DOI:10.1587/transfun.2022EAP1058

Publicized:2022/09/21

PAPER

Concatenated Permutation Codes under Chebyshev Distance

Motohiko KAWASUMI[†], *Nonmember and* Kenta KASAI[†], *Member*

SUMMARY Permutation codes are error-correcting codes over symmetric groups. We focus on permutation codes under Chebyshev (ℓ_∞) distance. A permutation code invented by Kløve et al. is of length n, size 2^{n-d} and, minimum distance d. We denote the code by $\phi_{n,d}$. This code is the largest known code of length n and minimum Chebyshev distance $d > n/2$ so far, to the best of the authors knowledge. They also devised efficient encoding and hard-decision decoding (HDD) algorithms that outperform the bounded distance decoding.

In this paper, we derive a tight upper bound of decoding error probability of HDD. By factor graph formalization, we derive an efficient maximum a-posterior probability decoding algorithm for $\phi_{n,d}$. We explore concatenating permutation codes of $\phi_{n,d=0}$ with binary outer codes for more robust error correction. A naturally induced pseudo distance over binary outer codes successfully characterizes Chebyshev distance of concatenated permutation codes. Using this distance, we upper-bound the minimum Chebyshev distance of concatenated codes. We discover how to concatenate binary linear codes to achieve the upper bound. We derive the distance distribution of concatenated permutation codes with random outer codes. We demonstrate that the sum-product decoding performance of concatenated codes with outer low-density parity-check codes outperforms conventional schemes.

key words: *permutation codes, Chebyshev distance, ℓ_∞ distance, concatenated codes*

1. Introduction

In this paper, we consider *permutation codes*. A permutation code is a subset of all permutations of fixed length n. The origin of permutation codes dates back to the 1960s [1]. Vinck et al. proposed applications of permutation codes for power-line communication and m-ary (frequency shift keying) FSK modulation system [2], [3] which renewed subsequent interest in permutation codes [4]–[6]. Frequencies in an m-ary FSK system are used in specific time slots to describe the permutation symbols. Time- and frequency-diversity are used to overcome various types of noise: background noise, impulse noise, and permanent frequency disturbances common in power-lines.

When we view a permutation code as a modulation, the map from an information message to a permutation codeword is called *rank modulation*. Rank modulation for flash memories is a significant application [7]–[9]. *Flash memory* is an electronic non-volatile computer memory storage medium that can be electrically erased and reprogrammed. Removing charge in flash memories can be done only by removing the entire charge (erase) from a large block of cells, while injecting charge (program) into a flash cell is a simple operation. Multi-level cells (MLC) store two or more bits per flash cell, while single-level cells (SLC) store a bit per flash cell. MLC flash devices cost less and allow for higher storage density than SLC but have short lifetimes as low as several thousands of erasures. Jiang et al. proposed the application of the rank modulation scheme for flash memories in [9]. Rank modulation eliminates the need for discrete cell levels. The standard (amplitude) modulation represents the information by their absolute levels of flash memory cells. In contrast, the rank modulation scheme represents information by relative levels.

DNA storage is another significant application of permutation codes. *Shotgun sequencing* is a reading technique of long DNA strings. In shotgun sequencing, DNA is broken up randomly into numerous small segments. Such process identifies a DNA string from a *profile vector*: a possibly inaccurate histogram of sub-strings of a DNA string. In [10], Kiah et al. introduced the DNA storage channel whose outputs are profile vectors. They suggested protecting the profile vector by permutation codes [10, Sec. VIII.B].

The purpose of this paper is not to make a direct contribution to the above applications of permutation codes. In fact, the proposed method is not always suitable for the above applications. Rather, the purpose of this paper is to make the fundamental contribution of introducing a philosophy of soft-decodable sparse-graph codes [11], which has been very successful in binary codes, to permutation codes.

In [12], Wadayama and Hagiwara invented permutation codes based on linearly constrained permutation matrices. A relaxed problem of the maximum likelihood decoding problem of the codes can be viewed as linear-programming (LP) since permutation matrices are vertices of the Birkhoff polytope. One can use efficient LP solvers based on simplex methods or interior point methods for decoding the codes. From the geometrical properties of a code polytope, they derived an upper bound on LP decoding error probability of randomly constrained permutation codes. However, although LP decoding is far efficient compared with ML decoding, in practice, it is not satisfactorily efficient. The reason is that the LP decoder deals with matrices of size $n \times n$ as its variables. It is hard to demonstrate LP decoding of long permutation codes, [12] uses codes of relatively small length 64 for demonstrating the performance.

The Hamming distance is the most important distance for binary vector codes. In contrast, there are various types of distances for permutation codes: Chebyshev distance [13], [14], Hamming distance [14], Kendall-tau distance [15], and

[†]The author is with Tokyo Institute of Technology

Ulam distance [16]. Since the concept of minimum distance leads to bounded distance decoding (BDD), it plays a central role in coding theory. Lots of efforts have been dedicated to designing permutation codes with large minimum distance.

Under the multilevel flash memory model, the distance induced by the ℓ_∞ norm, which is known as the Chebyshev distance, is appropriate for studying the recharging and error-correcting issues. Among the many distances for permutation codes, Chebyshev distance has been intensively studied: Gilbert–Varshamov bound and ball-packing bound [17]–[19], efficient encoding and decoding algorithms [13], [17], and systematic code constructions [20], [21].

It is important to note that Kløve et al. [13, Explicit Construction] and Tamo et al. [17, Construction 1] independently discovered a construction of permutation code under Chebyshev distance. The size of the code is given by $(\lceil n/d \rceil !)^{n \bmod d} (\lfloor n/d \rfloor !)^{d-(n \bmod d)}$ with linearly growing minimum Chebyshev distance d with code length n. For large minimum Chebyshev distance d such that $n/2 < d < n$, the size becomes 2^{n-d}. To the best of the authors knowledge, this is the largest known code construction of length n and linear growing minimum Chebyshev distance $d > n/2$ so far. They also devised very efficient encoding and decoding algorithms. The decoding algorithm realizes BDD, which corrects all errors within distance $d/2$ from the received word.

Kløve et al. discovered the permutation codes of length n, of size 2^{n-d} and minimum Chebyshev distance d [13, Sec. III D]. The size is the same as the largest codes we mentioned above for $d > n/2$. We denote the code by $\phi_{n,d}$. They also devised efficient encoding and hard-decision decoding (HDD) algorithms. In this paper, we show that HDD is better than BDD.

Consider binary binary codes of length n. The maximum size is $M_b = 2^n$. This means that each code bit has $\frac{1}{n} \log M_b = 1$ bits of information. In contrast, permutation codes can have a size of up to $M_p = n!$. This means that each code symbol has $\frac{1}{n} \log M_p = \frac{1}{n}(n \log n - n + O(\log n)) = \log n - 1 + o(1)$ bits of information. This is much larger than that of binary codes. This is the major advantage of using permutation codes. However, the permutation code $\phi_{n,d}$ has less than or equal to one bit of information per code symbol. In this paper, we deal with several types of n-ary input channels for evaluating the decoding performance. Every capacity of these channels is greater than one. The purpose of this paper is not to establish a permutation coding system which achieves the capacity of the channels. The purpose is to propose a permutation coding system which is more effective than the conventional method for a wide range of channels.

In conventional studies on permutation codes, a lot of effort has been put into mainly constructing methods that increase the minimum distance for a given distance metric. The minimum distance is the most important measure of error correction capability of BDD since BDD can correct all errors within half the minimum distance from the received word. In contrast, sparse graph codes have been achieved a great success in binary linear codes: the codes achieve the channel capacity over various types of binary-input channels [11]. Sparse graph codes are decoded by an efficient soft iterative message-passing algorithm called the sum-product (SP) algorithm. The SP decoder can usually correct errors more than half the minimum distance. This is the reason why minimum distance is not considered most important in sparse-graph coding-theory [22, §13.2. and §13.8.] and [11, §1.6.]. Although a lot of research has been done on permutation codes, the aspect of efficient soft decoding has not yet been extensively studied. To the best of the author's knowledge, the only study done for this purpose is [12]. Therefore, there is still a lot of room for research on permutation codes with efficient soft decoding algorithms. One of the goals of this paper is to give a sparse-graph coding-theory perspective to permutation codes. Specifically, we will formulate permutation codes using sparse graphs, and developed a soft iterative decoder that can correct more errors than half the minimum distance of conventional permutation codes.

In order to evaluate the performance of permutation codes, in this paper, we define three types of n-ary input channels. This does not mean that we are not only interested in these specific channels. We believe that the proposed method is not only effective for the three channels but also for other channels such as channels with memory. Our aim is to show that the proposed method is effective for a wide variety of channels. In this paper, we focus on the three channels that the most people will likely be interested in.

This paper consists of three parts. In the first part, we investigate $\phi_{n,d}$ under HDD and MAP decoding. In the second and third part, in order to make the codes robust for errors, we explore concatenated codes with $\phi_{n,d=0}$ and binary codes as inner and outer codes, respectively. In the second part, we study the distance distribution and minimum distance of concatenated codes. We mainly consider SP decoding for concatenated codes in this paper. However, the minimum distance of concatenated codes is important since it is the error correction capability of BDD with Chebyshev metric. Efficient BDD algorithms for proposed codes may be developed in the future. The third part deals with concatenated codes with low-density parity-check (LDPC) codes as outer codes. The contributions in this paper are summarized as follows:

1. Performance analysis on $\phi_{n,d}$
   a. We derive a tight upper bound of the error probability of HDD.
   b. We devise efficient bit-wise and block-wise maximum a-posterior probability (MAP) decoding algorithms for $\phi_{n,d}$.
2. Concatenation with binary codes: minimum distance analysis
   a. We introduce pseudo-distance over outer code space, which successfully characterizes the Chebyshev distance of concatenated permutation codes.

   b. We derive the distance distribution of concatenated permutation codes with outer random binary and linear codes.
   c. We upper-bound the minimum Chebyshev distance of concatenated codes. We discover how to concatenate outer binary linear codes that achieve the upper bound.

3. Concatenation with LDPC codes: SP decoding performance

   a. We formulate the SP decoder of concatenated codes with outer LDPC codes.
   b. We demonstrate the SP decoding performance of the codes and observe that the codes outperform conventional schemes: un-concatenated codes $\phi_{n,d}$ under HDD.

The remaining of the paper is organized as follows. Section 2 introduces notations and defines channels. Section 3 derives the BDD error probability of general permutation codes. Section 4 defines $\phi_{n,d}$ and reviews the efficient encoding and HDD algorithms for $\phi_{n,d}$. Section 5 derives an upper bound of the error probability of HDD and devises MAP decoding algorithms. Section 6 defines concatenated codes, introduces a pseudo distance, and derives the distance distribution of concatenated permutation codes with outer random binary and linear codes. Furthermore, we demonstrate the SP decoding performance of concatenated codes with outer LDPC codes. Section 7 concludes this paper.

2. Notations and Preliminaries

We define $\mathbb{1}[\mathsf{S}]$ as 1 if a statement S is true and as 0 otherwise. We denote a binary field by $\mathbb{F}_2 = \{0, 1\}$. Vectors are described like $u_1^n = (u_1, \ldots, u_n)$. When the components of a vector are clear from the context, we simply denote the vector underlined, e.g., $\underline{u} = u_1^n$. We denote random variables by capital, e.g., X. By $\mathbb{E}[X]$ and $\mathrm{Var}[X]$, we denote the mean and variance of X, respectively. For two integers a and b with $a \le b$, we denote $\{a, a+1, \ldots, b\}$ by $[a : b]$. We denote $\{1, \ldots, n\}$ by $[n]$. Define S_n as the set of all permutations of $[n]$. We denote a permutation on S_n as a vector: $\underline{\sigma} := (\sigma_1, \ldots, \sigma_n) := \begin{pmatrix} 1 & \cdots & n \\ \sigma(1) & \cdots & \sigma(n) \end{pmatrix}$. We denote the identity permutation by $\underline{\iota}$.

2.1 Chebyshev Distance and Permutation Codes

Definition 1 (Chebyshev Distance)**.** For two vectors $\underline{\pi}, \underline{\sigma} \in \mathbb{R}^n$, Chebyshev distance between $\underline{\pi}$ and $\underline{\sigma}$ is defined as follows:

$$d_\infty(\underline{\pi}, \underline{\sigma}) = \max_{j: 1 \le j \le n} |\pi_j - \sigma_j|.$$

Note that $0 \le d_\infty(\underline{\pi}, \underline{\sigma}) \le n-1$ for $\underline{\pi}, \underline{\sigma} \in S_n$. In other literature, this distance is known as ℓ_∞ distance defined over $\mathbb{R}^n$.

We give an example. Chebyshev distance between $\underline{\pi} = (12345)$ and $\underline{\sigma} = (25413)$ is

$$\begin{aligned} d_\infty(\underline{\pi}, \underline{\sigma}) &= \max\{|1-2|, |2-5|, |3-4|, |4-1|, |5-3|\} \\ &= \max\{1, 3, 1, 3, 2\} \\ &= 3. \end{aligned}$$

Definition 2 (Permutation Codes)**.** We say that a set of permutations $P \subset S_n$ is a *permutation code* of length n. We refer to the elements of a permutation code as codewords. Permutation codes are called permutation arrays in literature. We say the minimum Chebyshev distance of permutation code P is d and write $d_\infty(P) = d$ if $d_\infty(\underline{\pi}, \underline{\sigma}) \ge d$ for any two distinct permutations $\underline{\pi}, \underline{\sigma} \in P$ and $d_\infty(\underline{\pi}, \underline{\sigma}) = d$ for some $\underline{\pi}, \underline{\sigma} \in P$. We say $P \subset S_n$ is an (n, M, d) permutation code if P is of length n, size M, and minimum Chebyshev distance d.

We give an example of an (8,16,4) permutation code. Table 1 lists 16 codewords in the-right-most column. In this paper, we will investigate such a type of codes in detail. We will explain how this code is constructed and other columns in the following sections.

2.2 Channels

In this section, we define three types of n-ary input channels. Each channel deals with permutation $\underline{\pi} \in S_n$ as input and n-tuple $\underline{\sigma} \in \mathcal{Y}^n$ as output, where $\mathcal{Y}$ is the set of output symbols. The channels are all memoryless:

$$P_{\Sigma_1^n | \Pi_1^n}(\sigma_1^n | \pi_1^n) = \prod_{j=1}^{n} P_{\Sigma_j | \Pi_j}(\sigma_j | \pi_j).$$

Definition 3 (AWGN Channel)**.** We define an additive white Gaussian noise (AWGN) channel with input $\underline{\pi} \in S_n$ and output $\underline{\sigma} \in \mathbb{R}^n$ as follows. Each transition probability is defined as

$$P_{\Sigma_j | \Pi_j}(\sigma_j | \pi_j) \overset{\text{def}}{=} \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{(\pi_j - \sigma_j)^2}{2s^2}}, (j = 1, \ldots, n),$$

where $s^2 \ge 0$ is the variance of the noise. We denote the channel by AWGNC(s^2). Let us consider a generic modulation: $[n] \to C$, where C is a constellation of signals. In order to save transmitting power, for an even number $n > 0$, we employ a simple modulation: n-level PAM (pulse-amplitude modulation) with n level value constellation $C = \{\pm(\frac{1}{2} + j) \mid j = 0, 1, \ldots, n/2 - 1\}$ and we naturally map $j \in [n]$ to $-(n/2 - 1/2) + (j - 1) \in C$. For example, the 4-level PAM maps $1, 2, 3, 4$ to $-1.5, -0.5, 0.5, 1.5$, respectively. Of course, one can use more sophisticated modulation methods, but we do not go into details in this paper. The SNR is calculated as $\mathrm{SNR[dB]} = 10 \log_{10} \frac{\frac{1}{n}\sum_{j=0}^{n/2-1}(1/2+j)^2}{s^2}$.

Definition 4 (n-ary Symmetric Channel)**.** We define n-ary symmetric channel n-SC(δ) of error probability $\epsilon \in [0, (n-1)/n]$ with input $\underline{\pi} \in S_n$ and output $\underline{\sigma} \in [n]^n$ as follows:

$$P_{\Sigma_j|\Pi_j}(\sigma_j|\pi_j) \overset{\text{def}}{=} \begin{cases} 1-\epsilon & (\text{if } \sigma_j = \pi_j), \\ \frac{\epsilon}{n-1} & (\text{if } \sigma_j \neq \pi_j). \end{cases}$$

The channel is noiseless when $\epsilon = 0$, i.e., the conditional entropy of σ_j given π_j is 0. The channel is completely noisy when $\epsilon = (n-1)/n$, i.e., the conditional entropy of σ_j given π_j is $\log_2(n)$.

Discussion 1. In [13, §I.], Kløve et al. implied the reason why they use Chebyshev metric for AWGN channels. We briefly review the reason as follow. Suppose that $\underline{\pi}$ is sent and $\underline{\sigma}$ is received through AWGN(s^2). The likelihood is given as a function of the square of the Euclidean distance between $\underline{\pi}$ and $\underline{\sigma}$ as follows.

$$\begin{aligned} P_{\underline{\Sigma}|\underline{\Pi}}(\underline{\sigma}|\underline{\pi}) &= \prod_{j=1}^{n} P_{\Sigma_j|\Pi_j}(\sigma_j|\pi_j) \\ &= \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{(\sigma_j-\pi_j)^2}{2s^2}} \\ &= \frac{1}{\left(\sqrt{2\pi s^2}\right)^n} e^{-\frac{d_E(\underline{\sigma},\underline{\pi})^2}{2s^2}} \end{aligned}$$

The probability that $\underline{\sigma}$ contains a component that is significantly different from the one transmitted is very small. Chebyshev distance can be viewed as a metric so that the distance becomes large once such a component exists by replacing the $\prod$ in likelihood with a min as follows,

$$\begin{aligned} \min_{j\in[n]} P_{\Sigma_j|\Pi_j}(\sigma_j|\pi_j) &= \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{\max_{j\in[n]}(\sigma_j-\pi_j)^2}{2s^2}} \\ &= \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{(\max_{j\in[n]}|\sigma_j-\pi_j|)^2}{2s^2}} \\ &= \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{d_\infty(\underline{\sigma},\underline{\pi})^2}{2s^2}}. \end{aligned}$$

Definition 5 (Erasure Channel)**.** For some $\delta \in (0, 1]$, we define erasure channel EC(δ) with input $\underline{\pi} \in S_n$ and output $\underline{\sigma} \in ([n] \cup \{?\})^n$ as follows:

$$P_{\Sigma_j|\Pi_j}(\sigma_j|\pi_j) \overset{\text{def}}{=} \begin{cases} \delta & (\sigma_j = ?), \\ 1-\delta & (\sigma_j = \pi_j), \\ 0 & (\text{otherwise}). \end{cases}$$

The erasure channel can be viewed as a simplified channel model of n-ary FSK with impulsive noise [3].

3. Decoding Performance of Bounded Distance Decoder

Kløve et al. [13, Explicit Construction] and Tamo et al. [17, Construction 1] independently discovered a construction of an (n, M, d) permutation code with

$$M = (\lceil n/d \rceil !)^{n \bmod d} (\lfloor n/d \rfloor !)^{d-(n \bmod d)}.$$

For fixed d, the size M grows as $\Theta((n/d)!)$. For large d such that $n/2 < d < n$, the size M scales as $M = 2^{n-d}$. To the best of the authors knowledge, this is the largest known code construction of length n and linear growing[†] minimum Chebyshev distance $d > n/2$ so far. Note that there still remains a gap to the best-known upper bound [17, Theorem 24] of the size of codes with linear growing minimum Chebyshev distance. They also devised very efficient encoding and decoding algorithms. The decoding algorithm realizes BDD.

In this section, we evaluate the BDD performance of an (n, M, d) permutation code.

Definition 6 (Bounded Distance Decoder)**.** Let P be an (n, M, d) permutation code.

$$\underline{\hat{\pi}}^{(\mathrm{BD})}(\underline{\sigma}) = \begin{cases} \underline{\pi}' & \text{if } \exists \underline{\pi}' \in P, d_\infty(\underline{\sigma}, \underline{\pi}') < d/2 \\ \text{error} & \text{otherwise} \end{cases}$$

Uniqueness of $\underline{\hat{\pi}}(\underline{\sigma})$ is ensured from the assumption that the minimum Chebyshev distance of P is d.

Assume transmissions take places over AWGN(s^2) channels $P_{\Sigma_1^n|\Pi_1^n}(\sigma_1^n|\pi_1^n)$. We consider a question: how large does the minimum Chebyshev distance d need to scale with n so that the BDD error probability goes to zero? The BDD fails if and only if $d_\infty(\underline{\sigma}, \underline{\pi}) \geq d/2$. Therefore, it holds that $P(\underline{\hat{\Pi}}^{(\mathrm{BD})}(\underline{\Sigma}) \neq \underline{\Pi}) = P(d_\infty(\underline{\Sigma}, \underline{\Pi}) \geq d/2)$. There are independently and identically distributed (iid) random variables Z_j so that $\Sigma_j = \Pi_j + Z_j$ for $j = 1, \ldots, n$ each Z_j $(j = 1, \ldots, n)$ is a Gaussian variable with mean 0 and variance s^2. Note that $d_\infty(\underline{\Sigma}, \underline{\Pi}) = \max_j |Z_j|$. Let $W := d_\infty(\underline{\Sigma}, \underline{\Pi})$. From [23, Th. 3.12], it is known that $\mathbb{E}[W] \leq s\sqrt{2\ln(n)}$ and $\mathrm{Var}[W] \leq s^2$. For arbitrary $\epsilon \geq 0$, if we choose

$$d/2 = s(\sqrt{2\ln(n)} + 1/\epsilon) \geq \mathbb{E}[W] + \sqrt{s^2/\epsilon}, \tag{1}$$

then we have

$$\begin{aligned} P(W \geq d/2) &\leq P(|W - \mathbb{E}[W]| \geq d/2 - \mathbb{E}[W]) \\ &\leq \frac{\mathrm{Var}[W]}{(d/2 - \mathbb{E}[W])^2} \\ &\leq \epsilon, \end{aligned}$$

where we used Chebyshev inequality and the bounds on $\mathbb{E}[W]$ and $\mathrm{Var}[W]$. From this, we can see that W exists within $d/2$ from the mean under Chebyshev distance with high probability. In other words, if $d/2$ is larger than this, the decoding will succeed with high probability. This is the

[†]This code is not the largest for fixed d. We can see this from observing that $\{123, 312, 231\}$ is an $(n = 3, M = 3, d = 2)$ permutation code and $M > 2^{n-d}$.

reason why we set $d/2$ as given in (1).

We will give another simple derivation for a similar result. First, recall the distribution of the maximum value of independent random variables can be simply written as

$$\begin{aligned}P(W \geq d/2) &= P\big(\max_j\{|Z_j| : 1 \leq j \leq n\} \geq d/2\big)\\ &= 1 - P\big(\max_j\{|Z_j| : 1 \leq j \leq n\} < d/2\big)\\ &= 1 - P(|Z_1| < d/2)^n.\end{aligned}$$

The numerical evaluation of this will be given in Section 5.4. Using this, we have the following result. For arbitrary $\epsilon>0$, if we choose $d/2 \geq s(\sqrt{2\ln(2n/\epsilon)})$, then it holds that

$$\begin{aligned}P(W \geq d/2) &\overset{\text{(a)}}{=} 1 - \left(1 - 2Q\left(\frac{d/2}{\sqrt{s^2}}\right)\right)^n \qquad (2)\\ &\overset{\text{(b)}}{\leq} 2nQ\left(\frac{d/2}{\sqrt{s^2}}\right)\\ &\overset{\text{(c)}}{\leq} 2ne^{-\frac{d^2}{8s^2}}\\ &\overset{\text{(d)}}{\leq} \epsilon\end{aligned}$$

where we used Q-function $\left(Q(x) \overset{\text{def}}{=} \frac{1}{\sqrt{2\pi}}\int_x^\infty \exp\left(-\frac{u^2}{2}\right) du\right)$ in (a), Bernoulli inequality in (b), Chernoff bound in (c), and we used the assumption $d/2 \geq s(\sqrt{2\ln(2n/\epsilon)})$ in (d).

4. Efficiently Encodable/Decodable Permutation Code

Kløve et al. developed a recursive construction of permutation codes [13, Sec. III C]. They also devised very efficient encoding and hard-decision decoding algorithms for some of those codes [13, Sec.III D]. The codes are of length n, of minimum Chebyshev distance d and of size 2^{n-d}. We denote the code by $\phi_{n,d}$ in this paper. In this section, we review the encoder and decoder for $\phi_{n,d}$ devised in [13].

4.1 Encoding Algorithm of $\phi_{n,d}$

Definition 7 ([13, Sec.III D])**.** The following algorithm consisting of two steps encodes a binary information vector u_1^{n-d} into a codeword $\underline{\pi}_1^n \in \phi_{n,d} \subset S_n$. With a little abuse of notation, we also denote the encoder by $\phi_{n,d} : \mathbb{F}_2^{n-d} \ni u_1^{n-d} \mapsto \pi_1^n \in S_n$. With this notaion, the code $\phi_{n,d}$ is given by the image $\phi_{n,d}(\mathbb{F}_2^{n-d})$. Two vectors $x_1^n \in \mathbb{F}_2^n$ and $t_1^{n+1} \in [0 : n]^{n+1}$ will be used as auxiliary variables.

1 Append d zeros at the end of information vector $(u_1, \dots, u_{n-d})$. Denote the resulting vector by x_1^n. To be precise, $x_1^{n-d} := u_1^{n-d}$ and $x_{n-d+1} = \cdots = x_n = 0$. Set $t_1 = 0$.

2 Repeat the following for $j = 1, \dots, n$. This step determines π_j and t_{j+1} from x_j and t_j so that

$$t_{j+1} = \begin{cases} t_j + 1, & (x_j = 0),\\ t_j, & (x_j = 1),\end{cases} \qquad (3)$$

$$\pi_j = \begin{cases} t_j + 1, & (x_j = 0),\\ n - (j - t_j - 1), & (x_j = 1).\end{cases} \qquad (4)$$

We can see that the total complexity is $O(n)$. Denote the sets consisting of possible values that t_j and π_j take, respectively, by $\mathcal{T}_j := \{0, 1, \dots, j-1\}$ and $\mathcal{P}_j(t_j) := \{t_j + 1, n - j + t_j + 1\}$ for $j = 1, \dots, n$.

Remark 1. Note that $|\mathcal{T}_j| = j$ for every j and $|\mathcal{P}_j(t_j)| = 2$ for $j = 1, \dots, n-1$ and $|\mathcal{P}_n(t_n)| = 1$. We see that t_j and $j - t_j - 1$ count the number of 0's and 1's appearing in $(x_1, \dots, x_{j-1})$, respectively. Denote (3) and (4) by $t_{j+1}(x_j, t_j)$ and $\pi_j(x_j, t_j)$, respectively. We have

$$\pi_j(1, t_j) - \pi_j(0, t_j) = n - j \geq 0 \qquad (5)$$

for any $1 \leq j \leq n$. In words, π_j takes the larger and smaller value in $\mathcal{P}_j(t_j)$ when $x_j = 0$, $x_j = 1$, respectively.

Example 1. Consider $\phi_{n=8,d=4}$. Table 1 lists information vectors u_1^4, the corresponding auxiliary vectors x_1^8, counter vectors t_1^9 and codewords π_1^8. From this table, we confirm

Table 1 A permutation code $\phi_{8,4}$

u_1^4	x_1^8	t_1^9	π_1^8
0000	00000000	012345678	12345678
1000	10000000	001234567	81234567
0100	01000000	011234567	18234567
1100	11000000	000123456	87123456
0010	00100000	012234567	12834567
1010	10100000	001123456	81723456
0110	01100000	011123456	18723456
1110	11100000	000012345	87612345
0001	00010000	012334567	12384567
1001	10010000	001223456	81273456
0101	01010000	011223456	18273456
1101	11010000	000112345	87162345
0011	00110000	012223456	12873456
1011	10110000	001112345	81762345
0111	01110000	011112345	18762345
1111	11110000	000001234	87651234

that the minimum Chebyshev distance of $\phi_{n=8,d=4}$ is 4. In contrast, we observe that the minimum Hamming distance of $\phi_{n=8,d=4}$ is 2, where the nearest pair of codewords is 81234567 and 18234567.

4.2 Hard-Decision Decoding Algorithm of $\phi_{n,d}$

Lots of efforts have been devoted to constructing permutation codes with large minimum Chebyshev distance. Some of those codes are equipped with efficient encoding algorithms. However, few of them have efficient decoding algorithms. In this section, we review the recursive HDD algorithm [13, Sec.III D] for $\phi_{n,d}$. The algorithm efficiently and sequentially estimates each transmitted symbol π_j for $j = 1, \dots, n$. Below we describe a slightly modified version of [13, Sec. III D]. Specifically, we introduce a random tie-breaking rule for symmetry.

Definition 8 (Hard-decision Decoder)**.** Denote the received symbols $\sigma_1^n \in \mathbb{R}^n$. The decoder outputs estimated binary information bits u_1^{n-d} which is equal to x_1^{n-d} as $\hat{x}_1^{n-d}$. The decoder also estimates the transmitted symbols and auxiliary variables, respectively as $\hat{\pi}_1^{n-d}$ and $\hat{t}_1^{n-d+1}$.

(1) Set $j = 1$ and $\hat{t}_1 = 0$.
(2) Repeat the following for $j = 1, \ldots, n-d$. Let μ_j be the middle point between $\pi_j(0, \hat{t}_j)$ and $\pi_j(1, \hat{t}_j)$: $\mu_j := \big(\pi_j(0, \hat{t}_j) + \pi_j(1, \hat{t}_j)\big)/2$, where $\pi_j(x_j, t_j)$ is a function defined in (4). First, choose $\hat{x}_j \in \{0, 1\}$ so that $\pi_j(\hat{x}_j, \hat{t}_j)$ is closer to σ_j. To be precise,

$$\hat{x}_j = \begin{cases} 0 & (\sigma_j < \mu_j), \\ \mathrm{Ber}(1/2) & (\sigma_j = \mu_j), \\ 1 & (\sigma_j > \mu_j), \end{cases} \tag{6}$$

where $\mathrm{Ber}(1/2)$ is a random variable taking a value of 0 or 1 with a probability 1/2. For symmetry, we added such a tie-breaking rule on $\hat{x}_j$ (6). Next, substituting $\hat{x}_j$ and $\hat{t}_j$ into (4) and (3), we get $\hat{\pi}_j$ and $\hat{t}_{j+1}$ as follows: $\hat{\pi}_j = \pi_j(\hat{x}_j, \hat{t}_j), \hat{t}_{j+1} = t_{j+1}(\hat{x}_j, \hat{t}_j)$.

We can see that the total complexity is $O(n)$.

5. HDD Performance and MAP decoding of $\phi_{n,d}$

In this section, we first derive an upper bound of HDD error probability over AWGN(s^2). Numerical results demonstrate the tightness of the bound. Next, we devise an efficient MAP decoding algorithm based on factor graphs.

5.1 Decoding Performance of Hard-Decision Decoder over Channels

In this section, we analytically evaluate the HDD performance for $\phi_{n,d}$. First, we give a sufficient condition for successful estimation.

Lemma 1 (Sufficient Condition of Successful HDD)**.** Consider the HDD of $\phi_{n,d}$. We assume the same notations and conditions we use in Definition 8. Let $j \in [1 : n-d]$. Assume the decoder successfully estimates $x_1, \ldots x_{j-1}$, i.e., $\hat{x}_1^{j-1} = x_1^{j-1}$. Then it holds that

$$\sigma_j < \pi_j + (n-j)/2 \Longrightarrow \hat{x}_j = x_j, \text{ if } x_j = 0,$$

and

$$\sigma_j > \pi_j - (n-j)/2 \Longrightarrow \hat{x}_j = x_j, \text{ if } x_j = 1.$$

Proof. Recall that $\hat{x}_j \in \{0, 1\}$ is chosen so that $\pi_j(\hat{x}_j, \hat{t}_j)$ is closer to σ_j. From (5), we know that the difference between candidates is $\pi_j(1, t_j) - \pi_j(0, t_j) = n - j$. First, consider the case with $x_j = 0$. From the assumption $\hat{x}_1^{j-1} = x_1^{j-1}$, the decoder also successfully estimates the counter $\hat{t}_1^j = t_1^j$ and the transmitted symbol $\pi_j(0, \hat{t}_j) = \pi_j(0, t_j) = \pi_j$. Recalling the first rule of (6), we see that it is sufficient to show $\sigma_j < \mu_j$. This is true since $\sigma_j < \pi_j + (n-j)/2 = \pi_j(0, \hat{t}_j) + \big(\pi_j(1, \hat{t}_j) - \pi_j(0, \hat{t}_j)\big)/2 = \mu_j$. The case with $x_j = 1$ can be shown similarly. □

Using this Lemma, we will derive an upper bound of the error probability. The numerical evaluation of this bound will be given in Section 5.4.

Theorem 1 (HDD performance)**.** We assume the same notations and conditions we use in Definition 8. For transmissions over AWGNC(s^2), the word error probability $P(\hat{X}_1^{n-d} \neq X_1^{n-d})$ of HDD for $\phi_{n,d}$ is upper-bounded as follows:

$$P(\hat{X}_1^{n-d} \neq X_1^{n-d}) \leq \sum_{j=1}^{n-d} Q\Big(\frac{n-j}{2\sqrt{s^2}}\Big). \tag{7}$$

Furthermore, for arbitrary $\epsilon \geq 0$, if we choose $d \geq \sqrt{8s^2 \ln(n/\epsilon)}$, then we have $P(\hat{X}_1^{n-d} \neq X_1^{n-d}) \leq \epsilon$.

Proof. Let E_j be the event that the first decoding error occurs at the j-th position, more precisely, $\hat{X}_1 = X_1, \ldots, \hat{X}_{j-1} = X_{j-1}$ and $\hat{X}_j \neq X_j$. The decoding error event $\hat{X}_1^{n-d} = X_1^{n-d}$ is equal to the union of E_j for $j = 1, 2, \ldots, n-d$. Since E_j is disjoint, the union bound holds with equality:

$$\begin{aligned} &P(\hat{X}_1^{n-d} \neq X_1^{n-d}) \\ &= P\Big(\bigcup_{j=1}^{n-d} E_j\Big) \\ &= \sum_{j=1}^{n-d} P(E_j) \\ &= \sum_{j=1}^{n-d} P(\hat{X}_j \neq X_j, \hat{X}_1^{j-1} = X_1^{j-1}). \end{aligned} \tag{8}$$

Each term in (8) is bounded as follow:

$$\begin{aligned} &P(\hat{X}_j \neq X_j, \hat{X}_1^{j-1} = X_1^{j-1}) \\ &= P(\hat{X}_j \neq X_j | \hat{X}_1^{j-1} = X_1^{j-1}) P(\hat{X}_1^{j-1} = X_1^{j-1}) \\ &\leq P(\hat{X}_j \neq X_j | \hat{X}_1^{j-1} = X_1^{j-1}) \\ &\overset{(a)}{=} P(X_j = 0 | \hat{X}_1^{j-1} = X_1^{j-1}) P(\hat{X}_j \neq X_j | X_j = 0, \hat{X}_1^{j-1} = X_1^{j-1}) \\ &\quad + P(X_j = 1 | \hat{X}_1^{j-1} = X_1^{j-1}) P(\hat{X}_j \neq X_j | X_j = 1, \hat{X}_1^{j-1} = X_1^{j-1}) \\ &\overset{(b)}{=} P(X_j = 0) P(\hat{X}_j \neq X_j | X_j = 0, \hat{X}_1^{j-1} = X_1^{j-1}) \\ &\quad + P(X_j = 1) P(\hat{X}_j \neq X_j | X_j = 1, \hat{X}_1^{j-1} = X_1^{j-1}) \\ &\overset{(c)}{\leq} \frac{1}{2}\Big(P(\Sigma_j \geq \Pi_j + \frac{n-j}{2} | X_j = 0, \hat{X}_1^{j-1} = X_1^{j-1}) \\ &\quad + P(\Sigma_j \leq \Pi_j - \frac{n-j}{2} | X_j = 1, \hat{X}_1^{j-1} = X_1^{j-1})\Big) \end{aligned}$$

$$\overset{\text{(d)}}{=} \frac{1}{2}\Big(P(\Sigma_j - \Pi_j \geq \frac{n-j}{2}) + P(\Sigma_j - \Pi_j \leq -\frac{n-j}{2})\Big)$$

$$\overset{\text{(e)}}{=} P\big(\Sigma_j - \Pi_j \geq \frac{n-j}{2}\big)$$

$$= \sum_{j=1}^{n-d} Q\Big(\frac{n-j}{2\sqrt{s^2}}\Big).$$

The equality (a) uses a marginalization and (b) uses the fact that X_j is independent of whether $\hat{X}_1^{j-1} = X_1^{j-1}$ is true or false. In (c), we used Lemma 1. In (d), we used the fact that AWG noise $\Sigma_j - \Pi_j$ is statistically independent of $X_j, \hat{X}_1^{j-1}, X_1^{j-1}$. In (e), we used symmetry of Gaussian: $P\big(\Sigma_j - \Pi_j \geq \frac{n-j}{2}\big) = P\big(\Sigma_j - \Pi_j \leq -\frac{n-j}{2}\big)$. This concludes the first claim.

Furthermore, we can bound the error probability as follows.

$$P(\hat{X}_1^{n-d} \neq X_1^{n-d}) \leq (n-d)Q\Big(\frac{d}{2\sqrt{s^2}}\Big) \leq ne^{-d^2/8s^2}, \quad (9)$$

where we used that Q is a decreasing funciton and the Chernoff bound: $Q(x) \leq e^{-\frac{x^2}{2}}$. Substituting the assumption $d \geq \sqrt{8s^2 \ln(n/\epsilon)}$ into (9), we obtain $P(\hat{X}_1^{n-d} \neq X_1^{n-d}) \leq \epsilon$ which concludes the proof. □

5.2 Bit-wise MAP Decoding of $\phi_{n,d}$

In this section, we evaluate the bit-wise MAP decoding performance of $\phi_{n,d}$ for various channels. We assume that the information vector u_1^{n-d} is chosen uniformly at random. To this end, we take a standard approach. We first formulate the bit-wise MAP decoding problem as a marginal of factorized function. Next, we draw a factor graph: a graphical model representing a factorization of function consisting of many variables. Finally, we devise bit-wise MAP decoding as a SP algorithm on the factor graph that is a tree [24].

Let $\underline{x} = x_1^{n-d}$ be an information vector that is chosen uniformly at random. Then, $\underline{x}$ is encoded into a codeword $\pi_1^n = \phi_{n,d}(\underline{x})$. Let $\underline{\sigma} = \sigma_1^n$ be a received vector through one of the three channels. Define the bit-wise MAP decoder by

$$\hat{x}_k(\underline{\sigma}) \overset{\text{def}}{=} \operatorname*{argmax}_{x_k \in \mathbb{F}_2} p_{X_k|\underline{\Sigma}}(x_k|\underline{\sigma}) \text{ for } k = 1, \dots, n. \quad (10)$$

Let $\underline{t} = t_1^{n+1}$ be the auxiliary counters. We claim that the bit-wise MAP decoding can be accomplished by marginalizing a factorized function:

$$\hat{x}_k = \operatorname*{argmax}_{x_k \in \mathbb{F}_2} \sum_{\sim x_k} \mathbb{1}[t_1 = 0] \prod_{j=1}^{n} P_{\Sigma_j|\Pi_j}(\sigma_j|\pi_j)$$
$$\times P_{\Pi_j|X_j,T_j}(\pi_j|x_j,t_j) P_{T_{j+1}|T_j,X_j}(t_{j+1}|t_j,x_j) P_{X_1^n}(x_1^n). \quad (11)$$

From the assumption, we see that $P_{X_1^n}(x_1^n)$ is further factorized into $P_{X_1^n}(x_1^n) = \frac{1}{2^{n-d}} \prod_{j=n-d+1}^{n} \mathbb{1}[x_j = 0]$. We used the notation $\sum_{\sim x_k}$ as a summation over all variables $x_1^n, \pi_1^n, t_1^{n+1}$ except x_k. Note that σ_1^n is not a variable but a fixed value. In the remainder of this subsection, when it is clear from the context, we will abbreviate random variables and write $P(x)$ and $P(x|y)$ instead of $P_X(x)$ and $P(X = x|Y = y)$, respectively.

We will show (11). We start by writing

$$\hat{x}_k = \operatorname*{argmax}_{x_k \in \mathbb{F}_2} \sum_{\sim x_k} P_{\underline{\Pi},\underline{X},\underline{T}|\underline{\Sigma}}(\underline{\pi},\underline{x},\underline{t}|\underline{\sigma}). \quad (12)$$

Next, we factorize $P(\underline{\pi},\underline{x},\underline{t}|\underline{\sigma})$ as follows:

$$\begin{aligned} &P(\underline{\pi},\underline{x},\underline{t}|\underline{\sigma}) \\ &= P(\underline{\sigma}|\underline{\pi},\underline{x},\underline{t})P(\underline{\pi}|\underline{x},\underline{t})P(\underline{t}|\underline{x})P(\underline{x})/P(\underline{\sigma}) \\ &= P(\underline{\sigma}|\underline{\pi})P(\underline{\pi}|\underline{x},\underline{t})P(\underline{t}|\underline{x})P(\underline{x})/P(\underline{\sigma}). \end{aligned} \quad (13)$$

In the last equality, we used the fact that $(\underline{x},\underline{t}) \leftrightarrow \underline{\pi} \leftrightarrow \underline{\sigma}$ forms a Markov chain. We ignore the factor $1/P(\underline{\sigma})$ since it does not affect argmax for $\hat{\pi}_j$. Further, we will factorize those four factors. Since we assume memoryless channels, we have $P(\underline{\sigma}|\underline{\pi}) = \prod_{j=1}^n P(\sigma_j|\pi_j)$. Similarly, we have

$$P(\underline{\pi}|\underline{x},\underline{t}) \overset{\text{(a)}}{=} \prod_{j=1}^{n} P(\pi_j|\pi_1,\dots,\pi_{j-1},\underline{x},\underline{t})$$
$$\overset{\text{(b)}}{=} \prod_{j=1}^{n} P(\pi_j|x_j,t_j),$$

where (a) is due to the chain rule and (b) holds since π_j depends only on (x_j, t_j) from (4). Similarly we have,

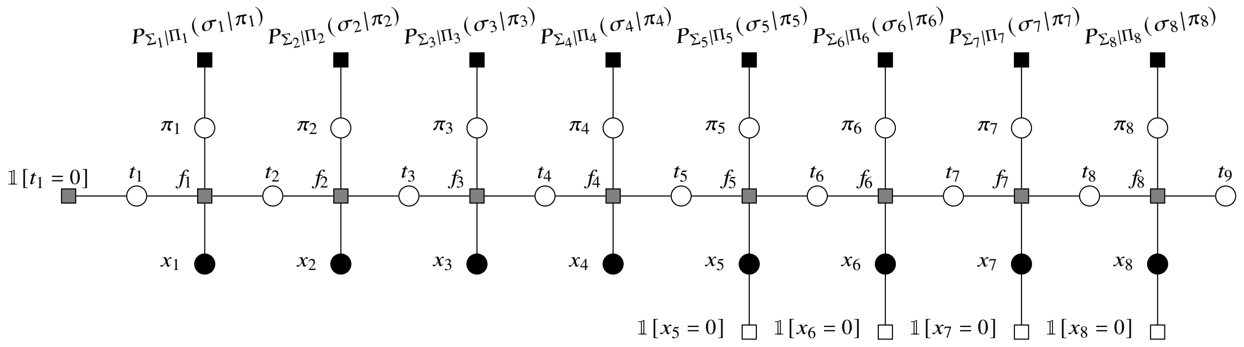
$$\begin{aligned} P(\underline{t}|\underline{x}) &= \mathbb{1}[t_1 = 0] \prod_{j=1}^{n-1} P(t_{j+1}|t_1,\dots,t_j,\underline{x}) \\ &= \mathbb{1}[t_1 = 0] \prod_{j=1}^{n-1} P(t_{j+1}|t_j,x_j). \end{aligned}$$

Since $\underline{x}$ is uniformly chosen with fixed $x_j = 0$ for $j = n-d+1,\dots,n$, we have $P(\underline{x}) = \frac{1}{2^{n-d}} \prod_{j=n-d+1}^{n} \mathbb{1}[x_j = 0]$. Substituting all factors into (13), we see the claim is true.

Figure 1 depicts the factor graph (tree) of factors inside argmax in (11) for $\phi_{8,4}$. It is known that the SP algorithm calculates the exact value of the marginal function when the factor graph is a tree [24]. Obviously, the factor graph forms a tree for general $\phi_{n,d}$. We see that the SP algorithm on the factor graph for $\phi_{n,d}$ realizes the bit-wise MAP decoding (10).

Let us write the decoding algorithm as forward and backward message updates as in the BCJR algorithm [25] or, more generally, in the Baum-Welch algorithm [26]. Denote the forward and backward messages by $\alpha_j(t_j)$ and $\beta_j(t_j)$, respectively. These are corresponding to the SP message from t_j to f_j and from t_{j+1} to f_j, respectively. The update rules for forward, backward, and estimate steps are given in (14), (15) and (16), respectively.

Let us consider the computation complexity of the algorithm. Since the number of multiplications and evaluating



$P_{\Sigma_1|\Pi_1}(\sigma_1|\pi_1)$
$P_{\Sigma_2|\Pi_2}(\sigma_2|\pi_2)$
$P_{\Sigma_3|\Pi_3}(\sigma_3|\pi_3)$
$P_{\Sigma_4|\Pi_4}(\sigma_4|\pi_4)$
$P_{\Sigma_5|\Pi_5}(\sigma_5|\pi_5)$
$P_{\Sigma_6|\Pi_6}(\sigma_6|\pi_6)$
$P_{\Sigma_7|\Pi_7}(\sigma_7|\pi_7)$
$P_{\Sigma_8|\Pi_8}(\sigma_8|\pi_8)$
π_1 π_2 π_3 π_4 π_5 π_6 π_7 π_8
$\mathbb{1}[t_1=0]$
t_1 f_1 t_2 f_2 t_3 f_3 t_4 f_4 t_5 f_5 t_6 f_6 t_7 f_7 t_8 f_8 t_9
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
$\mathbb{1}[x_5=0]$ $\mathbb{1}[x_6=0]$ $\mathbb{1}[x_7=0]$ $\mathbb{1}[x_8=0]$

Fig. 1 The factor graph of factors inside argmax in (11) for $\phi_{8,2}$. Each factor $P_{\Pi_j|X_j,T_j}(\pi_j|x_j,t_j)\,P_{T_{j+1}|T_j,X_j}(t_{j+1}|t_j,x_j)$ is denoted by f_j for $j=1,\ldots,8$.

`Forward Step:` $\alpha_1(t_1)=1$ for $t_1\in\mathcal{T}_1=\{0\}$

$$\alpha_{j+1}(t_{j+1})=\sum_{t_j\in\mathcal{T}_j}\sum_{\pi_j\in\mathcal{P}_j(t_j)}\sum_{x_j\in\mathbb{F}_2}P(\pi_j|x_j,t_j)P(t_{j+1}|t_j,x_j)P(\pi_j|\sigma_j)g_j(x_j)\alpha_j(t_j)\text{ for }t_{j+1}\in\mathcal{T}_{j+1},\,j=1,\ldots,n, \tag{14}$$

`Backward Step:` $\beta_{n+1}(t_{n+1})=1$ for $t_{n+1}\in\mathcal{T}_{n+1}=\{1,\ldots,n\}$

$$\beta_j(t_j)=\sum_{t_{j+1}\in\mathcal{T}_{j+1}}\sum_{\pi_j\in\mathcal{P}_j(t_j)}\sum_{x_j\in\mathbb{F}_2}P(\pi_j|x_j,t_j)P(t_{j+1}|t_j,x_j)P(\pi_j|\sigma_j)g_j(x_j)\beta_{j+1}(t_{j+1})\text{ for }t_j\in\mathcal{T}_j,\,j=n,\ldots,1, \tag{15}$$

`Estimate Step:`

$$\hat{x}_j=\operatorname*{argmax}_{x_j\in\mathbb{F}_2}\sum_{t_j\in\mathcal{T}_j}\sum_{t_{j+1}\in\mathcal{T}_{j+1}}\sum_{\pi_j\in\mathcal{P}_j(t_j)}P(\pi_j|x_j,t_j)P(t_{j+1}|t_j,x_j)P(\pi_j|\sigma_j)\alpha_j(t_j)\beta_{j+1}(t_{j+1})\text{ for }x_j\in\mathbb{F}_2,\,j=1,\ldots,n. \tag{16}$$

probabilities in (14) are proportional to that of additions, we focus on additions. For each j, a naive implementation of (14) needs about $\#\mathcal{T}_j\times\#\mathcal{T}_{j+1}\times\#\mathcal{P}_j\times\#\mathbb{F}_2=j(j+1)2^2$ additions. For $j=1,\ldots,n$, the total number of additions amount to $O(n^3)$. This can be reduced to $O(n^2)$ with the following alternative algorithm.

We can write $\mathbb{1}[\text{eq. (3) holds}]=P(t_{j+1}|t_j,x_j)$ and $\mathbb{1}[\text{eq. (4) holds}]=P(\pi_j|x_j,t_j)$ since they are deterministic. From this observation, we see that $\alpha_{j+1}(t_{j+1})$ can be calculated as follows:

```
1 First, set $\alpha_{j+1}(t_{j+1}) := 0$ for all $t_{j+1}\in\mathcal{T}_{j+1}$.
2 Next, for all $t_j\in\mathcal{T}_j, x_j\in\mathbb{F}_2$, repeat the following.
    i   Calculate $t_{j+1}(t_j,x_j)$ by (3).
    ii  Calculate $\pi_j(t_j,x_j)$ by (4).
    iii Accumulate $\alpha_{j+1}(t_{j+1})$ += $P(\pi_j|\sigma_j)g_j(x_j)\alpha_j(t_j)$.
```

This needs $\#\mathcal{T}_j\times\#\mathbb{F}_2=2j$ additions for each $j=1,\ldots,n$. The total number of additions amount to $O(n^2)$. Similarly, we can calculate backward and estimate steps with $O(n^2)$ additions. Consequently, we see that the total computational complexity of the algorithm is $O(n^2)$.

5.3 Viterbi Decoding Algorithm of $\phi_{n,d}$

In this section, we evaluate the block-wise MAP decoding performance of $\phi_{n,d}$ for various channels. We assume that the information vector u_1^{n-d} is chosen uniformly at random. In this section, we follow the method of realizing Viterbi decoding described in [24, IV. B] and [11, 2.5.5] with message passing over factor graphs. Consider block-wise MAP decoding:

$$\underline{\hat{x}}:=\operatorname*{argmax}_{\underline{x}\in\mathbb{F}_2^n}P_{\underline{X}|\underline{\Sigma}}(\underline{x}|\underline{\sigma})$$

Write $\underline{t}$ and $\underline{\pi}$ determined by (3) and (4) from $\underline{x}$ as $\underline{t}(\underline{x})$ and $\underline{\pi}(\underline{x})$, respectively.

$$P_{\underline{X}|\underline{\Sigma}}(\underline{x}|\underline{\sigma})=P_{\underline{\Pi},\underline{T},\underline{X}|\underline{\Sigma}}\left(\underline{\pi}(\underline{x}),\underline{t}(\underline{x}),\underline{x}|\underline{\sigma}\right)$$

If $\underline{\pi}\neq\underline{\pi}(\underline{x})$ or $\underline{t}\neq\underline{t}(\underline{x})$, then $P_{\underline{\pi},\underline{t},\underline{x}|\underline{\Sigma}}\left(\underline{\pi},\underline{t},\underline{x}|\underline{\sigma}\right)=0$, so we have

$$\underline{\hat{x}}=\operatorname*{argmax}_{\underline{x}\in\mathbb{F}_2^n}\max_{\underline{t},\underline{\pi}}P_{\underline{\Pi},\underline{T},\underline{X}|\underline{\Sigma}}\left(\underline{\pi},\underline{t},\underline{x}|\underline{\sigma}\right).$$

The k-th element of $\underline{\hat{x}}$ is written as $\left(\underline{\hat{x}}\right)_k$. To emphasize the difference from bit-wise MAP decoding, we denote it as follows.

$$\left(\underline{\hat{x}}\right)_k=\operatorname*{argmax}_{x_k\in\mathbb{F}_2}\max_{\sim x_k}P_{\underline{\Pi},\underline{T},\underline{X}|\underline{\Sigma}}\left(\underline{\pi},\underline{t},\underline{x}|\underline{\sigma}\right)$$

This can be regarded as a replacement of the sum of (12) with a max. We conclude that the block-wise MAP decoding can be realized by replacing every sum in forward (14), backward (15) and estimation step (16) by a max.

5.4 Numerical Evaluation

In this subsection, we numerically evaluate the decoding

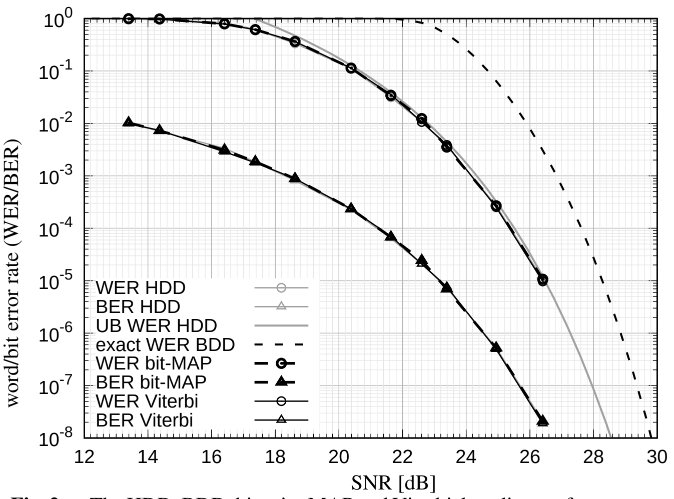


Fig. 2 The HDD, BDD, bit-wise MAP and Viterbi decoding performance evaluation of $\phi_{n,d}$ with $n = 512, d = 64$ over AWGN(s^2). The WER and BER of HDD, bit-wise MAP and Viterbi decoding are almost the same. The dashed black curve draws the exact word error probability of BDD, which is given in (2). The solid gray curve draws the upper bound of the HDD error probability given in (7). The SNR is calculated as $\mathrm{SNR[dB]} = 10\log_{10}\frac{1}{n}\sum_{j=0}^{n/2-1}(1/2+j)^2/s^2$.

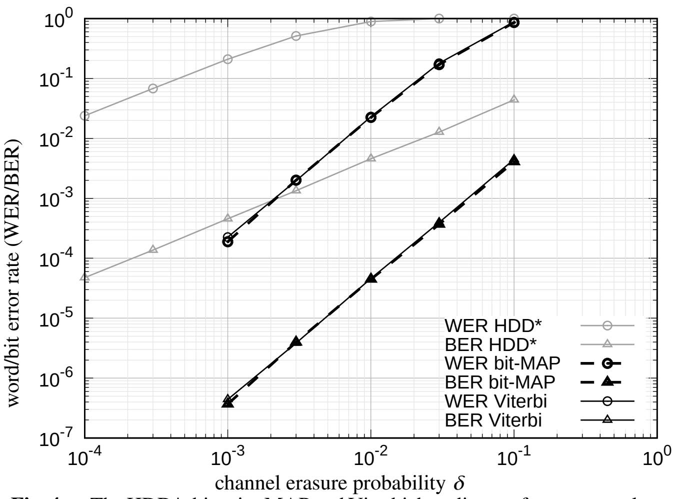


Fig. 4 The HDD*, bit-wise MAP and Viterbi decoding performance evaluation of $\phi_{n,d}$ with $n = 512, d = 64$ over EC(δ). The WER and BER of bit-wise MAP and Viterbi decoding are almost the same.

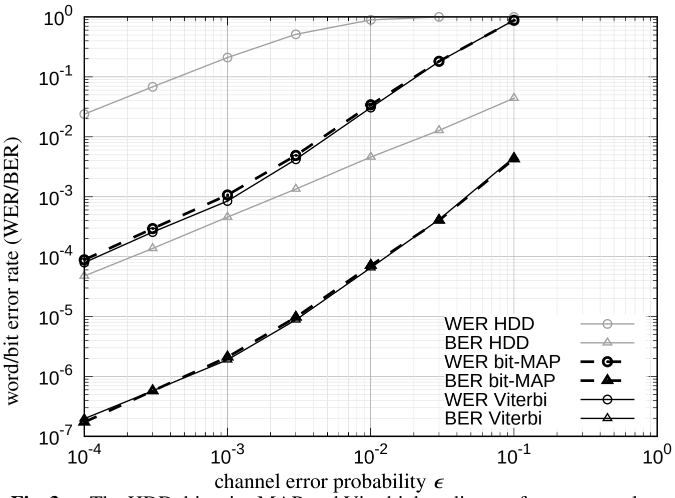


Fig. 3 The HDD, bit-wise MAP and Viterbi decoding performance evaluation of $\phi_{n,d}$ with $n = 512, d = 64$ over n-SC(ϵ). The WER and BER of HDD, bit-wise MAP and Viterbi decoding are almost the same.

performance of the code $\phi_{n,d}$ with $n = 512$ and $d = 64$. First, Fig. 2 evaluates the HDD, BDD, bit-wise MAP and Viterbi decoding performance over AWGN(s^2). The dashed black curve draws the exact word error probability of BDD, which is given in (2). We see that HDD outperforms BDD and is likely the same as the bit-wise MAP and Viterbi decoder. The solid gray curve draws the upper bound of the HDD error probability given in (7). We see that the upper bound is tight. Recall the computational complexity of HDD and the proposed bit-wise MAP decoding algorithm are as small as $O(n)$ and $O(n^2)$, respectively. This makes it possible to evaluate the error probabilities by Monte Carlo simulation.

Next, in Figures 3 and 4, we evaluate the performance of HDD, bit-wise MAP and Viterbi decoding of code $\phi_{n,d}$ with $n = 512$ and $d = 64$ over n-SC(ϵ), and EC(δ). The reason why we do not have HDD performance over EC(δ) is that HDD is not applicable to EC(δ). This is due the restriction that HDD assumes that outputs of a channel are real numbers. The output of EC(δ) can be an erasure symbol '?'. We can evaluate HDD performance over EC(δ) by forcibly regarding an erasure symbol as a uniformly chosen symbol in $[n]$. Denote such a decoder by HDD*, which results in the same result as HDD over n-SC($\epsilon = \delta$).

For AWGN(s^2), there is no difference between the performance of HDD and bit-wise MAP decoding, while it is observed that the difference is large for the n-SC(ϵ). These results imply that the Chebyshev distance is suitable for capturing statistical properties in AWGN(s^2). Conversely, on n-SC(ϵ) and EC(δ), MAP decoding takes full advantage of the properties of the channel that are not captured by Chebyshev distance.

In our experiment, only 448/512=0.875 information bits are transmitted per channel use. Despite such low efficiency, one may think the system is operated at high SNR. This is due to the high-level modulation (512-level PAM) and the lack of powerful error correction. It is natural to wonder if there is a way to improve decoding performance further while keeping the structure of efficiently decodable permutation codes. For this purpose, we consider concatenated codes in the next section.

6. Concatenation

In this section, we investigate concatenated permutation codes. We use a binary code C of size $\#C$ as the outer code. We use the permutation code $\phi_{n,d=0}$ discussed in Section 4 as the inner code. In the rest of this paper, for simplicity, we denote $\phi := \phi_{n,d=0}$. With a little abuse of notation, we denote the encoder of C also by C.

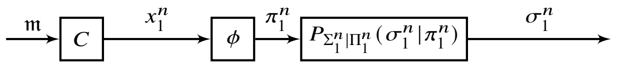


Fig. 5 Data diagram of concatenated permutation code.

6.1 Definition of Concatenation

Let us define the concatenated codes precisely with a data diagram. Take a look at Fig. 5. Let C be a binary code of size M. First, a message $\mathfrak{m} \in [M]$ is encoded into a binary codeword $x_1^n := C(\mathfrak{m}) \in C$. Next, the codeword is used as an input to the permutation encoder ϕ. The resulting permutation codeword $\pi_1^n = \phi(x_1^n)$ is transmitted over the channels. We denote the resulting concatenated permutation code by $\phi(C) := \{\phi(x_1^n) \mid x_1^n \in C\}$.

Example 2. The encoder $\phi_{n,d}$ can be viewed as a special case of the concatenated code with $C = C_{n,d}$, where $C_{n,d}$ maps $\mathfrak{m} := u_1^{n-d} \in \mathbb{F}_2^{n-d}$ to x_1^n by appending d zeros:

$$C_{n,d} : u_1 \cdots u_{n-d} \mapsto x_1^n = u_1 \cdots u_{n-d} \overbrace{0 \cdots 0}^{d}. \tag{17}$$

Example 3. We give another example. Let C be a binary extended Hamming code of length 8. Table 2 lists information vectors $\mathfrak{m} = u_1^4 \in \mathbb{F}_2^4$, binary codewords $x_1^8 \in C$, counters t_1^9 and permutation codewords $\pi_1^8 = \phi(x_1^8)$.

Table 2 An example of concatenated permutation code $\phi(C)$ with the extended Hamming code C.

$\mathfrak{m} = u_1^4$	x_1^8	t_1^9	π_1^8
0000	00000000	012345678	12345678
0001	00011110	012333334	12387654
0010	00101011	012233444	12837465
0011	00110101	012223344	12873645
0100	01000111	011234444	18234765
0101	01011001	011222344	18276345
0110	01101100	011122234	18726534
0111	01110010	011112334	18762354
1000	10001101	001233344	81237645
1001	10010011	001223444	81273465
1010	10100110	001123334	81723654
1011	10111000	001111234	81765234
1100	11001010	000122334	87126354
1101	11010100	000112234	87162534
1110	11100001	000012344	87612345
1111	11111111	000000000	87654321

Note that, the map $\phi : \mathbb{F}_2^n \longrightarrow S_n$ is not injective. For example, $\phi_{n=4,d=0}$ maps both 0100 and 0101 to 1423. This problem can be quickly resolved as follows. We can make the whole mapping from $\mathfrak{m}$ to $\underline{\pi}$ injective by choosing C so that the minimum Hamming distance of C is greater than 1. We will investigate this in the following subsection.

6.2 Induced Pseudo Distance Space: Minimum Chebyshev Distance Characterization

In this subsection, we characterize the minimum Chebyshev distance of concatenated codes by binary outer codes. For this purpose, we introduce a pseudo distance on the binary outer code. This distance is naturally induced from Chebyshev distance of concatenated codes so that the concatenation preserves distance.

First, in the following theorem, we give a simple loose lower bound of the minimum Chebyshev distance of concatenated code $d_\infty(\phi(C))$. Since the proof of this theorem becomes simple with the notion that we will introduce later, we delegate the proof to the appendix.

Theorem 2. Let $C \subset \mathbb{F}_2^n$ be a binary code of length n, with the minimum Hamming distance $d_H(C)$. Then it holds that

$$d_\infty(\phi(C)) \geq d_H(C) - 1 \tag{18}$$

In words, the minimum Chebyshev distance of concatenated code $\phi(C)$ is roughly greater than the minimum Hamming distance of C. The natural thought is that concatenating codes with a larger minimum Hamming distance makes concatenated codes with a larger minimum Chebyshev distance. This idea is not so good because the bound is not tight. In fact, using the outer code $C_{n,d}$ we saw in Example 2, we can express $\phi_{n,d} = \phi(C_{n,d})$. It can be seen that the minimum Hamming distance is 1: $d_H(C_{n,d}) = 1$ and $d_\infty(\phi_{n,d}) = d$ while the bound gives $d_\infty(\phi_{n,d}) \geq d_H(C) - 1 = 0$.

The following pseudo distance over $\mathbb{F}_2^n$ essentially captures the structure of concatenated codes $\phi(C)$. We start from a definition.

Definition 9 (pseudo distance)**.** Consider two vectors $\underline{x}^{(1)}$ and $\underline{x}^{(2)}$ in $\mathbb{F}_2^n$. Let $\ell(\underline{x}^{(1)}, \underline{x}^{(2)})$ be the smallest $j \in [n]$ such that $x_j^{(1)} \neq x_j^{(2)}$. To be precise,

$$\begin{aligned} x_j^{(1)} &= x_j^{(2)} \text{ for } j = 1, 2, \ldots, \ell - 1, \\ x_\ell^{(1)} &\neq x_\ell^{(2)}. \end{aligned} \tag{19}$$

When $\underline{x}^{(1)} = \underline{x}^{(2)}$ we define $\ell = n$. Define $d_*\left(\underline{x}^{(1)}, \underline{x}^{(2)}\right) \overset{\text{def}}{=} n - \ell$. In Theorem 3, we will show that $(\mathbb{F}_2^n, d_*)$ is a pseudo distance space.

From this definition, it is easily seen that

$$d_*(\underline{x}^{(1)}, \underline{x}^{(2)}) = d_*(\underline{x}^{(1)} - \underline{x}^{(2)}, \underline{0}). \tag{20}$$

It follows that $d_*(\underline{x}^{(1)}, \underline{x}^{(2)})$ depends only on the difference between $\underline{x}^{(1)}$ and $\underline{x}^{(2)}$.

Lemma 2 (ϕ is distance preserving)**.** For $\underline{x}^{(1)}$ and $\underline{x}^{(2)}$ in $\mathbb{F}_2^n$, denote the corresponding permutation codewords by $\underline{\pi}^{(1)} := \phi(\underline{x}^{(1)})$ and $\underline{\pi}^{(2)} := \phi(\underline{x}^{(2)})$, respectively. It holds that ϕ preserves distance between $(\mathbb{F}_2^n, d_*)$ and (S_n, d_∞). To be precise,

$$d_\infty(\underline{\pi}^{(1)}, \underline{\pi}^{(2)}) = d_*(\underline{x}^{(1)}, \underline{x}^{(2)}).$$

Proof. Write $\ell = \ell(\underline{x}^{(1)}, \underline{x}^{(2)})$. It is sufficient to show that $d_\infty(\underline{\pi}^{(1)}, \underline{\pi}^{(2)}) = n - \ell$. Recall Definition 1: $d_\infty(\underline{\pi}^{(1)}, \underline{\pi}^{(2)}) =$

$\max_{1\le j\le n}|\pi_j^{(1)}-\pi_j^{(2)}|$. First, we partition $[n]$ into three sections: $\{1,\dots,\ell-1\},\{\ell\}$ and $\{\ell+1,\dots,n\}$. Next, we will evaluate $|\pi_j^{(1)}-\pi_j^{(2)}|$ with j in the three regions as follows.

1. For $j=1,\dots,\ell-1$, it holds that $x_j^{(1)}=x_j^{(2)}$ from (19). Recalling the definition of the encoder ϕ, we know $\pi_j^{(1)}$ and $\pi_j^{(2)}$ depend only on $x_1^{(1)},\dots,x_j^{(1)}$ and $x_1^{(2)},\dots,x_j^{(2)}$, respectively. It follows that $\pi_j^{(1)}-\pi_j^{(2)}=0$.
2. For $j=\ell$, from (4), we know that $\pi_j^{(2)}-\pi_j^{(1)}=n-\ell$.
3. Consider the case with $j=\ell+1,\dots,n$. Recall Remark 1. Let $t_j^{(1)}$ and $t_j^{(2)}$ for $j=0,\dots,n$ be counters that count the number of 0's in $(x_1^{(1)},\dots,x_{j-1}^{(1)})$ and $(x_1^{(2)},\dots,x_{j-1}^{(2)})$, respectively. Recall (4). It holds that
$$|\pi_j^{(1)}-\pi_j^{(2)}|=\begin{cases}|t_j^{(1)}-t_j^{(2)}| & (x_j^{(1)}=x_j^{(2)}),\\ |n-j+t_j^{(1)}-t_j^{(2)}| & (x_j^{(1)}\neq x_j^{(2)}).\end{cases}\tag{21}$$
First, consider the case with $x_j^{(1)}=x_j^{(2)}$. It holds $|\pi_j^{(1)}-\pi_j^{(2)}|=|t_j^{(1)}-t_j^{(2)}|$. Note that $|t_j^{(1)}-t_j^{(2)}|$ is equal to the number of distinct components between $(x_1^{(1)},\dots,x_{j-1}^{(1)})$ and $(x_1^{(2)},\dots,x_{j-1}^{(2)})$. Since $x_1^{(1)}=x_1^{(2)},\dots,x_{\ell-1}^{(1)}=x_{\ell-1}^{(2)}$, we have $|t_j^{(1)}-t_j^{(2)}|\le j-\ell\le n-\ell$. Using this and the triangle inequality for absolute values, we have $|n-j+t_j^{(1)}-t_j^{(2)}|\le(n-j)+|t_j^{(1)}-t_j^{(2)}|\le n-\ell$. Hence, for both cases in (21), it holds that $|\pi_j^{(1)}-\pi_j^{(2)}|\le n-\ell$.

Hence, we have $|\pi_j^{(1)}-\pi_j^{(2)}|\le n-\ell$ for any $j\in[n]$ and the equality holds with $j=\ell$. This concludes the proof. □

Theorem 3. $(\mathbb{F}_2^n,d_*)$ is a pseudo distance space.

Proof. Let us choose arbitrary $\underline{x},\underline{y},\underline{z}\in\mathbb{F}_2^n$. Obviously, $d_*(\underline{x},\underline{y})=d_*(\underline{y},\underline{x})\ge 0$ and $d_*(\underline{x},\underline{x})=0$. The function $d_*(\cdot,\cdot)$ satisfies the triangle inequality: for arbitrary $\underline{x},\underline{y},\underline{z}\in\mathbb{F}_2^n$, it holds that
$$\begin{aligned}d_*(\underline{x},\underline{z})&=d_\infty(\phi(\underline{x}),\phi(\underline{z}))\\&\le d_\infty(\phi(\underline{x}),\phi(\underline{y}))+d_\infty(\phi(\underline{y}),\phi(\underline{z}))\\&=d_\infty(\underline{x},\underline{y})+d_*(\underline{y},\underline{z}),\end{aligned}$$
where the equalities are due to Lemma 2 and the inequality is due to the triangle inequality of $d_\infty(\cdot,\cdot)$. This concludes the proof. □

However, $d_*(\cdot,\cdot)$ is not a distance but a pseudo distance since $d_*(\underline{x}^{(1)},\underline{x}^{(2)})=0$ does not imply $\underline{x}^{(1)}=\underline{x}^{(2)}$, e.g., $d_*(0000,0001)=0$.

Table 6.2 demonstrates Lemma 2: $d_*(\underline{x}^{(1)},\underline{x}^{(2)})=d_\infty(\underline{\pi}^{(1)},\underline{\pi}^{(2)})$ for some $\underline{x}^{(1)},\underline{x}^{(2)}\in\mathbb{F}_2^n$ for $n=4$.

Consider the pseudo distance space $(\mathbb{F}_2^n,d_*)$. For a binary code $C\subset\mathbb{F}_2^n$, we define the minimum pseudo-distance of C by the minimum value of pseudo distance of any two distinct codewords and denote it by $d_*(C)=\min\{d_*(\underline{x}^{(1)},\underline{x}^{(2)}):\underline{x}^{(1)},\underline{x}^{(2)}\in C,\underline{x}^{(1)}\neq\underline{x}^{(2)}\}$. We say $C\subset\mathbb{F}_2^n$ is an (n,M,d) code on $(\mathbb{F}_2^n,d_*)$ if $\#C=M$ and $d=d_*(C)$. As a direct consequence of Lemma 2, we have the following Theorem.

Table 3 Demonstration of Lemma 2: $d_*(\underline{x}^{(1)},\underline{x}^{(2)})=d_\infty(\underline{\pi}^{(1)},\underline{\pi}^{(2)})$

$\underline{x}^{(1)}$	$\underline{x}^{(2)}$	ℓ	d_*	$\underline{\pi}^{(1)}$	$\underline{\pi}^{(2)}$	d_∞
0000	0000	4	0	1234	1234	0
0000	1000	1	3	1234	4123	3
0000	0100	2	2	1234	1423	2
0000	0010	3	1	1234	1243	1
0000	0001	4	0	1234	1234	0

Theorem 4. Let $C\subset\mathbb{F}_2^n$ and $d\ge 1$. It holds that C is an (n,M,d) code on $(\mathbb{F}_2^n,d_*)$ if and only if $\phi(C)$ is an (n,M,d) permutation code on (S_n,d_∞).

Theorem 4 states that for any d such that $d\ge 1$, any proposition about $\phi(C)$ with $d_\infty(C)=d$ can be replaced by a proposition about the outer code C with $d_*(C)=d$. Although the subject of our study is the concatenated code $\phi(C)$ under Chebyshev distance, in the remainder of this paper, for the sake of brevity and to avoid redundancy, we will refer only to statements in terms of a binary code C under distance d_*.

6.3 Bounds on Minimum Chebyshev Distance of Concatenated Permutation Codes

We define a Chebyshev ball $B_*(\underline{x},r)$ centered at $\underline{x}\in\mathbb{F}_2^n$ of radius r by
$$B_*(\underline{x},r)=\{\underline{y}\in\mathbb{F}_2^n\mid d_*(\underline{x},\underline{y})\le r\}.$$
The size of ball is given by $\#B_*(\underline{x},r)=2^{r+1}$ for $0\le r<n$. Note that this does not depend on $\underline{x}$. Hence, we drop $\underline{x}$ and simply write $B_*(\underline{x},r)=:B_*(r)$. This leads to the Gilbert–Varshamov (GV)-like bound on the size of code as follows.

Theorem 5. An (n,M,d) code C exists on $(\mathbb{F}_2^n,d_*)$, if $M\le 2^n/\#B_*(d-1)=2^{n-d}$.

Proof. The claim follows from the standard arguments of GV bound by counting the size of balls. We omit the proof. □

As a corollary of Theorem 5, we have that an (n,M,d) permutation code $\phi(C)$ with $C\subset\mathbb{F}_2^n$ exists on (S_n,d_∞), if $M\le 2^n/\#B_*(d-1)=2^{n-d}$. The following theorem explains $C_{n,d}$ is the outer code that maximizes the size of concatenated codes of minimum Chebyshev distance d.

Theorem 6. If an (n,M,d) code C exists on $(\mathbb{F}_2^n,d_*)$, then $M\le 2^{n-d}$.

Proof. Let $\underline{x},\underline{y}\in C$ be two distinct codewords. Since the minimum pseudo distance is d, it holds that $\ell(\underline{x}, \underline{y}) \le n - d$. In words, $\underline{x}$ and $\underline{y}$ have at least one different component in the first $n-d$ components. The maximum number of codewords such that any two distinct codewords satisfy such property is 2^{n-d}. □

6.4 Concatenation with Outer Linear Codes

Consider the case $C \subset \mathbb{F}_2^n$ is linear. The minimum pseudo distance can be evaluated by minimizing the distance from the zero-vector.

Theorem 7. The minimum pseudo distance of a linear code C on $(\mathbb{F}_2^n, d_*)$ is given by the minimum value of the distance from the all-zero vector. To be precise, $d_*(C) = \min_{\underline{x} \in C : \underline{x} \neq 0} d_*(\underline{x}, \underline{0})$.

Proof. This can be seen from

$$\begin{aligned} d_*(C) &= \min_{\underline{x}^{(1)}, \underline{x}^{(2)} \in C : \underline{x}^{(1)} \neq \underline{x}^{(2)}} d_*(\underline{x}^{(1)}, \underline{x}^{(2)}) \\ &\overset{(a)}{=} \min_{\underline{x}^{(1)}, \underline{x}^{(2)} \in C : \underline{x}^{(1)} \neq \underline{x}^{(2)}} d_*(\underline{x}^{(1)} - \underline{x}^{(2)}, \underline{0}) \\ &\overset{(b)}{=} \min_{\underline{x}^{(1)} - \underline{x}^{(2)} \in C : \underline{x}^{(1)} - \underline{x}^{(2)} \neq \underline{0}} d_*(\underline{x}^{(1)} - \underline{x}^{(2)}, \underline{0}) \\ &= \min_{\underline{x} \in C : \underline{x} \neq 0} d_*(\underline{x}, \underline{0}), \end{aligned}$$

where (a) is due to (20) and (b) is due to the linearity of C. □

Example 4. For example, consider the code $C_{n,d}$ defined in (17). It is easy to check that $C_{n,d}$ is linear. From Theorem 7, we have

$$d_*(C_{n,d}) = \min_{\underline{x} \in C : \underline{x} \neq 0} d_*(\underline{x}, \underline{0}) = \max_{\underline{x} \in C : \underline{x} \neq 0} n - \ell(\underline{x}, \underline{0}).$$

Recall $\ell(\underline{x}, \underline{0})$ is the smallest j such that $x_j \neq 0$. The minimum pseudo distance is given by $d_*(\underline{x}, \underline{0})$ with $\underline{x} = \overbrace{0 \cdots 0}^{n-d-1} 1 \overbrace{0 \cdots 0}^{d}$.

Next, we will construct linear codes of minimum pseudo distance k on $(\mathbb{F}_2^n, d_*)$, from arbitrary linear codes C of dimension $n-k$ and length n. For a permutation $\tau \in S_n$ and a vector x_1^n of length n, denote $(x_{\tau(1)}, \dots, x_{\tau(n)})$ by $\underline{x}^\tau$. For example, $(x_1, x_2, x_3)^\tau = (x_3, x_2, x_1)$ with $\tau = (3, 2, 1)$. Similarly, for a code C of length n, we define $C^\tau := \{\underline{x}^\tau \mid \underline{x} \in C\}$. Obviously, the dimension and Hamming distance remain the same, that is, $\dim(C^\tau) = \dim(C)$ and $d_H(C^\tau) = d_H(C)$ for any $\tau \in S_n$.

Theorem 8. Let C be a binary linear code of length n and dimension k. There exists a permutation $\tau \in S_n$ such that $d_*(C^\tau) = n - k$.

Proof. From Theorem 5, it holds that $d_*(C^\tau) \le n-k$ for any $\tau \in S_n$. We will construct $\tau \in S_n$ such that $d_*(C^\tau) \ge n - k$.

Let G be a $k \times n$ generator matrix of C. The rank of G should be k. Recall the rank of a matrix is the maximum number of linearly independent columns. Therefore, there exist k linearly independent columns in G. We denote the set of column indices by $\mathcal{I}$. Let $G_{\mathcal{I}}$ be the submatrix of G restricted to the columns indexed by $\mathcal{I}$. Similarly, denote the corresponding subvector of $\underline{x} \in \mathbb{F}_2^n$ by $\underline{x}_{\mathcal{I}}$.

Let τ be chosen so that τ moves columns in $\mathcal{I}$ to the left, that is, $G^\tau = (G_{\mathcal{I}} | G_{\overline{\mathcal{I}}})$, where $\overline{\mathcal{I}} = [n] \setminus \mathcal{I}$. The first k components of codewords of C^τ takes any values: $C^\tau_{[k]} = \mathbb{F}_2^k$. This can be seen as follows. Let $\underline{u}$ be an information vector of length k. A codeword of C^τ is generated by $\underline{x}^\tau = \underline{u} G^\tau = \underline{u}(G_{\mathcal{I}} | G_{\overline{\mathcal{I}}}) = (\underline{u} G_{\mathcal{I}}, \underline{u} G_{\overline{\mathcal{I}}})$. We see that $\underline{u} G_{\mathcal{I}}$ is the first k components of the codeword. This takes any value of $\mathbb{F}_2^k$ by letting $\underline{u}$ run over $\mathbb{F}_2^k$ since $G_{\mathcal{I}}$ is invertible.

Consider two codewords $\underline{x}, \underline{x}'$ in C^τ. The first k components of $\underline{x}$ and $\underline{x}'$ must be different, i.e., $\underline{x}_{[k]} \neq \underline{x}'_{[k]}$. Recalling the definition of $\ell(\cdot, \cdot)$, we have $\ell(\underline{x}_i, \underline{x}_j) \le k$. Therefore, we obtain $d_*(\underline{x}, \underline{x}') = n - \ell(\underline{x}, \underline{x}') \ge n - k$. □

As a corollary of Theorem 8, it holds that from any linear code C of dimension k, one can construct a concatenated permutation code $\phi(C)$ of minimum Chebyshev distance $n - k$.

In the following subsections, we will evaluate the minimum pseudo distance of random codes.

6.5 Minimum Pseudo Distance of Random Binary Codes

In this subsection, we evaluate the minimum pseudo distance of random binary codes. The proof relies on the technique used in [27]. We write $f(n) = \Psi_n(g(n))$ if

$$\lim_{n \to \infty} \frac{1}{n} \log_2 f(n) = \lim_{n \to \infty} \frac{1}{n} \log_2 g(n).$$

Let C be a binary code of length n and size M. Let us write $C := \{\underline{X}^{(1)}, \dots, \underline{X}^{(M)}\} \subset \mathbb{F}_2^n$. Each component of $X^{(i)}$ for $i = 1, \dots, M$ is independently and uniformly chosen from $\mathbb{F}_2$. Here we view the code as a multiset that may have multiple instances. There are 2^{nM} possible codes. We call C a random binary code. We denote the number of codeword pairs of pseudo distance d by $S(C, d)$. To be precise,

$$S(C, d) = \sum_{i, j \in [M] : i < j} \mathbb{1}[d_*(\underline{X}^{(i)}, \underline{X}^{(j)}) = d].$$

Theorem 9. For $r \in (0, 1)$, let $M := 2^{\lceil nr \rceil}$. Let C be a random binary code of size M. For $\delta \in (0, 1)$, it holds that

$$\begin{aligned} &\mathbb{E}[S(C, \lfloor \delta n \rfloor)] = \Psi_n(2^{n(2r-1+\delta)}), \\ &P\Big[d_*(C) \le \delta n\Big] \le \Psi_n(2^{-n(1-2r-\delta)}), \\ &P\Big[\frac{S(C, \lfloor \delta n \rfloor)}{\mathbb{E}[S(C, \lfloor \delta n \rfloor)]} \notin (1-\alpha, 1+\alpha)\Big] \le \frac{1}{\alpha^2} \Psi_n\big(2^{-n(2r-1+\delta)}\big). \end{aligned}$$

The second and third claims are valid for all $r, \delta \in (0, 1)$, but have meaning only for $\delta < 1-2r$ and $\delta > 1-2r$, respectively. From the first and third claim, we see that $S(C, \lfloor\delta n\rfloor)$ grows exponentially with n if $\delta > 1-2r$. We see that, from this and the second claim, for $r \leq 1/2$, C has the minimum pseudo distance approximately $n(1-2r)$ with probability $1-2^{-\Omega(n)}$.

Proof. Recall (20). We see that shifting codewords preserves distance: $d_*(\underline{X}^{(i)}, \underline{X}^{(j)}) = d_*(\underline{X}^{(i)} - \underline{X}^{(j)}, \underline{0})$ for $i, j \in [M]$. Moreover, we see that

$$P\big(d_*(\underline{X}^{(i)}, \underline{X}^{(j)}) = d\big) = \begin{cases} 1/2^{n-1} & (d = 0), \\ 1/2^{n-d} & (d \in [n-1]). \end{cases} \tag{22}$$

We drop i, j and denote (22) by q_d, since (22) does not depend on (i, j). From this and how the code is constructed, we see that $d_*(\underline{X}^{(i)}, \underline{X}^{(j)})$ for (i, j) with $i < j$ are independent and identically distributed (iid). Hence, we have $\mathbb{E}[S(C, d)] = \binom{M}{2}\mathbb{E}[\mathbb{1}_{i,j}]$ for some arbitrarily fixed (i, j) with $i < j$, where $\mathbb{1}_{i,j} := \mathbb{1}[d_*(\underline{X}^{(i)}, \underline{X}^{(j)}) = d]$. Recalling the definition of d_*, we see that $P[\mathbb{1}_{i,j} = 1] = q_d$ for arbitrarily fixed (i, j). Hence, for $d = \lfloor\delta n\rfloor$, we have

$$\mathbb{E}[S(C, d)] = \binom{M}{2} q_d = \Psi_n(2^{n(2r-1+\delta)}).$$

Let us move to the second claim. The minimum pseudo distance $d_*(C)$ is equal to the minimum value of $d \in [0 : n]$ such that $S(C, d) \neq 0$. Note that the minimum pseudo distance can be 0, since C is allowed to have multiple instances. Consequently, $d_*(C) \leq \delta n$ iff $\sum_{d=0}^{\lfloor n\delta\rfloor} S(C, d) \geq 1$. It follows that

$$\begin{aligned} P\Big[d_*(C) \leq \delta n\Big] &= P\Big[\sum_{d=0}^{\lfloor n\delta\rfloor} S(C, d) \geq 1\Big] \\ &\overset{\text{(a)}}{\leq} \mathbb{E}\Big[\sum_{d=0}^{\lfloor n\delta\rfloor} S(C, d)\Big] \\ &= \sum_{d=0}^{\lfloor n\delta\rfloor} \mathbb{E}[S(C, d)] \\ &\leq (n\delta + 1) \max_{d\in[0:\delta n]} \mathbb{E}[S(C, d)] \\ &= \Psi_n(2^{N(2r-1+\delta)}). \end{aligned}$$

In (a), we used Markov inequality. This concludes the second claim.

Now, we move to the final claim. For $i, j \in [M]$ with $i < j$, the variance of $\mathbb{1}_{i,j}$ can be calculated via $\mathbb{E}[\mathbb{1}_{i,j}]$ as follows:

$$\begin{aligned} \mathrm{Var}[\mathbb{1}_{i,j}] &= \mathbb{E}[\mathbb{1}_{i,j}^2] - (\mathbb{E}[\mathbb{1}_{i,j}])^2 \\ &= \mathbb{E}[\mathbb{1}_{i,j}] - (\mathbb{E}[\mathbb{1}_{i,j}])^2 \leq \mathbb{E}[\mathbb{1}_{i,j}], \end{aligned}$$

where we used that $\mathbb{1}_{i,j} \in \{0, 1\}$ and $0 \leq \mathbb{E}[\mathbb{1}_{i,j}] \leq 1$. Using this and the fact that $\mathbb{1}_{i,j}$ for (i, j) with $i < j$ are $\binom{M}{2}$ iid random variables, we have

$$\mathrm{Var}(S(C, d)) = \binom{M}{2} \mathrm{Var}(\mathbb{1}_{i,j}) \leq \mathbb{E}[S(C, d)].$$

with arbitrarily fixed (i, j). By the Chebyshev inequality, for any $\alpha > 0$, we have

$$\begin{aligned} &P\{S(C, d)/\mathbb{E}[S(C, d)] \notin (1-\alpha, 1+\alpha)\} \\ &= P\{|S(C, d) - \mathbb{E}[S(C, d)]| \geq \alpha\mathbb{E}[S(C, d)]\} \\ &\leq \frac{\mathbb{E}[S(C, d)]}{\alpha^2\mathbb{E}[S(C, d)]^2} \\ &= \frac{1}{\alpha^2\mathbb{E}[S(C, d)]} \end{aligned} \tag{23}$$

Together with the first claim, the third claim follows. □

6.6 Minimum Pseudo Distance of Random Linear Codes

In this subsection, we evaluate the minimum pseudo distance of random linear codes. The proof relies on the technique used in [11, Problem 1.17].

Let G be a random binary matrix of size $k \times n$. Each entry of G is chosen from $\mathbb{F}_2$ identically and uniformly at random. Let $M := 2^k$. Consider a code C, which is generated by G: $C = \{\underline{X}^{(0)}, \ldots, \underline{X}^{(M-1)}\}$, where we wrote $\underline{X}^{(i)} := \underline{u}^{(i)}G$ and $\underline{u}^{i}$ for $i = 0, \ldots, M-1$ are chosen so that $\{\underline{u}^{(0)}, \ldots, \underline{u}^{(M-1)}\} = \mathbb{F}_2^k$. Particularly, we choose $\underline{u}^{(0)} := \underline{0} \in \mathbb{F}_2^k$. It follows $\underline{x}^{(0)} = \underline{0} \in \mathbb{F}_2^n$. Note that C is defined as a multiset, which may have multiple instances. We call C a random linear code. There are 2^{kn} possible choices of G. We call G a random generator matrix and C a random linear code, respectively.

Since C is linear, it contains at least an all-zero vector $\underline{0}$. Note that C may have multiple all-zero vectors. Let $A(C, d; i)$ be the number of codewords of pseudo distance d from $\underline{x}^{(i)}$. Precisely,

$$A(C, d; i) = \sum_{j\in[0:M-1]:j\neq i} \mathbb{1}[d_*(\underline{X}^{(i)}, \underline{X}^{(j)}) = d].$$

It follows that

$$\begin{aligned} A(C, d; i) &\overset{\text{(a)}}{=} \sum_{j\in[0:M-1]:j\neq i} \mathbb{1}[d_*(\underline{X}^{(i)} - \underline{X}^{(j)}, \underline{0}) = d] \\ &\overset{\text{(b)}}{=} \sum_{j\in[1:M-1]} \mathbb{1}[d_*(\underline{X}^{(j)}, \underline{0}) = d], \end{aligned} \tag{24}$$

where we used (20) in (a). From the linearity of C, it follows that $\{\underline{x}^{(i)} - \underline{x}^{(j)} \mid j \in [0 : M-1], j \neq i\} = \{x^{(1)}, \ldots, x^{(M-1)}\}$. We used this in (b). It can be seen that (24) does not depend on i. Hence, we drop i and write (24) simply by $A(C, d)$.

Example 5. For $n = 3, k = 3$, $G = \begin{pmatrix} 011 \\ 011 \\ 101 \end{pmatrix}$, and $\mathbb{F}_2^3 = \{000, 100, 010, 110, 001, 101, 011, 111\}$, then we have

$$\begin{aligned} C &= \{000, 011, 011, 000, 101, 110, 110, 101\}, \\ P &= \{123, 132, 132, 123, 312, 321, 321, 312\}, \end{aligned}$$

$$A(C,0)=1, A(C,1)=2, A(C,2)=4.$$

Theorem 10. For an arbitrary fixed $r \in (0,1)$, let $k := \lfloor rn \rfloor$ and $M := 2^k$. Let G be the random generator matrix of size $k \times n$. Let C be the random linear code of size M, which is generated by G. For any $\delta \in (0,1)$, it holds that

$$\mathbb{E}_G[A(C,\lfloor \delta n \rfloor)] = \Psi_n(2^{n(r-1+\delta)}), \tag{25}$$

$$P\Big[d_*(C) \le \delta n\Big] \le \Psi_n(2^{n(r-1+\delta)}),$$

$$P\Big[\frac{A(C,\lfloor \delta n \rfloor)}{\mathbb{E}[A(C,\lfloor \delta n \rfloor)]} \notin (1-\alpha, 1+\alpha)\Big] \le \frac{1}{\alpha^2}\Psi_n\big(2^{-n(r-1+\delta)}\big).$$

Proof. Taking $\mathbb{E}$ on both sides of (24), we have

$$\mathbb{E}[A(C,d)] = \sum_{i\in[1:M-1]} P[d_*(\underline{X}^{(i)},\underline{0}) = d].$$

Recalling the definition of d_*, we have that $d_*(\underline{X}^{(i)},\underline{0}) = 0$ iff $\underline{X}^{(i)} = \overbrace{0\cdots0*}^{n-1}$ for $d=0$, and $\underline{X}^{(i)} = \overbrace{0\cdots0}^{n-d}1\overbrace{*\cdots*}^{d-1}$ for $d \neq 0$. where $*$ means we do not care whether it is 0 or 1. Note that $i \neq 0$. Since $\underline{u}^{(i)} \neq \underline{0}$, each element of $\underline{X}^{(i)} = \underline{u}^{(i)}G$ is an independently and uniformly distributed random binary variable: $P[\underline{X}^{(i)} = \underline{X}] = 1/2^n$ for any $\underline{X} \in \mathbb{F}_2^n$. Consequently, we have

$$P[d_*(\underline{X}^{(i)},\underline{0}) = d] = \begin{cases} 1/2^{n-1} & (d=0), \\ 1/2^{n-d+1} & (d\neq 0). \end{cases} \tag{26}$$

Note that the right hand side of (26) does not depend on i. Therefore, we have

$$\mathbb{E}_G[A(C,d)] = (M-1)P[d_*(\underline{X}^{(i)},\underline{0}) = d]. \tag{27}$$

Substituting (26) and $d = \lfloor \delta n \rfloor$ to this, we obtain the first claim (25).

Noting that C may contain multiple zero codewords, we see that the minimum pseudo distance $d_*(C)$ can be zero. Hence, it holds that $d_*(C)$ is equal to the minimum value of $d \in [0:n-1]$ such that $A(C,d) \neq 0$. Consequently, $d_*(C) \le \delta n$ iff $\sum_{d=0}^{\lfloor n\delta \rfloor} A(C,d) \ge 1$. It follows that

$$\begin{aligned} &P\Big[d_*(C) \le \delta n\Big] \\ &= P\Big[\sum_{d=0}^{\lfloor n\delta \rfloor} A(C,d) \ge 1\Big] \\ &\overset{\text{(a)}}{\le} \mathbb{E}\Big[\sum_{d=0}^{\lfloor n\delta \rfloor} A(C,d)\Big] \\ &= \sum_{d=0}^{\lfloor n\delta \rfloor} \mathbb{E}[A(C,d)] \\ &\le (n\delta+1) \max_{d\in[0:\delta n]} \mathbb{E}[A(C,d)] \\ &\overset{\text{(b)}}{=} \Psi_n(2^{n(r-1+\delta)}), \end{aligned}$$

where we used Markov inequality and (25) in (a) and in (b), respectively. This concludes the second claim.

For $0 \le d \le n-1$, we write $\mathbb{1}[d_*(\underline{X}^{(i)},\underline{0}) = d]$ by $\mathbb{1}_i$. With this notation, (24) is simply written by $A(C,d) = \sum_{i=1}^{M-1} \mathbb{1}_i$. It follows that

$$\begin{aligned} \mathbb{E}\left[A^2(C,d)\right] &= \mathbb{E}\Big[\Big(\sum_{i=1}^{M-1} \mathbb{1}_i\Big)\Big(\sum_{j=1}^{M-1} \mathbb{1}_j\Big)\Big] \\ &= \mathbb{E}\Big[\sum_{i,j:i=j} \mathbb{1}_i\mathbb{1}_j\Big] + \mathbb{E}\Big[\sum_{i,j:i\neq j} \mathbb{1}_i\mathbb{1}_j\Big] \\ &= \mathbb{E}[A(C,d)] + \mathbb{E}\Big[\sum_{i,j:i\neq j} \mathbb{1}_i\mathbb{1}_j\Big]. \end{aligned}$$

We see that $\mathbb{1}_1,\dots,\mathbb{1}_{M-1}$ are not independent but pairwise independent, since, for $i,j \in [M-1]$ with $i \neq j$, $P[\underline{X}^{(i)} = \underline{x} | \underline{X}^{(j)} = \underline{x}'] = P[(\underline{u}^{(i)} - \underline{u}^{(j)})G = \underline{x} - \underline{x}' | \underline{X}^{(j)} = \underline{x}'] = 1/2^n$ for any $\underline{x},\underline{x}' \in \mathbb{F}_2^n$. Consequently, we have

$$\begin{aligned} \mathbb{E}\Big[\sum_{i,j:i\neq j} \mathbb{1}_i\mathbb{1}_j\Big] &= (M-1)\,(M-2)P[d_*(\underline{X}^{(i)},\underline{0}) = d]^2 \\ &\overset{\text{(a)}}{=} \mathbb{E}[A(C,d)]^2 \frac{M-2}{M-1} \\ &\le \mathbb{E}[A(C,d)]^2, \end{aligned}$$

where we used (27) in (a). Now we evaluate the variance by

$$\begin{aligned} \mathrm{Var}[A(C,d)] &= \mathbb{E}[A(C,d)^2] - \mathbb{E}[A(C,d)]^2 \\ &\le \mathbb{E}[A(C,d)]. \end{aligned}$$

The third claim follows by using this and the same arguments as in (23). □

6.7 Concatenation with LDPC codes

In this section, we demonstrate SP decoding of concatenated permutation codes with outer binary linear codes. Specifically, we employ LDPC codes. Let the binary outer linear code of length n be defined by a parity-check matrix H of size $m \times n$. Denote the code by C_H. We have $C_H = \{\underline{x} \in \mathbb{F}_2^n | H\underline{x}^{\mathrm{T}} = \underline{0}\}$. Assume that H is a full-rank matrix. First, a uniformly picked message $\mathfrak{m} \in [\#C_H]$ is encoded into a binary codeword $x_1^n := C_H(\mathfrak{m}) \in C_H$. Consequently, $\underline{x}$ is uniformly distributed in C_H. Next, $\underline{x}$ is encoded into a permutation codeword $\phi(\underline{x}) \in \phi(C_H)$, which is transmitted through channels.

At the receiver, let $\underline{\sigma} = \sigma_1^n$ be a received vector through channels. From the same argument in Section 5.2, the bit-wise MAP decoding $\hat{x}_k(\underline{\sigma}) \overset{\text{def}}{=} \operatorname{argmax}_{x_k\in\mathbb{F}_2} P_{X_k|\underline{\Sigma}}(x_k|\underline{\sigma})$ for $k = 1,\dots,n$ can be written by the same as (11), except that $P_{X_1^n}(x_1^n)$ is factorized into $P_{X_1^n}(x_1^n) =$

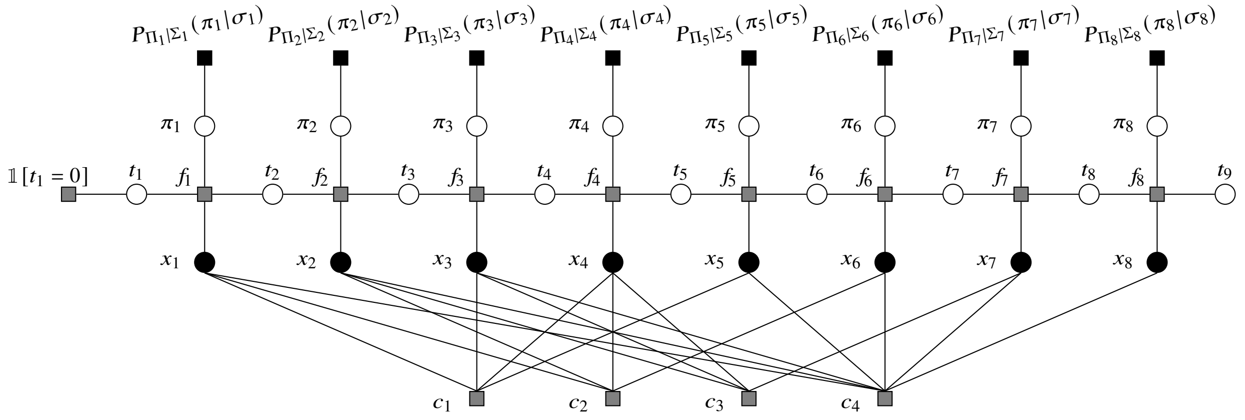

$P_{\Pi_1|\Sigma_1}(\pi_1|\sigma_1)$
$P_{\Pi_2|\Sigma_2}(\pi_2|\sigma_2)$
$P_{\Pi_3|\Sigma_3}(\pi_3|\sigma_3)$
$P_{\Pi_4|\Sigma_4}(\pi_4|\sigma_4)$
$P_{\Pi_5|\Sigma_5}(\pi_5|\sigma_5)$
$P_{\Pi_6|\Sigma_6}(\pi_6|\sigma_6)$
$P_{\Pi_7|\Sigma_7}(\pi_7|\sigma_7)$
$P_{\Pi_8|\Sigma_8}(\pi_8|\sigma_8)$
π_1 π_2 π_3 π_4 π_5 π_6 π_7 π_8
$\mathbb{1}[t_1 = 0]$
t_1 f_1 t_2 f_2 t_3 f_3 t_4 f_4 t_5 f_5 t_6 f_6 t_7 f_7 t_8 f_8 t_9
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
c_1 c_2 c_3 c_4

Fig. 6 Factor graph of factors inside argmax in (11) with $P_{X_1^n}(x_1^n) = \frac{1}{|C|}\prod_{i=1}^{m} \mathbb{1}[\sum_{j\in\partial i} x_j = 0]$ and H in (28). Each factor $\mathbb{1}[\sum_{j\in\partial i} x_j = 0]$ is denoted by c_i for $i = 1, \ldots, 4$.

$\frac{1}{\#C_H}\prod_{i=1}^{m} \mathbb{1}[\sum_{j\in\partial i} x_j = 0]$, where ∂i is the set of column indices j such that $H_{i,j} = 1$.

We give an example. Assume that we are given C_H defined by a parity-check matrices H:

$$H = \begin{pmatrix} 10111000 \\ 11010100 \\ 01110010 \\ 11111111 \end{pmatrix}. \quad (28)$$

Figure 6 depicts the factor graph of factors inside argmax in (11) with $P_{X_1^n}(x_1^n) = \frac{1}{|16|}\mathbb{1}[x_1+x_3+x_4+x_5 = 0]\mathbb{1}[x_1+x_2+x_4+x_6 = 0]\mathbb{1}[x_2+x_3+x_4+x_6 = 0]\mathbb{1}[x_1+x_2+x_3+x_4+x_5+x_6+x_7+x_8 = 0]$ and H in (28). As can be seen from this figure, the factor graph is not a tree in general. This implies that applying the SP algorithm no longer realizes MAP decoding.

We demonstrate the proposed method: concatenated permutation codes with (3,24)-regular LDPC codes of length $n = 512$ decoded by the SP algorithm. The parity-check matrix H is of size 64×512. We compare with the conventional method [13]: non-concatenated permutation code $\phi_{n,d}$ with $n = 512$ and $d = 64$ with HDD. We chose these parameters for making the comparison fair so that both concatenated codes are of size 2^{512-64} and of length 512. Figures 7, 8 and 9 plot the word and bit error rate of the conventional and proposed methods over AWGN(s^2), n-SC(ϵ) and EC(δ), respectively. The SP decoder is evaluated with 50 iterations. It is observed that the SP decoding of concatenated codes vastly outperform the conventional methods for all three channels.

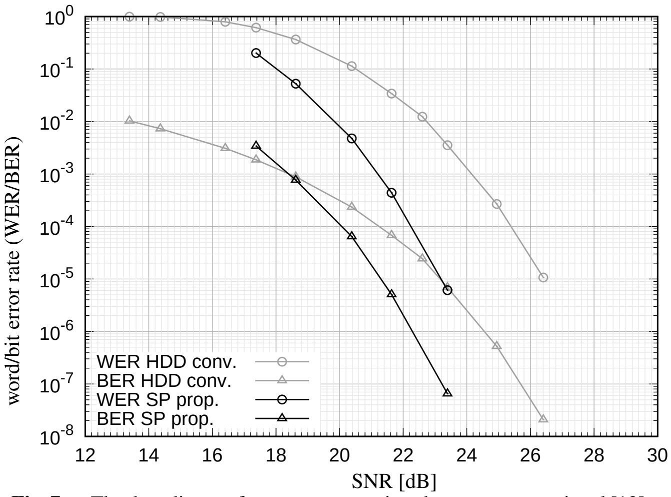


Fig. 7 The decoding performance comparison between conventional [13] and proposed methods over AWGN(s^2). The conventional method [13] uses $\phi_{n,d}$ with $n = 512, d = 64$ and HDD. The proposed method uses concatenated permutation codes with a (3,24)-regular LDPC code of length $n = 512$ and SP decoder.

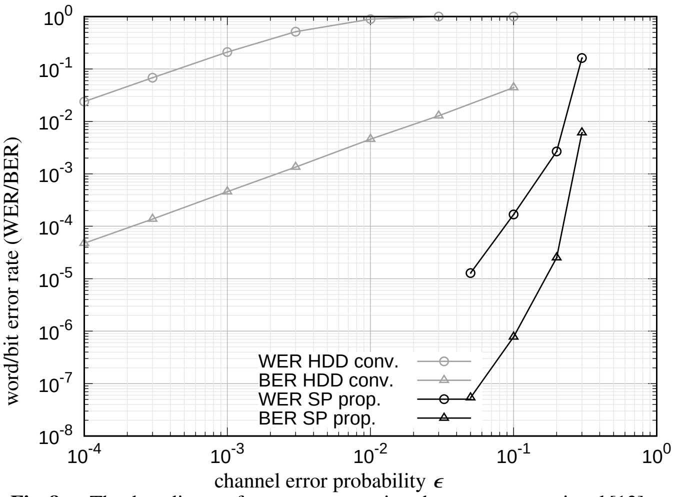


Fig. 8 The decoding performance comparison between conventional [13] and proposed methods over n-SC(ϵ). The conventional method [13] uses $\phi_{n,d}$ with $n = 512, d = 64$ and HDD. The proposed method uses concatenated permutation codes with a (3,24)-regular LDPC code of length $n = 512$ and SP decoder.

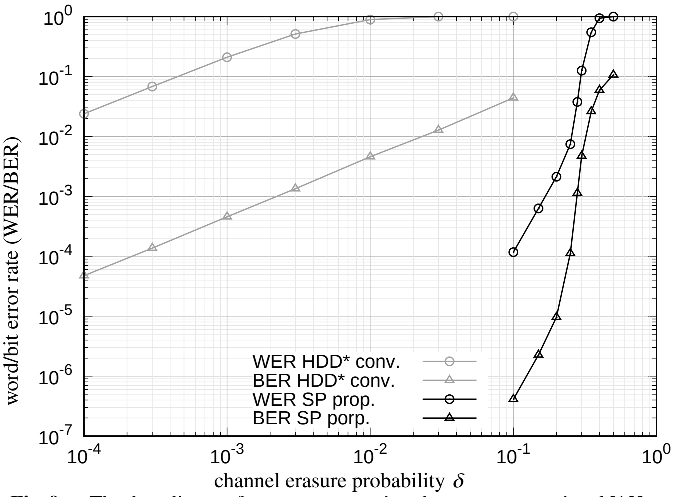


Fig. 9 The decoding performance comparison between conventional [13] and proposed method over EC(δ). The conventional method [13] uses $\phi_{n,d}$ with $n = 512$, $d = 64$ and HDD*. The proposed method method uses concatenated permutation codes with a (3,24)-regular LDPC code of length $n = 512$ and SP decoder.

7. Conclusions

We have studied permutation codes on Chebyshev distance, particularly on the permutation code $\phi_{n,d}$, which is the largest known code with linear growing minimum Chebyshev distance. We derived a tight upper bound of the error probability of HDD. We devised an efficient MAP decoding algorithm for $\phi_{n,d}$.

We also explored concatenation with binary codes with $\phi_{n,d=0}$. We introduced pseudo distance over outer code space, which successfully characterizes the Chebyshev distance of concatenated permutation codes. We derived the distance distribution of concatenated permutation codes with outer random binary and linear codes. We upper-bounded the minimum Chebyshev distance of concatenated codes. We discover how to concatenate binary linear codes to achieve the upper bound. We demonstrated concatenated codes with outer LDPC codes outperform conventional schemes.

Appendix A: Proof of Theorem 2

From Lemma 2, we see that it is sufficient to show that $d_*(x^{(1)}, x^{(2)}) \geq d_H(C) - 1$ for any two distinct codewords $\underline{x}^{(1)}$ and $\underline{x}^{(2)}$ in C. Assume this is not true, i.e., $d_*(x^{(1)}, x^{(2)}) \leq d_H(C) - 2$ for some distinct $\underline{x}^{(1)}$ and $\underline{x}^{(2)}$ in C. It follows that $\ell \geq n - d_H(C) + 2$, where $\ell = \ell(\underline{x}^{(1)}, \underline{x}^{(2)})$. Then it holds that $x_j^{(1)} = x_j^{(2)}$ for $j = 1, 2, \ldots, n - d_H(C) + 1$. This implies $d_H(x^{(1)}, x^{(2)}) \leq d_H(C) - 1$. This contradicts the condition that the minimum Hamming distance of C is $d_H(C)$.

It follows that equality of (18) holds if and only if $\ell = d_H(C) - 1$, in other words, there are codewords $\underline{x}^{(1)}$ and $\underline{x}^{(2)}$ in C such that the first $n - d$ components are identical and the last $n - d$ components are different. To be precise,

$$\begin{aligned} x_j^{(1)} &= x_j^{(2)} && (1 \leq j \leq n - d), \\ x_j^{(1)} &\neq x_j^{(2)} && (n - d + 1 \leq j \leq n). \end{aligned}$$

Acknowledgment

The authors would like to thank the Associate Editor and the anonymous reviewers for their careful reading of our manuscript and their many insightful comments and suggestions.

References

[1] D. Slepian, "Permutation modulation," *Proceedings of the IEEE*, vol. 53, no. 3, pp. 228–236, March 1965.

[2] A. J. H. Vinck, "Coded modulation for powerline communications," in *Proc. Int. J. Elec. Commun.*, no. 1, 2000, pp. 45–49.

[3] A. J. H. Vinck, J. Haering, and T. Wadayama, "Coded m-FSK for power line communications," in *Proc. 2000 IEEE Int. Symp. Inf. Theory (ISIT)*, 2000, p. 137.

[4] I. F. Blake, G. Cohen, and M. Deza, "Coding with permutations," *Information and Control*, vol. 43, no. 1, pp. 1 – 19, 1979.

[5] C. J. Colbourn, T. Kløve, and A. C. H. Ling, "Permutation arrays for powerline communication and mutually orthogonal latin squares," *IEEE Trans. Inf. Theory*, vol. 50, no. 6, pp. 1289–1291, June 2004.

[6] T. G. Swart and H. C. Ferreira, "Decoding distance-preserving permutation codes for power-line communications," in *Proc. IEEE AFRICON 2007*. Windhoek: IEEE, 2007, pp. 1–7.

[7] Z. Wang, A. Jiang, and J. Bruck, "On the capacity of bounded rank modulation for flash memories," in *Proc. 2009 IEEE Int. Symp. Inf. Theory (ISIT)*, June 2009, pp. 1234–1238.

[8] N. Papandreou, H. Pozidis, T. Mittelholzer, G. F. Close, M. Breitwisch, C. Lam, and E. Eleftheriou, "Drift-tolerant multilevel phase-change memory," in *2011 3rd IEEE International Memory Workshop (IMW)*, May 2011, pp. 1–4.

[9] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, "Rank modulation

for flash memories," *IEEE Trans. Inf. Theory*, vol. 55, no. 6, pp. 2659–2673, June 2009.

[10] H. M. Kiah, G. J. Puleo, and O. Milenkovic, "Codes for DNA sequence profiles," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3125–3146, 2016.

[11] T. Richardson and R. Urbanke, *Modern Coding Theory*. New York, NY, USA: Cambridge University Press, Mar. 2008.

[12] T. Wadayama and M. Hagiwara, "LP-decodable permutation codes based on linearly constrained permutation matrices," *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5454–5470, Aug. 2012.

[13] T. Kløve, T. Lin, S. Tsai, and W. Tzeng, "Permutation arrays under the Chebyshev distance," *IEEE Trans. Inf. Theory*, vol. 56, no. 6, pp. 2611–2617, 2010.

[14] M. Z. Shieh and S. C. Tsai, "Decoding frequency permutation arrays under Chebyshev distance," *IEEE Trans. Inf. Theory*, vol. 56, no. 11, pp. 5730–5737, Nov 2010.

[15] S. Vijayakumaran, "Largest permutation codes with the kendall τ-metric in s_5 and s_6," *IEEE Communications Letters*, vol. 20, no. 10, pp. 1912–1915, Oct 2016.

[16] F. Göloğlu, J. Lember, A. E. Riet, and V. Skachek, "New bounds for permutation codes in ulam metric," in *Proc. 2015 IEEE Int. Symp. Inf. Theory (ISIT)*, June 2015, pp. 1726–1730.

[17] I. Tamo and M. Schwartz, "Correcting limited-magnitude errors in the rank-modulation scheme," *IEEE Trans. Inf. Theory*, vol. 56, no. 6, pp. 2551–2560, 2010.

[18] F. Farnoud Hassanzadeh, M. Schwartz, and J. Bruck, "Bounds for permutation rate-distortion," *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 703–712, 2016.

[19] M. Schwartz and P. O. Vontobel, "Improved lower bounds on the size of balls over permutations with the infinity metric," *IEEE Trans. Inf. Theory*, vol. 63, no. 10, pp. 6227–6239, 2017.

[20] H. Han, J. Mu, Y. He, and X. Jiao, "Coset partitioning construction of systematic permutation codes under the Chebyshev metric," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 3842–3851, 2019.

[21] H. Zhou, M. Schwartz, A. A. Jiang, and J. Bruck, "Systematic error-correcting codes for rank modulation," *IEEE Trans. Inf. Theory*, vol. 61, no. 1, pp. 17–32, 2015.

[22] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[23] P. Massart and J. Picard, *Concentration Inequalities and Model Selection: Ecole d'Eté de Probabilités de Saint-Flour XXXIII - 2003*, ser. Lecture Notes in Mathematics. Springer Berlin Heidelberg, 2007.

[24] F. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[25] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (corresp.)," *IEEE Trans. Inf. Theory*, vol. 20, no. 2, pp. 284–287, Mar. 1974.

[26] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164 – 171, 1970.

[27] A. Barg and G. Forney, "Random codes: minimum distances and error exponents," *IEEE Trans. Inf. Theory*, vol. 48, no. 9, pp. 2568–2573, 2002.

Motohiro KAWASUMI received B.E. and M.E. from Tokyo Institute of Technology in 2017 and 2019, respectively.

Kenta KASAI received B.E., M.E. and Ph.D. degrees from Tokyo Institute of Technology in 2001, 2003 and 2006, respectively. From 2008 to 2009, he was a visiting researcher at ETIS laboratory, Ecole Nationale Supérieure de l'Electronique et de ses Applications, Cergy-Pontoise, France. Since April 2012, he has been an associate professor in Tokyo Institute of Technology. His current research interests include codes on graphs and iterative decoding algorithms.