The Local Weight Distributions of Transitive Invariant Codes and Their Punctured Codes

Kenji Yasunaga Toru Fujiwara

Graduate School of Information Science and Technology, Osaka University, 1-5, Yamadaoka, Suita, Osaka 565-0871, Japan.
E-mail: {k-yasunaga, fujiwara}@ist.osaka-u.ac.jp

Abstract The local weight distribution is the weight distribution of zero neighbors in a code. A zero neighbor is a codeword whose Voronoi region shares a facet with that of the all-zero codeword. The local weight distribution is valuable for an error performance analysis of the code. In this paper, some relations are shown for the local weight distributions of transitive invariant codes and their punctured codes. Extended primitive BCH codes and Reed-Muller codes are transitive invariant codes. *Only-odd decomposable codewords* are key codewords in those relations. A method for finding the only-odd decomposable codewords is also presented.

Key words local weight distribution, transitive invariant code, punctured code, only-odd decomposable codeword

1 Introduction

In a binary linear code, a zero neighbor is a codeword whose Voronoi region shares a facet with that of the all-zero codeword [1]. The local weight distribution [2, 3] (or local distance profile [1, 4, 5, 6, 7]) of a binary linear code is defined as the weight distribution of zero neighbors in the code. Knowledge of the local weight distribution of a code is valuable for the error performance analysis of the code. For example, the local weight distribution gives a tighter upper bound on error probability for soft decision decoding over AWGN channel than the usual union bound [7].

Formulas for local weight distribution are only known for certain classes of codes, Hamming codes and second-order Reed-Muller codes. Although an efficient method to examine zero neighborship of codeword is presented in [1], the computation for obtaining the local weight distribution is a very time-consuming task. As Agrell noted in [1], the automorphism group of the code can help reduce the complexity. Using the automorphism group of cyclic codes, i.e. cyclic permutations, Mohri et al. devised an algorithm generating the representative codewords of cyclic permutations and examining each of them whether it is a zero neighbor or not. By the algorithm, they obtained the local weight distributions of the binary primitive BCH codes of length 63 [5, 6].

When the automorphism group properly contains cyclic permutations, for example the affine group, finding the representative codewords is difficult. We proposed an algorithm for computing the local weight distribution of a code using the automorphism group of the set of the all cosets of a subcode in the code [4]. Using the algorithm, we obtained the local weight distributions of the $(128, k)$ extended primitive BCH codes for $k \leq 50$ and the third-order Reed-Muller code of length 128 [3, 4]. For extended primitive BCH codes, which is closed under the affine group of permutations, our proposed algorithm has considerably smaller complexity than the algorithm in [5, 6].

However, for cyclic codes, the complexity is not reduced. Then, the local weight distributions of the $(127, k)$ primitive BCH codes for $k \geq 36$ were not obtained although those of the corresponding $(128, k)$ extended primitive BCH codes are known.

In this paper, relations between local weight distributions of a binary linear transitive invariant code and its punctured code are given. Extended binary primitive BCH codes and Reed-Muller codes are transitive invariant codes. First, relations between the local weigh distributions of a code and its extended code are presented. After that, a more concrete relation is shown for the case that an extended code is a transitive invariant code and contains no *only-odd decomposable* codewords. *Only-odd decomposable* codewords are key codewords in considering relations between the local weight distribution of a code and its extended code.

By using relations, the local weight distributions of the $(127, k)$ binary primitive BCH codes for $36 \leq k \leq 50$ are obtained from those of the corresponding $(128, k)$ extended primitive BCH codes for $36 \leq k \leq 50$.

Finally, we mention a method for finding only-odd decomposable codewords in a code.

2 Local Weight Distribution

Let C be a binary (n, k) linear code. Define a mapping s from $\{0, 1\}$ to $\mathbf{R}$ as $s(0) = 1$ and $s(1) = -1$. The mapping s is naturally extended to one from $\{0, 1\}^n$ to $\mathbf{R}^n$. A zero neighbor of C is defined [1] as follows:

Definition 1 (Zero neighbor). *For $\boldsymbol{v} \in C$, define $\boldsymbol{m}_0 \in \mathbf{R}^n$ as $\boldsymbol{m}_0 = \frac{1}{2}(s(\mathbf{0}) + s(\boldsymbol{v}))$ where $\mathbf{0} =$*

$(0, 0, \ldots, 0)$. *The codeword* $\boldsymbol{v}$ *is a zero neighbor if and only if*

$$d_E(\boldsymbol{m}_0, s(\boldsymbol{v})) = d_E(\boldsymbol{m}_0, s(\boldsymbol{0})) < d_E(\boldsymbol{m}_0, s(\boldsymbol{v}')), \quad \text{for any } \boldsymbol{v}' \in C \setminus \{\boldsymbol{0}, \boldsymbol{v}\}, \quad (1)$$

where $d_E(\boldsymbol{x}, \boldsymbol{y})$ *is the squared Euclidean distance between* $\boldsymbol{x}$ *and* $\boldsymbol{y}$ *in* $\mathbf{R}^n$.

A zero neighbor is called a minimal codeword in [8]. The following lemma is useful to check whether a given codeword is a zero neighbor or not [1].

Lemma 1. $\boldsymbol{v} \in C$ *is a zero neighbor if and only if there does not exist* $\boldsymbol{v}' \in C \setminus \{\boldsymbol{0}\}$ *such that* $\mathrm{Supp}(\boldsymbol{v}') \subsetneq \mathrm{Supp}(\boldsymbol{v})$. *Note that* $\mathrm{Supp}(\boldsymbol{v})$ *is the set of support of* $\boldsymbol{v}$*, which is the set of positions of nonzero elements in* $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)$.

The local weight distribution is defined as follows:

Definition 2 (Local weight distribution). *Let* $L_w(C)$ *be the number of zero neighbors with weight* w *in* C*. The local weight distribution of* C *is defined as the* $(n+1)$*-tuple* $(L_0(C), L_1(C), \ldots, L_n(C))$.

On the local weight distribution, we have the following lemma [2, 8].

Lemma 2. *Let* $A_w(C)$ *be the number of codewords with weight* w *in* C *and* d *be the minimum distance of* C.

$$L_w(C) = \begin{cases} A_w(C), & w < 2d, \\ 0, & w > n - k + 1. \end{cases} \quad (2)$$

When the (global) weight distribution $(A_0(C), A_1(C), \ldots, A_n(C))$ is known, only $L_w(C)$ with $2d \leq w \leq n-k+1$ need to be computed to obtain the local weight distribution. Generally, the complexity for computing the local weight distribution is larger than that for computing the weight distribution. Therefore, Lemma 2 is useful for obtaining local weight distributions. Moreover, when all the weights w in a code is confined in $w < 2d$ and $w > n - k + 1$, the local weight distribution can be obtained from the weight distribution straightforwardly. For example, the local weight distribution of the (n, k) primitive BCH code of length 63 for $k \leq 18$, of length 127 for $k \leq 29$, and of length 255 for $k \leq 45$ can be obtained from their weight distributions.

3 Relations of Local Weight Distributions

Let C be a binary linear code of length n and C_{punc} be a punctured code of length $n-1$ obtained from C by deleting one bit out of n bits. Puncturing a code is the inverse process to extending a code. First, we describe punctured transitive invariant codes. Next, some relations between local weight distributions of a code and its extended code are presented. After that, we consider local weight distributions of a transitive invariant code and its punctured code.

3.1 Punctured transitive invariant codes

A Transitive invariant code is the code which is invariant under a transitive group of permutations. A group of permutations is said to be transitive if for any two bits in a codeword there exists a permutation that interchanges them [10]. Extended primitive BCH codes and Reed-Muller codes are transitive invariant codes. Now we consider puncturing a transitive invariant code. Puncturing a code C of length n generates a code C_{punc} of length $n-1$ by deleting one bit out of n bits for all codewords in C. C_{punc} varies depending on the deleted bit position. However, the following theorem holds for transitive invariant codes.

Theorem 1. *Punctured transitive invariant codes obtained by deleting any bit position are the same.*

(*Proof*) Let C be a transitive invariant code of length n. Assume that the i-th $(1 \leq i \leq n)$ bits of all codewords in C are deleted for puncturing. Since C is invariant under a transitive group of permutations, there exists a permutation $\pi \in \mathrm{Aut}(C)$ that interchanges the bit positions i and $j\,(1 \leq j \leq n)$, where $\mathrm{Aut}(C)$ is the automorphism group of C. Then, deleting the i-th bits of all codewords in C is equivalent to deleting the j-th bits of all codewords in πC. $C = \pi C$ since $\pi \in \mathrm{Aut}(C)$. Therefore, deleting i-th bits of all codewords in C is equivalent to deleting the j-th bits of all codewords in C for a transitive invariant code C. □

From this theorem, for example, puncturing an extended primitive BCH code by deleting any bit position generates the (unextended) primitive BCH code.

3.2 Codes and their extended codes

We consider general relations between local weight distributions of a code and its extended code. Let C be a binary linear code and C_{ex} be its extended code. For a codeword $\boldsymbol{v} \in C$, let $\mathrm{wt}(\boldsymbol{v})$ be the Hamming weight of $\boldsymbol{v}$ and $\boldsymbol{v}^{(\mathrm{ex})}$ be the corresponding codeword in C_{ex}, that is, $\boldsymbol{v}^{(\mathrm{ex})}$ is obtained from $\boldsymbol{v}$ by adding the over-all parity bit.

We define a *decomposable* codeword.

Definition 3 (Decomposable). $\boldsymbol{v} \in C$ *is called decomposable if* $\boldsymbol{v}$ *can be represented as* $\boldsymbol{v} = \boldsymbol{v}_1 + \boldsymbol{v}_2$ *where* $\boldsymbol{v}_1, \boldsymbol{v}_2 \in C$ *and* $\mathrm{Supp}(\boldsymbol{v}_1) \cap \mathrm{Supp}(\boldsymbol{v}_2) = \emptyset$.

From Lemma 1, $\boldsymbol{v}$ is not a zero neighbor if and only if $\boldsymbol{v}$ is decomposable. For even weight codewords, we introduce *only-odd-decomposable* and *even-decomposable*.

Definition 4. *Let* $\boldsymbol{v} \in C$ *be a decomposable codeword with even* $\mathrm{wt}(\boldsymbol{v})$*. That is,* $\boldsymbol{v}$ *is not a zero neighbor in* C*.* $\boldsymbol{v}$ *is said to be only-odd-decomposable, if all the decomposition of* $\boldsymbol{v}$ *is of the form* $\boldsymbol{v}_1 + \boldsymbol{v}_2$ *with the odd weight codewords* $\boldsymbol{v}_1, \boldsymbol{v}_2 \in C$*. Otherwise,* $\boldsymbol{v}$ *is said to be even-decomposable.*

When $\boldsymbol{v}$ is even-decomposable, there is a decomposition of $\boldsymbol{v}$, $\boldsymbol{v}_1 + \boldsymbol{v}_2$ such that both $\mathrm{wt}(\boldsymbol{v}_1)$ and $\mathrm{wt}(\boldsymbol{v}_2)$ are even. The following corollary can be derived from the definition.

Corollary 1. *An only-odd decomposable codeword has only one decomposition.*

(*Proof*) Let $\boldsymbol{v}$ be an only-odd decomposable codeword having a decomposition $\boldsymbol{v}_1 + \boldsymbol{v}_2$, where $\mathrm{wt}(\boldsymbol{v}_1)$ and $\mathrm{wt}(\boldsymbol{v}_2)$ are odd. Assume that $\boldsymbol{v}$ has another decomposition $\boldsymbol{v}_3 + \boldsymbol{v}_4$ where $\boldsymbol{v}_3 \notin \{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ and both $\mathrm{wt}(\boldsymbol{v}_3)$ and $\mathrm{wt}(\boldsymbol{v}_4)$ are odd. Then, $\mathrm{Supp}(\boldsymbol{v}) \supsetneq \mathrm{Supp}(\boldsymbol{v}_1 + \boldsymbol{v}_3)$. Therfore, $\boldsymbol{v}$ has another decomposition $(\boldsymbol{v}_1 + \boldsymbol{v}_3) + \boldsymbol{v}_5$. Since $\mathrm{wt}(\boldsymbol{v}_1)$ and $\mathrm{wt}(\boldsymbol{v}_3)$ are odd, $\mathrm{wt}(\boldsymbol{v}_1 + \boldsymbol{v}_3)$ is even, that is, $\boldsymbol{v}$ has an even weight decomposition. This contradicts the fact that $\boldsymbol{v}$ is only-odd decomposable codeword. □

The relation between C and C_{ex} with respect to zero neighborship is given in the following theorem, which is also summarized in Table 1.

Theorem 2. 1. For a zero neighbor $\boldsymbol{v}$ in C, $\boldsymbol{v}^{(\mathrm{ex})}$ is a zero neighbor in C_{ex}.

2. For a codeword $\boldsymbol{v}$ which is not a zero neighbor in C, the following (a) and (b) hold.
   (a) When $\mathrm{wt}(\boldsymbol{v})$ is odd, $\boldsymbol{v}^{(\mathrm{ex})}$ is not a zero neighbor in C_{ex}.
   (b) When $\mathrm{wt}(\boldsymbol{v})$ is even, $\boldsymbol{v}^{(\mathrm{ex})}$ is a zero neighbor in C_{ex} if and only if $\boldsymbol{v}$ is only-odd-decomposable in C.

(*Proof*) 1. Suppose that $\boldsymbol{v}^{(\mathrm{ex})}$ is not a zero neighbor in C_{ex}. $\boldsymbol{v}^{(\mathrm{ex})}$ is decomposable into $\boldsymbol{v}_1^{(\mathrm{ex})} + \boldsymbol{v}_2^{(\mathrm{ex})}$, then $\boldsymbol{v}$ is decomposable into $\boldsymbol{v}_1 + \boldsymbol{v}_2$. This contradicts the indecomposability of $\boldsymbol{v}$.

2. Suppose that $\boldsymbol{v}$ is decomposed into $\boldsymbol{v} = \boldsymbol{v}_1 + \boldsymbol{v}_2$. (a) Since $\mathrm{wt}(\boldsymbol{v})$ is odd, the sum of the parity bits in $\boldsymbol{v}_1^{(\mathrm{ex})}$ and $\boldsymbol{v}_2^{(\mathrm{ex})}$ is one. Also, the parity bit in $\boldsymbol{v}^{(\mathrm{ex})}$ is one. Then, $\boldsymbol{v}^{(\mathrm{ex})}$ is decomposable into $\boldsymbol{v}_1^{(\mathrm{ex})} + \boldsymbol{v}_2^{(\mathrm{ex})}$, and $\boldsymbol{v}^{(\mathrm{ex})}$ is not a zero neighbor in C_{ex}. (b) Since $\mathrm{wt}(\boldsymbol{v})$ is even, the parity bit in $\boldsymbol{v}^{(\mathrm{ex})}$ is zero. (If part) Suppose that $\boldsymbol{v}^{(\mathrm{ex})}$ is not a zero neighbor in C_{ex}. Then, there exists a decomposition $\boldsymbol{v}^{(\mathrm{ex})} = \boldsymbol{v}_1^{(\mathrm{ex})} + \boldsymbol{v}_2^{(\mathrm{ex})}$. Because the parity bit in $\boldsymbol{v}^{(\mathrm{ex})}$ is zero, the parity bits in $\boldsymbol{v}_1^{(\mathrm{ex})}$ and $\boldsymbol{v}_2^{(\mathrm{ex})}$ must be zero. This implies that $\boldsymbol{v}$ is even-decomposable into $\boldsymbol{v}_1 + \boldsymbol{v}_2$, and contradicts the assumption that $\boldsymbol{v}$ is only-odd-decomposable. (Only if part) Because $\boldsymbol{v}$ is even-decomposable, there is a decomposition such that the parity bits in both $\boldsymbol{v}_1^{(\mathrm{ex})}$ and $\boldsymbol{v}_2^{(\mathrm{ex})}$ are zero. For such the decomposition, $\boldsymbol{v}^{(\mathrm{ex})}$ is decomposable into $\boldsymbol{v}_1^{(\mathrm{ex})} + \boldsymbol{v}_2^{(\mathrm{ex})}$, and $\boldsymbol{v}^{(\mathrm{ex})}$ is not a zero neighbor in C_{ex}. □

From 2 - (b) of Theorem 2, there may be codewords that are not zero neighbors in C although their extended codewords are zero neighbors in C_{ex}. Such codewords are the only-odd decomposable codewords. For investigating relations of local weight distributions between a code and its extended code, only-odd decomposable codewords are important.

The following theorem is a direct consequence of Theorem 2.

Corollary 2. *For a code C of length n,*

$$L_{2i}(C_{\mathrm{ex}}) = L_{2i-1}(C) + L_{2i}(C) + N_{2i},\ 0 \le i \le n/2, \quad (3)$$

where N_j is the number of only-odd decomposable codewords with weight j in C.

From Corollary 2, if there is no only-odd decomposable codeword in C, the local weight distributions of C_{ex} are obtained from that of C. Next, we give a useful sufficient condition under which no only-odd-decomposable codeword exists.

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

(*Proof*) If $\boldsymbol{v} \in C$ is an only-odd-decomposable codeword and decomposed into $\boldsymbol{v}_1 + \boldsymbol{v}_2$, the weights of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ can be represented as $\mathrm{wt}(\boldsymbol{v}_1) = 4i - 1$ and $\mathrm{wt}(\boldsymbol{v}_2) = 4j - 1$ where i and j are integers. Then, $\mathrm{wt}(\boldsymbol{v}) = \mathrm{wt}(\boldsymbol{v}_1 + \boldsymbol{v}_2) = \mathrm{wt}(\boldsymbol{v}_1) + \mathrm{wt}(\boldsymbol{v}_2) = (4i - 1) + (4j - 1) = 4i + 4j - 2$. This contradicts the fact that $\mathrm{wt}(\boldsymbol{v})$ is a multiple of four. □

For example, all the weights of codewords in the $(128, k)$ extended primitive BCH code with $k \le 57$ are multiples of four. The parameters of Reed-Muller codes with which all the weights of codewords are multiples of four are given by Corollary 13 of Chapter 15 in [9]. The third-order Reed-Muller code of length 128, 256, and 512 are true for the case.

The local weight distribution of C_{ex} for these codes can be obtained from that of C by using Corollary 2. However, the local weight distribution of C can not be obtained from that of C_{ex} because the number of zero neighbors with parity bit one is unknown. If C_{ex} is a class of transitive invariant codes, we could know the number. Next, we describe the local weight distributions of transitive invariant codes and their punctured codes.

3.3 Transitive invariant codes and their punctured codes

We consider relations of local weight distributions between a transitive invariant code C and its punctured code C_{punc}. Generally, the punctured code varies depending on the deleted bit position. However, as described in Section 3.1, punctured transitive invariant codes by deleting any bit position are the same.

For a transitive invariant code C_{ex}, a relation between the (global) weight distributions of C_{ex} and its unextended code C is presented in Theorem 8.15 in [10]. A similar relation holds for local weight distribution. The following lemma can be proved in a similar way as the proof of Theorem 8.15.

Table 1: Zero neighborship of $\boldsymbol{v}$ in a code C and $\boldsymbol{v}^{(\mathrm{ex})}$ in its extended code C_{ex}.

$\boldsymbol{v}$ in C			$\boldsymbol{v}^{(\mathrm{ex})}$ in C_{ex}	
Zero neighborship	Weight	Decomposability	Zero neighborship	Theorem 2
Yes	Odd	Not decomposable	Yes	(1)
	Even			
No	Odd	Decomposable	No	(2) - (a)
	Even	Only-odd-decomposable	Yes	(2) - (b)
	Even	Even-decomposable	No	

Lemma 3. *If C_{ex} is a transitive invariant code of length n, the number of zero neighbors with parity bit one is $\frac{w}{n}L_w(C_{\mathrm{ex}})$.*

(*Proof*) Arrange all zero neighbors with weight w in a column. Next, interchange the j-th column and the last column, which is the parity bit column, for all these codewords with the permutation. All the resulting codewords have weight w and must be the same as the original set of codewords. Thus, the number of ones in the j-th column and that in the last column are the same. Denote this number l_w, which is the same as the number of zero neighbors of weight w with parity bit one. Then, the total ones in the original set of codewords is $n\,l_w$, or $L_w(C_{\mathrm{ex}})$ times the weight w. Thus, $n\,l_w = wL_w(C_{\mathrm{ex}})$, and $l_w = \frac{w}{n}L_w(C_{\mathrm{ex}})$. □

It is clear that there are $\frac{n-w}{n}L_w(C_{\mathrm{ex}})$ zero neighbors with weight w whose parity bit is zero from this lemma. The following theorem is obtained from Theorem 2 and Lemma 3.

Theorem 4. *If C is a transitive invariant code of length n,*

$$L_w(C_{\mathrm{punc}}) = \frac{w+1}{n}L_{w+1}(C), \quad \text{for odd } w, \tag{4}$$

$$L_w(C_{\mathrm{punc}}) = \frac{n-w}{n}L_w(C) - N_w, \tag{5}$$

$$\leq \frac{n-w}{n}L_w(C), \quad \text{for even } w, \tag{6}$$

where N_w is the number of only-odd decomposable codewords with weight w in C_{punc}. If there is no only-odd-decomposable codeword in a transitive invariant code C, the equality of (6) holds. That is, in this case, we have that

$$L_w(C_{\mathrm{punc}}) = \frac{n-w}{n}L_w(C), \quad \text{for even } w. \tag{7}$$

Therefore, for a transitive invariant code C having no only-odd-decomposable codeword in C_{punc}, the local weight distributions of C_{punc} can be obtained from that of C by using (4) and (7) in Theorem 4.

4 Obtained Local Weight Distributions

As discussed in the previous section, the local weight distributions of the $(127, k)$ primitive BCH codes for $k \leq 57$ and the punctured third-order Reed-Muller codes of length 127, 255, and 511 are obtained from those of the corresponding extended codes by using Corollary 2 and Theorem 4. However, the local weight distribution of the $(128, 57)$ extended primitive BCH code and the third-order Reed-Muller code of length 256 and 512 are unknown. The local weight distributions of the $(127, k)$ primitive BCH codes for $k = 36, 43, 50$ are presented in Table 2.

5 Finding Only-Odd Decomposable Codewords

Let C be a binary linear code and C_{ex} be its extended code. As presented in Section 3, if we know the number of only-odd decomposable codewords for each weight, the local weight distribution of C_{ex} can be obtained from that of C by using Corollary 2. In addition, in the case that C_{ex} is a transitive invariant code, the local weight distribution of C can be obtained from that of C_{ex} by using Theorem 4.

We show a method for finding only-odd decomposable codewords for cyclic codes. This method is an expansion of an algorithm for computing the local weight distribution for cyclic codes presented in [6]. The method for finding only-odd decomposable codewords is based on the following Lemma;

Lemma 4. *Let C be a binary code and Π be the automorphism group of C. If $\boldsymbol{v} \in C$ is an only-odd decomposable codeword, a permuted codeword $\pi\boldsymbol{v}$ is an only-odd decomposable codeword for $\pi \in \Pi$.*

(*Proof*) $\pi\boldsymbol{v}$ is a decomposable codeword and $\mathrm{wt}(\pi\boldsymbol{v})$ is even. If $\pi\boldsymbol{v}$ is not an only-odd decomposable codeword, $\pi\boldsymbol{v}$ has a decomposition $\boldsymbol{v}_1 + \boldsymbol{v}_2$ such that $\mathrm{wt}(\boldsymbol{v}_1)$ and $\mathrm{wt}(\boldsymbol{v}_1)$ are even. Since Π is a group, there exists $\boldsymbol{v}'_1$ and $\boldsymbol{v}'_2$ such that $\boldsymbol{v}_1 = \pi\boldsymbol{v}'_1$ and $\boldsymbol{v}_2 = \pi\boldsymbol{v}'_2$. Then, $\pi\boldsymbol{v} = \pi\boldsymbol{v}'_1 + \pi\boldsymbol{v}'_2$, and $\boldsymbol{v} = \boldsymbol{v}'_1 + \boldsymbol{v}'_2$. Thus, $\boldsymbol{v}$ has an even weight decomposition $\boldsymbol{v}'_1 + \boldsymbol{v}'_2$. This contradict the fact that $\boldsymbol{v}$ is an only-odd decomposable codeword. □

Table 2: The local weight distributions of the $(127, k)$ primitive BCH codes for $k = 36, 43,$ and 50.

$(127, 36)$ BCH code

w	L_w
31	2,667
32	8,001
35	4,572
36	11,684
39	640,080
40	1,408,176
43	12,220,956
44	23,330,916
47	132,560,568
48	220,934,280
51	823,921,644
52	1,204,193,172
55	3,157,059,472
56	4,059,076,464
59	7,022,797,740
60	7,959,170,772
63	9,742,066,368
64	9,742,066,368
67	7,959,170,772
68	7,022,797,740
71	4,059,071,892
72	3,157,055,916
75	1,204,193,172
76	823,921,644
79	217,627,200
80	130,576,320
83	23,330,916
84	12,220,956
87	1,408,176
88	640,080

$(127, 43)$ BCH code

w	L_w
31	31,115
32	93,345
35	2,478,024
36	6,332,728
39	82,356,960
40	181,185,312
43	1,554,145,736
44	2,967,005,496
47	16,837,453,752
48	28,062,422,920
51	106,485,735,720
52	155,632,998,360
55	400,716,792,672
56	515,207,304,864
59	905,612,814,120
60	1,026,361,189,336
63	1,238,334,929,472
64	1,238,334,929,472
67	1,026,345,592,720
68	905,599,052,400
71	515,097,101,376
72	400,631,078,848
75	155,191,535,184
76	106,183,681,968
79	26,980,367,680
80	16,188,220,608
83	1,617,588,840
84	847,308,440

$(127, 50)$ BCH code

w	L_w
27	40,894
28	146,050
31	4,853,051
32	14,559,153
35	310,454,802
36	793,384,494
39	10,538,703,840
40	23,185,148,448
43	199,123,183,160
44	380,144,258,760
47	2,154,195,406,104
48	3,590,325,676,840
51	13,633,106,229,288
52	19,925,309,104,344
55	51,285,782,220,204
56	65,938,862,854,548
59	115,927,157,830,260
60	131,384,112,207,628
63	158,486,906,385,472
64	158,486,906,385,472
67	131,258,388,369,668
68	115,816,225,032,060
71	64,917,266,933,304
72	50,491,207,614,792
75	15,345,182,164,032
76	10,499,335,164,864

For $\boldsymbol{v} \in C$, let $C(\boldsymbol{v}) = \{\boldsymbol{u} \mid \boldsymbol{u} \in C, \mathrm{Supp}(\boldsymbol{u}) \subseteq \mathrm{Supp}(\boldsymbol{v})\}$. Then, $\boldsymbol{v}$ is a zero neighbor if and only if the dimension of $C(\boldsymbol{v})$ is one [1]. This property is used for examining codewords for zero neighborship [1, 4, 6]. By modifying a process of examining codewords for zero neighborship, the representatives of only-odd decomposable codewords can be obtained. The following theorem is derived from Lemma 1 and Corollary 1.

Theorem 5. *For $\boldsymbol{v} \in C$ with even weight, let $C(\boldsymbol{v}) = \{\boldsymbol{u} \mid \boldsymbol{u} \in C, \mathrm{Supp}(\boldsymbol{u}) \subseteq \mathrm{Supp}(\boldsymbol{v})\}$. $\boldsymbol{v}$ is a zero neighbor if and only if the dimension of $C(\boldsymbol{v})$ is one. If the dimension of $C(\boldsymbol{v})$ is two, that is, $C(\boldsymbol{v}) = \{\boldsymbol{0}, \boldsymbol{v}, \boldsymbol{u}, \boldsymbol{v}+\boldsymbol{u}\}$ and the weights of $\boldsymbol{u}$ and $\boldsymbol{v}+\boldsymbol{u}$ are odd, then $\boldsymbol{v}$ is an only-odd decomposable codeword.*

Using the above corollary, we can obtain the number of only-odd decomposable codewords for cyclic codes in computing the local weight distribution. Then, by using the algorithm presented in [6] with the above modified property, the local weight distributions of a cyclic code and its extended code can be obtained at the same time.

However, the same way as this could not apply directly to the algorithm proposed in [4]. This algorithm could reduce the complexity more than the algorithm presented in [6] for extended primitive BCH codes or Reed-Muller codes. For example, the $(127, 57)$ primitive BCH code has no only-odd decomposable codewords. On the other hand, the $(127, k)$ primitive BCH codes for $k \geq 64$ probably have only-odd decomposable codewords. If we could obtain the local weight distributions of the $(128, k)$ extended primitive BCH codes for $k \geq 57$, the local weight distributions of the $(127, k)$ codes for $k \geq 64$ could not be obtained although that of the $(127, 57)$ code could be obtained.

6 Conclusion

In this paper, relations between local weight distributions of a transitive invariant code and its punctured code are presented. The local weight distributions of the $(127, k)$ primitive BCH codes for $k = 36, 43, 50$ are shown. The method for finding only-odd decomposable codewords is presented.

References

[1] E. Agrell, "Voronoi regions for binary linear block codes," *IEEE Trans. Inform. Theory*, vol.42, no.1, pp.310–316, Jan. 1996.

[2] E. Agrell, "On the Voronoi Neighbor Ratio for Binary Linear Block Codes," *IEEE Trans. Inform. Theory*, vol.44, no.7, pp.3064–3072, Nov. 1998.

[3] K. Yasunaga and T. Fujiwara, "The local weight distributions of the (128,50) extended binary primitive BCH code and (128,64) Reed-Muller code," *IEICE Technical Report*, IT2004-19, Jul. 2004.

[4] K. Yasunaga and T. Fujiwara, "An algorithm for computing the local distance profile of binary linear codes closed under a group of permutations," *IEICE Technical Report*, IT2003-47, Sept. 2003.

[5] M. Mohri, and M. Morii, "On computing the local distance profile of binary cyclic codes," *Proc. of ISITA2002*, pp.415–418, Oct. 2002.

[6] M. Mohri, Y. Honda, and M. Morii,"A method for computing the local weight distribution of binary cyclic codes," *IEICE Trans.Fundamentals (Japanese Edition)*, vol.J86-A, no.1, pp.60–74, Jan. 2003.

[7] G.D. Forney, Jr., "Geometrically uniform codes," *IEEE Trans. Inform. Theory*, vol.37, no.5, pp.1241–1260, Sept, 1991.

[8] A. Ashikhmin and A. Barg, "Minimal vectors in linear codes," *IEEE Trans. Inform. Theory*, vol.44, no.5, pp.2010–2017, Sept. 1998.

[9] F.J. MacWilliams and N.J.A. Sloane, *The theory of error-correcting codes*, North-Holland, 1977.

[10] W.W. Peterson and E.J. Weldon, Jr., *Error-correcting codes, 2nd Edition*, MIT Press, 1972.