On Binary Linear (16, 8) Code with the Maximum Minimum Distance

By

Shuzo YAJIMA* and Youichi SHIMADA*

(Received December 24, 1985)

Abstract

It is known that the maximum value of minimum distances of binary linear (16, 8) codes is 5. This paper shows that the binary linear (16, 8) code with the minimum distance 5 is unique up to the permutation of bit positions. It is also confirmed that there is no binary linear (16, 8) code with the minimum distance 6, which implies there is no binary linear (15, 8) code with the minimum distance 5, although there is a well-known binary non-linear (15, 8) code of Preparata with the minimum distance 5. The above facts are obtained by finding novel search algorithms in parity-check matrices for binary linear codes.

1. Introduction

With the success of applications of error-correcting codes in digital systems, more precise details of codes of much higher error-correcting capabilities are being discussed recently from the standpoint of actual implementations. Not only single-error-correcting, double-error-detecting codes, but also double-error-correcting codes may be soon used in memories and so on.

There is an extremely well-studied table for minimum bounds for binary linear codes[1]. In contrast with the table making we focus our attention only on the binary linear (16, 8) code.

The binary linear (16, 8) code attracts attention since the information bits are of one byte and the check bits are also of one byte, which can be easily processed by current byte-oriented computers. There is a well-known cyclic code of (17, 9) with the minimum distance 5, or of a double-error-correcting capability, having the self-reciprocal generator polynomial $g(x)=x^8+x^5+x^4+x^3+1$[2]. It is apparent that the shortened (16, 8) code of this cyclic code has the minimum distance 5. There are also the very famous binary BCH code of (15, 7) with the minimum distance 5, and the Preparata non-linear (15, 8) code with the minimum distance 5[3]. Moreover, by adding an overall parity bit to the Preparata non-linear (15, 8) code, a non-

* Department of Information Science.

linear (16, 8) code with the minimum distance 6 is obtained. Hence, it is interesting to know whether there is a binary linear (16, 8) code with the minimum distance 6. It is confirmed in this paper that there is no such code with the minimum distance 6. Moreover, the maximum value of minimum distances of the binary linear (16, 8) code is 5. We show that the code with this minimum distance 5 is unique up to the permutation of bit positions. This fact implies that there exists no binary linear (15, 8) code with the minimum distance 5.

This paper presents the above results together with novel search algorithms in parity-check matrices for binary linear codes for finding these facts.

Chapter 2 describes definitions and several properties which are necessary for making search algorithms. Chapter 3 presents the search algorithms. Computational results are shown in Chapter 4. Concluding remarks are given in Chapter 5.

2. Definitions and Properties

In general, two binary linear codes that are the same except for a permutation of bit positions are called *equivalent codes*, and every binary linear (n, k) code is equivalent to a binary systematic linear (n, k) code whose parity-check matrix H and generator matrix G are respectively given as follows:

$$H=[I_m\ P],\quad G=[P^T\ I_k],\quad m=n-k,$$

where I_r is an r by r identity matrix, P is an m by k matrix and Q^T is the transpose of a matrix Q. Binary linear codes that are equivalent have the same minimum distance, so it is enough to obtain binary linear codes that have minimum distances not less than a given value to search for solutions of the matrix P in H.

In Theorem 2.1, Theorem 2.2, Corollary 2.3 and Corollary 2.4, several well-known properties and proofs for binary linear codes are given for preparations. Every operation with respect to vectors and matrices is over $GF(2)$ unless otherwise specified.

[**Theorem 2.1**][4] For a binary linear code, the minimum distance is equal to the minimum weight of all the codewords except zero. □

(*Proof*)[4] Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be distinct codewords of a binary linear code. Then, $\boldsymbol{x}-\boldsymbol{y}$ is also a codeword, because $H\boldsymbol{x}^T=\mathbf{0}$ and $H\boldsymbol{y}^T=\mathbf{0}$ imply $H(\boldsymbol{x}-\boldsymbol{y})^T=\mathbf{0}$. Hence, if the distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ is minimum, then the distance between $\boldsymbol{x}-\boldsymbol{y}$ and zero is minimum and is also equal to the weight of $\boldsymbol{x}-\boldsymbol{y}$. That is, the minimum distance is equal to the minimum weight of all the codewords except zero. □

[**Theorem 2.2**][4] A binary linear code has the minimum distance d_{min} not less than d if, and only if, every set of $d-1$ column vectors of H is linearly independent. □

(*Proof*)[4] From Theorem 2.1, there exists a codeword $\boldsymbol{x}$ which has the weight w equal to d_{min}. Suppose that $d_{min}<d$, then the left side of $H\boldsymbol{x}^T=\mathbf{0}$ is equal to the

sum of w column vectors of H. Hence, H has a linearly dependent set of w column vectors. That is, H has a linearly dependent set of $d-1$ column vectors.

Conversely, if H has a linearly dependent set of $d-1$ column vectors, then the sum of at most $d-1$ column vectors of the set is equal to zero. This shows that there exists a codeword $\boldsymbol{x}$ which has a weight of $d-1$ or less. Hence, the relation $d_{min}<d$ is derived from Theorem 2.1. □

[**Corollary 2.3**] If $d_{min}\geqq 3$, then all of the column vectors of H are distinct. □

(*Proof*) Suppose that two column vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ of H are the same, then the set $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ is linearly dependent, because $\boldsymbol{v}_1+\boldsymbol{v}_2=\boldsymbol{0}$. Hence, the relation $d_{min}<3$ is derived from Theorem 2.2. That is, if $d_{min}\geqq 3$, then all of the column vectors of H are distinct. □

[**Corollary 2.4**] The weight of every column vector of P in $H=[I_m\ P]$ is not less than $d-1$, where d is the lower bound of d_{min}. □

(*Proof*) Suppose that the weight of a column vector $\boldsymbol{v}$ of P is w less than $d-1$, then we can select w proper column vectors $\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_w$ out of I_m that satisfy $\boldsymbol{e}_1+\boldsymbol{e}_2+\cdots+\boldsymbol{e}_w+\boldsymbol{v}=\boldsymbol{0}$. Hence, the set $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_w, \boldsymbol{v}\}$ of at most $d-1$ column vectors of H is linearly dependent. Therefore, from Theorem 2.2, the weight of every column vector of P is not less than $d-1$. □

[**Definition 2.1**] A square matrix is called *a permutation matrix* if, and only if, every row and column contains just one component equal to 1 and others equal to 0. □

It is well-known that every permutation matrix is an orthogonal matrix. So, if Q is an r by r permutation matrix, then $QQ^T=I_r$.

[**Theorem 2.5**] Two binary linear (n, k) codes which have respectively the parity-check matrices $H=[I_m\ P]$ and $H'=[I_m\ P']$ are equivalent, if P and P' are the same, except for permutations of rows and columns. □

(*Proof*) Let $P'=UPV$, where U and V are respectively m by m and k by k permutation matrices, then we find the relation:

$$H'=[I_m\ P']=[I_m\ UPV]=U[I_m\ P]\begin{bmatrix} U^T & O \\ O & V \end{bmatrix}=UHW,\quad W=\begin{bmatrix} U^T & O \\ O & V \end{bmatrix}.$$

This relation implies

$$H'\boldsymbol{x}^T=\boldsymbol{0} \quad \text{if, and only if,} \quad H(\boldsymbol{x}W^T)^T=\boldsymbol{0},$$

where $\boldsymbol{x}$ and $\boldsymbol{x}W^T$ are the same, except for a permutation of bit positions, because W^T is an n by n permutation matrix. Hence, two binary linear codes which have respectively the parity-check matrices H and H' are equivalent. □

From Definition 2.2 to the end of this chapter, every operation is over $\boldsymbol{R}$ or the real field.

[**Definition 2.2**] A row vector whose components are ordered ascendingly (descendingly, non-descendingly, non-ascendingly) from left to right is called *an ascending* (*descending, non-descending, non-ascending*) *vector.* That is, for $v=[c_1, c_2, \cdots, c_r]$,

if $c_1<c_2<\cdots<c_r$, then v is an ascending vector;
if $c_1>c_2>\cdots>c_r$, then v is a descending vector;
if $c_1\leqq c_2\leqq\cdots\leqq c_r$, then v is a non-descending vector;

and

if $c_1\geqq c_2\geqq\cdots\geqq c_r$, then v is a non-ascending vector. □

[**Definition 2.3**] An m by k matrix P whose components are 0 or 1 is called *a row-non-descending matrix with respect to* $\boldsymbol{w}_R$ if, and only if, $\boldsymbol{w}_R P^T$ is a non-descending vector, where $\boldsymbol{w}_R$ is a given row vector with the dimension k.

Similarly, P is called *a column-non-descending matrix with respect to* $\boldsymbol{w}_C$ if, and only if, $\boldsymbol{w}_C P$ is a non-descending vector, where $\boldsymbol{w}_C$ is a given row vector with the dimension m. □

[**Lemma 2.6**] Let Q be an r by r permutation matrix, then for the two given row vectors v_1 and v_2 with the dimension r, where v_2 is a descending (ascending) vector, the following proposition is satisfied:

$v_1 Q v_2^T$ *is minimum* (*maximum*)
when only $v_1 Q$ *is a non-descending vector.* □

(*Proof*) We shall show the case whereby v_2 is a descending vector. If $v_1 Q$ is not a non-descending vector, then, for some i and j, there exist the i-th component a_i and the j-th component a_j in $v_1 Q$ that satisfy the relations $i<j$ and $a_i>a_j$. Now, let T_{ij} be an r by r permutation matrix which transposes the i-th and the j-th columns, then we find the relation:

$$\begin{aligned} &v_1 Q v_2^T - v_1 Q T_{ij} v_2^T \\ &= v_1 Q (I_r - T_{ij}) v_2^T \\ &= (a_i - a_j) b_i + (a_j - a_i) b_j \\ &= (a_i - a_j)(b_i - b_j) > 0, \end{aligned}$$

where b_i and b_j are respectively the i-th and the j-th components of a descending vector v_2, which satisfy the relation $b_i>b_j$. That is,

$$v_1 Q v_2^T > v_1 Q T_{ij} v_2^T,$$

and QT_{ij} is an r by r permutation matrix. This relation shows that $v_1 Q v_2^T$ is not minimum when $v_1 Q$ is not a non-descending vector. Further, the domain for Q is finite, and $v_1 Q$ which is a non-descending vector is unique. Hence, $v_1 Q v_2^T$ is minimum when only $v_1 Q$ is a non-descending vector.

Similarly, we can prove the case whereby v_2 is an ascending vector. □

[**Theorem 2.7**] Two descending (ascending) vectors $\boldsymbol{w}_R$ with the dimension k and $\boldsymbol{w}_C$ with the dimension m are given. Then, for every m by k matrix P whose components are 0 or 1, there exists at least one m by k matrix P' that is a row-non-descending and column-non-descending matrix with respect to $\boldsymbol{w}_R$ and $\boldsymbol{w}_C$ respectively, where P and P' are the same, except for permutations of rows and columns. □

(*Proof*) In the case whereby $\boldsymbol{w}_R$ and $\boldsymbol{w}_C$ are descending vectors, for an m by m permutation matrix U and a k by k permutation matrix V, the following propositions:

$(\boldsymbol{w}_C UP)V\boldsymbol{w}_R^T$ *is minimum*
when only $(\boldsymbol{w}_C UP)V$ *is a non-descending vector,*

and

$(\boldsymbol{w}_R V^T P^T)U^T \boldsymbol{w}_C^T$ *is minimum*
when only $(\boldsymbol{w}_R V^T P^T)U^T$ *is a non-descending vector,*

are derived from Lemma 2.6. Since $\boldsymbol{w}_C UPV\boldsymbol{w}_R^T = \boldsymbol{w}_R V^T P^T U^T \boldsymbol{w}_C^T$, if we select U and V which minimize $\boldsymbol{w}_C UPV\boldsymbol{w}_R^T$, then both $\boldsymbol{w}_C(UPV)$ and $\boldsymbol{w}_R(UPV)^T$ are non-descending vectors. Hence, let $P'=UPV$, P' is an m by k row-non-descending and column-non-descending matrix with respect to $\boldsymbol{w}_R$ and $\boldsymbol{w}_C$ respectively, and P and P' are the same, except for permutations of rows and columns.

Similarly, we can prove the case whereby $\boldsymbol{w}_R$ and $\boldsymbol{w}_C$ are ascending vectors. □

[**Theorem 2.8**] For every binary linear (n, k) code, there is at least one equivalent code that has a row-non-descending and column-non-descending matrix P with respect to the descending (ascending) vectors $\boldsymbol{w}_R$ with the dimension k and $\boldsymbol{w}_C$ with the dimension m respectively, where P is in $H=[I_m\ P]$. □

(*Proof*) This proposition is derived from Theorem 2.5 and Theorem 2.7. □

The result of Theorem 2.8 is very useful in searching for binary linear codes, because we can extremely reduce the space where we should search for the solutions of P in H.

[**Example 2.1**] The descending vector

$$\boldsymbol{w}_C=[2^{m-1},\ 2^{m-2},\ \cdots,\ 2,\ 1]$$

gives *the non-descending-binary order* (*NDB*) to the column-non-descending order in P. That is, for any binary column vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$,

$\boldsymbol{v}_1$ *is at the left of* $\boldsymbol{v}_2$ *in* P
if, and only if, $b(\boldsymbol{v}_1) \leqq b(\boldsymbol{v}_2)$,

where $b(\boldsymbol{v})$ represents the binary value of a column vector $\boldsymbol{v}$, which is obtained by regarding the components of $\boldsymbol{v}^T$ as a bit sequence of a positional notation of a binary number.

Similarly, the ascending vector

$$\boldsymbol{w}_C=[-2^{m-1},\ -2^{m-2},\ \cdots,\ -2,\ -1]$$

gives *the non-ascending-binary order* (*NAB*) to the column-non-descending order in P. □

[**Example 2.2**] The descending vectors

$$w_C=[2^m+2^{m-1},\ 2^m+2^{m-2},\ \cdots,\ 2^m+2,\ 2^m+1]$$

and

$$w_C=[-2^m+2^{m-1},\ -2^m+2^{m-2},\ \cdots,\ -2^m+2,\ -2^m+1]$$

give respectively *the non-descending-weight-non-descending-binary order* (*NDWNDB*) and *the non-ascending-weight-non-descending-binary order* (*NAWNDB*) to the column-non-descending order in *P*. That is, in the case of *NDWNDB*, for any binary column vectors v_1 and v_2,

v_1 is at the left of v_2 in P

if, and only if,

(i) $w(v_1)<w(v_2)$,

or

(ii) $w(v_1)=w(v_2)$ *and* $b(v_1)\leqq b(v_2)$,

where $w(v)$ represents the weight of a column vector v.

Similarly, the ascending vectors

$$w_C=[2^m-2^{m-1},\ 2^m-2^{m-2},\ \cdots,\ 2^m-2,\ 2^m-1],$$

and

$$w_C=[-2^m-2^{m-1},\ -2^m-2^{m-2},\ \cdots,\ -2^m-2,\ -2^m-1]$$

give respectively *the non-descending-weight-non-ascending-binary order* (*NDWNAB*) and *the non-ascending-weight-non-ascending-binary order* (*NAWNAB*) to the column-non-descending order in *P*. □

[**Example 2.3**] The descending vector

$$w_R=[2^{k-1},\ 2^{k-2},\ \cdots,\ 2,\ 1]$$

gives *NDB* to the row-non-descending order in *P*. That is, for any binary row vectors v_1 and v_2,

v_1 is above v_2 in P

if, and only if, $b(v_1^T)\leqq b(v_2^T)$.

Similarly, the ascending vector

$$w_R=[-2^{k-1},\ -2^{k-2},\ \cdots,\ -2,\ -1]$$

gives *NAB* to the row-non-descending order in *P*. □

The orders in Example 2.3 have a good property that is shown in the next theorem.

[**Theorem 2.9**] The descending vector

$$w_R=[2^{k-1},\ 2^{k-2},\ \cdots,\ 2,\ 1]$$

satisfies the following proposition:

'*P is a row-non-descending matrix with respect to* w_R'

implies

'*P_1 is a row-non-descending matrix with respect to* w_{R1}', where

$$\boldsymbol{w}_R=[\boldsymbol{w}_{R1}\ \boldsymbol{w}_{R2}],\ P=[P_1\ P_2],$$

and $\boldsymbol{w}_{R1}$ has a dimension equal to the number of columns of P_1.

Similarly, the ascending vector

$$\boldsymbol{w}_R=[-2^{k-1},\ -2^{k-2},\ \cdots,\ -2,\ -1]$$

satisfies also the above proposition. □

(*Proof*) We shall show the case whereby $\boldsymbol{w}_R$ is the descending vector. Let $\boldsymbol{w}_{R2}$ have the dimension r. Then, in the following relation:

$$\boldsymbol{w}_R P^T=[\boldsymbol{w}_{R1}\ \boldsymbol{w}_{R2}]\begin{bmatrix}P_1^T\\ P_2^T\end{bmatrix}=\boldsymbol{w}_{R1}P_1^T+\boldsymbol{w}_{R2}P_2^T,$$

$\boldsymbol{w}_{R2}P_2^T$ does not affect the order of the components of $\boldsymbol{w}_R P^T$, because all the components of $\boldsymbol{w}_{R1}P_1^T$ are multiples of 2^r, and because all the components of $\boldsymbol{w}_{R2}P_2^T$ are less than 2^r. Hence, the above proposition is satisfied.

Similarly, we can prove the case whereby $\boldsymbol{w}_R$ is the ascending vector. □

In searching for binary linear codes, we can construct the matrix P by adding successively new binary column vectors at the right of the established columns of P. This property is also very useful in making search algorithms.

3. Search Algorithms

In this chapter, we give two search algorithms. One is an algorithm to search for binary linear $(n,\ k)$ codes that have minimum distances not less than a given value d, which we call $BL(n,\ k,\ d)$ codes. The other is to prove the equivalence of all $BL(n,\ k,\ d)$ codes.

The properties described in the preceding chapter show that it is enough to obtain $BL(n,\ k,\ d)$ codes to search for solutions of P in a parity-check matrix $H=[I_m\ P]$, for which the following conditions are satisfied:

(1) *every set of* $d-1$ *column vectors of* H *is linearly independent,*

and

(2) P *is a row-non-descending and column-non-descending matrix.*

To begin with, we give a criterion for whether a new vector which is added at the right of the established columns of P satisfies the condition (1) or not. Hereafter, the word '*vector*' means a '*binary column vector with the dimension* m'.

[**Definition 3.1**] Let $P=[\boldsymbol{p}_1\ \boldsymbol{p}_2\ \cdots\ \boldsymbol{p}_k]$, where $\boldsymbol{p}_1,\ \boldsymbol{p}_2,\ \cdots,\ \boldsymbol{p}_k$ are vectors. Then the matrix

$$[I_m\ \boldsymbol{p}_1\ \boldsymbol{p}_2\ \cdots\ \boldsymbol{p}_i]$$

is called H_i, where $i=0,\ 1,\ \cdots,\ k$. Especially, $H_0=I_m$ and $H_k=H$. □

[**Definition 3.2**] For a vector $\boldsymbol{v}$ and H_i, *the degree of dependency* $dep(\boldsymbol{v},\ H_i)$ is defined as follows:

$$\mathrm{dep}(\boldsymbol{v},\ H_i)=\min\{r\,|\,\boldsymbol{v}=\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_r,$$
$$\text{and } \boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_r \text{ are vectors of } H_i\}.\ \square$$

〔**Theorem 3.1**〕 Suppose that every set of $d-1$ vectors of H_{i-1} is linearly independent. Then, for $H_i=[H_{i-1}\ \boldsymbol{p}_i]$, where $i=1,\ 2,\ \cdots,\ k$, the following proposition is satisfied:

every set of $d-1$ *vectors of* H_i *is linearly independent*
if, and only if, $dep(\boldsymbol{p}_i,\ H_{i-1})\geqq d-1$. □

(*Proof*) If $\mathrm{dep}(\boldsymbol{p}_i,\ H_{i-1})=r\leqq d-2$, then there exists a set of r vectors $\boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_r$ of H_{i-1} that satisfies the relation $\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_r=\boldsymbol{p}_i$. Hence, the sum of $r+1$ vectors $\boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_r,\ \boldsymbol{p}_i$ of H_i is equal to zero. That is, there exists a linearly dependent set of $d-1$ vectors of H_i.

Conversely, if H_i has a linearly dependent set of $d-1$ vectors, then there exist in H_i at most $d-1$ vectors $\boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_s$ that satisfy the relation $\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_s=\boldsymbol{0}$. Further, since every set of $d-1$ vectors of H_{i-1} is linearly independent, not all of $\boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_s$ are in H_{i-1}. Hence, $\boldsymbol{p}_i$ belongs to $\{\boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_s\}$. Suppose that $\boldsymbol{p}_i=\boldsymbol{h}_s$, then $\boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_{s-1}$ are in H_{i-1} and the relation $\boldsymbol{p}_i=\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_{s-1}$ is satisfied. This shows that $\mathrm{dep}(\boldsymbol{p}_i,\ H_{i-1})\leqq s-1$ or $\mathrm{dep}(\boldsymbol{p}_i,\ H_{i-1})\leqq d-2$. □

The value of $\mathrm{dep}(\boldsymbol{v},\ H_i)$ can be calculated recursively.

〔**Theorem 3.2**〕 For every vector $\boldsymbol{v}$, the following relations are satisfied:

(a) $\mathrm{dep}(\boldsymbol{v},\ H_0)=w(\boldsymbol{v})$.

(b) for $i=1,\ 2,\ \cdots,\ k$,

$$\mathrm{dep}(\boldsymbol{v},\ H_i)=\min\{\mathrm{dep}(\boldsymbol{v},\ H_{i-1}),\ \mathrm{dep}(\boldsymbol{v}+\boldsymbol{p}_i,\ H_{i-1})+1\},$$
where $H_i=[H_{i-1}\ \boldsymbol{p}_i]$. □

(*Proof*) Since $H_0=I_m$, H_0 consists of all the unit vectors, and $\boldsymbol{v}$ is equal to the sum of the $w(\boldsymbol{v})$ unit vectors. Hence, the above relation (a) is satisfied.

Suppose that

$$\boldsymbol{S}=\{r\,|\,\boldsymbol{v}=\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_r,$$
$$\text{and } \boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_r \text{ are vectors of } H_i\},$$
$$\boldsymbol{S}_1=\{r\,|\,\boldsymbol{v}=\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_r,$$
$$\text{and } \boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_r \text{ are vectors of } H_{i-1}\},$$

and

$$\boldsymbol{S}_2=\{r\,|\,\boldsymbol{v}=\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_{r-1}+\boldsymbol{p}_i,$$
$$\text{and } \boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_{r-1} \text{ are vectors of } H_{i-1}\}.$$

These sets satisfy the relation $\boldsymbol{S}=\boldsymbol{S}_1\cup\boldsymbol{S}_2$. Then, the relation (b) is derived as follows:

$$\mathrm{dep}(\boldsymbol{v},\ H_i)=\min(\boldsymbol{S})$$
$$=\min\{\min(\boldsymbol{S}_1),\ \min(\boldsymbol{S}_2)\},$$

where

$$\min(S_1)=\mathrm{dep}(\boldsymbol{v},\ H_{i-1}),$$

and

$$\begin{aligned}\min(S_2)&=\min\{r\,|\,\boldsymbol{v}+\boldsymbol{p}_i=\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_{r-1},\\&\qquad\text{and } \boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_{r-1} \text{ are vectors of } H_{i-1}\}\\&=\min\{r-1\,|\,\boldsymbol{v}+\boldsymbol{p}_i=\boldsymbol{h}_1+\boldsymbol{h}_2+\cdots+\boldsymbol{h}_{r-1},\\&\qquad\text{and } \boldsymbol{h}_1,\ \boldsymbol{h}_2,\ \cdots,\ \boldsymbol{h}_{r-1} \text{ are vectors of } H_{i-1}\}+1\\&=\mathrm{dep}(\boldsymbol{v}+\boldsymbol{p}_i,\ H_{i-1})+1.\ \square\end{aligned}$$

Concerning the condition (2), we adopt *NDB* (*NAB*) as the row-non-descending order in P in order to use the property described in Theorem 2.9. Further, we adopt a proper order as the column-non-descending order, for example *NDB*, *NDWNDB*, and *NAWNDB* (*NAB*, *NDWNAB*, and *NAWNAB*). We call these paired orders *NDB-NDB*, *NDB-NDWNDB*, and *NDB-NAWNDB* (*NAB-NAB*, *NAB-NDWNAB*, and *NAB-NAWNAB*).

Now, we give an algorithm to search for the solutions of P.

[**Algorithm A**] This is an algorithm to search for the solutions of P to find *BL*(n, k, d) codes.

Input : the values of n, k, and d.
Output: the solutions of P.

This algorithm is constructed with a main routine and a subroutine. The order in the columns of P corresponds to the order in the columns of a matrix M in this algorithm.

```
*Main routine MAIN:
begin
    Read the values of $n$, $k$, and $d$;
    Prepare an area for $P$;
    Make a column-non-descending matrix $M$ consisting of all the vectors that have
        weights not less than $d-1$;
    For every vector $\boldsymbol{v}$,
        $DEP(\boldsymbol{v},\ 0)=w(\boldsymbol{v})$;
    Let a pointer $A(0)$ point the first vector of $M$;
    Call the subroutine $SEARCH(1)$
end.

*Subroutine SEARCH($i$):
begin
    do while a pointer $A(i)$ is pointing to a vector of $M$
        begin
            Let $\boldsymbol{u}$ be the vector pointed to by $A(i)$;
```

```
      if $DEP(\mathbf{u}, i-1) \geqq d-1$ then
        begin
          Put $\mathbf{u}$ on the $i$-th column of $P$;
          if rows of the matrix of the left $i$ columns of $P$ are arranged in
            NDB (NAB) then
            if $i<k$ then
              begin
                For every vector $\mathbf{v}$,
                  $DEP(\mathbf{v}, i) = min\{DEP(\mathbf{v}, i-1),$
                    $DEP(\mathbf{v}+\mathbf{u}, i-1)+1\}$;
                $A(i+1) = A(i)$;
                Call the subroutine $SEARCH(i+1)$ recursively
              end
              else
                Write $P$ as a solution (*)
        end;
      Let $A(i)$ point to the next of the vector to which $A(i)$ is pointing currently
  end
end. □
```

The next theorem gives a method to prove the equivalence of all *BL*(*n*, *k*, *d*) codes.

[**Theorem 3.3**] For *BL*(*n*, *k*, *d*) codes, if the following propositions (a) and (b) are satisfied, then all of those codes are equivalent.

(a) Every *BL*(*n*, *k*, *d*) code has a generator matrix consisting of k codewords with the weight d which can be transformed into the form [Q I_k] by a permutation of columns, where Q is a k by m matrix.

(b) All solutions of P that consist of column vectors with the weight $d-1$ are the same, except for permutations of rows and columns. □

(*Proof*) Permutation of columns of a generator matrix corresponds to transforming a code into an equivalent code, and does not vary the weights of its row vectors. Further, the code which has a generator matrix [Q I_k] consisting of k codewords with the weight d has a parity-check matrix [I_m Q^T], where every column vector of Q^T has the weight $d-1$. Hence, from Theorem 2.5, if the above propositions (a) and (b) are satisfied, then all *BL*(*n*, *k*, *d*) codes are equivalent. □

Note that we use Theorem 3.3 to prove the equivalence of all *BL*(16, 8, 5) codes, because the condition '(a) and (b)' in Theorem 3.3 is *not necessary but sufficient*. For example, all *BL*(7, 4, 3) codes are equivalent but the condition is not satisfied.

Now, we give an algorithm to prove the equivalence of all $BL(n, k, d)$ codes.
[**Algorithm B**] This is an algorithm to prove the equivalence of all $BL(n, k, d)$ codes.

Input : the values of n, k, and d, where $d \geqq 3$.
Output: *YES* or *NO*.

If this program responds "*YES*", then the propositions in Theorem 3.3 are satisfied.

This algorithm is composed of the main routine *MAIN* and the subroutine *SEARCH(i)* in Algorithm A and the following subroutine *TEST*, where the following statement:

Write "YES"

is inserted at the last of *MAIN*, and the statement (*) in *SEARCH(i)* is replaced by the following statement:

Call the subroutine TEST.

```
*Subroutine TEST:
begin
    if every column of $P$ has the weight $d-1$ then
        if this step is executed for the first time then
                Keep the set of column vectors of $P$ in an area $PS$ as a model of
                    solutions of $P$
            else begin
                if the set of column vectors of no permutation of $P$ of rows is the same
                    as $PS$ then (**)
                    begin
                        Write "NO";
                        Exit this program
                    end
            end
        else begin
            Generate a table $T$ consisting of codewords with the weight $d$ from the
                generator matrix $G=[P^T\ I_k]$;
            If possible, select $k$ codewords with the weight $d$ from $T$ that construct a
                generator matrix which can be transformed into the form $[Q\ I_k]$ by
                a permutation of columns, where $Q$ is a $k$ by $m$ matrix;
            if the above statement is not executable then
                begin
                    Write "NO";
                    Exit this program
                end
        end
end. □
```

From Corollary 2.3, all of the column vectors of P are distinct when $d \geqq 3$. Hence, the comparison of the sets (⁎⁎) in *TEST* is acceptable.

4. Computational Results

For $BL(16, 8, 6)$ codes, we have no solution on Algorithm A.

For $BL(16, 8, 5)$ codes, on Algorithm A we have 4207 solutions in adopting *NDB-NAWNDB*, and 8468 solutions in adopting *NAB-NAWNAB*. However, we have many more solutions in adopting other orders.

By means of Algorithm B, we can prove that all $BL(16, 8, 5)$ codes are equivalent. In adopting *NDB-NAWNDB* or *NAB-NAWNAB*, we have only 16 solutions for P that consist of column vectors with the weight 4, for which we should calculate all permutations of rows.

The programs based on Algorithm A and Algorithm B are written in about 60 and 180 statements of *PL/I* respectively, and run on HITAC 240H or about 2.3 MIPS machine of the department of information science of Kyoto University for about 10 minutes and 20 minutes respectively in adopting *NDB-NAWNDB* for $BL(16, 8, 5)$ codes.

5. Concluding Remarks

The maximum value of minimum distances of binary linear (16, 8) codes is 5, and the binary linear (16, 8) code with the minimum distance 5 is unique up to the permutation of bit positions. A shortened cyclic (16, 8) code having the generator polynomial $g(x) = x^8 + x^5 + x^4 + x^3 + 1$ is one representation for it, whereas another primitive polynomial $g'(x) = x^8 + x^7 + x^6 + x^4 + x^2 + x + 1$, both of which are the factors of $x^{17} - 1$, generates the equivalent (16, 8) code.

From the discussions in the paper the following interesting problems are immediately open.

(1) To list equivalent codes with the maximum minimum distance up to the modest code length.

(2) To clarify what codes have a unique equivalence class.

(3) To find the conditions of such uniqueness.

Acknowledgement

This paper has evolved from the homework of a master course lecture on coding theory by Yajima, who asked the students to write search programs for this binary linear (16, 8) code, where Shimada succeeded in finding the facts in the paper by those search programs. We are indebted to Professor Yasuo Sugiyama of Setsunan University for providing reference 1). We want to extend our hearty thanks to the course students for various discussions which contributed variously in completing this paper. Lastly, but not least, our gratitude is to Professor Shuji Doshita for his support of this study.

References

1) H. J. Helgert and R. D. Stinaff; "Minimum-Distance Bounds for Binary Linear Codes", IEEE Transactions on Information Theory, vol. **IT-19**, No. 3, pp. 344-356 (1973).

2) E. R. Berlekamp; "Algebraic Coding Theory", McGraw-Hill Book Company, pp. 139-141 (1968).

3) F. P. Preparata; "A Class of Optimum Non-linear Double-Error-Correcting Codes", Information and Control **13**, pp. 378-400 (1968).

4) R. E. Blahut; "Theory and Practice of Error Control Codes", Addison-Wesley Publishing Company, pp. 47-50 (1983).