}{N_0}}\right)}_{\text{union bound}} \geq \underbrace{\sum_{i=1}^{n} L_i(C)\, Q\left(\sqrt{i\frac{2E_b}{N_0}}\right)}_{\text{a tighter bound}} \geq P_e$$

$A_i(C)$:= #(codewords with weight i in C)
$L_i(C)$:= #(zero neighbors with weight i in C)

$$Q(x) = \int_x^{\infty} (2\pi)^{-1/2} \exp(-z^2/2)dz.$$

Theorem 2 :

For a code C of length n,

$$L_{2i}(C_{\mathrm{ex}}) = L_{2i-1}(C) + L_{2i}(C) + N_{2i}(C),\ 0 \le i \le n/2.$$

$N_w(C)$:= #(only-odd decomposable codewords in C with weight w)

If there is no only-odd decomposable codewords in C, LWD of C_{ex} is determined from LWD of C.

Relations between LWDs of C and C_{ex}

1. $\boldsymbol{v} \in C$ is an *only-odd decomposable codeword*.
   $\Leftrightarrow$ Zero neighborship of $\boldsymbol{v}$ and $\boldsymbol{v}^{(\mathrm{ex})} \in C_{\mathrm{ex}}$ differs
   ($\boldsymbol{v}$ is not a zero neighbor, $\boldsymbol{v}^{(\mathrm{ex})}$ is a zero neighbor).

2. If all the weights of codewords in C_{ex} are multiples of four, there is no only-odd decomposable codewords in C (Theorem 3).

3. If C_{ex} is a transitive invariant code and there is no only-odd decomposable codewords in C, LWD of C is determined from LWD of C_{ex} (Theorem 6).