Codes for Improving Lifetime, Speed, and Reliability of Flash Memories

March 2019

Graduate School of Engineering

Gifu University

Akira Yamawaki

Abstract

Codes for flash memory applications are investigated. In this dissertation, we consider three codes for flash memories: index-less indexed flash code with inversion cells (I-ILIFC), parallel random I/O (P-RIO) code, and permutation code.

Index-less indexed flash code (ILIFC) is a coding scheme to prolong the lifetime of flash memory. I-ILIFC improves the average performance of index-less indexed flash code (ILIFC). The ILIFC is originally designed in terms of the worst-case performance. We analyze the worst-case performance of I-ILIFC. We derive an upper bound on the worst-case number of writes by ILIFC and lower bounds on the number of writes by I-ILIFC. The results show that the worst-case performance of I-ILIFC is better than that of ILIFC when the code length is sufficiently large.

Random I/O (RIO) code is a coding scheme that minimizes the number of read thresholds required to read one of pages in multilevel flash memory. P-RIO code is RIO code in which all pages are encoded in parallel. We construct P-RIO codes using coset coding with Hamming codes. When $(7, 4)$ Hamming code and $(15, 11)$ Hamming code are used, we provide P-RIO codes with 4 pages and 8 pages, respectively. Our P-RIO codes have parameters for which RIO codes do not exist.

Rank modulation is a coding scheme to correct errors in flash memory. In rank modulation, information is stored in the permutation induced by the charge levels of the cells. Permutation codes that can correct errors are investigated for applications in the rank modulation. The generalized Cayley distance is one of distances that are considered in permutation codes. We derive a tighter upper bound on the generalized Cayley distance using the block permutation distance that is simple to compute. We employ our upper bound to derive an upper bound on the optimal code rate with the generalized Cayley distance.

Contents

Chapter 1

Introduction

Flash memory is the widely used type of non-volatile memory at present, and flash devices are employed in universal serial bus (USB) memory technology, solid state drives (SSD), mobile applications, and so on. A flash memory consists of an array of cells in which information bits are stored in the form of an amount of charge. In conventional flash memory, one cell stores a single bit, and the information is read using a single read threshold. Recently, multilevel flash memory technology has been introduced. In multilevel flash memory, each cell can represent one of more than two levels, and these levels are distinguished by multiple read thresholds.

One of the most notable characteristics of flash memory is the asymmetry of charging and discharging operations. That is, the charge level of the cell can be increased in a cell-by-cell manner but cannot be decreased in this manner. Instead, discharging is achieved by way of a special operation known as block erasure, which discharges the cells in a long block simultaneously. The disadvantage of the block erasure operation is that it partially destroys cells in the flash memory and thus increases the error probability. This necessitates the use of an error correcting code. However, the cells invariably become highly unreliable after block erasure is executed a certain number of times.

1.1 Flash Codes

Designing coding schemes that are useful for prolonging the lifetime of flash memory has been important. Such coding schemes, known as flash codes, were given in [1, 2]. The objective of flash codes is to write the information into flash memory without discharging the cells, that is, block erasure. In flash codes, a write is supposed to change one of the information bits only by transitions from a lower level to a higher level in the cells. The performance of flash codes is characterized in terms of a write deficiency, defined as the difference between the total number of available level transitions and the number of such writes that can be accommodated.

Rank modulation codes are very related to flash codes but the objective of rank modulation codes, which is to improve the reliability of flash memories, is slightly different from that of the flash codes [3]. In other works, efficient data movement in flash memories was studied [4] and algorithms to achieve high write speed were presented [5].

Index-less indexed flash code (ILIFC) is one of flash codes that approaches the best known lower bound on the write deficiency [6]. In the ILIFC, both the value of one of the information bits and the index of the bit are stored in one slice. Several modified ILIFC schemes capable of improving the performance of the ILIFC have since been proposed [7, 8, 9].

ILIFC with inversion cells (I-ILIFC), which is one of such modified ILIFC schemes, was proposed to increase the number of writes between two block erasures [7]. Here, a write means that the current information bits are changed to new information bits. That is, multiple bits among the information bits can be changed by a single write. This write is more practical than the conventional write in flash codes. Computer simulation was used to show that the I-ILIFC improves the average performance of the ILIFC in many cases [7, 10].

In this dissertation, we focus on the analysis of the worst-case performance of the I-ILIFC.

1.2 Random I/O Codes

In multilevel flash memory, in general, multiple read thresholds are required to read a single logical page. For example, we consider the triple-level cell (TLC) that is currently being utilized in multilevel flash memory. A TLC can represent one of eight levels from 0 to 7, that is, it stores three bits. Each level corresponds to a vertical three-bit sequence, as listed in Table 1.1, and each bit represents one logical page. Then, a group of such cells stores three logical

Table 1.1: Triple-level cell (TLC)

Level	0	1	2	3	4	5	6	7
Page 1	0	0	0	0	1	1	1	1
Page 2	0	0	1	1	1	1	0	0
Page 3	0	1	1	0	0	1	1	0

pages, referred to as pages 1–3. Each logical page is supposed to be retrieved independently. To read page 1, the read threshold between levels 3 and 4 is required. To read page 2, the read threshold between levels 1 and 2 is required. Moreover, the threshold between levels 5 and 6 is also required. Similarly, to read page 3, four thresholds are required. Hence, on average, the number of read thresholds that are required to read one page is $(1+2+4)/3 = 2.33$. The use of a large number of read thresholds degrades the read performance of the flash memory device.

In order to solve this problem, random I/O (RIO) code was proposed by Sharon and Alrod [11]. This is a coding scheme in which one logical page can be read using a single read threshold in multilevel flash memory. Further, Sharon and Alrod showed that the construction of RIO codes is equivalent to the construction of well-studied write-once memory (WOM) codes [11]. WOM code, introduced by Rivest and Shamir, is a coding scheme that permits writing data bits into binary cells several times without decreasing the levels [12].

In WOM codes, the data are stored sequentially and are not all known in advance. Therefore, each encoding depends on the current data and the previous data. However, in RIO codes, the data of all logical pages may be known in advance. Yaakobi and Motwani proposed a family of RIO codes called parallel RIO (P-RIO) code [13]. In P-RIO codes, the encoding of each page is performed in parallel and depends on the data of all logical pages. These researchers demonstrated P-RIO codes with parameters for which WOM codes or RIO codes do not exist. In addition, they proposed an algorithm to construct a P-RIO code via a computer search [13]. However, the complexity of this algorithm increases exponentially with the code length or the number of pages.

In this dissertation, we focus on the construction of P-RIO codes using coset coding technique [14] with Hamming codes.

1.3 Permutation Codes

A coding scheme for flash memory using the rank modulation was proposed [3]. In this scheme, a block of cells stores the information in the permutation induced by the charge levels of the cells. For example, we consider a block consisting of 6 cells. The relative levels of the cells in the blocks are shown in Figure 1.1. The cells in the block are supposed to be

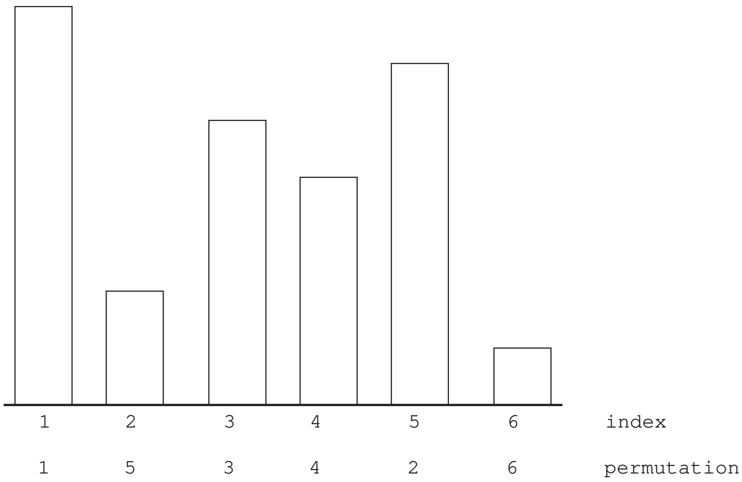

Figure 1.1: Example of the relative levels of cells in a block

indexed as shown in Figure 1.1. In this example, the permutation induced by the levels of the cells is obtained as follows. The charge level of cell 1 is the highest in the block. Thus, the first component of the permutation is 1. The cell of which the level is the second highest is cell 5. Hence, the second component of the permutation is 5. Similarly, the remaining components of the permutation are obtained. As a result, the permutation is $(1, 5, 3, 4, 2, 6)$.

In the rank modulation, uniform offsets do not cause any errors because uniform offsets do not change the relative level of any cell as shown in Figure 1.2.

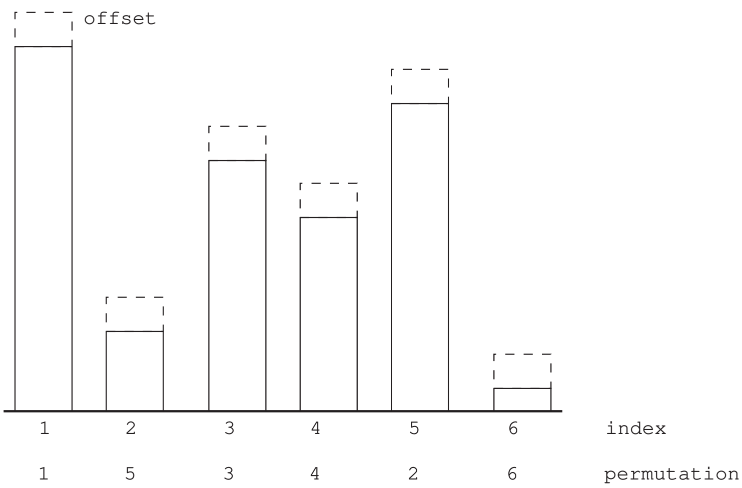

Figure 1.2: The effect of uniform offsets

On the other hand, since high density flash memories are used recently, we must take into account the inter-cell interference. The inter-cell interference is caused by unintended capacitance among adjacent cells. Local offsets, non-uniform offsets can be caused by the inter-cell interference. For example, by neighboring cells, the charge levels of the first, third, and fifth cells may be decreased as shown in Figure 1.3. In this example, the charge level of the third cell is lower than that of the second cell and the charge levels of the first and fifth cells are lower than that of the third cell. Then, by swapping two subsequences $(1, 5)$ and

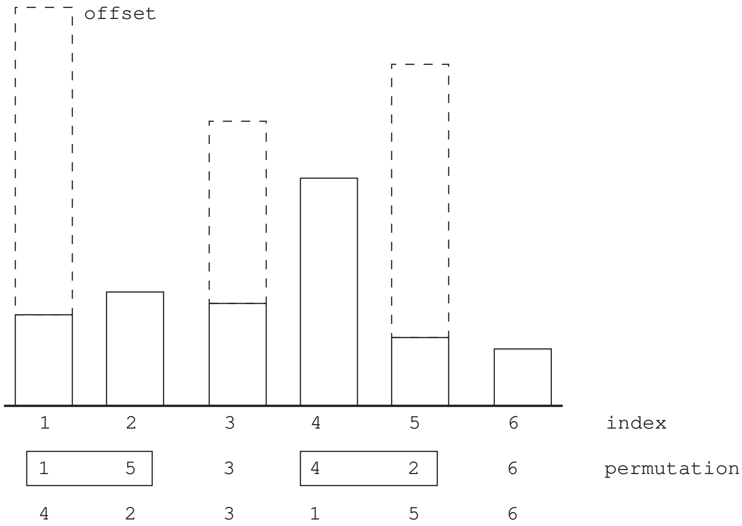

Figure 1.3: The effect of non-uniform offsets

$(4, 2)$, the permutation $(1, 5, 3, 4, 2, 6)$ may be changed into $(4, 2, 3, 1, 5, 6)$, which is an error.

Permutation codes have recently been studied for flash memory applications [15, 16]. In permutation codes, various metrics, such as the Cayley distance, Kendall tau distance, and Ulam distance, have been considered [17, 18]. The generalized Cayley distance, introduced recently by Chee and Vu, is defined as the minimum number of generalized transpositions between two permutations [19]. Generalized transpositions include transpositions and translocations, of which the numbers are captured by the Cayley, Kendall tau, and Ulam distances. Permutation codes with the generalized Cayley distance were studied using a breakpoint analysis by Chee and Vu [19]. However, the rate of these codes is much smaller than the optimal rate when the code length is large. On the other hand, Yang, Schoeny, and Dolecek introduced a new distance, called the block permutation distance, to construct order-optimal codes with the generalized Cayley distance [20]. They derived a relation between these two distances, and showed that the construction of codes with one of the distances can be transformed into the construction with the other.

Christie showed an algorithm to compute the block-interchange distance, which is exactly the generalized Cayley distance [21]. On the other hand, Yang, Schoeny, and Dolecek derived two constants in an inequality that shows that the generalized Cayley and block permutation distances are strongly equivalent on the space of permutations [20].

In this dissertation, we derive a tighter upper bound on the generalized Cayley distance, using the block permutation distance. Furthermore, we employ our upper bound to derive a tighter upper bound on the optimal rate for codes with the generalized Cayley distance when the code length satisfies a condition.

The remaining part of this dissertation is organized as follows. In Chapter 2, we give a preliminary of ILIFC and I-ILIFC. In Chapter 3, the analysis of the worst-case performance of I-ILIFC is described. In Chapter 4, we give a construction of P-RIO codes using coset coding with Hamming codes. In Chapter 5, the derivation of a tighter upper bound on the generalized Cayley distance is described. Finally, in chapter 6, we close this dissertation by remarking a brief conclusion.

Chapter 2

Index-less indexed flash code with inversion cells

This chapter describes index-less indexed flash code (ILIFC) and ILIFC with inversion cells (I-ILIFC).

2.1 Index-less indexed flash code (ILIFC)

In this dissertation, it is assumed that the level of electric charge in a cell of a NAND flash memory is in the range $A_q = \{0, 1, \ldots, q-1\}$. A block of data bits (information bits) of length k is encoded and stored in a block of cells of length n. An ILIFC that satisfies these conditions is denoted by ILIFC(n, k, q).

In the ILIFC, the block of cells of length n is divided into slices consisting of k cells and, therefore, the number of slices in the block is $m = \lfloor n/k \rfloor$. If n is not a multiple of k, the remaining cells in the block are unused. Each slice represents one bit of the k data bits. Since k slices are used to store k data bits, we require $m \geq k$, that is, $n \geq k^2$. For example, when $n = 17$ and $k = 4$, the block of 17 cells is divided into 4 slices and one cell is unused as shown in Figure 2.1. In this example, q is assumed to be 3 and the level of one cell in the range $\{0, 1, 2\}$ is also shown in Figure 2.1. The state of m slices is denoted by

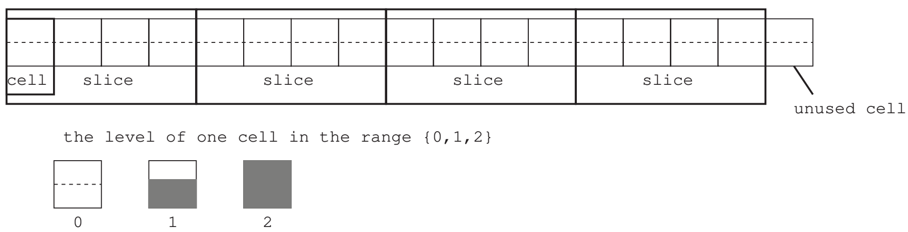

Figure 2.1: Cells and slices in ILIFC$(17, 4, 3)$

$(\boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m)$, where $\boldsymbol{x}_j \in A_q^k$ for $1 \leq j \leq m$. When $n = 17$, $k = 4$ and $q = 3$, an example of the state of slices is shown in Figure 2.2. In this example, the state of 4 slices is denoted by $((2, 2, 2, 2) \mid (2, 1, 0, 0) \mid (0, 0, 1, 0) \mid (0, 0, 0, 0))$. For a slice $\boldsymbol{x} = (x_1, x_2, \ldots, x_k)$, we define $wt(\boldsymbol{x}) = \sum_{i=1}^{k} x_i$ and $bv(\boldsymbol{x}) = wt(\boldsymbol{x}) \bmod 2$. $wt(\boldsymbol{x})$ is termed the weight of the slice $\boldsymbol{x}$. For example, for the second slice $\boldsymbol{x} = (2, 1, 0, 0)$ from the left in the Figure 2.2, $wt(\boldsymbol{x}) = 3$ and $bv(\boldsymbol{x}) = 1$. A slice $\boldsymbol{x} = (x_1, x_2, \ldots, x_k)$ is said to be full if $x_1 = x_2 = \cdots = x_k = q - 1$ and to be empty if $x_1 = x_2 = \cdots = x_k = 0$. The slice is said to be active if it is neither full

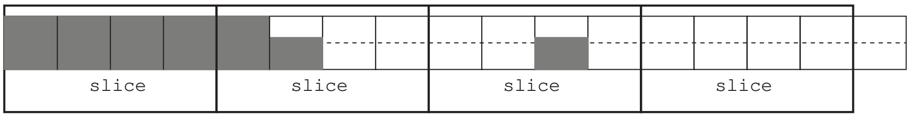

Figure 2.2: The state of slices in ILIFC$(17, 4, 3)$

nor empty. In Figure 2.2, the first slice from the left is full, the second and third slices are active, and the fourth slice is empty.

In the ILIFC, the value of the i-th bit in the k data bits and the index i of the bit are stored in a slice as follows (see [6] for details). In the initial state, it is assumed that all slices are empty and all data bits are 0.

Suppose that the value of the i-th bit is changed. If none of the slices represent the i-th bit, an empty slice is reserved for the bit and then the level of the i-th cell in the slice is changed to 1. In the case that no empty slices exist, block erasure is incurred.

On the other hand, if there is a slice representing the i-th bit, the weight of the slice is increased by 1. In the beginning, the level of the i-th cell in the slice is increased. If the level of the i-th cell is $q-1$, the level of the i'-th cell is increased, where $i' = (i \bmod k) + 1$. Similarly, if the level of the i'-th cell is also $q-1$, the level of the i''-th cell is increased, where $i'' = (i' \bmod k) + 1$. This procedure enables the value of the bit, which is represented by $bv(\boldsymbol{x})$, to be obtained for the active slice $\boldsymbol{x}$. Additionally, the index of the bit is represented by the position of the first updated cell in $\boldsymbol{x}$. This updating procedure is performed until the slice gets full.

For example, let $n = 16$, $k = 4$ and $q = 3$. The initial state of $n/k = 4$ slices is as follows.

$$((0, 0, 0, 0) \mid (0, 0, 0, 0) \mid (0, 0, 0, 0) \mid (0, 0, 0, 0)).$$

Then all data bits are 0, that is, the current data is $(0, 0, 0, 0)$.

Suppose that the data $(0, 0, 0, 0)$ are changed into $(1, 0, 0, 1)$, that is, the first and fourth bits are changed. First, two empty slices are reserved for these two bits. In the slice for the first bit, the level of the first cell is changed to 1. Similarly, in the slice for the fourth bit, the level of the fourth cell is changed to 1. The state of 4 slices is changed as follows.

$$(\underbrace{(1, 0, 0, 0)}_{\text{1st bit}} \mid \underbrace{(0, 0, 0, 1)}_{\text{4th bit}} \mid (0, 0, 0, 0) \mid (0, 0, 0, 0)).$$

Next, suppose that the data $(1, 0, 0, 1)$ are changed into $(1, 1, 1, 0)$, that is, the last three bits are changed. Since the leftmost slice represents the fourth bit, the level of the fourth cell in the slice is increased by 1, that is, changed to 2. Because there are not slices representing the second or three bits, empty slices are reserved for the bits and the level of one cell in each slice is changed to 1 as follows.

$$(\underbrace{(1, 0, 0, 0)}_{\text{1st bit}} \mid \underbrace{(0, 0, 0, 2)}_{\text{4th bit}} \mid \underbrace{(0, 1, 0, 0)}_{\text{2nd bit}} \mid \underbrace{(0, 0, 1, 0)}_{\text{3rd bit}}).$$

Moreover, if the data are changed from $(1, 1, 1, 0)$ to $(1, 1, 1, 1)$, that is, the fourth bit is changed, the i'-th cell in the slice representing the fourth bit is changed to 1, where $i' = (4 \bmod 4) + 1 = 1$, because the level of the fourth cell is $q - 1 = 2$ and cannot be increased. Then the state of slices is as follows.

$$(\underbrace{(1, 0, 0, 0)}_{\text{1st bit}} \mid \underbrace{(1, 0, 0, 2)}_{\text{4th bit}} \mid \underbrace{(0, 1, 0, 0)}_{\text{2nd bit}} \mid \underbrace{(0, 0, 1, 0)}_{\text{3rd bit}}).$$

We consider another example in which $n = 20$, $k = 4$ and $q = 3$. Suppose that the state of 4 slices is as follows.

$$(\underbrace{(2,2,2,1)}_{\text{1st bit}} \mid \underbrace{(2,1,0,2)}_{\text{4th bit}} \mid \underbrace{(0,2,2,2)}_{\text{2nd bit}} \mid \underbrace{(2,0,2,2)}_{\text{3rd bit}} \mid (0,0,0,0)).$$

The data represented by this state are $(1,0,0,1)$. If the data are changed from $(1,0,0,1)$ to $(0,0,0,1)$, the state is changed as follows.

$$(\underbrace{(2,2,2,2)}_{\text{full}} \mid \underbrace{(2,1,0,2)}_{\text{4th bit}} \mid \underbrace{(0,2,2,2)}_{\text{2nd bit}} \mid \underbrace{(2,0,2,2)}_{\text{3rd bit}} \mid (0,0,0,0)).$$

As shown in this example, note that any full slice, such as the leftmost slice in the above state, cannot represent the index. In the ILIFC, the value of a bit without any corresponding slice is considered to be 0. Therefore, for the full slice $\boldsymbol{x}' \in A_q^k$, $wt(\boldsymbol{x}') = k(q-1)$ should be even. Thus, in this dissertation it is assumed that k or $q-1$ is even.

In the above state, if the data are changed from $(0,0,0,1)$ to $(1,1,0,1)$, that is, the first and second bits are changed, the empty slice is reserved for the first bit because there is not a slice representing the first bit. Then the state of slices is changed as follows.

$$(\underbrace{(2,2,2,2)}_{\text{full}} \mid \underbrace{(2,1,0,2)}_{\text{4th bit}} \mid \underbrace{(1,2,2,2)}_{\text{2nd bit}} \mid \underbrace{(2,0,2,2)}_{\text{3rd bit}} \mid \underbrace{(1,0,0,0)}_{\text{1st bit}}).$$

Lastly, suppose that the state of slices is as follows.

$$(\underbrace{(2,2,2,2)}_{\text{full}} \mid \underbrace{(2,1,0,2)}_{\text{4th bit}} \mid \underbrace{(2,2,2,2)}_{\text{full}} \mid \underbrace{(2,0,2,2)}_{\text{3rd bit}} \mid \underbrace{(1,0,0,0)}_{\text{1st bit}}).$$

The data represented by this state are $(1,0,0,1)$. If the data are changed from $(1,0,0,1)$ to $(1,1,0,0)$, none of the slices represent the second bit that is changed and, in this case, no empty slices exist. Therefore, block erasure is incurred.

The state of slices $(\boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m)$ enables the k data bits $(s_1, s_2, \ldots, s_k)$ to be obtained as follows. For each i, $s_i = bv(\boldsymbol{x}_j)$ if there is a slice $\boldsymbol{x}_j$ representing the i-th bit; otherwise, $s_i = 0$. The function that maps $(\boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m)$ to $(s_1, s_2, \ldots, s_k)$ is denoted by $\mathcal{D}_s(\boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m)$.

The ILIFC is one of flash codes and, thus, in the ILIFC one write means changing one of information bits [6]. Let an (n, k, q, t) flash code be a coding scheme for storing k data bits in n cells with q levels such that any t writes can be accommodated without block erasure. The performance of an (n, k, q, t) flash code is characterized in terms of its write deficiency. The write deficiency of an (n, k, q, t) flash code is defined as

$$n(q-1) - t.$$

The write deficiency of ILIFC(n, k, q) is at most

$$(k-1)\left((k+1)(q-1) - 1\right). \tag{2.1}$$

Flash codes proposed in [22] achieve a write deficiency of $O(qk^2)$, which is the best previously known result. From (2.1), the ILIFC also achieves the write deficiency of $O(qk^2)$.

As stated in Chapter 1, practically, it is preferable that multiple bits among the information bits can be changed by one write. In this dissertation, one write operation means that the current data bits are changed to new data bits. If the new data bits are equal to the current data bits, then we assume that no write operation has occurred. The number of such write operations that can occur between two consecutive block erasures is referred to simply as the number of writes. The number of writes depends on the sequence of data bits to be stored. The minimum number of writes is termed the worst-case number of writes.

Assume that the state of m slices $(\boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m)$ is changed to $(\boldsymbol{x}'_1 \mid \boldsymbol{x}'_2 \mid \cdots \mid \boldsymbol{x}'_m)$ by one write, where $\boldsymbol{x}_j, \boldsymbol{x}'_j \in A_q^k$ for $1 \leq j \leq m$. Then $\sum_{j=1}^{m} \Big(wt(\boldsymbol{x}'_j) - wt(\boldsymbol{x}_j) \Big)$ is termed the total number of cell level changes.

2.2 ILIFC with inversion cells

Suppose that a write operation in which the current data $\boldsymbol{v}$ are changed to the new data $\boldsymbol{v}'$ is conducted by ILIFC. If such a write can be achieved without block erasure, the total number of cell level changes is equal to the Hamming distance between $\boldsymbol{v}$ and $\boldsymbol{v}'$. An ILIFC with inversion cells (I-ILIFC) was proposed in order to reduce the total number of cell level changes and increase the number of writes [7]. The I-ILIFC has two storing modes, a normal mode and an inverted mode, information about which is contained in the inversion cells.

In this dissertation, it is assumed that k data bits are stored in a block of n q-ary cells including r inversion cells. Such an I-ILIFC is denoted by I-ILIFC(n, k, q, r). In the I-ILIFC(n, k, q, r), a block of $(n - r)$ cells excluding the r inversion cells is divided into slices consisting of k cells. These cells, which are grouped into slices, are termed data cells. Hence, there are $m = \lfloor (n - r)/k \rfloor$ slices. The restriction of the ILIFC scheme, $m \geq k$, determines that $n \geq k^2 + r$ should hold.

For $\boldsymbol{w} = (w_1, w_2, \ldots, w_l) \in \{0, 1\}^l$, we define $\overline{\boldsymbol{w}} = (\overline{w_1}, \overline{w_2}, \ldots, \overline{w_l})$, where $\overline{w_i}$ is 1 if $w_i = 0$, and 0 if $w_i = 1$. For $\boldsymbol{w}, \boldsymbol{w}' \in \{0, 1\}^l$, let $d_H(\boldsymbol{w}, \boldsymbol{w}')$ be the Hamming distance between $\boldsymbol{w}$ and $\boldsymbol{w}'$.

In the I-ILIFC, the storing mode is represented by the r inversion cells. We denote the state of these r inversion cells by $\boldsymbol{b} = (b_1, b_2, \ldots, b_r) \in A_q^r$. We denote the state of the inversion cells and m slices by $\boldsymbol{c} = (\boldsymbol{b} \mid \boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m)$, where $\boldsymbol{x}_j \in A_q^k$ for $1 \leq j \leq m$. Suppose that the data $\boldsymbol{v} \in \{0, 1\}^k$ are stored in the cell state $\boldsymbol{c}$. If $bv(\boldsymbol{b}) = 0$, the cell is in the normal mode, and $\mathcal{D}_s(\boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m) = \boldsymbol{v}$ is satisfied. If $bv(\boldsymbol{b}) = 1$, the cell is in the inverted mode, and $\mathcal{D}_s(\boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m) = \overline{\boldsymbol{v}}$ is satisfied. If there is an i that satisfies $b_i < q - 1$, the mode is changed by increasing b_i by 1.

For example, let $n = 20, k = 4, q = 3$ and $r = 4$. Suppose that the state of $(20 - 4)/4 = 4$ slices and 4 inversion cells is as follows.

$$(\underbrace{(2,2,0,0)}_{\text{inversion cells}} \mid \underbrace{(2,2,1,0)}_{\text{1st bit}} \mid \underbrace{(2,0,2,2)}_{\text{3rd bit}} \mid \underbrace{(2,2,0,2)}_{\text{4th bit}} \mid \underbrace{(0,2,1,0)}_{\text{2nd bit}}).$$

Then the cell is in the normal mode and the stored bits $(1, 1, 0, 0)$ are exactly the data.

On the other hand, suppose that the state of 4 slices and 4 inversion cells is as follows.

$$(\underbrace{(2,2,1,0)}_{\text{inversion cells}} \mid \underbrace{(2,2,2,2)}_{\text{full}} \mid \underbrace{(2,1,0,2)}_{\text{4th bit}} \mid \underbrace{(2,2,0,0)}_{\text{1st bit}} \mid \underbrace{(1,0,2,2)}_{\text{3rd bit}}).$$

Then the cell is in the inverted mode and the stored bits $(0, 0, 1, 1)$ are the inverted data, that is, the data are $(1, 1, 0, 0)$.

When new data are given, two rules can be considered for the encoding of the I-ILIFC. We refer to these rules as rule 1 and rule 2. Rule 1 is to change the storing mode and store the data in the new mode. Rule 2 is to store the data in the current mode.

Assume that the state $(\boldsymbol{b} \mid \boldsymbol{x}_1 \mid \boldsymbol{x}_2 \mid \cdots \mid \boldsymbol{x}_m)$ is changed to $(\boldsymbol{b}' \mid \boldsymbol{x}'_1 \mid \boldsymbol{x}'_2 \mid \cdots \mid \boldsymbol{x}'_m)$ by one write operation. Then $\sum_{j=1}^{m} \Big(wt(\boldsymbol{x}'_j) - wt(\boldsymbol{x}_j) \Big)$ is termed the sum of the data cell level changes and $(wt(\boldsymbol{b}') - wt(\boldsymbol{b}))$ is termed the sum of the inversion cell level changes. The sum of these two values is referred to as the total number of cell level changes.

In the I-ILIFC, when new data are given, one of two rules is applied during encoding such that the total number of cell level changes is minimized. If the rule cannot be applied without block erasure, erasure takes place.

Let $n = 20, k = 4, q = 3$ and $r = 4$. Suppose that the state of 4 slices and 4 inversion cells is as follows.

$$(\underbrace{(1,0,0,0)}_{\text{inversion cells}} \mid \underbrace{(0,0,1,0)}_{\text{3rd bit}} \mid \underbrace{(0,0,0,0)}_{\text{empty}} \mid \underbrace{(0,0,0,0)}_{\text{empty}} \mid \underbrace{(0,0,0,0)}_{\text{empty}}).$$

The data represented by this state are $(1,1,0,1)$ (Note that the cell is in the inverted mode). Suppose that the data are changed from $(1,1,0,1)$ to $(0,1,0,0)$. If rule 1 is applied, that is, the storing mode is changed and the data are stored in the new mode, the state of slices and inversion cells is as follows.

$$(\underbrace{(2,0,0,0)}_{\text{inversion cells}} \mid \underbrace{(0,0,2,0)}_{\text{3rd bit}} \mid \underbrace{(0,1,0,0)}_{\text{2nd bit}} \mid \underbrace{(0,0,0,0)}_{\text{empty}} \mid \underbrace{(0,0,0,0)}_{\text{empty}}).$$

Then, the total number of cell level changes is 3. On the other hand, If rule 2 is applied, that is, the data are stored in the current mode, the state of slices and inversion cells is as follows.

$$(\underbrace{(1,0,0,0)}_{\text{inversion cells}} \mid \underbrace{(0,0,1,0)}_{\text{3rd bit}} \mid \underbrace{(1,0,0,0)}_{\text{1st bit}} \mid \underbrace{(0,0,0,1)}_{\text{4th bit}} \mid \underbrace{(0,0,0,0)}_{\text{empty}}).$$

Then, the total number of cell level changes is 2. Therefore, in this example, rule 2 is applied such that the total number of cell level changes is minimized.

The following theorem holds [7].

Theorem 1. *Suppose a write operation, in which the current data $\boldsymbol{v}$ are changed to the new data $\boldsymbol{v}'$, is carried out by I-ILIFC(n,k,q,r), where $\boldsymbol{v}, \boldsymbol{v}' \in \{0,1\}^k$. Rule 1 is applied during encoding if and only if $d_H(\boldsymbol{v}, \boldsymbol{v}') > (k+1)/2$.*

Proof. We denote $d_H(\boldsymbol{v}, \boldsymbol{v}')$ by d. If rule 1 is applied, the sum of the data cell level changes is $d_H(\boldsymbol{v}, \overline{\boldsymbol{v}'}) = d_H(\overline{\boldsymbol{v}}, \boldsymbol{v}') = k - d$, and the sum of the inversion cell level changes is 1. Hence, the total number of cell level changes is $(k - d + 1)$. On the other hand, if rule 2 is applied, the total number of cell level changes is equal to the sum of the data cell level changes, $d_H(\boldsymbol{v}, \boldsymbol{v}') = d_H(\overline{\boldsymbol{v}}, \overline{\boldsymbol{v}'}) = d$. Therefore, if $d > k - d + 1$, that is, $d > (k+1)/2$, rule 1 is applied during encoding. Additionally, if $d \le (k+1)/2$, the above discussion shows that rule 2 is applied during encoding. □

For the state of r inversion cells $\boldsymbol{b} = (b_1, b_2, \ldots, b_r) \in A_q^r$, the inversion cells are said to be exhausted if $b_1 = b_2 = \cdots = b_r = q - 1$, that is, $wt(\boldsymbol{b}) = r(q-1)$. In this case, rule 2 is applied until the next block erasure takes place.

Computer simulation is used to show that the average number of writes by I-ILIFC(n,k,q,r) is greater than that by ILIFC(n,k,q) in many cases when the number r of inversion cells is optimized [7, 10]. We would like to show this result theoretically, but conducting an analysis of the average performance may not be a simple matter. Therefore, in this dissertation we analyze the worst-case performance and specify a threshold for the code length that determines whether the I-ILIFC improves the worst-case performance of the ILIFC.

Chapter 3

Worst-Case Performance of ILIFC and ILIFC with Inversion Cells

This chapter describes the analysis of the worst-case performance of the I-ILIFC and compare the performance of the I-ILIFC with that of the ILIFC.

3.1 Upper bound on the worst-case number of writes by ILIFC

Under the definition of one write operation in [6], that is, write operation changing one of data bits, it is shown that the worst-case number of writes by ILIFC(n, k, q) is

$$k(\lfloor n/k \rfloor - k + 1)(q-1) + k - 1$$

[6]. Let t_w be the worst-case number of writes by ILIFC(n, k, q) under the definition of one write in this dissertation. In this section, we derive the upper bound on t_w.

We denote $(0, 0, \ldots, 0)$ and $(1, 1, \ldots, 1)$ by $\mathbf{0}$ and $\mathbf{1}$, respectively. Let T be the number of writes by ILIFC(n, k, q) when the data sequence is $\mathbf{1}, \mathbf{0}, \mathbf{1}, \mathbf{0}, \ldots$. Then $t_w \leq T$ holds. If the state of slices after T such writes is $(\boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$, $\sum_{j=1}^{m} wt(\boldsymbol{y}_j) = kT$ holds. Since the total number of cell levels is $n(q-1)$, $kT \leq n(q-1)$ holds. We denote the upper bound on t_w by t_{ub}. Then from $t_w \leq T \leq n(q-1)/k$, we have

$$t_{ub} = n(q-1)/k. \tag{3.1}$$

3.2 Maximum number of unused cell levels in I-ILIFC

In the I-ILIFC, when all data bits are changed, the sum of the data cell level changes is 0. In this dissertation, we consider the worst-case performance of the I-ILIFC. Therefore, suppose that the sum of the data cell level changes caused by one write is greater than 0.

In the following, it is assumed that a sufficient number r of inversion cells are reserved such that the inversion cells are not entirely consumed whenever block erasure takes place in the worst case. Then, $r(q-1)$ is bounded above by the maximum number of writes $(n-r)(q-1)$. That is, $r \leq n/2$. The strict bound on r is obtained in the next section.

When erasure takes place, we denote the state of slices by $(\boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$, where $m = \lfloor (n-r)/k \rfloor$. Then the weight of each of the slices $wt(\boldsymbol{y}_j)$ can be increased $(k(q-1) - wt(\boldsymbol{y}_j))$ more times. The sum of such unused cell levels $\sum_{j=1}^{m} (k(q-1) - wt(\boldsymbol{y}_j))$ is termed the number of unused cell levels.

Since the number of unused cell levels decreases as the number of writes increases, it is apparent that the number of unused cell levels in the worst case is greater than in the other cases. Therefore, in this section we determine the maximum number of unused cell levels.

For $\boldsymbol{v}, \boldsymbol{v}' \in \{0,1\}^k (\boldsymbol{v} \neq \boldsymbol{v}')$, let d be the Hamming distance between $\boldsymbol{v}$ and $\boldsymbol{v}'$. If the write operation, in which the data $\boldsymbol{v}$ are changed to $\boldsymbol{v}'$, is executed, then from Theorem 1, the sum of the data cell level changes is d if $d \leq (k+1)/2$, and $(k-d)$ if $d > (k+1)/2$. Hence, if k is even, the maximum sum of the data cell level changes $\delta(k)$ is as follows.

$$\begin{aligned}\delta(k) &= \max\{\max_{1\leq d\leq k/2} d, \max_{k/2+1\leq d\leq k}(k-d)\} \\ &= \max\{k/2, k/2-1\} = k/2.\end{aligned}$$

Similarly, if k is odd,

$$\begin{aligned}\delta(k) &= \max\{\max_{1\leq d\leq (k+1)/2} d, \max_{(k+1)/2+1\leq d\leq k}(k-d)\} \\ &= \max\{(k+1)/2, (k+1)/2-2\} = (k+1)/2.\end{aligned}$$

Therefore,

$$\delta(k) = \begin{cases} k/2 & (k \text{ is even}) \\ (k+1)/2 & (k \text{ is odd}) \end{cases}.$$

For the state of slices $(\boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$, let α_1 be the number of bits without any corresponding slice, and let α_2 be the number of empty slices. Then the next write operation for any new data can be carried out if and only if the changes to any $\delta(k)$ bits among the k data bits can be stored in slices, that is,

$$\begin{gathered}(\alpha_1 < \delta(k) \text{ and } \alpha_2 \geq \alpha_1) \text{ or} \\ (\alpha_1 \geq \delta(k) \text{ and } \alpha_2 \geq \delta(k)).\end{gathered}$$

This condition is equivalent to the following condition.

$$\alpha_2 \geq \min\{\alpha_1, \delta(k)\}.$$

Therefore, block erasure may take place if and only if

$$\alpha_2 < \min\{\alpha_1, \delta(k)\}. \tag{3.2}$$

The number of bits that have a corresponding slice is $(k-\alpha_1)$. Let $\boldsymbol{y}_{j_i}$ be the slice corresponding to the i-th bit from the left among $(k-\alpha_1)$ such bits. Since $\boldsymbol{y}_{j_i}$ is active, $1 \leq wt(\boldsymbol{y}_{j_i}) \leq k(q-1)-1$. Then the number of unused cell levels is as follows:

$$\sum_{i=1}^{k-\alpha_1}(k(q-1) - wt(\boldsymbol{y}_{j_i})) + \alpha_2 \cdot k(q-1). \tag{3.3}$$

When (3.2) holds, the maximum number of unused cell levels is derived. For fixed values of α_1 and α_2, the number of unused cell levels is maximized when $wt(\boldsymbol{y}_{j_1}) = \cdots = wt(\boldsymbol{y}_{j_{k-\alpha_1}}) = 1$. Hence, the maximum is expressed as follows.

$$(k-\alpha_1)(k(q-1)-1) + \alpha_2 \cdot k(q-1).$$

When $\alpha_1 < \delta(k)$ holds:

From (3.2), $\alpha_2 < \alpha_1$ holds. For fixed α_1, when $\alpha_2 = \alpha_1 - 1$, the maximum is expressed as follows.

$$\begin{aligned} & (k-\alpha_1)(k(q-1)-1) + (\alpha_1 - 1)\cdot k(q-1) \\ = \; & (k-1)\cdot k(q-1) - k + \alpha_1. \end{aligned}$$

Therefore, when $\alpha_1 = \delta(k) - 1$, the maximum is as follows.

$$(k-1)\cdot k(q-1) - k + \delta(k) - 1. \tag{3.4}$$

When $\alpha_1 \geq \delta(k)$ holds:

From (3.2), $\alpha_2 < \delta(k)$ holds. Similarly, for a constant value of α_1, when $\alpha_2 = \delta(k) - 1$, the maximum is expressed as follows.

$$(k-\alpha_1)(k(q-1)-1) + (\delta(k)-1)\cdot k(q-1).$$

Hence, when $\alpha_1 = \delta(k)$, the maximum is as follows.

$$(k-1)\cdot k(q-1) - k + \delta(k). \tag{3.5}$$

From (3.4) and (3.5), the maximum number of unused cell levels is given by (3.5). We have the following theorem.

Theorem 2. *In the I-ILIFC(n,k,q,r), when block erasure takes place, the number u of unused cell levels satisfies the following inequality:*

$$u \leq (k-1)\cdot k(q-1) - k + \delta(k).$$

3.3 Lower bound on the number of writes by I-ILIFC

In this section, we use Theorem 2 to show the lower bound on the worst-case number of writes by I-ILIFC.

Let $(\boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$ be the state of slices, where $m = \lfloor (n-r)/k \rfloor$. Then $\sum_{j=1}^{m} wt(\boldsymbol{y}_j)$ is termed the number of used cell levels.

The number of used cell levels increases as the number of writes increases. Therefore, in order to consider the worst-case number of writes, we derive the minimum number of used cell levels when block erasure takes place.

Theorem 3. *In the I-ILIFC(n,k,q,r), when block erasure takes place, the number u' of used cell levels satisfies the following inequality:*

$$u' \geq (\lfloor (n-r)/k \rfloor - k + 1)\cdot k(q-1) + k - \delta(k).$$

Proof. Let $(\boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$ be the state of slices. From Theorem 2,

$$\sum_{j=1}^{m} (k(q-1) - wt(\boldsymbol{y}_j)) \leq (k-1)\cdot k(q-1) - k + \delta(k).$$

Therefore,

$$u' = \sum_{j=1}^{m} wt(\boldsymbol{y}_j) \geq (m-k+1)\cdot k(q-1) + k - \delta(\check{k}),$$

where $m = \lfloor (n-r)/k \rfloor$. $\square$

Next, we use Theorem 3 to show a sufficient condition for the next write operation to be possible for any new data.

Corollary 1. *If the number u' of used cell levels satisfies the inequality*

$$u' < (\lfloor (n-r)/k \rfloor - k + 1) \cdot k(q-1) + k - \delta(k),$$

then the next write operation for any new data can be executed without block erasure.

Proof. This is the contraposition of Theorem 3. □

For given n, k, and q, we define

$$\begin{aligned} U_1(r) &= (\lfloor (n-r)/k \rfloor - k + 1) \cdot k(q-1) + k - \delta(k), \\ U_1'(r) &= ((n-r)/k - k + 1) \cdot k(q-1) + k - \delta(k). \end{aligned}$$

Then $U_1(r) \leq U_1'(r)$. We define $t_1(r) = \lceil U_1(r)/\delta(k) \rceil$. Let r_1^* be the minimum integer r that satisfies $r(q-1) \geq U_1'(r)/\delta(k) + 1$. That is, r_1^* is the integer r that satisfies $R_1 \leq r < R_1 + 1$, where

$$R_1 = \frac{n - k^2 + k + k/(q-1)}{\delta(k) + 1}. \tag{3.6}$$

As mentioned in Section 2.2, the restriction on the I-ILIFC(n, k, q, r_1^*) scheme requires $n \geq k^2 + r_1^*$ to be satisfied. Note that this inequality is satisfied if

$$n \geq k^2 + R_1 + 1 \tag{3.7}$$

holds. We substitute (3.6) for R_1 in (3.7) and arrange the inequality so that n appears only on the left side. The result is as follows.

$$n \geq k^2 + \frac{k + 1 + k/(q-1)}{\delta(k)} + 1. \tag{3.8}$$

In the following, we consider only values of n, k, and q that satisfy (3.8) because the I-ILIFC scheme cannot be defined if (3.8) is not satisfied. Then from (3.6) and (3.8), $R_1 > 0$. That is, $r_1^* \geq 1$.

We use the sufficient condition of Corollary 1 to derive the lower bound on the number of writes.

Theorem 4. *Suppose that* (3.8) *is satisfied. Let t_1^* be the number of writes by I-ILIFC (n, k, q, r_1^*). Then*

$$t_1^* \geq t_1(r_1^*).$$

Proof. We prove this theorem by induction on the number of writes t.

Since (3.8) is satisfied, $n \geq k^2 + r_1^*$; that is, the number of slices is at least k. Additionally, $r_1^* \geq 1$. Therefore, it is apparent that the first write operation for any new data can be executed.

For $t < t_1(r_1^*)$, we suppose that t write operations for any data sequence of length t can be executed without block erasure. Let $(\boldsymbol{b} \mid \boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$ be the state of inversion cells and slices after t writes. Then

$$wt(\boldsymbol{b}) \leq t \tag{3.9}$$

because when one write operation has occurred, the maximum sum of the inversion cell level changes is 1. From the definitions,

$$t < t_1(r_1^*) = \lceil U_1(r_1^*)/\delta(k) \rceil < U_1(r_1^*)/\delta(k) + 1 \leq U_1'(r_1^*)/\delta(k) + 1 \leq r_1^*(q-1). \tag{3.10}$$

From (3.9) and (3.10), $wt(\boldsymbol{b}) < r_1^*(q-1)$. Hence, r_1^* inversion cells are not used in their entirety. Additionally, since the maximum sum of the data cell level changes is $\delta(k)$,

$$\sum_{j=1}^{m} wt(\boldsymbol{y}_j) \leq \delta(k) \cdot t. \tag{3.11}$$

$t_1(r_1^*)$ is the minimum integer t' such that $t' \geq U_1(r_1^*)/\delta(k)$, and t is the integer that satisfies $t < t_1(r_1^*)$. Hence, $t < U_1(r_1^*)/\delta(k)$ holds. Therefore,

$$\delta(k) \cdot t < U_1(r_1^*). \tag{3.12}$$

From (3.11) and (3.12), $\sum_{j=1}^{m} wt(\boldsymbol{y}_j) < U_1(r_1^*)$. Therefore, according to Corollary 1, the next $(t+1)$-th write operation for any new data can be executed.

The above discussion serves to confirm that $t_1(r_1^*)$ write operations for any data sequence of length $t_1(r_1^*)$ can be carried out without erasure. □

We define

$$\begin{aligned} U_{lb1}(r) &= ((n-r)/k - 1 - k + 1) \cdot k(q-1) + k - \delta(k) \\ &= (n - k^2 - r)(q-1) + k - \delta(k). \end{aligned}$$

Then $U_1(r) > U_{lb1}(r)$. Hence,

$$t_1(r_1^*) = \lceil U_1(r_1^*)/\delta(k) \rceil \geq U_1(r_1^*)/\delta(k) > U_{lb1}(r_1^*)/\delta(k). \tag{3.13}$$

From $r_1^* < R_1 + 1$,

$$U_{lb1}(r_1^*) > U_{lb1}(R_1 + 1). \tag{3.14}$$

From (3.13) and (3.14),

$$t_1(r_1^*) > U_{lb1}(R_1 + 1)/\delta(k). \tag{3.15}$$

We define $t_{lb1}^* = U_{lb1}(R_1+1)/\delta(k)$. Let t_{w1}^* be the worst-case number of writes by I-ILIFC(n,k,q,r_1^*). Since $t_{w1}^* \geq t_1(r_1^*) > t_{lb1}^*$, t_{lb1}^* is the lower bound on t_{w1}^*.

If k is even,

$$t_{lb1}^* = 2\left(\frac{n-k^2-2}{k+2} - \frac{1}{k}\right)(q-1) + \frac{2k}{k+2} - 1. \tag{3.16}$$

If k is odd,

$$t_{lb1}^* = 2\left(\frac{n-k^2-3}{k+3}\right)(q-1) + \frac{2k}{k+3} - 1. \tag{3.17}$$

We compare t_{lb1}^* and t_{ub}, where t_{ub} is the upper bound on the worst-case number t_w of writes by ILIFC(n,k,q). From (3.1), $t_{ub} = n(q-1)/k$. If $t_{ub} < t_{lb1}^*$, then $t_w \leq t_{ub} < t_{lb1}^* < t_{w1}^*$; that is, the worst-case number of writes by I-ILIFC(n,k,q,r_1^*) is greater than that by ILIFC(n,k,q). Therefore, $t_{lb1}^* > t_{ub}$ is the sufficient condition for improving the worst-case performance of ILIFC(n,k,q). For $k \geq 4$, we can show that $t_{lb1}^* > t_{ub}$ if and only if $n > p_1$. From (3.16) and (3.17),

$$p_1 = \begin{cases} \frac{2(k^3+3k+2)}{k-2} - \frac{k}{q-1} & (k \text{ is even}) \\ \frac{2k(k^2+3)}{k-3} - \frac{k}{q-1} & (k \text{ is odd}) \end{cases}. \tag{3.18}$$

Then p_1 is a threshold of the code length n that determines whether I-ILIFC(n,k,q,r_1^*) improves the performance of ILIFC(n,k,q) in the worst case. In this dissertation, p_1 is referred to simply as the threshold. The results show that I-ILIFC(n,k,q,r_1^*) improves the worst-case performance of ILIFC(n,k,q) if n is sufficiently large.

3.4 Another lower bound on the number of writes by I-ILIFC

Thus far, we have assumed that block erasure takes place if the rule that minimizes the total number of cell level changes cannot be applied during encoding. However, at the moment it remains possible to apply another rule, that is, the rule that does not minimize the total number of cell level changes.

For example, let $n = 20, k = 4, q = 3$ and $r = 4$. Suppose that the state of 4 slices and 4 inversion cells is as follows.

$$(\underbrace{(2,2,1,0)}_{\text{inversion cells}} \mid \underbrace{(2,2,2,2)}_{\text{full}} \mid \underbrace{(2,2,2,0)}_{\text{1st bit}} \mid \underbrace{(2,2,1,2)}_{\text{4th bit}} \mid \underbrace{(0,2,2,1)}_{\text{2nd bit}}).$$

Then the cell is in the inverted mode and the stored bits are $(0, 1, 0, 1)$, that is, the data are $(1, 0, 1, 0)$. Suppose that the data are changed from $(1, 0, 1, 0)$ to $(0, 1, 1, 1)$. Since the Hamming distance between $(1, 0, 1, 0)$ and $(0, 1, 1, 1)$ is 3, from Theorem 1, rule 1 is applied, that is, the storing mode is changed to the normal mode and the stored bits $(0, 1, 0, 1)$ are changed to $(0, 1, 1, 1)$. However, rule 1 cannot be applied because there is not a slice representing the changed third bit and no empty slices exist. On the other hand, rule 2 can be applied, that is, the stored bits $(0, 1, 0, 1)$ can changed to the inverted data $(1, 0, 0, 0)$ as follows.

$$(\underbrace{(2,2,1,0)}_{\text{inversion cells}} \mid \underbrace{(2,2,2,2)}_{\text{full}} \mid \underbrace{(2,2,2,1)}_{\text{1st bit}} \mid \underbrace{(2,2,2,2)}_{\text{full}} \mid \underbrace{(0,2,2,2)}_{\text{2nd bit}}).$$

As shown in this example, the number of block erasures can be reduced by ensuring that if the rule that minimizes the total number of cell level changes cannot be applied but another rule can be applied during encoding, the latter rule is applied before erasure takes place. Therefore, in this section it is assumed that block erasure takes place if neither of the two rules can be applied during encoding. Under this assumption, we derive another lower bound on the number of writes. This lower bound cannot simply be compared with the first lower bound because, as will be seen, the number of inversion cells is different from that for the case described in the previous section.

3.4.1 Maximum number of unused cell levels

In this subsection, we determine the maximum number of unused cell levels when block erasure takes place. We show a necessary and sufficient condition for the next write operation to cause block erasure when new data are given.

Theorem 5. *For the state of slices* $(\boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$*, let* β_1 *be the number of bits that do not have a corresponding slice, and let* β_2 *be the number of empty slices. Then block erasure may take place if and only if* $\lfloor \beta_1/2 \rfloor > \beta_2$*.*

Proof. Necessity: We show that if $\lfloor \beta_1/2 \rfloor \le \beta_2$ holds, then either rule 1 or rule 2 can be applied without erasure during the encoding of any new data.

When some new data are given, let l be the number of bits without any corresponding slice that are changed on the data (not on the sequence stored in the slices), where $0 \le l \le \beta_1$. Then rule 2 can be applied if the changes to l such bits can be stored in l empty slices. On the other hand, the number of bits without any slice that are not changed on the data is $\beta_1 - l$. Therefore, rule 1 can be applied if the changes to $\beta_1 - l$ such bits can be stored in $\beta_1 - l$ empty slices.

When $l \le \lfloor \beta_1/2 \rfloor$ holds:

The inequality $l \le \beta_2$ determines that the changes that are made to l bits can be stored in l slices from among β_2 empty slices. Hence, rule 2 can be applied. On the other hand, the inequality $\beta_1 - l > \beta_2$ may hold. If this inequality holds, although the level of one inversion cell can be increased, rule 1 cannot be applied because the number of empty slices is less than the number of bits for which an empty slice should be reserved.

When $l > \lfloor \beta_1/2 \rfloor$ holds:

If β_1 is even,

$$\beta_1 - l < \beta_1 - \beta_1/2 = \lfloor \beta_1/2 \rfloor \le \beta_2. \tag{3.19}$$

If β_1 is odd,

$$\beta_1 - l < \beta_1 - (\beta_1 - 1)/2 = (\beta_1 - 1)/2 + 1 = \lfloor \beta_1/2 \rfloor + 1.$$

That is,

$$\beta_1 - l \le \lfloor \beta_1/2 \rfloor \le \beta_2. \tag{3.20}$$

Hence, from (3.19) and (3.20), the changes related to $(\beta_1 - l)$ bits can be stored in $(\beta_1 - l)$ slices from among β_2 empty slices. Therefore, rule 1 can be applied.

The above discussion shows that either rule 1 or rule 2 can be applied during encoding if $\lfloor \beta_1/2 \rfloor \le \beta_2$.

Sufficiency: We show that if $\lfloor \beta_1/2 \rfloor > \beta_2$ holds, then neither rule 1 nor rule 2 can be applied during the encoding of some new data.

Let $\boldsymbol{v}$ be the current data and let $\boldsymbol{v}'$ be the new data. We consider β_1 bits without any corresponding slice among k data bits $\boldsymbol{v}$. It is assumed that $\boldsymbol{v}'$ is the data after $\lfloor \beta_1/2 \rfloor$ bits among β_1 such bits in $\boldsymbol{v}$ have been changed. We show that neither rule 1 nor rule 2 can be applied during the encoding of $\boldsymbol{v}'$.

When rule 2 is applied, the number of bits for which an empty slice should be reserved is $\lfloor \beta_1/2 \rfloor$. The inequality $\lfloor \beta_1/2 \rfloor > \beta_2$ determines that rule 2 cannot be applied because there is a bit for which a slice cannot be reserved. On the other hand, when rule 1 is applied, the number of bits for which an empty slice should be reserved is $\beta_1 - \lfloor \beta_1/2 \rfloor$. If β_1 is even,

$$\beta_1 - \lfloor \beta_1/2 \rfloor = \beta_1/2 = \lfloor \beta_1/2 \rfloor > \beta_2. \tag{3.21}$$

If β_1 is odd,

$$\beta_1 - \lfloor \beta_1/2 \rfloor = \beta_1 - (\beta_1 - 1)/2 = (\beta_1 - 1)/2 + 1 = \lfloor \beta_1/2 \rfloor + 1 > \lfloor \beta_1/2 \rfloor > \beta_2. \tag{3.22}$$

From (3.21) and (3.22), $\beta_1 - \lfloor \beta_1/2 \rfloor > \beta_2$. Hence, since there is a bit for which an empty slice cannot be reserved, rule 1 cannot be applied.

The above discussion indicates that neither rule 1 nor rule 2 can be applied during encoding if $\lfloor \beta_1/2 \rfloor > \beta_2$ when the new data $\boldsymbol{v}'$ are given. □

The number of unused cell levels is as follows:

$$\sum_{i=1}^{k-\beta_1} (k(q-1) - wt(\boldsymbol{y}_{j_i})) + \beta_2 \cdot k(q-1), \tag{3.23}$$

where $\boldsymbol{y}_{j_1}, \dots, \boldsymbol{y}_{j_{k-\beta_1}}$ are $(k - \beta_1)$ active slices, that is, $1 \le wt(\boldsymbol{y}_{j_i}) \le k(q-1) - 1$. When $\lfloor \beta_1/2 \rfloor > \beta_2$ holds, we derive the maximum number of unused cell levels. For fixed values of β_1 and β_2, the number of unused cell levels is maximized when $wt(\boldsymbol{y}_{j_i}) = \cdots = wt(\boldsymbol{y}_{j_{k-\beta_1}}) = 1$. Hence, the maximum is expressed as follows.

$$(k - \beta_1)\,(k(q-1) - 1) + \beta_2 \cdot k(q-1).$$

When β_1 is even:

When β_1 is even, $\beta_1/2 > \beta_2$ holds. For fixed β_1, when $\beta_2 = \beta_1/2 - 1$, the maximum is expressed as follows.

$$\beta_1 \left(1 - k(q-1)/2\right) + k\left((k-1)(q-1) - 1\right). \tag{3.24}$$

Since $1 - k(q-1)/2 \leq 0$ and $\beta_1/2 - 1 = \beta_2 \geq 0$ hold, when $\beta_2 = 0$ and $\beta_1 = 2$, the maximum is as follows.

$$(k-2) \cdot k(q-1) - k + 2. \tag{3.25}$$

When β_1 is odd:

When β_1 is odd, $(\beta_1 - 1)/2 > \beta_2$ holds. For a constant value of β_1, when $\beta_2 = (\beta_1 - 1)/2 - 1$, the maximum is expressed as follows.

$$\beta_1 \left(1 - k(q-1)/2\right) + (k - 3/2) \cdot k(q-1) - k.$$

Similarly, since $(\beta_1 - 1)/2 - 1 = \beta_2 \geq 0$ holds, when $\beta_2 = 0$ and $\beta_1 = 3$, the maximum is as follows.

$$(k-3) \cdot k(q-1) - k + 3. \tag{3.26}$$

From (3.25) and (3.26), we have the following theorem.

Theorem 6. *In the I-ILIFC(n, k, q, r), when block erasure takes place, the number u of unused cell levels satisfies the following inequality:*

$$u \leq (k-2) \cdot k(q-1) - k + 2.$$

3.4.2 Another lower bound on the number of writes

In this subsection, we derive another lower bound on the number of writes. From Theorem 6, we obtain the minimum number of used cell levels when block erasure takes place.

Theorem 7. *In the I-ILIFC(n, k, q, r), when block erasure takes place, the number u' of used cell levels satisfies the following inequality:*

$$u' \geq (\lfloor (n-r)/k \rfloor - k + 2) \cdot k(q-1) + k - 2.$$

Proof. Let $(\boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$ be the state of slices. From Theorem 6,

$$\sum_{j=1}^{m} \left(k(q-1) - wt(\boldsymbol{y}_j)\right) \leq (k-2) \cdot k(q-1) - k + 2.$$

Therefore,

$$u' = \sum_{j=1}^{m} wt(\boldsymbol{y}_j) \geq (m - k + 2) \cdot k(q-1) + k - 2,$$

where $m = \lfloor (n-r)/k \rfloor$. □

From Theorem 7, we have a sufficient condition for the next write operation to be possible for any new data.

Corollary 2. *If the number u' of used cell levels satisfies the inequality*

$$u' < (\lfloor (n-r)/k \rfloor - k + 2) \cdot k(q-1) + k - 2,$$

then the next write operation for any new data can be executed without erasure.

Proof. This is the contraposition of Theorem 7. $\square$

For given n, k, and q, we define

$$\begin{aligned} U_2(r) &= (\lfloor (n-r)/k \rfloor - k + 2) \cdot k(q-1) + k - 2, \\ U_2'(r) &= ((n-r)/k - k + 2) \cdot k(q-1) + k - 2. \end{aligned}$$

Then $U_2(r) \leq U_2'(r)$. We define

$$t_2(r) = \lceil (U_2(r) - U_1(r) - \delta(k) + 1)/(k-1) \rceil.$$

Clearly, $t_2(r) > 0$. Let r_2^* be the minimum integer r that satisfies

$$r(q-1) \geq \frac{U_1'(r)}{\delta(k)} + \frac{U_2'(r) - U_1'(r) - \delta(k) + 1}{k-1} + 2.$$

That is, r_2^* is the integer r that satisfies $R_2 \leq r < R_2 + 1$, where

$$R_2 = \frac{1}{\delta(k)+1} \times \left(n - k^2 + k + \frac{k + \delta(k)}{q-1} + \frac{k\delta(k)}{k-1} - \frac{\delta(k)}{(q-1)(k-1)} \right). \tag{3.27}$$

From the restriction on the I-ILIFC(n, k, q, r_2^*) scheme, $n \geq k^2 + r_2^*$ should hold. This inequality is satisfied if

$$n \geq k^2 + R_2 + 1. \tag{3.28}$$

We substitute (3.27) for R_2 in (3.28) and arrange the inequality so that n appears only on the left side. The result is

$$n \geq k^2 + \frac{1}{\delta(k)} \left(k + \frac{k + \delta(k)}{q-1} + \frac{k\delta(k)}{k-1} - \frac{\delta(k)}{(q-1)(k-1)} \right) + \frac{\delta(k)+1}{\delta(k)}. \tag{3.29}$$

In the following, we consider only values of n, k, and q that satisfy (3.29). Then, from (3.27) and (3.29), $R_2 > 0$; that is, $r_2^* \geq 1$. From Corollary 2, we obtain another lower bound on the number of writes by I-ILIFC(n, k, q, r_2^*).

Theorem 8. *Suppose that (3.29) is satisfied. Let t_2^* be the number of writes by I-ILIFC (n, k, q, r_2^*). Then*

$$t_2^* \geq t_1(r_2^*) + t_2(r_2^*).$$

Proof. We use induction on the number of writes t to prove this theorem.

Since (3.29) is satisfied, the first write operation for any new data can be executed.

For $t < t_1(r_2^*)$, we suppose that each of t write operations for any data sequence of length t can be executed by applying the rule that minimizes the total number of cell level changes. Let $(\boldsymbol{b} \mid \boldsymbol{y}_1 \mid \boldsymbol{y}_2 \mid \cdots \mid \boldsymbol{y}_m)$ be the state of inversion cells and slices after t writes. Then

$$wt(\boldsymbol{b}) \leq t < t_1(r_2^*) < t_1(r_2^*) + t_2(r_2^*). \tag{3.30}$$

From the definitions,

$$
\begin{aligned}
t_1(r_2^*) + t_2(r_2^*) &= \left\lceil \frac{U_1(r_2^*)}{\delta(k)} \right\rceil + \left\lceil \frac{U_2(r_2^*) - U_1(r_2^*) - \delta(k) + 1}{k-1} \right\rceil \\
&< \left(\frac{U_1(r_2^*)}{\delta(k)} + 1 \right) + \left(\frac{U_2(r_2^*) - U_1(r_2^*) - \delta(k) + 1}{k-1} + 1 \right) \\
&\leq \frac{U_1'(r_2^*)}{\delta(k)} + \frac{U_2'(r_2^*) - U_1'(r_2^*) - \delta(k) + 1}{k-1} + 2 \\
&\leq r_2^*(q-1). \qquad (3.31)
\end{aligned}
$$

From (3.30) and (3.31),

$$wt(\boldsymbol{b}) < r_2^*(q-1).$$

Hence, r_2^* inversion cells are not used in their entirety. Additionally,

$$\sum_{j=1}^{m} wt(\boldsymbol{y}_j) < U_1(r_2^*).$$

(See the proof of Theorem 4 for the details.) Therefore, according to Corollary 1, the next $(t+1)$-th write operation for any new data can be executed by applying the rule that minimizes the total number of cell level changes.

The above discussion indicates that each of $t_1(r_2^*)$ write operations for any data sequence of length $t_1(r_2^*)$ can be carried out by applying the rule that minimizes the total number of cell level changes.

Let $(\boldsymbol{b}' \mid \boldsymbol{y}_1' \mid \boldsymbol{y}_2' \mid \cdots \mid \boldsymbol{y}_m')$ be the state of inversion cells and slices after $t_1(r_2^*)$ writes. Then

$$
\begin{aligned}
\sum_{j=1}^{m} wt(\boldsymbol{y}_j') &\leq \delta(k) \cdot t_1(r_2^*) = \delta(k) \cdot \left\lceil \frac{U_1(r_2^*)}{\delta(k)} \right\rceil \\
&< \delta(k) \cdot \left(\frac{U_1(r_2^*)}{\delta(k)} + 1 \right) = U_1(r_2^*) + \delta(k).
\end{aligned}
$$

Hence,

$$\sum_{j=1}^{m} wt(\boldsymbol{y}_j') \leq U_1(r_2^*) + \delta(k) - 1. \qquad (3.32)$$

For $t_1(r_2^*) \leq t' < t_1(r_2^*) + t_2(r_2^*)$, we suppose that t' write operations for any data sequence of length t' can be carried out by applying rule 1 or rule 2. Let $(\boldsymbol{b}'' \mid \boldsymbol{y}_1'' \mid \boldsymbol{y}_2'' \mid \cdots \mid \boldsymbol{y}_m'')$ be the state of inversion cells and slices after t' writes. Then, from (3.31),

$$wt(\boldsymbol{b}'') \leq t' < t_1(r_2^*) + t_2(r_2^*) < r_2^*(q-1).$$

Therefore, r_2^* inversion cells are not used in their entirety. We consider the sum δ' of the data cell level changes when a write operation is executed by applying rule 1 or rule 2. The write operation such that $\delta' = k$ is not executed because such a write operation can be executed only by changing the mode. Hence, $\delta' \leq k - 1$. Therefore,

$$\sum_{j=1}^{m} wt(\boldsymbol{y}_j'') \leq \sum_{j=1}^{m} wt(\boldsymbol{y}_j') + (k-1) \cdot (t' - t_1(r_2^*)). \qquad (3.33)$$

From $t' - t_1(r_2^*) < t_2(r_2^*)$,

$$\begin{aligned}(k-1)\cdot(t'-t_1(r_2^*)) &\leq (k-1)\cdot(t_2(r_2^*)-1)\\ &= (k-1)\cdot\left(\left\lceil\frac{U_2(r_2^*)-U_1(r_2^*)-\delta(k)+1}{k-1}\right\rceil-1\right)\\ &< (k-1)\cdot\frac{U_2(r_2^*)-U_1(r_2^*)-\delta(k)+1}{k-1}\\ &= U_2(r_2^*)-U_1(r_2^*)-\delta(k)+1. \qquad (3.34)\end{aligned}$$

From (3.32) and (3.34),

$$\begin{aligned}\sum_{j=1}^{m} wt(\boldsymbol{y}'_j)+(k-1)\cdot(t'-t_1(r_2^*)) &< (U_1(r_2^*)+\delta(k)-1)+(U_2(r_2^*)-U_1(r_2^*)-\delta(k)+1)\\ &= U_2(r_2^*). \qquad (3.35)\end{aligned}$$

From (3.33) and (3.35),

$$\sum_{j=1}^{m} wt(\boldsymbol{y}''_j) < U_2(r_2^*).$$

Therefore, Corollary 2 implies that the next $(t'+1)$-th write operation for any new data can be executed.

Therefore, $(t_1(r_2^*)+t_2(r_2^*))$ write operations for any data sequence of length $(t_1(r_2^*)+t_2(r_2^*))$ can be carried out by applying rule 1 or rule 2. □

From the definitions,

$$\begin{aligned}t_1(r_2^*)+t_2(r_2^*) &= \left\lceil\frac{U_1(r_2^*)}{\delta(k)}\right\rceil+\left\lceil\frac{U_2(r_2^*)-U_1(r_2^*)-\delta(k)+1}{k-1}\right\rceil\\ &\geq \frac{U_1(r_2^*)}{\delta(k)}+\frac{U_2(r_2^*)-U_1(r_2^*)-\delta(k)+1}{k-1}\\ &> \frac{U_{lb1}(r_2^*)}{\delta(k)}+\frac{k(q-1)+\delta(k)-2-\delta(k)+1}{k-1}\\ &= \frac{U_{lb1}(r_2^*)}{\delta(k)}+\frac{k(q-1)-1}{k-1}\\ &> \frac{U_{lb1}(R_2+1)}{\delta(k)}+\frac{k(q-1)-1}{k-1}. \qquad (3.36)\end{aligned}$$

We define

$$t_{lb2}^* = \frac{U_{lb1}(R_2+1)}{\delta(k)}+\frac{k(q-1)-1}{k-1}.$$

Let t_{w2}^* be the worst-case number of writes by I-ILIFC(n,k,q,r_2^*). From (3.36), $t_{w2}^* \geq t_1(r_2^*)+t_2(r_2^*) > t_{lb2}^*$. Therefore, t_{lb2}^* is the lower bound on t_{w2}^*. t_{lb2}^* is as follows. If k is even,

$$t_{lb2}^* = \frac{2}{k+2}\left(n-k^2+\frac{k^3-6k^2+2k+4}{2k(k-1)}\right)(q-1)+\frac{k^2-6k+4}{(k-1)(k+2)}. \qquad (3.37)$$

If k is odd,

$$t_{lb2}^* = \frac{2}{k+3}\left(n-k^2+\frac{k^3-4k^2+k+6}{2(k+1)(k-1)}\right)(q-1)+\frac{k^2-7k+4}{(k+3)(k-1)}. \qquad (3.38)$$

Table 3.1: Our lower bounds and experimental results for I-ILIFC$(n, 4, 4, r_2^*)$

n	22	38	54	70	86	102
Lower bound	4.9	20.9	36.9	52.9	68.9	84.9
Experimental result	14	25	43	61	73	91

If $t_{lb2}^* > t_{ub}$, the worst-case performance of I-ILIFC(n, k, q, r_2^*) is better than that of ILIFC(n, k, q). For $k \geq 4$, we can show that $t_{lb2}^* > t_{ub}$ if and only if $n > p_2$. From (3.1), (3.37), and (3.38), the threshold p_2 is as follows.

$$p_2 = \begin{cases} \frac{2k^4-3k^3+6k^2-2k-4}{(k-1)(k-2)} - \frac{k(k^2-6k+4)}{(k-1)(k-2)(q-1)} & (k \text{ is even}) \\ \frac{k(2k^4-k^3+2k^2-k-6)}{(k+1)(k-1)(k-3)} - \frac{k(k^2-7k+4)}{(k-1)(k-3)(q-1)} & (k \text{ is odd}) \end{cases}. \tag{3.39}$$

From (3.18) and (3.39),

$$p_1 - p_2 = \begin{cases} \frac{k^3(q-1)-3k^2+2k}{(k-1)(k-2)(q-1)} & (k \text{ is even}) \\ \frac{(k^4+2k^3+k^2)(q-1)-3k^3-2k^2+k}{(k+1)(k-1)(k-3)(q-1)} & (k \text{ is odd}) \end{cases}.$$

Therefore, for $k \geq 4$, we have $p_1 - p_2 > 0$, that is, $p_1 > p_2$. This result shows that I-ILIFC(n, k, q, r_2^*) improves the worst-case performance of ILIFC(n, k, q) also for $p_2 < n \leq p_1$.

Table 3.1 shows our lower bounds on I-ILIFC$(n, 4, 4, r_2^*)$ and experimental results for some values of n. In our experiments for this dissertation, the write operation, in which two bits randomly selected from four data bits are changed, was repeated until 100,000 block erasures took place, and the minimum number of writes was calculated. Therefore, the worst-case number of writes may be less than the experimental result.

3.5 Asymptotic analysis

In this section, we use our lower bound on the number of writes to analyze the asymptotic performance for $q = 2$.

In this dissertation, multiple data bits can be changed by each write. Therefore, the I-ILIFC in this dissertation is actually a write-once memory (WOM) code instead of a flash code. As stated in Chapter 1, WOM code is a coding scheme that permits writing data bits into binary cells several times without decreasing the levels, which is exactly the I-ILIFC in this dissertation when $q = 2$.

We compare the asymptotic performance of the I-ILIFC and the WOM code. The sum-rate of the WOM code that stores k data bits in n binary cells t times is kt/n. Then,

$$\frac{kt}{n} \leq \log_2(t+1) \tag{3.40}$$

For fixed k, let t_n be the maximum integer t that satisfies (3.40) for each n. Then t_n is an upper bound on the worst-case number of writes for the code rate k/n.

Let $q = 2$. From (3.37) and (3.38), for fixed k, there exists α such that $t_{lb2}^* < \alpha n$ for sufficiently large n. Therefore,

$$\frac{t_n}{t_{lb2}^*} = \frac{t_n/n}{t_{lb2}^*/n} > \frac{t_n/n}{\alpha} \approx \frac{\log_2(t+1)/k}{\alpha} \tag{3.41}$$

As n increases, t_n also increases. That is, from (3.41), t_{lb2}^* is much smaller than t_n even if the code length n goes to infinity. Unfortunately, it can be seen that the asymptotic performance of the I-ILIFC is not better than that of the WOM code.

3.6 Conclusion

We have presented our derivation of the lower bound on the number of write operations by the I-ILIFC and specified the threshold for the code length that determines whether the I-ILIFC improves the worst-case performance of the ILIFC. The results show that the I-ILIFC performance is better than that of the ILIFC in the worst case if the code length is sufficiently large. Additionally, we have considered whether another rule should be applied if the rule selected during encoding cannot be applied. Then, we have derived another lower bound thereon. Consequently, the threshold was able to be made smaller than that in the first lower bound.

Chapter 4

Parallel Random I/O Codes

As described in Chapter 1, in multilevel flash memory, on the average, more than a single read threshold is required to read a single logical page. In order to solve this problem, random I/O (RIO) code was proposed by Sharon and Alrod [11]. This chapter describes write-once memory (WOM) codes [11], of which construction is equivalent to that of RIO codes. RIO codes and parallel RIO (P-RIO) codes [13], which is a family of RIO codes, are also described. Then, our construction of P-RIO codes using coset coding is described.

4.1 Preliminaries

We first present some preliminary definitions and notation. For a positive integer n, we define $[n] = \{1, \ldots, n\}$. In addition, for two vectors $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_n)$, we denote $\boldsymbol{x} \leq \boldsymbol{y}$ if $x_i \leq y_i$ for any $i \in [n]$. For a binary vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, we define $I(\boldsymbol{x}) = \{i \mid i \in [n], x_i = 1\}$. For a positive integer n and any $j \in [n]$, we define $\boldsymbol{e}_j \in \{0,1\}^n$ such that $I(\boldsymbol{e}_j) = \{j\}$.

4.1.1 WOM Code

In a WOM, a cell represents one of two levels $\{0, 1\}$, and the level of each cell can only increase. Rivest and Shamir proposed a WOM code that allows writing into WOM multiple times [12].

Definition 1. *An $[n, l, t]$ WOM code is a coding scheme that permits the writing of l data bits into n binary cells t times without decreasing the levels. This scheme is defined by t pairs of encoding and decoding maps $(\mathcal{E}_i, \mathcal{D}_i)$ for $i \in [t]$. Encoding map $\mathcal{E}_i$ is defined by*

$$\mathcal{E}_i : \{0,1\}^l \times Im(\mathcal{E}_{i-1}) \to \{0,1\}^n,$$

where $Im(\mathcal{E}_0) = \{(0, \ldots, 0)\}$. For all $(\boldsymbol{d}, \boldsymbol{c}) \in \{0,1\}^l \times Im(\mathcal{E}_{i-1})$ with $i \in [t]$, $\boldsymbol{c} \leq \mathcal{E}_i(\boldsymbol{d}, \boldsymbol{c})$. Further, decoding map $\mathcal{D}_i$ is defined by

$$\mathcal{D}_i : Im(\mathcal{E}_i) \to \{0,1\}^l,$$

such that $\mathcal{D}_i(\mathcal{E}_i(\boldsymbol{d}, \boldsymbol{c})) = \boldsymbol{d}$ for all $(\boldsymbol{d}, \boldsymbol{c}) \in \{0,1\}^l \times Im(\mathcal{E}_{i-1})$ with $i \in [t]$.

Example 1. *Rivest and Shamir presented the $[3, 2, 2]$ WOM code, which is shown in Table 4.1 [12]. For example, when the first and second data are 10 and 01, respectively, from Table 4.1, $\mathcal{E}_1(10, 000) = 010, \mathcal{E}_2(01, 010) = 011$.*

Table 4.1: $[3, 2, 2]$ WOM code [12]

Data bits	First write	Second write
00	000	111
01	100	011
10	010	101
11	001	110

4.1.2 Coset Coding

In this dissertation, a linear binary code of length n and dimension k is referred to as an (n, k) code. Coset coding with an (n, k) code C is used to construct an $[n, n-k, t]$ WOM code, where t depends on C [14].

Let H be the parity check matrix of C. For all $(\boldsymbol{d}, \boldsymbol{c}) \in \{0,1\}^{n-k} \times Im(\mathcal{E}_{i-1})$ with $i \in [t]$, encoding map $\mathcal{E}_i$ is as follows.

$$\mathcal{E}_i(\boldsymbol{d}, \boldsymbol{c}) = \boldsymbol{c} + \boldsymbol{x},$$

where $\boldsymbol{x} \in \{\boldsymbol{v} \in \{0,1\}^n \mid \boldsymbol{v}H^T = \boldsymbol{d} - \boldsymbol{c}H^T, I(\boldsymbol{c}) \cap I(\boldsymbol{v}) = \emptyset\}$. Further, for all $\boldsymbol{c} \in Im(\mathcal{E}_i)$ with $i \in [t]$, the decoding map $\mathcal{D}_i$ is

$$\mathcal{D}_i(\boldsymbol{c}) = \boldsymbol{c}H^T.$$

The following theorem was proven in [14].

Theorem 9. *When using coset coding with $(7, 4)$ Hamming code, the $[7, 3, 3]$ WOM code can be constructed. Furthermore, a $[2^r - 1, r, 2^{r-2} + 2]$ WOM code can be constructed via coset coding with a $(2^r - 1, 2^r - r - 1)$ Hamming code for $r \geq 4$.*

4.1.3 RIO Code

In this chapter, it is assumed that each cell of the flash memory represents one of $(t+1)$ levels $\{0, 1, \ldots, t\}$. These levels are distinguished by t read thresholds. For each $i \in [t]$, we denote a read threshold between levels $(i-1)$ and i by the i-th threshold. Let $\boldsymbol{v} = (v_1, \ldots, v_n) \in \{0, \ldots, t\}^n$ be the state of n cells. The operation of reading the i-th threshold from $\boldsymbol{v}$ is denoted by $RT_i(\boldsymbol{v})$ and is defined as follows:

$$RT_i(\boldsymbol{v}) = (r_1, \ldots, r_n) \in \{0,1\}^n,$$

where for each $j \in [n]$, r_j is 1 if $v_j \geq i$, and 0 otherwise. The next proposition is obtained from the definition of this operation.

Proposition 1. *For any $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_t \in \{0,1\}^n$ such that $\boldsymbol{c}_1 \leq \boldsymbol{c}_2 \leq \cdots \leq \boldsymbol{c}_t$, the following property holds: For each $i \in [t]$,*

$$RT_{t+1-i}\left(\sum_{j=1}^{t} \boldsymbol{c}_j\right) = \boldsymbol{c}_i.$$

RIO code is a coding scheme that permits reading one page using a single read threshold [11]. In RIO codes, t pages are stored in $(t+1)$-level cells as follows. For each $i \in [t]$, if the codeword of the $(i-1)$-th page is $\boldsymbol{c}_{i-1} \in \{0,1\}^n$, the data of the i-th page is encoded into a codeword $\boldsymbol{c}_i \in \{0,1\}^n$ such that $\boldsymbol{c}_{i-1} \leq \boldsymbol{c}_i$, where $\boldsymbol{c}_0 = (0, \ldots, 0)$. Then, the state of the cells is $\sum_{j=1}^{t} \boldsymbol{c}_j \in \{0, \ldots, t\}^n$. From Proposition 1, the i-th page is read using the $(t+1-i)$-th threshold from the state of the cells.

Table 4.2: $[3, 2, 2]$ RIO code

Data of page 2 \ Data of page 1	00	01	10	11
00	000	211	121	112
01	100	200	021	012
10	010	201	020	102
11	001	210	120	002

Definition 2. *An $[n, l, t]$ RIO code is a coding scheme in which t pages of l data bits are stored into n $(t+1)$-level cells such that each page is read using a single read threshold. This scheme is defined by t pairs of encoding and decoding maps $(\mathcal{E}_i, \mathcal{D}_i)$ for $i \in [t]$. Encoding map $\mathcal{E}_i$ is defined by*

$$\mathcal{E}_i : \{0,1\}^l \times Im(\mathcal{E}_{i-1}) \to \{0,1\}^n,$$

where $Im(\mathcal{E}_0) = \{(0, \dots, 0)\}$. For all $(\boldsymbol{d}_i, \boldsymbol{c}_{i-1}) \in \{0,1\}^l \times Im(\mathcal{E}_{i-1})$, $\boldsymbol{c}_i = \mathcal{E}_i(\boldsymbol{d}_i, \boldsymbol{c}_{i-1}) \geq \boldsymbol{c}_{i-1}$ is satisfied. The cell state is $\sum_{j=1}^{t} \boldsymbol{c}_j$. Further, decoding map $\mathcal{D}_i$ is defined by

$$\mathcal{D}_i : Im(\mathcal{E}_i) \to \{0,1\}^l,$$

such that $\mathcal{D}_i(\boldsymbol{c}_i) = \mathcal{D}_i(\mathcal{E}_i(\boldsymbol{d}_i, \boldsymbol{c}_{i-1})) = \boldsymbol{d}_i$ for all $(\boldsymbol{d}_i, \boldsymbol{c}_{i-1}) \in \{0,1\}^l \times Im(\mathcal{E}_{i-1})$. The data $\boldsymbol{d}_i$ is read using $\boldsymbol{c}_i = RT_{t+1-i}(\sum_{j=1}^{t} \boldsymbol{c}_j)$.

From Definitions 1 and 2, the construction of an $[n, l, t]$ RIO code is clearly equivalent to that of an $[n, l, t]$ WOM code [11].

Example 2. *The $[3, 2, 2]$ RIO code based on the $[3, 2, 2]$ WOM code in Example 1 is shown in Table 4.2. As an example, let the data of pages 1 and 2 be 10 and 01, respectively. From Example 1, the data 10 of page 1 are encoded into $\boldsymbol{c}_1 = 010$, and then the data 01 of page 2 are encoded into $\boldsymbol{c}_2 = 011$ such that $\boldsymbol{c}_1 \leq \boldsymbol{c}_2$. The state of the cells is $\boldsymbol{c}_1 + \boldsymbol{c}_2 = 021$.*

4.1.4 P-RIO Code

In RIO codes, the encoding of each page depends on the data of the page and the previous pages. However, if the data of all pages are known in advance, the information of those data can be leveraged to encode each page. P-RIO code is a family of RIO codes in which the encoding of each page depends on the data of all pages [13].

Definition 3. *An $[n, l, t]$ P-RIO code is an $[n, l, t]$ RIO code in which the encoding of each page is performed in parallel. This P-RIO code is defined by encoding map $\mathcal{E}$ and t decoding maps $\mathcal{D}_i$, $i \in [t]$. The encoding map $\mathcal{E}$ is defined by*

$$\mathcal{E} : \prod_{j=1}^{t} \{0,1\}^l \to \prod_{j=1}^{t} \{0,1\}^n.$$

For all $(\boldsymbol{d}_1, \dots, \boldsymbol{d}_t) \in \prod_{j=1}^{t} \{0,1\}^l$, $\boldsymbol{c}_1 \leq \boldsymbol{c}_2 \leq \cdots \leq \boldsymbol{c}_t$ is satisfied, where $(\boldsymbol{c}_1, \dots, \boldsymbol{c}_t) = \mathcal{E}(\boldsymbol{d}_1, \dots, \boldsymbol{d}_t)$. For each $i \in [t]$, decoding map $\mathcal{D}_i$ is defined by

$$\mathcal{D}_i : \{0,1\}^n \to \{0,1\}^l$$

such that $\mathcal{D}_i(\boldsymbol{c}_i) = \boldsymbol{d}_i$ for all $(\boldsymbol{d}_1, \dots, \boldsymbol{d}_t) \in \prod_{j=1}^{t} \{0,1\}^l$, where $(\boldsymbol{c}_1, \dots, \boldsymbol{c}_t) = \mathcal{E}(\boldsymbol{d}_1, \dots, \boldsymbol{d}_t)$.

An algorithm to construct P-RIO codes was proposed in a previous study [13], and was run to yield P-RIO codes in which two pages are stored with moderate code lengths. These codes have parameters for which RIO codes do not exist [13]. In this dissertation, we use coset coding to construct P-RIO codes. When using coset coding with $(7, 4)$ and $(15, 11)$ Hamming codes, from Theorem 9, $[7, 3, 3]$ RIO code and $[15, 4, 6]$ RIO code based on WOM codes are constructed, respectively. Then, we leverage the information of the data of all pages to construct P-RIO codes in which more pages are stored than these RIO codes. Note that the number of levels that each cell can represent must be increased when the number of pages increases.

4.2 Construction of P-RIO Codes using Coset Coding

Prior to the construction of P-RIO codes using coset coding, we discuss several properties.

4.2.1 Properties

We have the following theorem:

Theorem 10. *Let H be the patiry check matrix of (n, k) code C. A sufficient condition that ensures the construction of an $[n, n-k, t]$ P-RIO code using coset coding with code C is as follows: For any $\boldsymbol{s}_1, \dots, \boldsymbol{s}_t \in \{0,1\}^{n-k}$, there exist $\boldsymbol{x}_1, \dots, \boldsymbol{x}_t \in \{0,1\}^n$ that satisfy the following conditions:*

1. *For all $i \in [t]$, $\boldsymbol{x}_i H^T = \boldsymbol{s}_i$;*

2. *For all $i, i' \in [t]$ with $i \neq i'$, $I(\boldsymbol{x}_i) \cap I(\boldsymbol{x}_{i'}) = \emptyset$.*

Proof. For any $\boldsymbol{d}_1, \dots, \boldsymbol{d}_t \in \{0,1\}^{n-k}$, we define $\boldsymbol{s}_i = \boldsymbol{d}_i - \boldsymbol{d}_{i-1}$ for each $i \in [t]$, where $\boldsymbol{d}_0 = (0, \dots, 0)$. Suppose that there exist $\boldsymbol{x}_1, \dots, \boldsymbol{x}_t \in \{0,1\}^n$ that satisfy the above conditions for $\boldsymbol{s}_1, \dots, \boldsymbol{s}_t$. We define $\boldsymbol{c}_i = \sum_{j=1}^{i} \boldsymbol{x}_j$ for each $i \in [t]$. Then, we have $\boldsymbol{c}_i H^T = \boldsymbol{d}_i$ for each $i \in [t]$ and $\boldsymbol{c}_1 \leq \boldsymbol{c}_2 \leq \cdots \leq \boldsymbol{c}_t$. Therefore, an $[n, n-k, t]$ P-RIO code can be constructed, where $\mathcal{E}(\boldsymbol{d}_1, \dots, \boldsymbol{d}_t) = (\boldsymbol{c}_1, \dots, \boldsymbol{c}_t)$ and $\mathcal{D}(\boldsymbol{c}_i) = \boldsymbol{c}_i H^T$. □

For $r \geq 3$, we denote the parity check matrix of the $(2^r - 1, 2^r - r - 1)$ Hamming code by

$$H = \begin{pmatrix} \boldsymbol{h}_1^T & \boldsymbol{h}_2^T & \cdots & \boldsymbol{h}_{2^r-1}^T \end{pmatrix},$$

where $\boldsymbol{h}_j \in \{0,1\}^r \setminus \{(0, \dots, 0)\}$ for all $j \in [2^r - 1]$ and $\boldsymbol{h}_j \neq \boldsymbol{h}_{j'}$ for all $j, j' \in [2^r - 1]$ with $j \neq j'$. Note that for any $\boldsymbol{s} \in \{0,1\}^r \setminus \{(0, \dots, 0)\}$, there exists a unique integer $j \in [2^r - 1]$ such that $\boldsymbol{e}_j H^T = \boldsymbol{h}_j = \boldsymbol{s}$ and there exist $(2^{r-1} - 1)$ pairs of $j_1 \in [2^r - 1]$ and $j_2 \in [2^r - 1]$ such that $(\boldsymbol{e}_{j_1} + \boldsymbol{e}_{j_2}) H^T = \boldsymbol{h}_{j_1} + \boldsymbol{h}_{j_2} = \boldsymbol{s}$. For any $\boldsymbol{s} \in \{0,1\}^r \setminus \{(0, \dots, 0)\}$, we define $V(\boldsymbol{s}) = \{I(\boldsymbol{v}) \subseteq [2^r - 1] \mid \boldsymbol{v} \in \{0,1\}^{2^r-1}, \boldsymbol{v} H^T = \boldsymbol{s}, |I(\boldsymbol{v})| \leq 2\}$. We have the following theorem:

Theorem 11. *Let r be an integer such that $r \geq 3$. For any $\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3, \boldsymbol{s}_4 \in \{0,1\}^r \setminus \{(0, \dots, 0)\}$, where $\boldsymbol{s}_i \neq \boldsymbol{s}_{i'}$ for all $i, i' \in [4]$ with $i \neq i'$, we have the permutation σ of $[2^r - 1]$, which satisfies the following conditions:*

Case 1: $r \geq 4$ and $\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3,$ and $\boldsymbol{s}_4$ are linearly independent.

$$\begin{aligned}
& V(\boldsymbol{s}_1) \\
= \;& \{\{\sigma(1)\}, \{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+1)\}, \\
& \{\sigma(8i+2), \sigma(8i+3)\}, \{\sigma(8i+4), \sigma(8i+5)\}, \{\sigma(8i+6), \sigma(8i+7)\}\}), \\
& V(\boldsymbol{s}_2) \\
= \;& \{\{\sigma(2)\}, \{\sigma(1), \sigma(3)\}, \{\sigma(4), \sigma(6)\}, \{\sigma(5), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+2)\}, \\
& \{\sigma(8i+1), \sigma(8i+3)\}, \{\sigma(8i+4), \sigma(8i+6)\}, \{\sigma(8i+5), \sigma(8i+7)\}\}), \\
& V(\boldsymbol{s}_3) \\
= \;& \{\{\sigma(4)\}, \{\sigma(1), \sigma(5)\}, \{\sigma(2), \sigma(6)\}, \{\sigma(3), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+4)\}, \\
& \{\sigma(8i+1), \sigma(8i+5)\}, \{\sigma(8i+2), \sigma(8i+6)\}, \{\sigma(8i+3), \sigma(8i+7)\}\}), \\
& V(\boldsymbol{s}_4) \\
= \;& \{\{\sigma(8)\}, \{\sigma(1), \sigma(9)\}, \{\sigma(2), \sigma(10)\}, \{\sigma(3), \sigma(11)\}, \{\sigma(4), \sigma(12)\}, \{\sigma(5), \sigma(13)\}, \\
& \{\sigma(6), \sigma(14)\}, \{\sigma(7), \sigma(15)\}\} \cup (\bigcup_{i=1}^{2^{r-4}-1} \{\{\sigma(16i), \sigma(16i+8)\}, \{\sigma(16i+1), \sigma(16i+9)\}, \\
& \{\sigma(16i+2), \sigma(16i+10)\}, \{\sigma(16i+3), \sigma(16i+11)\}, \{\sigma(16i+4), \sigma(16i+12)\}, \\
& \{\sigma(16i+5), \sigma(16i+13)\}, \{\sigma(16i+6), \sigma(16i+14)\}, \{\sigma(16i+7), \sigma(16i+15)\}\}).
\end{aligned}$$

Case 2: $\boldsymbol{s}_1 + \boldsymbol{s}_2 = \boldsymbol{s}_3$.

$$\begin{aligned}
& V(\boldsymbol{s}_1) \\
= \;& \{\{\sigma(1)\}, \{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+1)\}, \\
& \{\sigma(8i+2), \sigma(8i+3)\}, \{\sigma(8i+4), \sigma(8i+5)\}, \{\sigma(8i+6), \sigma(8i+7)\}\}), \\
& V(\boldsymbol{s}_2) \\
= \;& \{\{\sigma(2)\}, \{\sigma(1), \sigma(3)\}, \{\sigma(4), \sigma(6)\}, \{\sigma(5), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+2)\}, \\
& \{\sigma(8i+1), \sigma(8i+3)\}, \{\sigma(8i+4), \sigma(8i+6)\}, \{\sigma(8i+5), \sigma(8i+7)\}\}), \\
& V(\boldsymbol{s}_3) \\
= \;& \{\{\sigma(3)\}, \{\sigma(1), \sigma(2)\}, \{\sigma(4), \sigma(7)\}, \{\sigma(5), \sigma(6)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+3)\}, \\
& \{\sigma(8i+1), \sigma(8i+2)\}, \{\sigma(8i+4), \sigma(8i+7)\}, \{\sigma(8i+5), \sigma(8i+6)\}\}), \\
& V(\boldsymbol{s}_4) \\
= \;& \{\{\sigma(4)\}, \{\sigma(1), \sigma(5)\}, \{\sigma(2), \sigma(6)\}, \{\sigma(3), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+4)\}, \\
& \{\sigma(8i+1), \sigma(8i+5)\}, \{\sigma(8i+2), \sigma(8i+6)\}, \{\sigma(8i+3), \sigma(8i+7)\}\}).
\end{aligned}$$

Case 3: $\boldsymbol{s}_1 + \boldsymbol{s}_2 = \boldsymbol{s}_3 + \boldsymbol{s}_4$.

$$\begin{aligned}
& V(\boldsymbol{s}_1) \\
= \; & \{\{\sigma(1)\}, \{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+1)\}, \\
& \{\sigma(8i+2), \sigma(8i+3)\}, \{\sigma(8i+4), \sigma(8i+5)\}, \{\sigma(8i+6), \sigma(8i+7)\}\}), \\
& V(\boldsymbol{s}_2) \\
= \; & \{\{\sigma(2)\}, \{\sigma(1), \sigma(3)\}, \{\sigma(4), \sigma(6)\}, \{\sigma(5), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+2)\}, \\
& \{\sigma(8i+1), \sigma(8i+3)\}, \{\sigma(8i+4), \sigma(8i+6)\}, \{\sigma(8i+5), \sigma(8i+7)\}\}), \\
& V(\boldsymbol{s}_3) \\
= \; & \{\{\sigma(4)\}, \{\sigma(1), \sigma(5)\}, \{\sigma(2), \sigma(6)\}, \{\sigma(3), \sigma(7)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+4)\}, \\
& \{\sigma(8i+1), \sigma(8i+5)\}, \{\sigma(8i+2), \sigma(8i+6)\}, \{\sigma(8i+3), \sigma(8i+7)\}\}), \\
& V(\boldsymbol{s}_4) \\
= \; & \{\{\sigma(7)\}, \{\sigma(1), \sigma(6)\}, \{\sigma(2), \sigma(5)\}, \{\sigma(3), \sigma(4)\}\} \cup (\bigcup_{i=1}^{2^{r-3}-1} \{\{\sigma(8i), \sigma(8i+7)\}, \\
& \{\sigma(8i+1), \sigma(8i+6)\}, \{\sigma(8i+2), \sigma(8i+5)\}, \{\sigma(8i+3), \sigma(8i+4)\}\}).
\end{aligned}$$

Example 3. *Let* $r = 4$ *and*

$$\begin{aligned}
H &= \begin{pmatrix} \boldsymbol{h}_1^T & \boldsymbol{h}_2^T & \cdots & \boldsymbol{h}_{15}^T \end{pmatrix} \\
&= \begin{pmatrix} 1&0&1&0&1&0&1&0&1&0&1&0&1&0&1 \\ 0&1&1&0&0&1&1&0&0&1&1&0&0&1&1 \\ 0&0&0&1&1&1&1&0&0&0&0&1&1&1&1 \\ 0&0&0&0&0&0&0&1&1&1&1&1&1&1&1 \end{pmatrix}.
\end{aligned}$$

Suppose $\boldsymbol{s}_1 = 0110$, $\boldsymbol{s}_2 = 1100$, $\boldsymbol{s}_3 = 1101$, *and* $\boldsymbol{s}_4 = 0010$. *Note that* $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_4$ *are linearly independent (Case 1). Let* $\sigma(1) = 6$, *then*

$$\boldsymbol{e}_{\sigma(1)} H^T = \boldsymbol{h}_6 = 0110 = \boldsymbol{s}_1.$$

Let $\sigma(2) = 3$, *then*

$$\boldsymbol{e}_{\sigma(2)} H^T = \boldsymbol{h}_3 = 1100 = \boldsymbol{s}_2.$$

Let $\sigma(3) = 5$, *then*

$$(\boldsymbol{e}_{\sigma(2)} + \boldsymbol{e}_{\sigma(3)}) H^T = \boldsymbol{h}_3 + \boldsymbol{h}_5 = 0110 = \boldsymbol{s}_1,$$

$$(\boldsymbol{e}_{\sigma(1)} + \boldsymbol{e}_{\sigma(3)}) H^T = \boldsymbol{h}_6 + \boldsymbol{h}_5 = 1100 = \boldsymbol{s}_2,$$

and

$$\boldsymbol{e}_{\sigma(3)} H^T = \boldsymbol{h}_5 = 1010 \neq \boldsymbol{s}_3.$$

Let $\sigma(4) = 11$, *then*

$$\boldsymbol{e}_{\sigma(4)} H^T = \boldsymbol{h}_{11} = 1101 = \boldsymbol{s}_3.$$

Let $\sigma(5) = 13$, *then*

$$(\boldsymbol{e}_{\sigma(4)} + \boldsymbol{e}_{\sigma(5)}) H^T = \boldsymbol{h}_{11} + \boldsymbol{h}_{13} = 0110 = \boldsymbol{s}_1$$

and
$$(\boldsymbol{e}_{\sigma(1)}+\boldsymbol{e}_{\sigma(5)})H^T=\boldsymbol{h}_6+\boldsymbol{h}_{13}=1101=\boldsymbol{s}_3.$$
Let $\sigma(6)=8$*, then*
$$(\boldsymbol{e}_{\sigma(4)}+\boldsymbol{e}_{\sigma(6)})H^T=\boldsymbol{h}_{11}+\boldsymbol{h}_8=1100=\boldsymbol{s}_2$$
and
$$(\boldsymbol{e}_{\sigma(2)}+\boldsymbol{e}_{\sigma(6)})H^T=\boldsymbol{h}_3+\boldsymbol{h}_8=1101=\boldsymbol{s}_3.$$
Let $\sigma(7)=14$*, then*
$$(\boldsymbol{e}_{\sigma(6)}+\boldsymbol{e}_{\sigma(7)})H^T=\boldsymbol{h}_8+\boldsymbol{h}_{14}=0110=\boldsymbol{s}_1,$$
$$(\boldsymbol{e}_{\sigma(5)}+\boldsymbol{e}_{\sigma(7)})H^T=\boldsymbol{h}_{13}+\boldsymbol{h}_{14}=1100=\boldsymbol{s}_2,$$
$$(\boldsymbol{e}_{\sigma(3)}+\boldsymbol{e}_{\sigma(7)})H^T=\boldsymbol{h}_5+\boldsymbol{h}_{14}=1101=\boldsymbol{s}_3,$$
$$\boldsymbol{e}_{\sigma(5)}H^T=\boldsymbol{h}_{13}=1011\neq\boldsymbol{s}_4,$$
$$\boldsymbol{e}_{\sigma(6)}H^T=\boldsymbol{h}_8=0001\neq\boldsymbol{s}_4,$$
and
$$\boldsymbol{e}_{\sigma(7)}H^T=\boldsymbol{h}_{14}=0111\neq\boldsymbol{s}_4.$$
Let $\sigma(8)=4$*, then*
$$\boldsymbol{e}_{\sigma(8)}H^T=\boldsymbol{h}_4=0010=\boldsymbol{s}_4.$$
Similarly, we can obtain $\sigma(9),\ldots,\sigma(15)$*. The permutation* σ *is obtained as follows:*
$$(\sigma(1),\ldots,\sigma(15))=(6,3,5,11,13,8,14,4,2,7,1,15,9,12,10).$$
$V(\boldsymbol{s}_1)$ *is obtained as follows:*
$$\begin{aligned}V(\boldsymbol{s}_1)&=\{\{\sigma(1)\},\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\},\{\sigma(8),\sigma(9)\},\{\sigma(10),\sigma(11)\},\\&\quad\{\sigma(12),\sigma(13)\},\{\sigma(14),\sigma(15)\}\}\\&=\{\{6\},\{3,5\},\{11,13\},\{8,14\},\{4,2\},\{7,1\},\{15,9\},\{12,10\}\}.\end{aligned}$$
Similarly, $V(\boldsymbol{s}_2),\ldots,V(\boldsymbol{s}_4)$ *are obtained as follows:*
$$\begin{aligned}V(\boldsymbol{s}_2)&=\{\{3\},\{6,5\},\{11,8\},\{13,14\},\{4,7\},\{2,1\},\{15,12\},\{9,10\}\},\\V(\boldsymbol{s}_3)&=\{\{11\},\{6,13\},\{3,8\},\{5,14\},\{4,15\},\{2,9\},\{7,12\},\{1,10\}\},\\V(\boldsymbol{s}_4)&=\{\{4\},\{6,2\},\{3,7\},\{5,1\},\{11,15\},\{13,9\},\{8,12\},\{14,10\}\}.\end{aligned}$$
Proof of Theorem 11. Suppose that $r\geq 4$ and $\boldsymbol{s}_1,\boldsymbol{s}_2,\boldsymbol{s}_3$, and $\boldsymbol{s}_4$ are linearly independent. Let $\sigma(1)\in[2^r-1]$ such that
$$\boldsymbol{e}_{\sigma(1)}H^T=\boldsymbol{h}_{\sigma(1)}=\boldsymbol{s}_1.$$
Let $\sigma(2)\in[2^r-1]\setminus\{\sigma(1)\}$ such that
$$\boldsymbol{e}_{\sigma(2)}H^T=\boldsymbol{h}_{\sigma(2)}=\boldsymbol{s}_2.$$
Let $\sigma(3)\in[2^r-1]\setminus\{\sigma(1),\sigma(2)\}$ such that
$$(\boldsymbol{e}_{\sigma(2)}+\boldsymbol{e}_{\sigma(3)})H^T=\boldsymbol{h}_{\sigma(2)}+\boldsymbol{h}_{\sigma(3)}=\boldsymbol{s}_1.$$
Then,
$$(\boldsymbol{e}_{\sigma(1)}+\boldsymbol{e}_{\sigma(3)})H^T=\boldsymbol{h}_{\sigma(1)}+\boldsymbol{h}_{\sigma(3)}=\boldsymbol{h}_{\sigma(1)}+\boldsymbol{h}_{\sigma(2)}+(\boldsymbol{h}_{\sigma(2)}+\boldsymbol{h}_{\sigma(3)})=\boldsymbol{s}_1+\boldsymbol{s}_2+\boldsymbol{s}_1=\boldsymbol{s}_2.$$

In addition,

$$\boldsymbol{e}_{\sigma(3)}H^T = \boldsymbol{h}_{\sigma(3)} = \boldsymbol{h}_{\sigma(1)} + (\boldsymbol{h}_{\sigma(1)} + \boldsymbol{h}_{\sigma(3)}) = \boldsymbol{s}_1 + \boldsymbol{s}_2 \neq \boldsymbol{s}_3.$$

Let $\sigma(4) \in [2^r - 1] \setminus \{\sigma(1), \sigma(2), \sigma(3)\}$ such that

$$\boldsymbol{e}_{\sigma(4)}H^T = \boldsymbol{h}_{\sigma(4)} = \boldsymbol{s}_3.$$

Let $\sigma(5) \in [2^r - 1] \setminus \{\sigma(1), \ldots, \sigma(4)\}$ such that

$$(\boldsymbol{e}_{\sigma(4)} + \boldsymbol{e}_{\sigma(5)})H^T = \boldsymbol{h}_{\sigma(4)} + \boldsymbol{h}_{\sigma(5)} = \boldsymbol{s}_1.$$

Then,

$$(\boldsymbol{e}_{\sigma(1)} + \boldsymbol{e}_{\sigma(5)})H^T = \boldsymbol{h}_{\sigma(1)} + \boldsymbol{h}_{\sigma(5)} = \boldsymbol{h}_{\sigma(1)} + \boldsymbol{h}_{\sigma(4)} + (\boldsymbol{h}_{\sigma(4)} + \boldsymbol{h}_{\sigma(5)}) = \boldsymbol{s}_1 + \boldsymbol{s}_3 + \boldsymbol{s}_1 = \boldsymbol{s}_3.$$

Let $\sigma(6) \in [2^r - 1] \setminus \{\sigma(1), \ldots, \sigma(5)\}$ such that

$$(\boldsymbol{e}_{\sigma(4)} + \boldsymbol{e}_{\sigma(6)})H^T = \boldsymbol{h}_{\sigma(4)} + \boldsymbol{h}_{\sigma(6)} = \boldsymbol{s}_2.$$

Then,

$$(\boldsymbol{e}_{\sigma(2)} + \boldsymbol{e}_{\sigma(6)})H^T = \boldsymbol{h}_{\sigma(2)} + \boldsymbol{h}_{\sigma(6)} = \boldsymbol{h}_{\sigma(2)} + \boldsymbol{h}_{\sigma(4)} + (\boldsymbol{h}_{\sigma(4)} + \boldsymbol{h}_{\sigma(6)}) = \boldsymbol{s}_2 + \boldsymbol{s}_3 + \boldsymbol{s}_2 = \boldsymbol{s}_3.$$

Let $\sigma(7) \in [2^r - 1] \setminus \{\sigma(1), \ldots, \sigma(6)\}$ such that

$$(\boldsymbol{e}_{\sigma(6)} + \boldsymbol{e}_{\sigma(7)})H^T = \boldsymbol{h}_{\sigma(6)} + \boldsymbol{h}_{\sigma(7)} = \boldsymbol{s}_1.$$

Then,

$$\begin{aligned}(\boldsymbol{e}_{\sigma(5)} + \boldsymbol{e}_{\sigma(7)})H^T &= \boldsymbol{h}_{\sigma(5)} + \boldsymbol{h}_{\sigma(7)} = (\boldsymbol{h}_{\sigma(4)} + \boldsymbol{h}_{\sigma(5)}) + (\boldsymbol{h}_{\sigma(6)} + \boldsymbol{h}_{\sigma(7)}) + (\boldsymbol{h}_{\sigma(4)} + \boldsymbol{h}_{\sigma(6)})\\ &= \boldsymbol{s}_1 + \boldsymbol{s}_1 + \boldsymbol{s}_2 = \boldsymbol{s}_2.\end{aligned}$$

In addition,

$$\begin{aligned}(\boldsymbol{e}_{\sigma(3)} + \boldsymbol{e}_{\sigma(7)})H^T &= \boldsymbol{h}_{\sigma(3)} + \boldsymbol{h}_{\sigma(7)} = (\boldsymbol{h}_{\sigma(2)} + \boldsymbol{h}_{\sigma(3)}) + (\boldsymbol{h}_{\sigma(6)} + \boldsymbol{h}_{\sigma(7)}) + (\boldsymbol{h}_{\sigma(2)} + \boldsymbol{h}_{\sigma(6)})\\ &= \boldsymbol{s}_1 + \boldsymbol{s}_1 + \boldsymbol{s}_3 = \boldsymbol{s}_3.\end{aligned}$$

Here,

$$\begin{aligned}\boldsymbol{e}_{\sigma(5)}H^T &= \boldsymbol{h}_{\sigma(5)} = \boldsymbol{h}_{\sigma(1)} + (\boldsymbol{h}_{\sigma(1)} + \boldsymbol{h}_{\sigma(5)}) = \boldsymbol{s}_1 + \boldsymbol{s}_3 \neq \boldsymbol{s}_4.\\ \boldsymbol{e}_{\sigma(6)}H^T &= \boldsymbol{h}_{\sigma(6)} = \boldsymbol{h}_{\sigma(2)} + (\boldsymbol{h}_{\sigma(2)} + \boldsymbol{h}_{\sigma(6)}) = \boldsymbol{s}_2 + \boldsymbol{s}_3 \neq \boldsymbol{s}_4.\\ \boldsymbol{e}_{\sigma(7)}H^T &= \boldsymbol{h}_{\sigma(7)} = \boldsymbol{h}_{\sigma(1)} + (\boldsymbol{h}_{\sigma(1)} + \boldsymbol{h}_{\sigma(3)}) + (\boldsymbol{h}_{\sigma(3)} + \boldsymbol{h}_{\sigma(7)}) = \boldsymbol{s}_1 + \boldsymbol{s}_2 + \boldsymbol{s}_3 \neq \boldsymbol{s}_4.\end{aligned}$$

Let $\sigma(8) \in [2^r - 1] \setminus \{\sigma(1), \ldots, \sigma(7)\}$ such that

$$\boldsymbol{e}_{\sigma(8)}H^T = \boldsymbol{h}_{\sigma(8)} = \boldsymbol{s}_4.$$

Similarly, we obtain $\sigma(9), \ldots, \sigma(2^r - 1)$.

In the other two cases, we note that $\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3$, and $\boldsymbol{s}_4$ are linearly dependent and obtain the permutation σ that satisfies the conditions in a similar manner. □

4.2.2 Construction of $[7,3,4]$ P-RIO Code

We construct $[7,3,4]$ P-RIO code using coset coding with the $(7,4)$ Hamming code. Let H be the parity check matrix of the $(7,4)$ Hamming code. First, we encode some data of four pages using coset coding.

Example 4. *The parity check matrix H is as follows:*

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Let the data of pages 1–4 be $\boldsymbol{d}_1 = 001$, $\boldsymbol{d}_2 = 111$, $\boldsymbol{d}_3 = 011$, and $\boldsymbol{d}_4 = 010$, respectively. Then, $\boldsymbol{s}_1 = \boldsymbol{d}_1 = 001, \boldsymbol{s}_2 = \boldsymbol{d}_2 - \boldsymbol{d}_1 = 110, \boldsymbol{s}_3 = \boldsymbol{d}_3 - \boldsymbol{d}_2 = 100$, and $\boldsymbol{s}_4 = \boldsymbol{d}_4 - \boldsymbol{d}_3 = 001$. For these $\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3, \boldsymbol{s}_4$, we obtain $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4 \in \{0,1\}^7$ that satisfy the conditions of Theorem 10. For each $i \in [4]$, $V(\boldsymbol{s}_i)$ is as follows:

$$\begin{aligned} V(\boldsymbol{s}_1) = V(\boldsymbol{s}_4) &= \{\{4\},\{1,5\},\{2,6\},\{3,7\}\}, \\ V(\boldsymbol{s}_2) &= \{\{3\},\{1,2\},\{4,7\},\{5,6\}\}, \\ V(\boldsymbol{s}_3) &= \{\{1\},\{2,3\},\{4,5\},\{6,7\}\}. \end{aligned}$$

Let $I(\boldsymbol{x}_1) = \{4\}, I(\boldsymbol{x}_2) = \{3\}, I(\boldsymbol{x}_3) = \{1\}$, and $I(\boldsymbol{x}_4) = \{2,6\}$. That is, $\boldsymbol{x}_1 = \boldsymbol{e}_4, \boldsymbol{x}_2 = \boldsymbol{e}_3, \boldsymbol{x}_3 = \boldsymbol{e}_1$, and $\boldsymbol{x}_4 = \boldsymbol{e}_2 + \boldsymbol{e}_6$. Then, the codewords of pages 1–4 are $\boldsymbol{c}_1 = \boldsymbol{x}_1 = 0001000$, $\boldsymbol{c}_2 = \boldsymbol{x}_1 + \boldsymbol{x}_2 = 0011000$, $\boldsymbol{c}_3 = \boldsymbol{x}_1 + \boldsymbol{x}_2 + \boldsymbol{x}_3 = 1011000$, and $\boldsymbol{c}_4 = \boldsymbol{x}_1 + \boldsymbol{x}_2 + \boldsymbol{x}_3 + \boldsymbol{x}_4 = 1111010$, respectively.

Now, we show that for any $\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3, \boldsymbol{s}_4 \in \{0,1\}^3$, we can obtain $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4 \in \{0,1\}^7$ that satisfy the conditions of Theorem 10. In this dissertation, we show that we can obtain $I(\boldsymbol{x}_1), I(\boldsymbol{x}_2), I(\boldsymbol{x}_3), I(\boldsymbol{x}_4) \subseteq [7]$ such that

1. $I(\boldsymbol{x}_i) \in V(\boldsymbol{s}_i)$ for all $i \in [4]$
2. $I(\boldsymbol{x}_i) \cap I(\boldsymbol{x}_{i'}) = \emptyset$ for all $i, i' \in [4]$ with $i \neq i'$

instead of $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4$. If $\boldsymbol{s}_i = 000$ for some $i \in [4]$, let $I(\boldsymbol{x}_i) = \{\}$ ($\boldsymbol{x}_i = 0000000$) such that $\boldsymbol{x}_i H^T = \boldsymbol{s}_i$ and $I(\boldsymbol{x}_i) \cap I(\boldsymbol{x}_{i'}) = \emptyset$ for any $i' \in [4] \setminus \{i\}$. In the following, we assume $\boldsymbol{s}_i \neq 000$ for any $i \in [4]$. We consider the following three cases. Note that $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_4$ are sorted if needed.

Case 1: $\boldsymbol{s}_i \neq \boldsymbol{s}_{i'}$ for any $i, i' \in [4]$ with $i \neq i'$.

Let $I(\boldsymbol{x}_i) = \{j_i\} \in V(\boldsymbol{s}_i)$ for each $i \in [4]$.

Case 2: For some $m \in \{2,3,4\}$, $\boldsymbol{s}_1 = \cdots = \boldsymbol{s}_m$ and $\boldsymbol{s}_i \neq \boldsymbol{s}_{i'}$ for any $i, i' \in \{m, \ldots, 4\}$ with $i \neq i'$.

Let $I(\boldsymbol{x}_i) = \{j_i\} \in V(\boldsymbol{s}_i)$ for each $i \in \{m, \ldots, 4\}$. From Theorem 11, we have the permutation σ of $[7]$ such that

$$V(\boldsymbol{s}_1) = \{\{\sigma(1)\},\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\}\}.$$

Then $I(\boldsymbol{x}_1), \ldots, I(\boldsymbol{x}_{m-1}) \in \{\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\}\}$. We obtain $(m-1)$ sets $I(\boldsymbol{x}_1), \ldots, I(\boldsymbol{x}_{m-1}) \in V(\boldsymbol{s}_1)$ such that $I(\boldsymbol{x}_i) \cap (I(\boldsymbol{x}_m) \cup \cdots \cup I(\boldsymbol{x}_4)) = \emptyset$ for any $i \in [m-1]$ as follows. Here, $\sigma(1) = j_m$ because $\{\sigma(1)\} \in V(\boldsymbol{s}_1)$ and $\{j_m\} \in V(\boldsymbol{s}_m) = V(\boldsymbol{s}_1)$. Hence, there are at least $(m-1)$ tuples $(\sigma(\alpha_1), \sigma(\alpha_1+1)), \ldots, (\sigma(\alpha_{m-1}), \sigma(\alpha_{m-1}+1))$, where $\alpha_1, \ldots, \alpha_{m-1} \in \{2,4,6\}$, such that $\{\sigma(\alpha_i), \sigma(\alpha_i+1)\} \cap \{j_{m+1}, \ldots, j_4\} = \emptyset$ for any $i \in [m-1]$,

because $|\{(\sigma(2),\sigma(3)),(\sigma(4),\sigma(5)),(\sigma(6),\sigma(7))\}| = 3$ and $|\{j_{m+1},\dots,j_4\}| = 4-m$. Then, let $I(\boldsymbol{x}_1) = \{\sigma(\alpha_1),\sigma(\alpha_1+1)\},\dots,I(\boldsymbol{x}_{m-1}) = \{\sigma(\alpha_{m-1}),\sigma(\alpha_{m-1}+1)\}$.
Case 3: $\boldsymbol{s}_1 = \boldsymbol{s}_2, \boldsymbol{s}_3 = \boldsymbol{s}_4$, and $\boldsymbol{s}_1 \neq \boldsymbol{s}_3$.

From Theorem 11, we have the permutation σ of $[7]$ such that

$$\begin{aligned} V(\boldsymbol{s}_1) &= \{\{\sigma(1)\},\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\}\}, \\ V(\boldsymbol{s}_3) &= \{\{\sigma(2)\},\{\sigma(1),\sigma(3)\},\{\sigma(4),\sigma(6)\},\{\sigma(5),\sigma(7)\}\}. \end{aligned}$$

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}, I(\boldsymbol{x}_2) = \{\sigma(2),\sigma(3)\}, I(\boldsymbol{x}_3) = \{\sigma(4),\sigma(6)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(5),\sigma(7)\}$.

From the above, we have $\boldsymbol{x}_1,\dots,\boldsymbol{x}_4$ for any $\boldsymbol{s}_1,\dots,\boldsymbol{s}_4$. Note that $\boldsymbol{x}_1,\dots,\boldsymbol{x}_4$ are sorted as necessary.

Therefore, from Theorem 10, the $[7,3,4]$ P-RIO code can be constructed. In this code, more pages are stored than the $[7,3,3]$ RIO code, which is constructed using the same $(7,4)$ Hamming code.

4.2.3 Construction of $[15,4,8]$ P-RIO Code

In a similar manner, we construct $[15,4,8]$ P-RIO code using coset coding with the $(15,11)$ Hamming code. Let H be the parity check matrix of the $(15,11)$ Hamming code. We show that for any $\boldsymbol{s}_1,\dots,\boldsymbol{s}_8 \in \{0,1\}^4$, we can obtain $\boldsymbol{x}_1,\dots,\boldsymbol{x}_8 \in \{0,1\}^{15}$ that satisfy the conditions of Theorem 10. As with the previous section, we show that we can obtain $I(\boldsymbol{x}_1),\dots,I(\boldsymbol{x}_8) \subseteq [15]$ such that

1. $I(\boldsymbol{x}_i) \in V(\boldsymbol{s}_i)$ for all $i \in [8]$
2. $I(\boldsymbol{x}_i) \cap I(\boldsymbol{x}_{i'}) = \emptyset$ for all $i,i' \in [8]$ with $i \neq i'$

instead of $\boldsymbol{x}_1,\dots,\boldsymbol{x}_8$. Without loss of generality, we assume $\boldsymbol{s}_i \neq 0000$ for any $i \in [8]$. We consider the following five cases:
Case 1: $\boldsymbol{s}_i \neq \boldsymbol{s}_{i'}$ for any $i,i' \in [8]$ with $i \neq i'$.

Let $I(\boldsymbol{x}_i) = \{j_i\} \in V(\boldsymbol{s}_i)$ for each $i \in [8]$.
Case 2: For some $m \in \{2,3,\dots,8\}$, $\boldsymbol{s}_1 = \cdots = \boldsymbol{s}_m$ and $\boldsymbol{s}_i \neq \boldsymbol{s}_{i'}$ for any $i,i' \in \{m,\dots,8\}$ with $i \neq i'$.

Let $I(\boldsymbol{x}_i) = \{j_i\} \in V(\boldsymbol{s}_i)$ for each $i \in \{m,\dots,8\}$. From Theorem 11, we have the permutation σ of $[15]$ such that

$$\begin{aligned} V(\boldsymbol{s}_1) = \; &\{\{\sigma(1)\},\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\},\{\sigma(8),\sigma(9)\},\{\sigma(10),\sigma(11)\}, \\ &\{\sigma(12),\sigma(13)\},\{\sigma(14),\sigma(15)\}\}. \end{aligned}$$

Then

$$\begin{aligned} I(\boldsymbol{x}_1),\dots,I(\boldsymbol{x}_{m-1}) \in \{&\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\},\{\sigma(8),\sigma(9)\},\{\sigma(10),\sigma(11)\}, \\ &\{\sigma(12),\sigma(13)\},\{\sigma(14),\sigma(15)\}\} \end{aligned}$$

Clearly, $\sigma(1) = j_m$. Hence, we obtain distinct $I(\boldsymbol{x}_1),\dots,I(\boldsymbol{x}_{m-1})$ such that

$$I(\boldsymbol{x}_i) \cap \{j_{m+1},\dots,j_8\} = \emptyset$$

for any $i \in [m-1]$.
Case 3: For some $m_1, m_2 \in \{2,3,\dots,6\}$, where $m_1 \geq m_2$, $\boldsymbol{s}_1 = \cdots = \boldsymbol{s}_{m_1}, \boldsymbol{s}_{m_1+1} = \cdots = \boldsymbol{s}_{m_1+m_2}$, and $\boldsymbol{s}_i \neq \boldsymbol{s}_{i'}$ for any $i,i' \in \{m_1, m_1+m_2, m_1+m_2+1,\dots,8\}$ with $i \neq i'$.

Let $I(\boldsymbol{x}_i) = \{j_i\} \in V(\boldsymbol{s}_i)$ for each $i \in \{m_1 + m_2 + 1, \dots, 8\}$. From Theorem 11, we have the permutation σ of $[15]$ such that

$$\begin{aligned} V(\boldsymbol{s}_1) &= \{\{\sigma(1)\}, \{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}, \{\sigma(8), \sigma(9)\}, \{\sigma(10), \sigma(11)\}, \\ &\quad \{\sigma(12), \sigma(13)\}, \{\sigma(14), \sigma(15)\}\}, \\ V(\boldsymbol{s}_{m_1+1}) &= \{\{\sigma(2)\}, \{\sigma(1), \sigma(3)\}, \{\sigma(4), \sigma(6)\}, \{\sigma(5), \sigma(7)\}, \{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \\ &\quad \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}. \end{aligned}$$

We obtain m_1 sets $I(\boldsymbol{x}_1), \dots, I(\boldsymbol{x}_{m_1}) \in V(\boldsymbol{s}_1)$ and m_2 sets $I(\boldsymbol{x}_{m_1+1}), \dots, I(\boldsymbol{x}_{m_1+m_2}) \in V(\boldsymbol{s}_{m_1+1})$ such that $I(\boldsymbol{x}_i) \cap \{j_{m_1+m_2+1}, \dots, j_8\} = \emptyset$ for any $i \in [m_1+m_2]$ and $I(\boldsymbol{x}_i) \cap I(\boldsymbol{x}_{i'}) = \emptyset$ for any $i, i' \in [m_1 + m_2]$ with $i \neq i'$. Clearly, $\{\sigma(1), \sigma(2)\} \cap \{j_{m_1+m_2+1}, \dots, j_8\} = \emptyset$. We define $A_1 = \{j_i \mid i \in \{m_1 + m_2 + 1, \dots, 8\}, j_i \in \{\sigma(3), \dots, \sigma(7)\}\}$, $A_2 = \{j_i \mid i \in \{m_1 + m_2 + 1, \dots, 8\}, j_i \in \{\sigma(8), \dots, \sigma(15)\}\}$, $a_1 = |A_1|$, and $a_2 = |A_2|$. Then, $a_1 + a_2 = 8 - m_1 - m_2$.

Case 3-1: $m_1 = m_2 = 2$.

Case 3-1-1: $a_1 = 0$ and $a_2 = 4$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(6)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(5), \sigma(7)\}$.

Case 3-1-2: $a_1 = 1$ and $a_2 = 3$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_2) \in \{\{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_2) = \emptyset$. Let $I(\boldsymbol{x}_3) = \{\sigma(2)\}$ and $I(\boldsymbol{x}_4) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_4) = \emptyset$.

Case 3-1-3: $a_1 = a_2 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_2) \in \{\{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_2) = \emptyset$. Let $I(\boldsymbol{x}_3), I(\boldsymbol{x}_4) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_3) = \emptyset$, $A_2 \cap I(\boldsymbol{x}_4) = \emptyset$, and $I(\boldsymbol{x}_3) \cap I(\boldsymbol{x}_4) = \emptyset$.

Case 3-1-4: $a_1 = 3$ and $a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_2) \in \{\{\sigma(8), \sigma(9)\}, \{\sigma(10), \sigma(11)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_2) = \emptyset$. Let $I(\boldsymbol{x}_3) = \{\sigma(2)\}$ and $I(\boldsymbol{x}_4) \in \{\{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_4) = \emptyset$.

Case 3-1-5: $a_1 = 4$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_3) = \{\sigma(2)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(12), \sigma(14)\}$.

Case 3-2: $m_1 = 3$ and $m_2 = 2$.

Case 3-2-1: $a_1 = 0$ and $a_2 = 3$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(4), \sigma(5)\}$, and $I(\boldsymbol{x}_3) = \{\sigma(6), \sigma(7)\}$. Let $I(\boldsymbol{x}_4) = \{\sigma(2)\}$ and $I(\boldsymbol{x}_5) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_5) = \emptyset$.

Case 3-2-2: $a_1 = 1$ and $a_2 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_2), I(\boldsymbol{x}_3) \in \{\{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_2) = \emptyset$, $A_1 \cap I(\boldsymbol{x}_3) = \emptyset$, and $I(\boldsymbol{x}_2) \cap I(\boldsymbol{x}_3) = \emptyset$. Let

$$I(\boldsymbol{x}_4), I(\boldsymbol{x}_5) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$$

such that $A_2 \cap I(\boldsymbol{x}_4) = \emptyset$, $A_2 \cap I(\boldsymbol{x}_5) = \emptyset$, and $I(\boldsymbol{x}_4) \cap I(\boldsymbol{x}_5) = \emptyset$.

Case 3-2-3: $a_1 = 2$ and $a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_4) = \{\sigma(2)\}$ If $A_2 \subset \{\sigma(12), \sigma(13), \sigma(14), \sigma(15)\}$, $I(\boldsymbol{x}_2) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_3) = \{\sigma(10), \sigma(11)\}$, and $I(\boldsymbol{x}_5) \in \{\{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_5) = \emptyset$. If $A_2 \subset \{\sigma(8), \sigma(9), \sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_2) = \{\sigma(12), \sigma(13)\}$, $I(\boldsymbol{x}_3) = \{\sigma(14), \sigma(15)\}$, and $I(\boldsymbol{x}_5) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_5) = \emptyset$.

Case 3-2-4: $a_1 = 3$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_3) = \{\sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_4) = \{\sigma(2)\}$, and $I(\boldsymbol{x}_5) = \{\sigma(12), \sigma(14)\}$.

Case 3-3: $m_1 = m_2 = 3$.

Case 3-3-1: $a_1 = 0$ and $a_2 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(4), \sigma(5)\}$, and $I(\boldsymbol{x}_3) = \{\sigma(6), \sigma(7)\}$. Let $I(\boldsymbol{x}_4) = \{\sigma(2)\}$ and $I(\boldsymbol{x}_5), I(\boldsymbol{x}_6) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_5) = \emptyset$, $A_2 \cap I(\boldsymbol{x}_6) = \emptyset$, and $I(\boldsymbol{x}_5) \cap I(\boldsymbol{x}_6) = \emptyset$.

Case 3-3-2: $a_1 = a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_2), I(\boldsymbol{x}_3) \in \{\{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_2) = \emptyset$, $A_1 \cap I(\boldsymbol{x}_3) = \emptyset$, and $I(\boldsymbol{x}_2) \cap I(\boldsymbol{x}_3) = \emptyset$. Let $I(\boldsymbol{x}_4), I(\boldsymbol{x}_5), I(\boldsymbol{x}_6) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_i) = \emptyset$ and $I(\boldsymbol{x}_i) \cap I(\boldsymbol{x}_{i'}) = \emptyset$ for any $i, i' \in \{4, 5, 6\}$ with $i \neq i'$.

Case 3-3-3: $a_1 = 2$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_3) = \{\sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_4) = \{\sigma(2)\}$, $I(\boldsymbol{x}_5) = \{\sigma(12), \sigma(14)\}$, and $I(\boldsymbol{x}_6) = \{\sigma(13), \sigma(15)\}$.

Case 3-4: $m_1 = 4$ and $m_2 = 2$.

Case 3-4-1: $a_1 = 0$ and $a_2 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(5)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(6), \sigma(7)\}$. Let

$$I(\boldsymbol{x}_5), I(\boldsymbol{x}_6) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$$

such that $A_2 \cap I(\boldsymbol{x}_5) = \emptyset$, $A_2 \cap I(\boldsymbol{x}_6) = \emptyset$, and $I(\boldsymbol{x}_5) \cap I(\boldsymbol{x}_6) = \emptyset$

Case 3-4-2: $a_1 = a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_2) \in \{\{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_2) = \emptyset$. Let $\boldsymbol{x}_5 = \{\sigma(2)\}$. If $A_2 \subset \{\sigma(12), \sigma(13), \sigma(14), \sigma(15)\}$, $I(\boldsymbol{x}_3) = \{\sigma(8), \sigma(9)\}, I(\boldsymbol{x}_4) = \{\sigma(10), \sigma(11)\}$, and $I(\boldsymbol{x}_6) \in \{\{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_6) = \emptyset$. If $A_2 \subset \{\sigma(8), \sigma(9), \sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_3) = \{\sigma(12), \sigma(13)\}, I(\boldsymbol{x}_4) = \{\sigma(14), \sigma(15)\}$, and $I(\boldsymbol{x}_6) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_6) = \emptyset$.

Case 3-4-3: $a_1 = 2$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_3) = \{\sigma(10), \sigma(11)\}$, and

$$I(\boldsymbol{x}_4) \in \{\{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$$

such that $A_1 \cap I(\boldsymbol{x}_4) = \emptyset$. Let $I(\boldsymbol{x}_5) = \{\sigma(12), \sigma(14)\}$ and $I(\boldsymbol{x}_6) = \{\sigma(13), \sigma(15)\}$.

Case 3-5: $m_1 = 4$ and $m_2 = 3$.

Case 3-5-1: $a_1 = 0$ and $a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(5)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(6), \sigma(7)\}$. Let $I(\boldsymbol{x}_5)$, $I(\boldsymbol{x}_6), I(\boldsymbol{x}_7) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_i) = \emptyset$ and $I(\boldsymbol{x}_i) \cap I(\boldsymbol{x}_{i'}) = \emptyset$ for any $i, i' \in \{5, 6, 7\}$ with $i \neq i'$.

Case 3-5-2: $a_1 = 1$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_2) = \{\sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_3) = \{\sigma(12), \sigma(13)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(14), \sigma(15)\}$. Let $I(\boldsymbol{x}_5) = \{\sigma(2)\}$ and

$$I(\boldsymbol{x}_6), I(\boldsymbol{x}_7) \in \{\{\sigma(1), \sigma(3)\}, \{\sigma(4), \sigma(6)\}, \{\sigma(5), \sigma(7)\}\}$$

such that $A_1 \cap I(\boldsymbol{x}_6) = \emptyset$, $A_1 \cap I(\boldsymbol{x}_7) = \emptyset$, and $I(\boldsymbol{x}_6) \cap I(\boldsymbol{x}_7) = \emptyset$.

Case 3-6: $m_1 = m_2 = 4$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_4) = \{\sigma(6), \sigma(7)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_6) = \{\sigma(9), \sigma(11)\}$, $I(\boldsymbol{x}_7) = \{\sigma(12), \sigma(14)\}$, and

$$I(\boldsymbol{x}_8) = \{\sigma(13), \sigma(15)\}.$$

Case 3-7: $m_1 = 5$ and $m_2 = 2$.

Case 3-7-1: $a_1 = 0$ and $a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_3) = \{\sigma(6), \sigma(7)\}$, and $I(\boldsymbol{x}_6) = \{\sigma(2)\}$ If $A_2 \subset \{\sigma(12), \sigma(13), \sigma(14), \sigma(15)\}$, $I(\boldsymbol{x}_4) = \{\sigma(8), \sigma(9)\}, I(\boldsymbol{x}_5) = \{\sigma(10), \sigma(11)\}$, and $I(\boldsymbol{x}_7) \in \{\{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_7) = \emptyset$. If $A_2 \subset \{\sigma(8), \sigma(9), \sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_4) = \{\sigma(12), \sigma(13)\}, I(\boldsymbol{x}_5) = \{\sigma(14), \sigma(15)\}$, and $I(\boldsymbol{x}_7) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_7) = \emptyset$.

Case 3-7-2: $a_1 = 1$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}, I(\boldsymbol{x}_2) = \{\sigma(8), \sigma(9)\}, I(\boldsymbol{x}_3) = \{\sigma(10), \sigma(11)\}, I(\boldsymbol{x}_4) = \{\sigma(12), \sigma(13)\}$, and $I(\boldsymbol{x}_5) = \{\sigma(14), \sigma(15)\}$. Let $I(\boldsymbol{x}_6) = \{\sigma(2)\}$ and $I(\boldsymbol{x}_7) \in \{\{\sigma(4), \sigma(6)\}, \{\sigma(5), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_7) = \emptyset$.

Case 3-8: $m_1 = 5$ and $m_2 = 3$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_3) = \{\sigma(6), \sigma(7)\}$, $I(\boldsymbol{x}_4) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_5) = \{\sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_6) = \{\sigma(2)\}$, $I(\boldsymbol{x}_7) = \{\sigma(12), \sigma(14)\}$, and $I(\boldsymbol{x}_8) = \{\sigma(13), \sigma(15)\}$.

Case 3-9: $m_1 = 6$ and $m_2 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_4) = \{\sigma(6), \sigma(7)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_6) = \{\sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_7) = \{\sigma(12), \sigma(14)\}$, and

$$I(\boldsymbol{x}_8) = \{\sigma(13), \sigma(15)\}.$$

Case 4: For some $m_1, m_2, m_3 \in \{2, 3, 4\}$, where $m_1 \geq m_2 \geq m_3$, $\boldsymbol{s}_1 = \cdots = \boldsymbol{s}_{m_1}, \boldsymbol{s}_{m_1+1} = \cdots = \boldsymbol{s}_{m_1+m_2}$, $\boldsymbol{s}_{m_1+m_2+1} = \cdots = \boldsymbol{s}_{m_1+m_2+m_3}$, and $\boldsymbol{s}_i \neq \boldsymbol{s}_{i'}$ for any $i, i' \in \{m_1, m_1+m_2, m_1+m_2+m_3, m_1+m_2+m_3+1, \ldots, 8\}$ with $i \neq i'$.

Let $I(\boldsymbol{x}_i) = \{j_i\} \in V(\boldsymbol{s}_i)$ for each $i \in \{m_1+m_2+m_3+1, \ldots, 8\}$.

Case 4-1: $\boldsymbol{s}_1 + \boldsymbol{s}_{m_1+1} \neq \boldsymbol{s}_{m_1+m_2+1}$

From Theorem 11, we have the permutation σ of [15] such that

$$\begin{aligned}
V(\boldsymbol{s}_1) &= \{\{\sigma(1)\}, \{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}, \{\sigma(8), \sigma(9)\}, \\
&\quad \{\sigma(10), \sigma(11)\}, \{\sigma(12), \sigma(13)\}, \{\sigma(14), \sigma(15)\}\}, \\
V(\boldsymbol{s}_{m_1+1}) &= \{\{\sigma(2)\}, \{\sigma(1), \sigma(3)\}, \{\sigma(4), \sigma(6)\}, \{\sigma(5), \sigma(7)\}, \{\sigma(8), \sigma(10)\}, \\
&\quad \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}, \\
V(\boldsymbol{s}_{m_1+m_2+1}) &= \{\{\sigma(4)\}, \{\sigma(1), \sigma(5)\}, \{\sigma(2), \sigma(6)\}, \{\sigma(3), \sigma(7)\}, \{\sigma(8), \sigma(12)\}, \\
&\quad \{\sigma(9), \sigma(13)\}, \{\sigma(10), \sigma(14)\}, \{\sigma(11), \sigma(15)\}\}.
\end{aligned}$$

We obtain m_1 sets $I(\boldsymbol{x}_1), \ldots, I(\boldsymbol{x}_{m_1}) \in V(\boldsymbol{s}_1)$, m_2 sets $I(\boldsymbol{x}_{m_1+1}), \ldots, I(\boldsymbol{x}_{m_1+m_2}) \in V(\boldsymbol{s}_{m_1+1})$, and m_3 sets $I(\boldsymbol{x}_{m_1+m_2+1}), \ldots, I(\boldsymbol{x}_{m_1+m_2+m_3}) \in V(\boldsymbol{s}_{m_1+m_2+1})$ such that

$$I(\boldsymbol{x}_i) \cap \{j_{m_1+m_2+m_3+1}, \ldots, j_8\} = \emptyset$$

for any $i \in [m_1+m_2+m_3]$ and $I(\boldsymbol{x}_i) \cap I(\boldsymbol{x}_{i'}) = \emptyset$ for any $i, i' \in [m_1+m_2+m_3]$ with $i \neq i'$. Clearly, $\{\sigma(1), \sigma(2), \sigma(4)\} \cap \{j_{m_1+m_2+m_3+1}, \ldots, j_8\} = \emptyset$. We define $A_1 = \{j_i \mid i \in \{m_1+m_2+m_3+1, \ldots, 8\}, j_i \in \{\sigma(3), \sigma(5), \sigma(6), \sigma(7)\}\}$, $A_2 = \{j_i \mid i \in \{m_1+m_2+m_3+1, \ldots, 8\}, j_i \in \{\sigma(8), \ldots, \sigma(15)\}\}$, $a_1 = |A_1|$, and $a_2 = |A_2|$. Then, $a_1 + a_2 = 8 - m_1 - m_2 - m_3$.

Case 4-1-1: $m_1 = m_2 = m_3 = 2$.

Case 4-1-1-1: $a_1 = 0$ and $a_2 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}, I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}, I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(6)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(5), \sigma(7)\}$. Let $I(\boldsymbol{x}_5), I(\boldsymbol{x}_6) \in \{\{\sigma(8), \sigma(12)\}, \{\sigma(9), \sigma(13)\}, \{\sigma(10), \sigma(14)\}, \{\sigma(11), \sigma(15)\}\}$ such that

$$A_2 \cap I(\boldsymbol{x}_5) = \emptyset, A_2 \cap I(\boldsymbol{x}_6) = \emptyset,$$

and $I(\boldsymbol{x}_5) \cap I(\boldsymbol{x}_6) = \emptyset$.

Case 4-1-1-2: $a_1 = a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_3) = \{\sigma(2)\}$, and $I(\boldsymbol{x}_5) = \{\sigma(4)\}$.

Case 4-1-1-2-1: $A_2 = \{\sigma(8)\}$ or $A_2 = \{\sigma(15)\}$.

Let $I(\boldsymbol{x}_2) = \{\sigma(10), \sigma(11)\}$, $I(\boldsymbol{x}_4) = \{\sigma(12), \sigma(14)\}$, and $I(\boldsymbol{x}_6) = \{\sigma(9), \sigma(13)\}$.

Case 4-1-1-2-2: $A_2 = \{\sigma(9)\}$ or $A_2 = \{\sigma(14)\}$.

Let $I(\boldsymbol{x}_2) = \{\sigma(12), \sigma(13)\}$, $I(\boldsymbol{x}_4) = \{\sigma(8), \sigma(10)\}$, and $I(\boldsymbol{x}_6) = \{\sigma(11), \sigma(15)\}$.

Case 4-1-1-2-3: $A_2 = \{\sigma(10)\}$ or $A_2 = \{\sigma(13)\}$.

Let $I(\boldsymbol{x}_2) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_4) = \{\sigma(12), \sigma(14)\}$, and $I(\boldsymbol{x}_6) = \{\sigma(11), \sigma(15)\}$.

Case 4-1-1-2-4: $A_2 = \{\sigma(11)\}$ or $A_2 = \{\sigma(12)\}$.

Let $I(\boldsymbol{x}_2) = \{\sigma(14), \sigma(15)\}$, $I(\boldsymbol{x}_4) = \{\sigma(8), \sigma(10)\}$, and $I(\boldsymbol{x}_6) = \{\sigma(9), \sigma(13)\}$.

Case 4-1-1-3: $a_1 = 2$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(8), \sigma(9)\}$, $I(\boldsymbol{x}_3) = \{\sigma(2)\}$, $I(\boldsymbol{x}_4) = \{\sigma(12), \sigma(14)\}$, $I(\boldsymbol{x}_5) = \{\sigma(4)\}$, and $I(\boldsymbol{x}_6) = \{\sigma(11), \sigma(15)\}$.

Case 4-1-2: $m_1 = 3$ and $m_2 = m_3 = 2$.

Case 4-1-2-1: $a_1 = 0$ and $a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(4), \sigma(5)\}$ $I(\boldsymbol{x}_3) = \{\sigma(6), \sigma(7)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(2)\}$.

Case 4-1-2-1-1: $A_2 = \{\sigma(8)\}$ or $A_2 = \{\sigma(10)\}$.

Let $I(\boldsymbol{x}_5) = \{\sigma(12), \sigma(14)\}$, $I(\boldsymbol{x}_6) = \{\sigma(9), \sigma(13)\}$, and $I(\boldsymbol{x}_7) = \{\sigma(11), \sigma(15)\}$.

Case 4-1-2-1-2: $A_2 = \{\sigma(9)\}$ or $A_2 = \{\sigma(11)\}$.

Let $I(\boldsymbol{x}_5) = \{\sigma(13), \sigma(15)\}$, $I(\boldsymbol{x}_6) = \{\sigma(8), \sigma(12)\}$, and $I(\boldsymbol{x}_7) = \{\sigma(10), \sigma(14)\}$.

Case 4-1-2-1-3: $A_2 = \{\sigma(12)\}$ or $A_2 = \{\sigma(14)\}$.

Let $I(\boldsymbol{x}_5) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_6) = \{\sigma(9), \sigma(13)\}$, and $I(\boldsymbol{x}_7) = \{\sigma(11), \sigma(15)\}$.

Case 4-1-2-1-4: $A_2 = \{\sigma(13)\}$ or $A_2 = \{\sigma(15)\}$.

Let $I(\boldsymbol{x}_5) = \{\sigma(9), \sigma(11)\}$, $I(\boldsymbol{x}_6) = \{\sigma(8), \sigma(12)\}$, and $I(\boldsymbol{x}_7) = \{\sigma(10), \sigma(14)\}$.

Case 4-1-2-2: $a_1 = 1$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_2), I(\boldsymbol{x}_3) \in \{\{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_2) = \emptyset$, $A_1 \cap I(\boldsymbol{x}_3) = \emptyset$, and $I(\boldsymbol{x}_2) \cap I(\boldsymbol{x}_3) = \emptyset$. Let $I(\boldsymbol{x}_4) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_5) = \{\sigma(12), \sigma(14)\}$, $I(\boldsymbol{x}_6) = \{\sigma(9), \sigma(13)\}$, and $I(\boldsymbol{x}_7) = \{\sigma(11), \sigma(15)\}$.

Case 4-1-3: $m_1 = m_2 = 3$ and $m_3 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_3) = \{\sigma(6), \sigma(7)\}$, $I(\boldsymbol{x}_4) = \{\sigma(2)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_6) = \{\sigma(12), \sigma(14)\}$, $I(\boldsymbol{x}_7) = \{\sigma(9), \sigma(13)\}$, and $I(\boldsymbol{x}_8) = \{\sigma(11), \sigma(15)\}$.

Case 4-1-4: $m_1 = 4$ and $m_2 = m_3 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_4) = \{\sigma(6), \sigma(7)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_6) = \{\sigma(12), \sigma(14)\}$, $I(\boldsymbol{x}_7) = \{\sigma(9), \sigma(13)\}$, and

$$I(\boldsymbol{x}_8) = \{\sigma(11), \sigma(15)\}.$$

Case 4-2: $\boldsymbol{s}_1 + \boldsymbol{s}_{m_1+1} = \boldsymbol{s}_{m_1+m_2+1}$

From Theorem 11, we have the permutation σ of $[15]$ such that

$$\begin{aligned}
V(\boldsymbol{s}_1) &= \{\{\sigma(1)\}, \{\sigma(2), \sigma(3)\}, \{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}, \{\sigma(8), \sigma(9)\}, \\
&\quad \{\sigma(10), \sigma(11)\}, \{\sigma(12), \sigma(13)\}, \{\sigma(14), \sigma(15)\}\}, \\
V(\boldsymbol{s}_{m_1+1}) &= \{\{\sigma(2)\}, \{\sigma(1), \sigma(3)\}, \{\sigma(4), \sigma(6)\}, \{\sigma(5), \sigma(7)\}, \{\sigma(8), \sigma(10)\}, \\
&\quad \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}, \\
V(\boldsymbol{s}_{m_1+m_2+1}) &= \{\{\sigma(3)\}, \{\sigma(1), \sigma(2)\}, \{\sigma(4), \sigma(7)\}, \{\sigma(5), \sigma(6)\}, \{\sigma(8), \sigma(11)\}, \\
&\quad \{\sigma(9), \sigma(10)\}, \{\sigma(12), \sigma(15)\}, \{\sigma(13), \sigma(14)\}\}.
\end{aligned}$$

As with *Case 4-1*, we obtain m_1 sets

$$I(\boldsymbol{x}_1), \ldots, I(\boldsymbol{x}_{m_1}) \in V(\boldsymbol{s}_1),$$

m_2 sets

$$I(\boldsymbol{x}_{m_1+1}), \ldots, I(\boldsymbol{x}_{m_1+m_2}) \in V(\boldsymbol{s}_{m_1+1}),$$

and m_3 sets

$$I(\boldsymbol{x}_{m_1+m_2+1}), \ldots, I(\boldsymbol{x}_{m_1+m_2+m_3}) \in V(\boldsymbol{s}_{m_1+m_2+1})$$

such that the conditions described in *Case 4-1* are satisfied. Clearly, $\{\sigma(1), \sigma(2), \sigma(3)\} \cap \{j_{m_1+m_2+m_3+1}, \ldots, j_8\} = \emptyset$. We define $A_1 = \{j_i \mid i \in \{m_1+m_2+m_3+1, \ldots, 8\}, j_i \in \{\sigma(4), \sigma(5), \sigma(6), \sigma(7)\}\}$, $A_2 = \{j_i \mid i \in \{m_1+m_2+m_3+1, \ldots, 8\}, j_i \in \{\sigma(8), \ldots, \sigma(15)\}\}$, $a_1 = |A_1|$, and $a_2 = |A_2|$. Then, $a_1 + a_2 = 8 - m_1 - m_2 - m_3$.
Case 4-2-1: $m_1 = m_2 = m_3 = 2$.
Case 4-2-1-1: $a_1 = 0$ and $a_2 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(6)\}$, and $I(\boldsymbol{x}_4) = \{\sigma(5), \sigma(7)\}$. Let $I(\boldsymbol{x}_5), I(\boldsymbol{x}_6) \in \{\{\sigma(8), \sigma(11)\}, \{\sigma(9), \sigma(10)\}, \{\sigma(12), \sigma(15)\}, \{\sigma(13), \sigma(14)\}\}$ such that

$$A_2 \cap I(\boldsymbol{x}_5) = \emptyset, A_2 \cap I(\boldsymbol{x}_6) = \emptyset,$$

and $I(\boldsymbol{x}_5) \cap I(\boldsymbol{x}_6) = \emptyset$.
Case 4-2-1-2: $a_1 = a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$ and $I(\boldsymbol{x}_2) \in \{\{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_2) = \emptyset$. Let $I(\boldsymbol{x}_3) = \{\sigma(2)\}$ and $I(\boldsymbol{x}_4) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}, \{\sigma(12), \sigma(14)\}, \{\sigma(13), \sigma(15)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_4) = \emptyset$. Let $I(\boldsymbol{x}_5) = \{\sigma(3)\}$ and $I(\boldsymbol{x}_6) \in \{\{\sigma(8), \sigma(11)\}, \{\sigma(9), \sigma(10)\}, \{\sigma(12), \sigma(15)\}, \{\sigma(13), \sigma(14)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_6) = \emptyset$ and $I(\boldsymbol{x}_4) \cap I(\boldsymbol{x}_6) = \emptyset$.
Case 4-2-1-3: $a_1 = 2$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_4) = \{\sigma(9), \sigma(11)\}$, $I(\boldsymbol{x}_5) = \{\sigma(12), \sigma(15)\}$, $I(\boldsymbol{x}_6) = \{\sigma(13), \sigma(14)\}$.
Case 4-2-2: $m_1 = 3$ and $m_2 = m_3 = 2$.
Case 4-2-2-1: $a_1 = 0$ and $a_2 = 1$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_3) = \{\sigma(6), \sigma(7)\}$, $I(\boldsymbol{x}_4) = \{\sigma(2)\}$, and $I(\boldsymbol{x}_5) \in \{\{\sigma(8), \sigma(10)\}, \{\sigma(9), \sigma(11)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_5) = \emptyset$. Let $I(\boldsymbol{x}_6) = \{\sigma(3)\}$ and $I(\boldsymbol{x}_7) \in \{\{\sigma(12), \sigma(15)\}, \{\sigma(13), \sigma(14)\}\}$ such that $A_2 \cap I(\boldsymbol{x}_7) = \emptyset$.
Case 4-2-2-2: $a_1 = 1$ and $a_2 = 0$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, and $I(\boldsymbol{x}_3) \in \{\{\sigma(4), \sigma(5)\}, \{\sigma(6), \sigma(7)\}\}$ such that $A_1 \cap I(\boldsymbol{x}_3) = \emptyset$. Let $I(\boldsymbol{x}_4) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_5) = \{\sigma(9), \sigma(11)\}$, $I(\boldsymbol{x}_6) = \{\sigma(12), \sigma(15)\}$, and $I(\boldsymbol{x}_7) = \{\sigma(13), \sigma(14)\}$.
Case 4-2-3: $m_1 = m_2 = 3$ and $m_3 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_3) = \{\sigma(6), \sigma(7)\}$, $I(\boldsymbol{x}_4) = \{\sigma(2)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_6) = \{\sigma(9), \sigma(11)\}$, $I(\boldsymbol{x}_7) = \{\sigma(3)\}$, and $I(\boldsymbol{x}_8) = \{\sigma(12), \sigma(15)\}$.
Case 4-2-4: $m_1 = 4$ and $m_2 = m_3 = 2$.

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2), \sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4), \sigma(5)\}$, $I(\boldsymbol{x}_4) = \{\sigma(6), \sigma(7)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8), \sigma(10)\}$, $I(\boldsymbol{x}_6) = \{\sigma(9), \sigma(11)\}$, $I(\boldsymbol{x}_7) = \{\sigma(12), \sigma(15)\}$, and

$$I(\boldsymbol{x}_8) = \{\sigma(13), \sigma(14)\}.$$

Case 5: $\boldsymbol{s}_1 = \boldsymbol{s}_2, \boldsymbol{s}_3 = \boldsymbol{s}_4$, $\boldsymbol{s}_5 = \boldsymbol{s}_6$, $\boldsymbol{s}_7 = \boldsymbol{s}_8$, and $\boldsymbol{s}_i \neq \boldsymbol{s}_{i'}$ for any $i, i' \in \{1, 3, 5, 7\}$ with $i \neq i'$.
Case 5-1: $\boldsymbol{s}_1, \boldsymbol{s}_3, \boldsymbol{s}_5$, and $\boldsymbol{s}_7$ are linearly independent.

From Theorem 11, we have the permutation σ of $[15]$ such that

$$\begin{aligned}
V(\boldsymbol{s}_1) &= \{\{\sigma(1)\},\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\},\{\sigma(8),\sigma(9)\},\{\sigma(10),\sigma(11)\},\\
&\quad \{\sigma(12),\sigma(13)\},\{\sigma(14),\sigma(15)\}\},\\
V(\boldsymbol{s}_3) &= \{\{\sigma(2)\},\{\sigma(1),\sigma(3)\},\{\sigma(4),\sigma(6)\},\{\sigma(5),\sigma(7)\},\{\sigma(8),\sigma(10)\},\{\sigma(9),\sigma(11)\},\\
&\quad \{\sigma(12),\sigma(14)\},\{\sigma(13),\sigma(15)\}\},\\
V(\boldsymbol{s}_5) &= \{\{\sigma(4)\},\{\sigma(1),\sigma(5)\},\{\sigma(2),\sigma(6)\},\{\sigma(3),\sigma(7)\},\{\sigma(8),\sigma(12)\},\{\sigma(9),\sigma(13)\},\\
&\quad \{\sigma(10),\sigma(14)\},\{\sigma(11),\sigma(15)\}\},\\
V(\boldsymbol{s}_7) &= \{\{\sigma(8)\},\{\sigma(1),\sigma(9)\},\{\sigma(2),\sigma(10)\},\{\sigma(3),\sigma(11)\},\{\sigma(4),\sigma(12)\},\{\sigma(5),\sigma(13)\},\\
&\quad \{\sigma(6),\sigma(14)\},\{\sigma(7),\sigma(15)\}\}.
\end{aligned}$$

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2),\sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4),\sigma(6)\}$, $I(\boldsymbol{x}_4) = \{\sigma(9),\sigma(11)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8),\sigma(12)\}$, $I(\boldsymbol{x}_6) = \{\sigma(10),\sigma(14)\}$, $I(\boldsymbol{x}_7) = \{\sigma(5),\sigma(13)\}$, and

$$I(\boldsymbol{x}_8) = \{\sigma(7),\sigma(15)\}.$$

Case 5-2: $\boldsymbol{s}_1 + \boldsymbol{s}_3 = \boldsymbol{s}_5$.

From Theorem 11, we have the permutation σ of $[15]$ such that

$$\begin{aligned}
V(\boldsymbol{s}_1) &= \{\{\sigma(1)\},\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\},\{\sigma(8),\sigma(9)\},\{\sigma(10),\sigma(11)\},\\
&\quad \{\sigma(12),\sigma(13)\},\{\sigma(14),\sigma(15)\}\},\\
V(\boldsymbol{s}_3) &= \{\{\sigma(2)\},\{\sigma(1),\sigma(3)\},\{\sigma(4),\sigma(6)\},\{\sigma(5),\sigma(7)\},\{\sigma(8),\sigma(10)\},\{\sigma(9),\sigma(11)\},\\
&\quad \{\sigma(12),\sigma(14)\},\{\sigma(13),\sigma(15)\}\},\\
V(\boldsymbol{s}_5) &= \{\{\sigma(3)\},\{\sigma(1),\sigma(2)\},\{\sigma(4),\sigma(7)\},\{\sigma(5),\sigma(6)\},\{\sigma(8),\sigma(11)\},\{\sigma(9),\sigma(10)\},\\
&\quad \{\sigma(12),\sigma(15)\},\{\sigma(13),\sigma(14)\}\},\\
V(\boldsymbol{s}_7) &= \{\{\sigma(4)\},\{\sigma(1),\sigma(5)\},\{\sigma(2),\sigma(6)\},\{\sigma(3),\sigma(7)\},\{\sigma(8),\sigma(12)\},\{\sigma(9),\sigma(13)\},\\
&\quad \{\sigma(10),\sigma(14)\},\{\sigma(11),\sigma(15)\}\}.
\end{aligned}$$

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2),\sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4),\sigma(6)\}$, $I(\boldsymbol{x}_4) = \{\sigma(5),\sigma(7)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8),\sigma(11)\}$, $I(\boldsymbol{x}_6) = \{\sigma(12),\sigma(15)\}$, $I(\boldsymbol{x}_7) = \{\sigma(9),\sigma(13)\}$, and

$$I(\boldsymbol{x}_8) = \{\sigma(10),\sigma(14)\}.$$

Case 5-3: $\boldsymbol{s}_1 + \boldsymbol{s}_3 = \boldsymbol{s}_5 + \boldsymbol{s}_7$.

From Theorem 11, we have the permutation σ of $[15]$ such that

$$\begin{aligned}
V(\boldsymbol{s}_1) &= \{\{\sigma(1)\},\{\sigma(2),\sigma(3)\},\{\sigma(4),\sigma(5)\},\{\sigma(6),\sigma(7)\},\{\sigma(8),\sigma(9)\},\{\sigma(10),\sigma(11)\},\\
&\quad \{\sigma(12),\sigma(13)\},\{\sigma(14),\sigma(15)\}\},\\
V(\boldsymbol{s}_3) &= \{\{\sigma(2)\},\{\sigma(1),\sigma(3)\},\{\sigma(4),\sigma(6)\},\{\sigma(5),\sigma(7)\},\{\sigma(8),\sigma(10)\},\{\sigma(9),\sigma(11)\},\\
&\quad \{\sigma(12),\sigma(14)\},\{\sigma(13),\sigma(15)\}\},\\
V(\boldsymbol{s}_5) &= \{\{\sigma(4)\},\{\sigma(1),\sigma(5)\},\{\sigma(2),\sigma(6)\},\{\sigma(3),\sigma(7)\},\{\sigma(8),\sigma(12)\},\{\sigma(9),\sigma(13)\},\\
&\quad \{\sigma(10),\sigma(14)\},\{\sigma(11),\sigma(15)\}\},\\
V(\boldsymbol{s}_7) &= \{\{\sigma(7)\},\{\sigma(1),\sigma(6)\},\{\sigma(2),\sigma(5)\},\{\sigma(3),\sigma(4)\},\{\sigma(8),\sigma(15)\},\{\sigma(9),\sigma(14)\},\\
&\quad \{\sigma(10),\sigma(13)\},\{\sigma(11),\sigma(12)\}\}.
\end{aligned}$$

Let $I(\boldsymbol{x}_1) = \{\sigma(1)\}$, $I(\boldsymbol{x}_2) = \{\sigma(2),\sigma(3)\}$, $I(\boldsymbol{x}_3) = \{\sigma(4),\sigma(6)\}$, $I(\boldsymbol{x}_4) = \{\sigma(5),\sigma(7)\}$, $I(\boldsymbol{x}_5) = \{\sigma(8),\sigma(12)\}$, $I(\boldsymbol{x}_6) = \{\sigma(11),\sigma(15)\}$, $I(\boldsymbol{x}_7) = \{\sigma(9),\sigma(14)\}$, and

$$I(\boldsymbol{x}_8) = \{\sigma(10),\sigma(13)\}.$$

From the above, we have $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_8$ for any $\boldsymbol{s}_1,\ldots,\boldsymbol{s}_8$. Therefore, the $[15,4,8]$ P-RIO code, in which more pages are stored than in the $[15,4,6]$ RIO code, can be constructed.

4.2.4 Non-existence of $[7,3,4]$ RIO code and $[15,4,8]$ RIO code

In this subsection, we show the non-existence of $[7,3,4]$ RIO code and $[15,4,8]$ RIO code that have the same parameters as we constructed.

As described in Subsection 4.1.3, the existence of $[n,l,t]$ RIO code is equivalent to that of $[n,l,t]$ WOM code. For any l and t, Rivest and Shamir derived a lower bound $Z(l,t)$ on n for which an $[n,l,t]$ WOM code exists [12].

Theorem 12 (Rivest and Shamir [12]). *Let*

$$\delta(l,m) = \min\left\{ h \,\middle|\, \sum_{i=0}^{h} \binom{m+h}{i} \geq 2^l \right\}.$$

Suppose that for any l,t, $Z(l,t)$ satisfies $Z(l,0)=0$, and for $t \geq 0$,

$$Z(l,t+1) = Z(l,t) + \delta(l, Z(l,t)).$$

Then, if an $[n,l,t]$ WOM code exists,

$$n \geq Z(l,t).$$

From $Z(3,4)=8$ and $Z(4,8)=16$, $[7,3,4]$ WOM code and $[15,4,8]$ WOM code, that is, $[7,3,4]$ RIO code and $[15,4,8]$ RIO code do not exist. Our $[7,3,4]$ P-RIO code and $[15,4,8]$ P-RIO code have parameters for which RIO codes based on WOM codes do not exist.

4.3 Conclusion

In this dissertation, we constructed P-RIO codes using coset coding with Hamming codes of length 7 and 15. In our P-RIO codes, more pages are stored than in RIO codes constructed via coset coding with the same Hamming codes. Our P-RIO codes have parameters for which RIO codes cannot be constructed using any technique, including coset coding.

Zhang, Yaakobi, and Etzion verified that when using coset coding with $(31,26)$ Hamming code, $[31,5,16]$ P-RIO code can be constructed [23]. It can be seen that $[31,5,16]$ RIO code does not exist.

P-RIO codes with parameters for which RIO codes do not exist could be constructed using Hamming codes of length (2^r-1) for $r \geq 6$. The number of pages in such a P-RIO code would increase with r because $[2^r-1, r, 2^{r-2}+2]$ RIO codes can be constructed from WOM codes. However, in our approach, the number of cases required for the encoding increases with the number of pages. Therefore, we should consider developing another approach to encoding or checking whether the sufficient condition ensuring construction of a P-RIO code holds. Additionally, we should explore other linear codes to construct P-RIO codes with higher rates than previous codes. Through coset coding by using the $(23,12)$ Golay code, a $[23,11,3]$ WOM code was constructed in [14]. However, it is unknown whether a $[23,11,4]$ P-RIO code can be constructed.

Chapter 5

Generalized Cayley distance

This chapter describes the generalized Cayley distance, which is one of distances considered in permutation codes [19]. Then, the derivation of our upper bound on the generalized Cayley distance is described.

5.1 Preliminaries

For a positive integer n, we define $[n] = \{1, 2, \ldots, n\}$. We define $\mathbb{S}_n$ as the set of all permutations on $[n]$. For any $\sigma \in \mathbb{S}_n$, we denote the permutation σ by a vector $(\sigma(1), \sigma(2), \ldots, \sigma(n))$. The identity permutation $(1, 2, \ldots, n)$ is denoted by e. Let $\sigma \circ \pi$ denote the composition of two permutations $\sigma, \pi \in \mathbb{S}_n$. That is, $\sigma \circ \pi = (\sigma(\pi(1)), \sigma(\pi(2)), \ldots, \sigma(\pi(n)))$. We denote the inverse permutation of σ by σ^{-1}. For any $1 \leq i \leq j \leq n$, we define $\sigma[i; j] = (\sigma(i), \sigma(i+1), \ldots, \sigma(j))$.

5.1.1 Generalized Cayley Distance

A generalized transposition $\phi(i_1, j_1, i_2, j_2) \in \mathbb{S}_n$ is defined as follows:

$$\phi(i_1, j_1, i_2, j_2) = (1, \ldots, i_1 - 1, i_2, \ldots, j_2, j_1 + 1, \ldots, i_2 - 1, i_1, \ldots, j_1, j_2 + 1, \ldots, n),$$

where $1 \leq i_1 \leq j_1 < i_2 \leq j_2 \leq n$. This is a permutation obtained by swapping two subsequences, $e[i_1; j_1]$ and $e[i_2; j_2]$ of the identity permutation e. We define $\mathbb{T}_n$ as the set of all generalized transpositions in $\mathbb{S}_n$. For each $\pi \in \mathbb{S}_n$ and $\phi(i_1, j_1, i_2, j_2) \in \mathbb{T}_n$, $\pi \circ \phi(i_1, j_1, i_2, j_2)$ is the permutation obtained by swapping two subsequences $\pi[i_1; j_1]$ and $\pi[i_2; j_2]$ of π. That is,

$$\begin{aligned}\pi \circ \phi(i_1, j_1, i_2, j_2) \quad = \quad & (\pi(1), \ldots, \pi(i_1 - 1), \pi(i_2), \ldots, \pi(j_2), \pi(j_1 + 1), \\ & \ldots, \pi(i_2 - 1), \pi(i_1), \ldots, \pi(j_1), \pi(j_2 + 1), \ldots, \pi(n)).\end{aligned}$$

Definition 4. *For any $\pi_1, \pi_2 \in \mathbb{S}_n$, the generalized Cayley distance $d_G(\pi_1, \pi_2)$ is the minimum number of generalized transpositions required to transform π_1 into π_2:*

$$d_G(\pi_1, \pi_2) = \min\{k \mid \pi_2 = \pi_1 \circ \phi_1 \circ \phi_2 \circ \cdots \circ \phi_k, \phi_1, \phi_2, \ldots, \phi_k \in \mathbb{T}_n\}.$$

Theorem 13 (Chee et al. [19])**.** *For all $\pi_1, \pi_2, \pi_3 \in \mathbb{S}_n$, d_G satisfies the following conditions:*

1. $d_G(\pi_2, \pi_1) = d_G(\pi_1, \pi_2)$.

2. $d_G(\pi_3 \circ \pi_1, \pi_3 \circ \pi_2) = d_G(\pi_1, \pi_2)$.

3. $d_G(\pi_1, \pi_3) \le d_G(\pi_1, \pi_2) + d_G(\pi_2, \pi_3)$.

An algorithm for calculating the generalized Cayley distance is demonstrated [21].

Theorem 14 (Christie [21])**.** *For any* $\pi \in \mathbb{S}_n$*, let* $c(\pi)$ *be the number of cycles in the directed graph that satisfies the following conditions:*

- *The vertex set is* $\{0, 1, \ldots, n\}$.
- *For any* $0 \le i \le n$*, an edge connects vertex* i *to vertex* $\pi(\pi^{-1}(i+1)-1)$*, where* $\pi(0) = 0$ *and* $\pi(n+1) = n+1$.

Then, for any $\pi_1, \pi_2 \in \mathbb{S}_n$,

$$d_G(\pi_1, \pi_2) = \frac{1}{2}(n + 1 - c(\pi_2^{-1} \circ \pi_1)).$$

The exact value of the generalized Cayley distance is complicated to compute. In order to construct order-optimal permutation codes with the generalized Cayley distance, a new distance called block permutation distance was introduced [20].

5.1.2 Block Permutation Distance

A permutation $\pi \in \mathbb{S}_n$ is called a minimal permutation if $\pi(i+1) \ne \pi(i) + 1$ for any $1 \le i \le n-1$. We denote the set of all minimal permutations in $\mathbb{S}_n$ by $\mathbb{D}_n$.

Definition 5. *For any* $\pi_1, \pi_2 \in \mathbb{S}_n$*, the block permutation distance* $d_B(\pi_1, \pi_2)$ *is defined as* d *if for some* $0 = i_0 < i_1 < \cdots < i_d < i_{d+1} = n$ *and* $\sigma \in \mathbb{D}_{d+1}$ *it holds that*

$$\begin{aligned} \pi_1 &= (\psi_1, \psi_2, \ldots, \psi_{d+1}), \\ \pi_2 &= (\psi_{\sigma(1)}, \psi_{\sigma(2)}, \ldots, \psi_{\sigma(d+1)}), \end{aligned} \tag{5.1}$$

where $\psi_k = \pi_1[i_{k-1}+1; i_k]$ *for* $1 \le k \le d+1$*. Then,* σ *is called the characteristic permutation between* π_1 *and* π_2.

Theorem 15 (Yang et al. [20])**.** *The block permutation distance* d_B *also satisfies the three conditions described in Theorem 13.*

Definition 6. *For any* $\pi \in \mathbb{S}_n$*, the characteristic set* $A(\pi)$ *is defined as follows:*

$$A(\pi) = \{(\pi(i), \pi(i+1)) \mid 1 \le i \le n-1\}.$$

Definition 7. *For any* $\pi \in \mathbb{S}_n$*, the block permutation weight* $w_B(\pi)$ *is defined as follows:*

$$w_B(\pi) = |A(\pi) \setminus A(e)|,$$

where e *is the identity permutation.*

Theorem 16. *For any* $\pi \in \mathbb{S}_n$,

$$\pi \in \mathbb{D}_n \text{ if and only if } w_B(\pi) = n-1$$

and

$$\pi = e \text{ if and only if } w_B(\pi) = 0.$$

Theorem 17 (Yang et al. [20]). *For all $\pi_1, \pi_2 \in \mathbb{S}_n$, it holds that*

$$d_B(\pi_1, \pi_2) = |A(\pi_2) \setminus A(\pi_1)| = |A(\pi_1) \setminus A(\pi_2)|.$$

Theorem 18 (Yang et al. [20]). *For all $\pi \in \mathbb{S}_n$, it holds that*

$$w_B(\pi) = d_B(e, \pi) = d_B(\pi, e).$$

Yang, Schoeny, and Dolecek derived the following relation between d_G and d_B.

Theorem 19 (Yang et al. [20]). *For all $\pi_1, \pi_2 \in \mathbb{S}_n$,*

$$\frac{1}{4} d_B(\pi_1, \pi_2) \leq d_G(\pi_1, \pi_2) \leq d_B(\pi_1, \pi_2).$$

In this dissertation, we derive a tighter upper bound on d_G, using d_B.

5.2 A Tighter Upper Bound on the Generalized Cayley Distance

First, we derive an upper bound on the generalized Cayley distance $d_G(\pi, e)$ between any minimal permutation $\pi \in \mathbb{D}_n$ and the identity permutation e. The following theorem shows that for any minimal permutation, there exists a generalized transposition that decreases the block permutation weight by at least two.

Theorem 20. *Let $n \geq 3$. For any $\pi \in \mathbb{D}_n$, there exists $\phi \in \mathbb{T}_n$ such that*

$$w_B(\pi \circ \phi) \leq w_B(\pi) - 2.$$

To prove Theorem 20, we demonstrate the following lemma.

Lemma 1. *Any $\pi \in \mathbb{D}_n$ satisfies at least one of the following conditions:*

1. *For some $1 \leq a \leq n-2$,*
$$\pi^{-1}(a) < \pi^{-1}(a+2) < \pi^{-1}(a+1).$$

2. *For some $1 \leq a \leq n-2$,*
$$\pi^{-1}(a+1) < \pi^{-1}(a) < \pi^{-1}(a+2).$$

3. *For some $1 \leq a \leq n-2$,*
$$\pi^{-1}(a+2) < \pi^{-1}(a+1) < \pi^{-1}(a).$$

4. *For some $1 \leq b \leq n-1$ and $2 \leq c \leq n$,*
$$\pi^{-1}(c-1) < \pi^{-1}(b) < \pi^{-1}(c) < \pi^{-1}(b+1) \quad \text{and } \pi^{-1}(c) = \pi^{-1}(b) + 1.$$

5. *For some $1 \leq b \leq n-1$ and $2 \leq c \leq n$,*
$$\pi^{-1}(b) < \pi^{-1}(c) < \pi^{-1}(b+1) < \pi^{-1}(c-1) \quad \text{and } \pi^{-1}(c) = \pi^{-1}(b) + 1.$$

Proof. If $n = 3$, then

$$\pi = (1, 3, 2) \text{ or } \pi = (2, 1, 3) \text{ or } \pi = (3, 2, 1)$$

because $\pi \in \mathbb{D}_3$. Thus, π satisfies one of the first three conditions.

Suppose that $n = 4$ and none of the first three conditions are satisfied. Then, since $\pi \in \mathbb{D}_4$,

$$\pi = (3, 1, 4, 2).$$

Therefore, π satisfies the fourth condition.

In general, we show that one of the last two conditions is satisfied if none of the first three conditions are satisfied.

If $\pi^{-1}(1) < \pi^{-1}(2)$, then

$$\begin{aligned} \pi^{-1}(1) < \pi^{-1}(c^{(0)}) < \pi^{-1}(2) \\ \text{and } \pi^{-1}(c^{(0)}) = \pi^{-1}(1) + 1 \end{aligned}$$

for some $c^{(0)}$, because $\pi \in \mathbb{D}_n$. If $\pi^{-1}(2) < \pi^{-1}(1)$, then

$$\begin{aligned} \pi^{-1}(2) < \pi^{-1}(c^{(0)}) < \pi^{-1}(3) < \pi^{-1}(1) \\ \text{and } \pi^{-1}(c^{(0)}) = \pi^{-1}(2) + 1 \end{aligned}$$

for some $c^{(0)}$, because neither the second nor third condition is satisfied and $\pi \in \mathbb{D}_n$. Hence, for some $1 \le b^{(0)} \le 2$ and $c^{(0)}$,

$$\begin{aligned} \pi^{-1}(b^{(0)}) < \pi^{-1}(c^{(0)}) < \pi^{-1}(b^{(0)} + 1) \\ \text{and } \pi^{-1}(c^{(0)}) = \pi^{-1}(b^{(0)}) + 1 \end{aligned}$$

and "1" does not appear in $\pi[\pi^{-1}(c^{(0)}); \pi^{-1}(b^{(0)} + 1) - 1]$.

Next, we consider the location of "$(c^{(0)} - 1)$". Assume that $b^{(0)} = c^{(0)} - 1$. Thus, $\pi^{-1}(c^{(0)}) = \pi^{-1}(c^{(0)} - 1) + 1$. That is, $\pi \notin \mathbb{D}_n$, which is a contradiction. Assume that $b^{(0)} + 1 = c^{(0)} - 1$. That is, $\pi^{-1}(c^{(0)} - 2) < \pi^{-1}(c^{(0)}) < \pi^{-1}(c^{(0)} - 1)$. Then, the first condition is satisfied, which is also a contradiction. Therefore, $b^{(0)} \neq c^{(0)} - 1$ and $b^{(0)} + 1 \neq c^{(0)} - 1$.

Case 1: "$(c^{(0)} - 1)$" does not appear in $\pi[\pi^{-1}(c^{(0)}) + 1; \pi^{-1}(b^{(0)} + 1) - 1]$.

One of the last two conditions is clearly satisfied.

Case 2: $\pi^{-1}(b^{(0)}) < \pi^{-1}(c^{(0)}) < \pi^{-1}(c^{(0)} - 1) < \pi^{-1}(b^{(0)} + 1)$.

Case 2-1: $c^{(0)} = n$.

Because neither the first nor third condition is satisfied and $\pi \in \mathbb{D}_n$, it holds that

$$\begin{aligned} \pi^{-1}(b^{(0)}) < \pi^{-1}(n) < \pi^{-1}(n-2) < \pi^{-1}(c^{(1)}) < \pi^{-1}(n-1) < \pi^{-1}(b^{(0)} + 1) \\ \text{and } \pi^{-1}(c^{(1)}) = \pi^{-1}(n-2) + 1 \end{aligned}$$

for some $c^{(1)}$.

Case 2-2: $c^{(0)} \neq n$.

Because neither the second nor third condition is satisfied and $\pi \in \mathbb{D}_n$,

$$\begin{aligned} \pi^{-1}(b^{(0)}) < \pi^{-1}(c^{(0)}) < \pi^{-1}(c^{(1)}) < \pi^{-1}(c^{(0)} + 1) < \pi^{-1}(c^{(0)} - 1) < \pi^{-1}(b^{(0)} + 1) \\ \text{and } \pi^{-1}(c^{(1)}) = \pi^{-1}(c^{(0)}) + 1 \end{aligned}$$

for some $c^{(1)}$.

Therefore, for some $b^{(1)}$ and $c^{(1)}$ it holds that

$$\pi^{-1}(b^{(0)}) < \pi^{-1}(b^{(1)}) < \pi^{-1}(c^{(1)}) < \pi^{-1}(b^{(1)} + 1) < \pi^{-1}(b^{(0)} + 1).$$

Similarly, we can find $b^{(j)}$ and $c^{(j)}$ such that

$$\pi^{-1}(b^{(j-1)}) < \pi^{-1}(b^{(j)}) < \pi^{-1}(c^{(j)}) < \pi^{-1}(b^{(j)}+1) < \pi^{-1}(b^{(j-1)}+1)$$

for some $j \geq 2$ if $\pi^{-1}(b^{(i)}) < \pi^{-1}(c^{(i)}) < \pi^{-1}(c^{(i)}-1) < \pi^{-1}(b^{(i)}+1)$ for all $0 \leq i \leq j-1$. Because the length of $\pi[\pi^{-1}(b^{(j)}); \pi^{-1}(b^{(j)}+1)]$ is shorter than that of $\pi[\pi^{-1}(b^{(j-1)}); \pi^{-1}(b^{(j-1)}+1)]$ and π is a finite sequence, "$(c^{(m)}-1)$" is not located between "$c^{(m)}$" and "$(b^{(m)}+1)$" for some $m \geq 1$. Hence, one of the last two conditions is satisfied. □

Proof of Theorem 20. From Lemma 1, we consider the following five cases.
Case 1: $\pi^{-1}(a) < \pi^{-1}(a+2) < \pi^{-1}(a+1)$, where $1 \leq a \leq n-2$.

Suppose that the generalized transposition applied to π is

$$\phi(\pi^{-1}(a+2), \pi^{-1}(a+1)-1, \pi^{-1}(a+1), \pi^{-1}(a+1))$$

if $\pi^{-1}(a+2) = \pi^{-1}(a)+1$, and

$$\phi(\pi^{-1}(a)+1, \pi^{-1}(a+2)-1, \pi^{-1}(a+1), \pi^{-1}(a+1))$$

otherwise.
Case 2: $\pi^{-1}(a+1) < \pi^{-1}(a) < \pi^{-1}(a+2)$, where $1 \leq a \leq n-2$.

Suppose that the generalized transposition applied to π is

$$\phi(\pi^{-1}(a+1), \pi^{-1}(a+1), \pi^{-1}(a+1)+1, \pi^{-1}(a))$$

if $\pi^{-1}(a+2) = \pi^{-1}(a)+1$, and

$$\phi(\pi^{-1}(a+1), \pi^{-1}(a+1), \pi^{-1}(a)+1, \pi^{-1}(a+2)-1)$$

otherwise.
Case 3: $\pi^{-1}(a+2) < \pi^{-1}(a+1) < \pi^{-1}(a)$, where $1 \leq a \leq n-2$.

Suppose that the generalized transposition applied to π is

$$\phi(\pi^{-1}(a+2), \pi^{-1}(a+1)-1, \pi^{-1}(a+1)+1, \pi^{-1}(a)).$$

In any of the three cases stated above, we have that

$$(\pi \circ \phi)^{-1}(a+1) = (\pi \circ \phi)^{-1}(a)+1$$
$$\text{and } (\pi \circ \phi)^{-1}(a+2) = (\pi \circ \phi)^{-1}(a+1)+1.$$

Thus,

$$(a, a+1), (a+1, a+2) \in A(\pi \circ \phi) \cap A(e). \tag{5.2}$$

Case 4: $\pi^{-1}(c-1) < \pi^{-1}(b) < \pi^{-1}(c) < \pi^{-1}(b+1)$ and $\pi^{-1}(c) = \pi^{-1}(b)+1$, where $1 \leq b \leq n-1$ and $2 \leq c \leq n$.

Suppose that the generalized transposition applied to π is

$$\phi(\pi^{-1}(c-1)+1, \pi^{-1}(b), \pi^{-1}(c), \pi^{-1}(b+1)-1).$$

Case 5: $\pi^{-1}(b) < \pi^{-1}(c) < \pi^{-1}(b+1) < \pi^{-1}(c-1)$ and $\pi^{-1}(c) = \pi^{-1}(b)+1$, where $1 \leq b \leq n-1$ and $2 \leq c \leq n$.

Suppose that the generalized transposition applied to π is

$$\phi(\pi^{-1}(c), \pi^{-1}(b+1)-1, \pi^{-1}(b+1), \pi^{-1}(c-1)).$$

In either of the two cases stated above,

$$(\pi \circ \phi)^{-1}(b+1) = (\pi \circ \phi)^{-1}(b) + 1$$
$$\text{and } (\pi \circ \phi)^{-1}(c) = (\pi \circ \phi)^{-1}(c-1) + 1.$$

Hence,

$$(b, b+1), (c-1, c) \in A(\pi \circ \phi) \cap A(e). \tag{5.3}$$

From (5.2) and (5.3), in any of the above five cases it holds that $|A(\pi \circ \phi) \cap A(e)| \geq 2$. That is,

$$\begin{aligned} w_B(\pi \circ \phi) &= |A(\pi \circ \phi) \setminus A(e)| \\ &= |A(\pi \circ \phi)| - |A(\pi \circ \phi) \cap A(e)| \\ &\leq (n-1) - 2 = w_B(\pi) - 2. \end{aligned}$$

□

Remark 1. *For $\pi \in \mathbb{S}_n$, we define*

$$A'(\pi) = \{(\pi(i), \pi(i+1)) \mid 0 \leq i \leq n\},$$

where $\pi(0) = 0$ and $\pi(n+1) = n+1$. For any $\pi \in \mathbb{S}_n$, the number $b(\pi)$ of breakpoints is defined as follows:

$$b(\pi) = |A'(\pi) \setminus A'(e)|.$$

Note that the number of breakpoints is slightly different from the block permutation weight. Theorem 20 is similar to the next theorem in [21].

Theorem 21 (Christie [21]). *There exists $\phi \in \mathbb{T}_n$ such that*

$$b(\pi \circ \phi) \leq b(\pi) - 2$$

for any $\pi \in \mathbb{S}_n$.

Suppose that e is obtained from π by recursively applying the above generalized transposition k times. Christie showed that k is equal to the generalized Cayley distance $d_G(\pi, e)$ [21]. Whether the generalized transposition given in the proof of Theorem 20 has this property is unknown.

The next theorem shows that the generalized Cayley distance between any permutation and the identity permutation is bounded above by that between the characteristic permutation and the identity permutation. Recall from Definition 5 that the characteristic permutation between $\pi_1, \pi_2 \in \mathbb{S}_n$ is $\sigma \in \mathbb{D}_{d+1}$, which satisfies (5.1), where $d = d_B(\pi_1, \pi_2)$.

Theorem 22. *For any $\pi \in \mathbb{S}_n$, let $d_B(e, \pi) = k$, and suppose that $\sigma \in \mathbb{D}_{k+1}$ is the characteristic permutation between e and π. Then,*

$$d_G(\pi, e) \leq d_G(\sigma, e).$$

Proof. Suppose that $d_G(\sigma, e) = d$. That is,

$$e = \sigma \circ \phi_1 \circ \phi_2 \circ \cdots \circ \phi_d,$$

where $\phi_1, \phi_2, \ldots, \phi_d \in \mathbb{T}_{k+1}$. For $1 \leq t \leq d$, let

$$\phi_t = \phi(i_1^{(t)}, j_1^{(t)}, i_2^{(t)}, j_2^{(t)}),$$

where $1 \le i_1^{(t)} \le j_1^{(t)} < i_2^{(t)} \le j_2^{(t)} \le k+1$, and suppose that $\sigma = (\sigma^{(0)}(1), \sigma^{(0)}(2), \ldots, \sigma^{(0)}(k+1))$ and

$$\sigma \circ \phi_1 \circ \cdots \circ \phi_t = (\sigma^{(t)}(1), \sigma^{(t)}(2), \ldots, \sigma^{(t)}(k+1)).$$

From Definition 5, it holds that

$$\begin{aligned} e &= (\psi_1, \psi_2, \ldots, \psi_{k+1}), \\ \pi &= (\psi_{\sigma(1)}, \psi_{\sigma(2)}, \ldots, \psi_{\sigma(k+1)}), \end{aligned}$$

where ψ_i is a subsequence of e for $1 \le i \le k+1$. For $0 \le t \le d$ and $1 \le i \le k+1$, let $l_i^{(t)}$ be the length of $\psi_{\sigma^{(t)}(i)}$.

For $1 \le t \le d$, suppose that

$$\phi'_t = \phi(\sum_{i=1}^{i_1^{(t)}-1} l_i^{(t-1)} + 1, \sum_{i=1}^{j_1^{(t)}} l_i^{(t-1)}, \sum_{i=1}^{i_2^{(t)}-1} l_i^{(t-1)} + 1, \sum_{i=1}^{j_2^{(t)}} l_i^{(t-1)}).$$

Then,

$$\begin{aligned} &(\psi_{\sigma^{(t-1)}(1)}, \psi_{\sigma^{(t-1)}(2)}, \ldots, \psi_{\sigma^{(t-1)}(k+1)}) \circ \phi'_t \\ &\quad = (\psi_{\sigma^{(t)}(1)}, \psi_{\sigma^{(t)}(2)}, \ldots, \psi_{\sigma^{(t)}(k+1)}). \end{aligned}$$

Hence, it clearly follows that

$$e = \pi \circ \phi'_1 \circ \phi'_2 \circ \cdots \circ \phi'_d.$$

Therefore, $d_G(\pi, e) \le d = d_G(\sigma, e)$. $\square$

From Theorem 20 and Theorem 22, for any $\pi \in \mathbb{D}_n$ an upper bound on $d_G(\pi, e)$ is derived as follows.

Theorem 23. *For any $\pi \in \mathbb{D}_n$, it holds that*

$$d_G(\pi, e) \le \left\lceil \frac{1}{2}(n-1) \right\rceil. \tag{5.4}$$

Proof. We prove this by induction on n. Suppose that $n = 2$. Then, $\pi = (2, 1)$ and $d_G(\pi, e) = 1$, because $e = \pi \circ \phi(1, 1, 2, 2)$. Thus, (5.4) is satisfied.

Next, suppose that $n = 3$. From Theorem 20, we have that for some $\phi \in \mathbb{T}_3$

$$w_B(\pi \circ \phi) \le w_B(\pi) - 2 = 0$$

(Note that $w_B(\pi) = 2$ for any $\pi \in \mathbb{D}_3$.) Hence, $w_B(\pi \circ \phi) = 0$. That is, $\pi \circ \phi = e$. Therefore, $d_G(\pi, e) = 1$, which means that (5.4) is satisfied.

We assume that (5.4) is satisfied for $n \le k$. Suppose that $n = k+1$. For any $\pi \in \mathbb{D}_{k+1}$, it follows from Theorem 20 that there exists $\phi \in \mathbb{T}_{k+1}$ such that

$$w_B(\pi \circ \phi) \le w_B(\pi) - 2 = k - 2. \tag{5.5}$$

Let $h = w_B(\pi \circ \phi) = d_B(e, \pi \circ \phi)$. Suppose that $\sigma \in \mathbb{D}_{h+1}$ is the characteristic permutation between e and $\pi \circ \phi$. From Theorem 22, it holds that

$$d_G(\pi \circ \phi, e) \le d_G(\sigma, e). \tag{5.6}$$

It follows from (5.5) that $h \leq k-2$. That is, $h+1 \leq k-1$. For σ, it follows from the induction hypothesis that

$$d_G(\sigma, e) \leq \left\lceil \frac{1}{2}h \right\rceil \leq \left\lceil \frac{1}{2}(k-2) \right\rceil. \tag{5.7}$$

It follows from (5.6) and (5.7) that

$$d_G(\pi \circ \phi, e) \leq \left\lceil \frac{1}{2}(k-2) \right\rceil. \tag{5.8}$$

It follows from Theorem 13 and (5.8) that

$$\begin{aligned} d_G(\pi, e) &\leq d_G(\pi, \pi \circ \phi) + d_G(\pi \circ \phi, e) \\ &\leq 1 + \left\lceil \frac{1}{2}(k-2) \right\rceil = \left\lceil \frac{1}{2}k \right\rceil. \end{aligned}$$

Thus, (5.4) is satisfied for $n = k+1$. □

Remark 2. *Christie showed that inequality (5.4) holds for any $\pi \in \mathbb{S}_n$ [21, Section 5] but his proof needs Theorem 21, which is similar to Theorem 20 and observations on a graph model interpretation of permutations [21, Lemma 2 and Lemma 3].*

Next, we generalize Theorem 23 to any permutation in $\mathbb{S}_n$.

Theorem 24. *For any $\pi \in \mathbb{S}_n$, it holds that*

$$d_G(\pi, e) \leq \left\lceil \frac{1}{2}w_B(\pi) \right\rceil. \tag{5.9}$$

Proof. Let $h = w_B(\pi) = d_B(e, \pi)$. Suppose that $\sigma \in \mathbb{D}_{h+1}$ is the characteristic permutation between e and π. Then, we have from Theorem 22 that

$$d_G(\pi, e) \leq d_G(\sigma, e). \tag{5.10}$$

From Theorem 23, it holds for σ that

$$d_G(\sigma, e) \leq \left\lceil \frac{1}{2}h \right\rceil = \left\lceil \frac{1}{2}w_B(\pi) \right\rceil. \tag{5.11}$$

From (5.10) and (5.11), any $\pi \in \mathbb{S}_n$ satisfies (5.9). □

Finally, we derive a tighter upper bound on the generalized Cayley distance between any two permutations in $\mathbb{S}_n$.

Theorem 25. *For any $\pi_1, \pi_2 \in \mathbb{S}_n$, it holds that*

$$d_G(\pi_1, \pi_2) \leq \left\lceil \frac{1}{2}d_B(\pi_1, \pi_2) \right\rceil. \tag{5.12}$$

Proof. For $\pi_3 = \pi_2^{-1} \circ \pi_1 \in \mathbb{S}_n$, it follows from Theorem 24 that

$$d_G(\pi_3, e) \leq \left\lceil \frac{1}{2}w_B(\pi_3) \right\rceil = \left\lceil \frac{1}{2}d_B(\pi_3, e) \right\rceil. \tag{5.13}$$

It follows from Proposition 13 that

$$d_G(\pi_3, e) = d_G(\pi_2 \circ \pi_3, \pi_2 \circ e) = d_G(\pi_1, \pi_2). \tag{5.14}$$

Similarly, it follows from Proposition 15 that

$$d_B(\pi_3, e) = d_B(\pi_2 \circ \pi_3, \pi_2 \circ e) = d_B(\pi_1, \pi_2). \tag{5.15}$$

From (5.13), (5.14), and (5.15), we have that any $\pi_1, \pi_2 \in \mathbb{S}_n$ satisfy (5.12). □

We provide an example in which $d_G(\pi_1, \pi_2) = \lceil \frac{1}{2} d_B(\pi_1, \pi_2) \rceil$ is satisfied.

Example 5. *Let* $\pi_1 = (2, 8, 3, 1, 10, 5, 9, 4, 7, 6)$ *and* $\pi_2 = (3, 4, 10, 2, 7, 1, 5, 9, 6, 8)$. *Then,* $d_B(\pi_1, \pi_2) = 8$ *and* $\pi_2^{-1} \circ \pi_1 = (4, 10, 1, 6, 3, 7, 8, 2, 5, 9)$. *The directed graph of* $\pi_2^{-1} \circ \pi_1$ *described in Theorem 14 consists of the vertex set* $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ *and the edge set*

$$\{(0, 10), (1, 8), (2, 6), (3, 0), (4, 2), (5, 1), (6, 3), (7, 7), (8, 5), (9, 4), (10, 9)\}.$$

The cycles in this graph are as follows.

$$0 \to 10 \to 9 \to 4 \to 2 \to 6 \to 3 \to 0, 1 \to 8 \to 5 \to 1, 7 \to 7$$

Therefore, the number $c(\pi_2^{-1} \circ \pi_1)$ *of cycles is* 3. *From Theorem 14,*

$$d_G(\pi_1, \pi_2) = \frac{1}{2}(11 - c(\pi_2^{-1} \circ \pi_1)) = 4 = \left\lceil \frac{1}{2} d_B(\pi_1, \pi_2) \right\rceil.$$

In the next example, $d_G(\pi_1, \pi_2) = \lceil \frac{1}{2} d_B(\pi_1, \pi_2) \rceil$ is not satisfied.

Example 6. *Let* $\pi_1 = (9, 1, 6, 8, 5, 4, 10, 3, 2, 7)$ *and* $\pi_2 = (1, 5, 10, 2, 9, 6, 4, 8, 3, 7)$. *Then,* $d_B(\pi_1, \pi_2) = 9$. *Suppose* $\phi_1 = \phi(4, 5, 6, 6)$, $\phi_2 = \phi(2, 2, 3, 5)$, $\phi_3 = \phi(5, 7, 8, 8)$, *and* $\phi_4 = \phi(1, 5, 6, 9)$. *Then,*

$$\begin{aligned}
\pi_1 &= (9, 1, 6, 8, 5, 4, 10, 3, 2, 7),\\
\pi_1 \circ \phi_1 &= (9, 1, 6, \underline{4}, \underline{8, 5}, 10, 3, 2, 7),\\
\pi_1 \circ \phi_1 \circ \phi_2 &= (9, \underline{6, 4, 8}, \underline{1}, 5, 10, 3, 2, 7),\\
\pi_1 \circ \phi_1 \circ \phi_2 \circ \phi_3 &= (9, 6, 4, 8, \underline{3}, \underline{1, 5, 10}, 2, 7),\\
\pi_1 \circ \phi_1 \circ \phi_2 \circ \phi_3 \circ \phi_4 &= (\underline{1, 5, 10, 2}, \underline{9, 6, 4, 8, 3}, 7).
\end{aligned}$$

Since $\pi_1 \circ \phi_1 \circ \phi_2 \circ \phi_3 \circ \phi_4 = \pi_2$, $d_G(\pi_1, \pi_2) \le 4$. *That is,*

$$d_G(\pi_1, \pi_2) \neq 5 = \left\lceil \frac{1}{2} d_B(\pi_1, \pi_2) \right\rceil.$$

5.3 An Upper Bound on the Code Rate

In this section, we derive an upper bound on the optimal rate for codes with the generalized Cayley distance.

For a subset $C \subseteq \mathbb{S}_n$, we define

$$d_{G,min}(C) = \min\{d_G(\pi_1, \pi_2) \mid \pi_1, \pi_2 \in C, \pi_1 \neq \pi_2\}.$$

If a subset $C \subseteq \mathbb{S}_n$ satisfies

$$d_{G,min}(C) \ge 2t + 1,$$

C is a code of length n that can correct t generalized transpositions. For n and t, the optimal code $C_{G,opt}(n, t)$ is defined as follows.

$$C_{G,opt}(n, t) \in \mathop{\arg\max}_{C \subseteq \mathbb{S}_n, d_{G,min}(C) \ge 2t+1} |C|.$$

The optimal code rate $R_{G,opt}(n, t)$ for the code $C_{G,opt}(n, t)$ is defined as follows.

$$R_{G,opt}(n, t) = \frac{\log |C_{G,opt}(n, t)|}{\log n!}.$$

We define

$$B_G(n,t) = \{\pi \in \mathbb{S}_n \mid d_G(e,\pi) \le t\},$$
$$B_B(n,t) = \{\pi \in \mathbb{S}_n \mid d_B(e,\pi) \le t\}.$$

Let $b_G(n,t) = |B_G(n,t)|$ and $b_B(n,t) = |B_B(n,t)|$. Then, $R_{G,opt}(n,t)$ satisfies the following inequality.

Proposition 2.

$$1 - \frac{\log b_G(n,2t)}{\log n!} \le R_{G,opt}(n,t) \le 1 - \frac{\log b_G(n,t)}{\log n!}.$$

Yang et al. derived lower and upper bounds on $b_G(n,t)$ and $b_B(n,t)$.

Theorem 26 (Yang et al. [20])**.** *For all n and $t \le \min\{n - \sqrt{n} - 1, \frac{n-1}{4}\}$,*

$$\binom{n-1}{4t}\frac{(2t)!}{2^t t!} \le b_G(n,t) \le \prod_{k=0}^{4t}(n-k).$$

Theorem 27 (Yang et al. [20])**.** *For all n and $t \le n - \sqrt{n} - 1$,*

$$\prod_{k=1}^{t}(n-k) \le b_B(n,t) \le \prod_{k=0}^{t}(n-k).$$

From Proposition 2 and Theorem 26, the following inequality is satisfied.

Theorem 28 (Yang et al. [20])**.** *For all n and $t \le \min\{n - \sqrt{n} - 1, \frac{n-1}{4}\}$,*

$$R_{G,opt}(n,t) \le 1 - \frac{\log \binom{n-1}{4t}\frac{(2t)!}{2^t t!}}{\log n!}.$$

In this section, our upper bound on the generalized Cayley distance is employed to derive another upper bound on $R_{G,opt}(n,t)$.

We first derive another lower bound on $b_G(n,t)$.

Theorem 29. *For all n and $t \le \frac{n-\sqrt{n}-1}{2}$, it holds that*

$$b_G(n,t) \ge \prod_{k=1}^{2t}(n-k).$$

Proof. For every $\pi \in B_B(n,2t)$, $d_B(e,\pi) \le 2t$. From Theorem 25, we have that

$$\begin{aligned} d_G(e,\pi) &\le \left\lceil \frac{1}{2} d_B(e,\pi) \right\rceil < \frac{1}{2} d_B(e,\pi) + 1 \\ &\le \frac{1}{2} \cdot 2t + 1 = t + 1. \end{aligned}$$

Hence, $d_G(e,\pi) < t+1$, i.e., $d_G(e,\pi) \le t$. That is, $\pi \in B_G(n,t)$. Thus, $B_B(n,2t) \subseteq B_G(n,t)$. If $2t \le n - \sqrt{n} - 1$, i.e., $t \le \frac{n-\sqrt{n}-1}{2}$, it follows from Theorem 27 that

$$b_G(n,t) \ge b_B(n,2t) \ge \prod_{k=1}^{2t}(n-k).$$

□

When n and t satisfy $t \leq \min\{\frac{n-\sqrt{n}-1}{2}, \frac{n-1}{4}\}$ and

$$\prod_{k=2t+1}^{4t} (n-k) < 2^t \cdot t! \cdot \prod_{k=2t+1}^{4t} k, \tag{5.16}$$

our lower bound on $b_G(n,t)$ is tighter than the one given in Theorem 26.

From Theorem 29 and Proposition 2, another upper bound on $R_{G,opt}(n,t)$ can be derived.

Theorem 30. *For all n and $t \leq \frac{n-\sqrt{n}-1}{2}$, it holds that*

$$R_{G,opt}(n,t) \leq 1 - \frac{\sum_{k=1}^{2t} \log(n-k)}{\log n!}. \tag{5.17}$$

Our upper bound on $R_{G,opt}(n,t)$ is tighter than that in Theorem 28 when (5.16) is satisfied.

5.4 Conclusion

In this dissertation, we have derived the tighter upper bound on the generalized Cayley distance. Our upper bound is equal to nearly half of the block permutation distance. Furthermore, we employed our upper bound to derive an upper bound on the optimal rate for codes with the generalized Cayley distance. Our upper bound on the rate is tighter than that given by Yang et al. [20] when the code length is relatively small.

An explicit construction of order-optimal systematic codes with the generalized Cayley distance was developed by Yang et al. [24]. However, in their scheme, the code length must be sufficiently large. Our future work is to develop a construction of systematic codes with short code lengths.

Chapter 6

Concluding Remarks

In this dissertation, some kinds of coding schemes for flash memory applications are investigated.

Index-less indexed flash code with inversion cells (I-ILIFC) is a coding scheme to prolong the lifetime of flash memory. We analyzed the worst-case performance of I-ILIFC and specified a threshold for the code length that determines whether I-ILIFC improves the worst-case performance of index-less indexed flash code (ILIFC), which is the underlying scheme of I-ILIFC. Additionally, we modified the encoding scheme of I-ILIFC and derived another threshold for the code length comparing the performances of ILIFC and I-ILIFC. Consequently, we relaxed the sufficient condition for improving the performance of ILIFC.

Parallel random I/O (P-RIO) code is a coding scheme to increase the reading speed of multilevel flash memory. It is known that the construction of random I/O (RIO) code, which is the underlying scheme of P-RIO code, is equivalent to that of well-studied write-once memory (WOM) code. Coset coding is a technique to construct WOM codes using linear binary codes. In coset coding, we leveraged information of all stored pages to construct P-RIO codes that have parameters for which RIO codes do not exist. In this dissertation, we used $(7, 4)$ Hamming code and $(15, 11)$ Hamming code as linear codes in coset coding. It is difficult to generalize our approach to the general $(2^r - 1, 2^r - r - 1)$ Hamming code or other linear codes. This motivates us to develop another construction of P-RIO codes.

Permutation code is a coding scheme to correct errors in flash memory. In permutation code, different distances depending on the kind of errors are considered. The generalized Cayley distance is the number of generalized transposition errors that are needed to transform one permutation to another. Generalized transpositions are swapping any two substrings of a permutation and subsume some other errors such as transpositions and translocations. Since the exact value of the generalized Cayley distance is complicated to compute, another distance called block permutation distance was proposed to construct permutation codes with the generalized Cayley distance. In this dissertation, we derived a tighter upper bound on the generalized Cayley distance using the block permutation distance. Additionally, we employed our upper bound to derive an upper bound on the optimal rate for codes with generalized Cayley distance. An explicit construction of order-optimal systematic codes with the generalized Cayley distance was developed by Yang et al. Our future work is to develop a construction of systematic codes with short code lengths. Our theorem used to derive our upper bound on the distance may be useful to construct such codes.

Acknowledgments

I would like to express the deepest appreciation to my advisors, Professor Hiroshi Kamabe and Assistant Professor Shan Lu, for their continual and excellent guidance, caring, patience, and providing me with an excellent atmosphere for doing research. I would never have been able to finish my dissertation without their guidance. I would like to thank Professor Keiichirou Kusakari, Professor Miwako Mishima, and Doctor Hironori Uchikawa for their useful comments and suggestions.

Finally, I would like to thank my parents for supporting me and encouraging me with their best wishes.

Bibliography

[1] A. Jiang, V. Bohossian, and J. Bruck, "Floating codes for joint information storage in write asymmetric memories," *Proc. IEEE International Symposium on Information Theory, 2007 (ISIT 2007)*, pp. 1166-1170, Nice, France, June 2007.

[2] A. Jiang and J. Bruck, "Joint coding for flash memory storage," *Proc. IEEE International Symposium on Information Theory, 2008 (ISIT 2008)*, pp. 1741-1745, Toronto, Canada, July 2008.

[3] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, "Rank modulation for flash memories," *IEEE Trans. Inf. Theory*, vol. 55, pp. 2659-2673, May 2009.

[4] A. Jiang, M. Langberg, R. Mateescu, and J. Bruck, "Data movement and aggregation in flash memories," *Proc. IEEE International Symposium on Information Theory, 2010 (ISIT 2010)*, pp. 1918-1922, Austin, Texas, June 2010.

[5] E. Yaakobi, A. Jiang, P. H. Siegel, A. Vardy, and J. K. Wolf, "On the parallel programming of flash memory cells," *Proc. 2010 IEEE Information Theory Workshop (ITW)*, Dublin, Ireland, August-September 2010.

[6] H. Mahdavifar, P. H. Siegel, A. Vardy, J. K. Wolf, and E. Yaakobi, "A nearly optimal construction of flash codes," *Proc. IEEE International Symposium on Information Theory, 2009 (ISIT 2009)*, pp. 1239-1243, Seoul, South Korea, June-July 2009.

[7] M. Ohkawa and H. Kamabe, "Index-less indexed flash codes using inversion bit" (in Japanese), *Proc. 35th Symposium on Information Theory and Its Applications*, pp. 117-122, Oita, Japan, December 2012.

[8] Y. Kaji, "The expected write deficiency of index-less flash codes and their improvement," *Proc. 2011 IEEE Information Theory Workshop (ITW)*, pp. 35-39, Paraty, Brazil, October 2011.

[9] R. Suzuki and T. Wadayama, "Modified index-less indexed flash codes for improving average performance," *IEICE Trans. Fundamentals (Japanese Edition)*, vol. J94-A, no. 12, pp. 991-1000, 2011.

[10] A. Yamawaki and H. Kamabe, "Maximization of average number of rewriting of ILIFC with inversion operation," *Proc. 2016 International Symposium on Information Theory and Its Applications (ISITA 2016)*, pp. 136-140, Monterey, California, USA, October-November 2016.

[11] E. Sharon and I. Alrod, "Coding scheme for optimizing random I/O performance," arXiv preprint, arXiv:1202.6481, 2012, presented at *Non-Volatile Memories Workshop*, San Diego, CA, USA, Mar. 2013.

[12] R. L. Rivest and A. Shamir, "How to reuse a write-once memory," *Inf. Control*, vol. 55, no. 1-3, pp. 1-19, Dec. 1982.

[13] E. Yaakobi and R. Motwani, "Construction of random input-output codes with moderate block lengths," *IEEE Trans. Commun.*, vol. 64, no. 5, May 2016.

[14] G. D. Cohen, P. Godlewski, and F. Merkx, "Linear binary code for write-once memories," *IEEE Trans. Inf. Theory*, vol. IT-32, no. 5, pp. 697-700, Oct. 1986.

[15] R. Gabrys, E. Yaakobi, F. Farnoud, F. Sala, J. Bruck, and L. Dolecek, "Codes correcting erasures and deletions for rank modulation," *IEEE Trans. Inf. Theory*, Vol. 62, pp. 136-150, Jan. 2016.

[16] N. Raviv, E. Yaakobi, and M. Médard, "Coding for locality in reconstructing permutations," in *Proc. IEEE Int. Symp. Inf. Theory*, pp. 450-454, Barcelona, Spain, July 2016.

[17] Y. Zhang and G. Ge, "Snake-in-the-box codes for rank modulation under Kendall's τ-metric," *IEEE Trans. Inf. Theory*, Vol. 62, pp. 151-158, Jan. 2016.

[18] F. Farnoud, V. Skachek, and O. Milenkovic, "Error-correction in flash memories via codes in the Ulam metric," *IEEE Trans. Inf. Theory*, Vol. 59, pp. 3003-3020, Jan. 2013.

[19] Y. M. Chee and V. K. Vu, "Breakpoint analysis and permutation codes in generalized Kendall tau and Cayley metrics," in *Proc. IEEE Int. Symp. Inf. Theory*, pp. 2959-2963, Honolulu, HI, USA, June-July 2014.

[20] S. Yang, C. Schoeny, and L. Dolecek, "Order-optimal permutation codes in the generalized Cayley metric," in *Proc. IEEE Inf. Theory Workshop*, Kaohsiung, Taiwan, Nov. 2017.

[21] D. A. Christie, "Sorting permutations by block-interchanges," *Inform. Process. Lett.*, 60, pp. 165-169, 1996.

[22] E. Yaakobi, A. Vardy, P. H. Siegel, and J. K. Wolf, "Multidimensional flash codes," in *Proc. 46-th Allerton Conference on Communication, Control and Computing*, pp. 392-399, Monticello, IL, Sep. 2008.

[23] Y. Zhang, E. Yaakobi, and T. Etzion, "Bounds on the length of functional PIR and batch codes," arXiv preprint, arXiv:1901.01605, 2019.

[24] S. Yang, C. Schoeny, and L. Dolecek, "Theoretical bounds and constructions of codes in the generalized Cayley metric," arXiv preprint, arXiv:1803.04314, 2018.

Peer-Reviewed Publications

1. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, “Worst-case performance of ILIFC with inversion cells,” *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E100-A, No. 12, pp. 2662-2670, 2017.

2. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, “Construction of parallel random I/O codes using coset coding with Hamming codes,” *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E101-A, No. 12, pp. 2125-2134, 2018.

3. Shan Lu, Hiroshi Kamabe, Jun Cheng and Akira Yamawaki, “Unrestricted-rate parallel random input-output codes for multilevel flash memory,” *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E101-A, No. 12, pp. 2135-2140, 2018.

4. Akira Yamawaki and Hiroshi Kamabe, “Maximization of average number of rewriting of ILIFC with inversion operation,” Proc. of *the International Symposium on Information Theory and Its Applications (ISITA2016)*, pp.136-140, 2016.

5. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, “Lower bounds on the number of write operations by index-less indexed flash code with inversion cells,” Proc. of *IEEE International Symposium on Information Theory (ISIT2017)*, pp. 2488-2492, 2017.

6. Shan Lu, Hiroshi Kamabe, Jun Cheng, and Akira Yamawaki, “Construction of unrestricted-rate parallel random input-output code,” Proc. of *IEEE International Symposium on Information Theory (ISIT2017)*, pp. 629-633, 2017.

7. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, “Construction of parallel RIO codes using coset coding with Hamming codes,” Proc. of *IEEE Information Theory Workshop (ITW2017)*, 2017.

8. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, “An upper bound on the generalized Cayley distance,” Proc. of *the International Symposium on Information Theory and Its Applications (ISITA2018)*, pp. 632-636, 2018.

Technical Publications

1. Akira Yamawaki, Hironori Uchikawa, and Hiroshi Kamabe, "Maximization of number of rewritings for index-less indexed flash codes with bits for reversal," (in Japanese), *IEICE Technical Report*, vol. 115, no. 137, IT2015-32, pp. 89-94, 2015.

2. Akira Yamawaki and Hiroshi Kamabe, "Optimal ratio of data and reversal cells for ILIFC with reversal cells," (in Japanese), Proc. of *Symposium on Information Theory and Its Application (SITA 2015)*, pp. 697-702, 2015.

3. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, "Lower bounds on the number of rewriting by ILIFC with inversion cell," (in Japanese), Proc. of *Symposium on Information Theory and Its Application (SITA 2016)*, pp. 63-67, 2016.

4. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, "Construction of parallel RIO codes using coset coding with Hamming codes," (in Japanese), *IEICE Technical Report*, vol. 117, no. 39, IT2017-13, pp. 73-78, 2017.

5. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, "An upper bound on the generalized Cayley distance," (in Japanese), *IEICE Technical Report*, vol. 118, no. 205, IT2018-28, pp. 7-12, 2018.

6. Akira Yamawaki, Hiroshi Kamabe, and Shan Lu, "On construction of permutation codes with generalized Cayley distance," (in Japanese), Proc. of *Symposium on Information Theory and Its Application (SITA 2018)*, pp. 161-166, 2018.