String Concatenation Construction for Chebyshev Permutation Channel Codes

Yeow Meng Chee*, Han Mao Kiah*, San Ling*, Tuan Thanh Nguyen*, Van Khu Vu*, and Xiande Zhang†

*School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore

†School of Mathematical Sciences, University of Science and Technology of China, China

email: {ymchee, hmkiah, lingsan, nguyentu001, vankhu001}@ntu.edu.sg, drzhangx@ustc.edu.cn.

***Abstract*—We construct codes for the Chebyshev permutation channels whose study was initiated by Langberg *et al.* (2015). We establish several recursive code constructions and present efficient decoding algorithms for our codes. In particular, our constructions yield a family of binary codes of rate 0.643 when $r = 1$. The upper bound on the rate in this case is $2/3$ and the previous highest rate is 0.609.**

I. Introduction

Permutation channels have been proposed as a solution to transmission networks that provide no guarantees on the in-order delivery of information [1]. In addition to insertion, deletion, and substitution errors, these channels have the effect of delivering a random permutation of the message vector. Examples include mobile ad hoc networks, vehicular networks, delay tolerant networks, wireless sensor networks [2]. A variety of permutation channel models have been studied under different scenarios:

(i) timing channels [3] where information is being encoded in the transmission times of messages, such as in queuing theory [4] and in molecular communications [5];

(ii) degraded broadcast channels where input packets are randomly permuted by selecting a permutation according to a probability distribution [6], [7]; and

(iii) the bit-shift magnetic recording channels in standard high-density magnetic recording systems [8], [9].

Recently, Langberg *et al.* [10] proposed the study of the *Chebychev permutation channel*[1]. In this channel, general vectors are transmitted and all symbols in the vector can be displaced a limited number of r positions away from their origins. In the same paper, Langberg *et al.* studied the combinatorial properties of the channel and provided certain direct and recursive code constructions.

We continue this investigation and provide new code constructions that improve the previous rates. Our constructions make use of a recursive technique where we concatenate several seed codes together. Langberg *et al.* used this technique to construct a family of binary codes of rate 0.609 when $r = 1$ (see [10, Construction B]). We employ this technique in the most general form and propose a prefixing construction that yields a family of binary codes of rate 0.643 when $r = 1$. Furthermore, in the case $r = 1$, using certain computation techniques, we determine the optimal sizes of codes for lengths up to 15.

[1]Langberg *et al.* [10] used the name ℓ_∞*-limited permutation channel with zero error* for this channel.

II. Preliminaries

For integers $a \le b$, let $[a, b]$ denote the set $\{a, \ldots, b\}$. Let n be a positive integer and S_n be the set of all permutations on the set $[1, n]$. For a permutation $\pi \in S_n$, let π_i be the ith component of π, that is, $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$.

For any two permutations $\pi, \pi' \in S_n$, the ℓ_∞-distance is defined as $d_\infty(\pi, \pi') = \max_{i\in[1,n]} |\pi_i - \pi'_i|$. If we denote the identity permutation as $\mathrm{Id} = (1, 2, \ldots, n)$, then the *weight* of a permutation $\pi \in S_n$ is defined as $wt(\pi) = d_\infty(\pi, \mathrm{Id})$. The ℓ_∞-distance is also known as the Chebyshev distance and has been well studied, see for example [11].

The Chebyshev permutation channel (CPC) works as follows [10]. For a q-ary alphabet $\Sigma = [0, q-1]$, consider a transmitted vector $\boldsymbol{x} = x_1x_2\ldots x_n \in \Sigma^n$. The *$r$-bounded Chebyshev permutation channel* distorts $\boldsymbol{x}$ by applying to it a permutation of weight at most r. Thus, the received vector $\boldsymbol{y} = y_1y_2\ldots y_n \in \Sigma^n$ satisfies $\boldsymbol{y} = \pi\boldsymbol{x}$ for some permutation $\pi \in S_n$ with $wt(\pi) \le r$. For a vector $\boldsymbol{x} \in \Sigma^n$, the *ball of radius r centered at $\boldsymbol{x}$* is given by $B_r(\boldsymbol{x}) = \{\boldsymbol{y} \in \Sigma^n : \boldsymbol{y} = \pi\boldsymbol{x}, \pi \in S_n, wt(\pi) \le r\}$.

Definition 1. Given an r-bounded CPC, two vectors $\boldsymbol{x}, \boldsymbol{x}' \in \Sigma^n$ are said to be *confusable* if $B_r(\boldsymbol{x}) \cap B_r(\boldsymbol{x}')$ is nonempty. They are not confusable, otherwise.

Consider the alphabet $\Sigma = [0, q-1]$. A vector $\boldsymbol{x} \in \Sigma^n$ has *composition* $(w_0, \ldots, w_{q-1})$ if w_i is the number of occurrences of i in $\boldsymbol{x}$ for $i \in \Sigma$. If two vectors have different compositions, then they are not confusable. For vectors with the same composition, we consider the following mapping of vectors into permutations.

Suppose $\boldsymbol{x} \in \Sigma^n$ has composition $(w_0, \ldots, w_{q-1})$. For each $i \in \Sigma$, let $L_i(j; \boldsymbol{x})$ be the jth occurrence of i in $\boldsymbol{x}$ and define $\mathsf{supp}_i(x) = (L_i(1; \boldsymbol{x}), \ldots, L_i(w_i; \boldsymbol{x}))$. When $w_i = 0$, the vector $\mathsf{supp}_i(\boldsymbol{x})$ is the empty vector. Define $\mathsf{supp}(\boldsymbol{x}) \in S_n$ to be the concatenation of $\mathsf{supp}_i(\boldsymbol{x})$, $i \in [0, q-1]$, that is,

$$\mathsf{supp}(\boldsymbol{x}) = \mathsf{supp}_0(\boldsymbol{x})|\mathsf{supp}_1(\boldsymbol{x})|\cdots|\mathsf{supp}_{q-1}(\boldsymbol{x}).$$

Here, $\boldsymbol{y}|\boldsymbol{z}$ denotes the concatenation of the vectors $\boldsymbol{y}$ and $\boldsymbol{z}$.

Example 1. Let $\boldsymbol{x} = 001101000$, $\boldsymbol{x}' = 101000001$. Then

$$\mathsf{supp}_0(\boldsymbol{x}) = (1,2,5,7,8,9), \qquad \mathsf{supp}_1(\boldsymbol{x}) = (3,4,6),$$
$$\mathsf{supp}_0(\boldsymbol{x}') = (2,4,5,6,7,8), \qquad \mathsf{supp}_1(\boldsymbol{x}') = (1,3,9).$$

Therefore,

$$\mathsf{supp}(x) = (1,2,5,7,8,9,3,4,6),$$
$$\mathsf{supp}(x') = (2,4,5,6,7,8,1,3,9).$$

The following lemma provides a sufficient condition to check if two vectors with the same composition are confusable.

Lemma 1. *Let $\boldsymbol{x}, \boldsymbol{x}' \in \Sigma^n$ be two vectors with the same composition. If $d_\infty(\mathsf{supp}(\boldsymbol{x}), \mathsf{supp}(\boldsymbol{x}')) \geq 2r+1$, then $\boldsymbol{x}$ and $\boldsymbol{x}'$ are not confusable in an r-bounded CPC.*

In Example 1, $d_\infty(\mathsf{supp}(\boldsymbol{x}), \mathsf{supp}(\boldsymbol{x}')) \geq 3$. Hence, $\boldsymbol{x}$ and $\boldsymbol{x}'$ are not confusable in the 1-bounded CPC by Lemma 1.

Remark 1. Langberg *et al.* defined $d_\infty(\mathsf{supp}(\boldsymbol{x}), \mathsf{supp}(\boldsymbol{x}'))$ to be the LPC_∞*-distance* [10]. However, the condition in Lemma 1 is not necessary. For example, consider $r = 1$, $\boldsymbol{x} = 0011$ and $\boldsymbol{x}' = 1100$. Since $B_1(\boldsymbol{c}) = \{0011, 0101\}$ and $B_1(\boldsymbol{c}') = \{1100, 1010\}$, the vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ are not confusable by definition. However, $\mathsf{supp}(\boldsymbol{x}) = (1,2,3,4)$ and $\mathsf{supp}(\boldsymbol{x}') = (3,4,1,2)$. Thus $d_\infty(\mathsf{supp}(\boldsymbol{x}), \mathsf{supp}(\boldsymbol{x}')) = 2$, which does not satisfy the condition in Lemma 1.

The next lemma provides a necessary and sufficient condition for the confusability of two vectors.

Lemma 2. *Let $\boldsymbol{x} = \boldsymbol{y}|\boldsymbol{z}$ and $\boldsymbol{x}' = \boldsymbol{y}'|\boldsymbol{z}'$ be two different vectors of the same length. Suppose that $\boldsymbol{y}$ and $\boldsymbol{y}'$ are confusable. Then $\boldsymbol{x}$ and $\boldsymbol{x}'$ are confusable if and only if $\boldsymbol{z}$ and $\boldsymbol{z}'$ are confusable.*

Finally, we define the codes capable of correcting errors in Chebyshev permutation channels.

Definition 2. A nonempty subset $\mathcal{C} \subseteq \Sigma^n$ is called an $(n,r)_q$-*CPC code* if any two distinct vectors from $\mathcal{C}$ are not confusable in the r-bounded CPC. The *rate* of the code $\mathcal{C}$ is given by $\log_q |\mathcal{C}|/n$. For a family of codes $\mathcal{C}_n$, the *asymptotic rate* is given by $\lim_{n\to\infty} \log_q |\mathcal{C}_n|/n$.

Given n, r and q, let $A_q(n,r)$ denote the maximum size that an $(n,r)_q$-CPC code can have. A code attaining this size is called *optimal*.

A. Previous Work

Previously known upper and lower bounds of $A_q(n,r)$ for general values of r are given below.

Theorem 1 (Langberg *et al.* [10]). *Let n, $q \geq 2$ and $r \geq 1$ be integers. Then*

$$A_q(n,r) \leq \binom{q+r}{q-1}^{\lceil n/(r+1)\rceil}.$$

When $(2r+1)|q$,

$$A_q(n,r) \geq \left(\frac{q}{2r+1}\right)^n.$$

In particular, there is a code family whose asymptotic rate is at least $1 - \log_q(2r+1)$.

For $r = 1$, an improved upper bound is given below.

Theorem 2 (Langberg *et al.* [10])**.** *For $q \geq 2$, and all $3|n$,*

$$A_q(n,1) \leq \left(q + 2\binom{q}{2} + 2\binom{q}{3}\right)^{n/3}.$$

As a consequence, $A_2(n,1) \leq 2^{2n/3}$ for all $3|n$.

The bound in Theorem 2 is tight for $n = 3$. That is, $A_q(3,1) = q + 2\binom{q}{2} + 2\binom{q}{3}$, where the optimal code consists of codewords of types aaa, aba, bab, abc and cba for all possible distinct symbols a, b and c. No other values of $A_q(n,r)$ are known except when $q = 2$, $r = 1$ and $n \in \{1, \ldots, 6, 9\}$ [10].

When $q = 2$, $r = 1$, the highest known asymptotic rate is 0.609 [10].

B. Our Contributions

As mentioned earlier, we recursively build our codes by concatenating certain seed codes. More formally, let $\mathcal{C}$ and $\mathcal{D}$ be two codes. We define a new code $\mathcal{C}|\mathcal{D} \triangleq \{\boldsymbol{x}|\boldsymbol{y} : \boldsymbol{x} \in \mathcal{C}, \boldsymbol{y} \in \mathcal{D}\}$. If $\mathcal{C}$ has only one codeword $\boldsymbol{x}$, then we write $\boldsymbol{x}\mathcal{D}$ instead of $\{\boldsymbol{x}\}|\mathcal{D}$. Using this simple idea, we build the following code families with higher rates.

- In Section III, we introduce a prefixing construction to build $(n,1)_2$-CPC codes that have asymptotic rate which yields 0.643.
- In Section IV, we build two families of $(n,r)_q$-CPC codes for general q and r. The first family of codes improves the size by a factor of $(1.5)^{n/(2r-1)}$, as compared to codes given in Theorem 1. The second family of codes on the other hand inherits certain local properties.
- For all code constructions, we provide accompanying decoding algorithms that run in linear time.

III. Binary Codes with $r = 1$

In this section, we focus on the binary case when $r = 1$. We first give a prefixing construction for $(n,1)_2$-CPC codes. This family of codes has an asymptotic rate of 0.643 that is significantly higher than the previously known rate of 0.609 [10]. Next, we improve the lower bounds for $A_2(n,1)$ for small values of n via computer search.

A. Prefixing Construction

The main idea of our recursive construction is to attach prefixes carefully to shorter codewords so that the set of the longer words is a code in the 1-bounded CPC. The choice of prefixes is illustrated by the example below.

Example 2. Consider the following optimal $(6,1)_2$-CPC code $\mathcal{C}_6$ of size 16.

$$\begin{aligned}\mathcal{C}_6 = \{&000000, 000100, 000110, 000111,\\ &111000, 111100, 111110, 111111,\\ &100000, 100001, 100011,\\ &011100, 011101, 011111,\\ &001111, 110000\}.\end{aligned}$$

Using our notation, we may write $\mathcal{C}_6$ as,

$$\begin{aligned}\mathcal{C}_6 =& 000\mathcal{C}_3 \cup 111\mathcal{C}_3 \cup 1000\mathcal{C}_2 \cup 0111\mathcal{C}_2\\ &\cup \{001111\} \cup \{110000\},\end{aligned}$$

where $\mathcal{C}_2$ is a $(2,1)_2$-CPC code of size three, and $\mathcal{C}_3$ is a $(3,1)_2$-CPC code of size four.

The observation in Example 2 can be generalized to the following construction that builds up longer codes from shorter ones.

Construction 1. *Fix n. Suppose that $\mathcal{C}_k$ is a $(k,1)_2$-CPC code for $k < n$. Define $\mathcal{C}_n$ recursively as follows:*

$$\begin{aligned}\mathcal{C}_n \triangleq\ & 000\mathcal{C}_{n-3} \cup 111\mathcal{C}_{n-3} \cup 1000\mathcal{C}_{n-4} \cup 0111\mathcal{C}_{n-4}\\ &\cup 001111\mathcal{C}_{n-6} \cup 110000\mathcal{C}_{n-6}.\end{aligned}$$

Since $\mathcal{C}_n$ is the disjoint union of six component codes, the size of $\mathcal{C}_n$ is given by

$$|\mathcal{C}_n| = 2|\mathcal{C}_{n-3}| + 2|\mathcal{C}_{n-4}| + 2|\mathcal{C}_{n-6}|. \tag{1}$$

The next theorem shows that $\mathcal{C}_n$ is also an $(n,1)_2$-CPC code. Since optimal codes of length less than six are known [10], Construction 1 yields a family of codes for all $n \geq 6$.

Theorem 3. *For all $n \geq 6$, the code $\mathcal{C}_n$ from Construction 1 is an $(n,1)_2$-CPC code. The asymptotic rate of this code family $\mathcal{C}_n$ is $\log_2 \lambda \approx 0.643$, where λ is the largest real root of $x^6 - 2x^3 - 2x^2 - 2$.*

Proof. Consider two distinct codewords $\boldsymbol{c}, \boldsymbol{c}' \in \mathcal{C}_n$ and we demonstrate that they are not confusable in the 1-bounded CPC. For convenience, let $\mathcal{P}$ be the set of prefixes $\{000, 111, 1000, 0001, 001111, 110000\}$. If $\boldsymbol{c}$ and $\boldsymbol{c}'$ have same prefix in $\mathcal{P}$, then they are not confusable by Lemma 2.

Hence, it remains to consider the case when $\boldsymbol{c}$ and $\boldsymbol{c}'$ have different prefixes in $\mathcal{P}$. We only check the case where $\boldsymbol{c} \in 000\mathcal{C}_{n-3}$. The other cases can be similarly verified and we omit them in this proof.

- If $\boldsymbol{c}' \in 111\mathcal{C}_{n-3} \cup 1000\mathcal{C}_{n-4} \cup 110000\mathcal{C}_{n-6}$, then $d_\infty(\mathsf{supp}(\boldsymbol{c}), \mathsf{supp}(\boldsymbol{c}')) \geq 3$ since $L_1(1;\boldsymbol{c}) \geq 4$ and $L_1(1;\boldsymbol{c}') = 1$.
- If $\boldsymbol{c}' \in 0111\mathcal{C}_{n-4}$, then $d_\infty(\mathsf{supp}(\boldsymbol{c}), \mathsf{supp}(\boldsymbol{c}')) \geq 3$ since $L_0(2;\boldsymbol{c}) = 2$ and $L_0(2;\boldsymbol{c}') \geq 5$.
- If $\boldsymbol{c}' \in 001111\mathcal{C}_{n-6}$, then $d_\infty(\mathsf{supp}(\boldsymbol{c}), \mathsf{supp}(\boldsymbol{c}')) \geq 4$ since $L_0(3;\boldsymbol{c}) = 3$ and $L_0(3;\boldsymbol{c}') \geq 7$.

Therefore, for all possible $\boldsymbol{c}'$, we have that $d_\infty(\mathsf{supp}(\boldsymbol{c}), \mathsf{supp}(\boldsymbol{c}')) \geq 3$. By Lemma 1, $\boldsymbol{c}$ and $\boldsymbol{c}'$ are not confusable. Thus $\mathcal{C}_n$ is an $(n,1)_2$-CPC code.

Now, the size of $|\mathcal{C}_n|$ satisfies the linear recurrence relation (1). Therefore, following standard techniques (see for example [12]), the asymptotic rate of this code family is given by $\log_2(\lambda) \approx 0.642803$, where λ is the largest real root of $x^6 - 2x^3 - 2x^2 - 2$. ∎

From the proof of Theorem 3, to decode a received vector $\boldsymbol{y}$ to a codeword, we need to recursively determine the correct prefixes (that belong to $\mathcal{P}$) for $\boldsymbol{y}$. More concretely, we have the following decoding algorithm $\texttt{decode}_1$ for codes $\mathcal{C}_n$ that runs in linear time.

Algorithm 1 Linear time decoder for $\mathcal{C}_n$ (Construction 1)

Input: $\boldsymbol{y} \in B_1(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{C}_n$
Output: $\texttt{decode}_1(\boldsymbol{y}, n)$ such that $\texttt{decode}_1(\boldsymbol{y}, n) = \boldsymbol{x}$

1: **if** $n < 6$ **then**
2:     **return** $\boldsymbol{x} \in \mathcal{C}_n$ such that $\boldsymbol{y} \in B_1(\boldsymbol{x})$
3: **if** $y_1y_2 = 00$ and $y_3y_4 \neq 11$ **then**
4:     **return** $000|\texttt{decode}_1(y_3 + y_4|y_5 \dots y_n, n-3)$
5: **if** $y_1y_2 = 11$ and $y_3y_4 \neq 00$ **then**
6:     **return** $111|\texttt{decode}_1(y_3 + y_4 - 1|y_5 \dots y_n, n-3)$
7: **if** ($y_1y_2y_3 = 101$ and $y_4y_5 \neq 00$) or $y_1y_2y_3 = 011$ **then**
8:     **return** $0111|\texttt{decode}_1(y_4 + y_5 - 1|y_6 \dots y_n, n-4)$
9: **if** ($y_1y_2y_3 = 010$ and $y_4y_5 \neq 11$) or $y_1y_2y_3 = 100$ **then**
10:     **return** $1000|\texttt{decode}_1(y_4 + y_5|y_6 \dots y_n, n-4)$
11: **if** $y_1y_2y_3y_4 = 0011$ or $y_1y_2y_3y_4y_5 = 01011$ **then**
12:     **return** $001111|\texttt{decode}_1(y_6 + y_7 - 1|y_8 \dots y_n, n-6)$
13: **if** $y_1y_2y_3y_4 = 1100$ or $y_1y_2y_3y_4y_5 = 10100$ **then**
14:     **return** $110000|\texttt{decode}_1(y_6 + y_7|y_8 \dots y_n, n-6)$

Example 3. To illustrate Algorithm 1, consider the received word $\boldsymbol{y} = 001110101001$ of length 12. Since $\boldsymbol{y}$ has prefix 0011, we go to line 14 and compute $\texttt{decode}_1(\boldsymbol{y}', 6)$ with $\boldsymbol{y}' = 001001$. Next, since $\boldsymbol{y}'$ has prefix 00 with $y_3'y_4' \neq 11$, we go to line 4 and compute $\texttt{decode}_1(\boldsymbol{y}'', 3)$ with $\boldsymbol{y}'' = 101$. Finally, since $\boldsymbol{y}'' \in B_1(110)$, we have $\texttt{decode}_1(\boldsymbol{y}'', 3) = 110$.

In summary, we have that

$$\begin{aligned}&\texttt{decode}_1(001110101001, 12)\\ &= 001111|\texttt{decode}_1(001001, 6)\\ &= 001111|000|\texttt{decode}_1(101, 3)\\ &= 001111|000|110.\end{aligned}$$

Direct application of Construction 1 with optimal codes of length at most six [10, Table 1] yields two new optimal codes of lengths eight and nine. From (1), we have $|\mathcal{C}_8| = 2|\mathcal{C}_5| + 2|\mathcal{C}_4| + 2|\mathcal{C}_2| = 46$ and $|\mathcal{C}_9| = 2|\mathcal{C}_6| + 2|\mathcal{C}_5| + 2|\mathcal{C}_3| = 64$, and these values match the upper bounds provided in [10, Table 1]. Therefore, we have $A_2(8,1) = 46$ and $A_2(9,1) = 64$.

B. Computation Results from Finding Maximum Cliques

In this subsection, we update the values of lower bound for $A_2(n,1)$ via computer search. Langberg *et al.* first constructed a table of upper and lower bounds for $A_2(n,1)$ [10, Table 1], where many values were obtained from computer search. Langberg *et al.* then conjectured that optimal asymptotic rate for $r = 1$ is $2/3$ in [10], and that $A_2(n,1) = 2^{2n/3}$ for all $3|n$. From their computations, they verified the conjecture for $n \in \{3, 6, 9\}$.

We continue this line of investigation and improve the lower bounds on $A_2(n,1)$ for $7 \leqslant n \leqslant 16$ (see Table I). In particular, we *determine* $A_2(n,1)$ *for* $n \leqslant 15$. To do so, we set up a specific program that searches for the largest clique in a graph.

For a fixed value of n, we define a family of graphs parametrized by the weight w, where $0 \leqslant w \leqslant n$. In particular, the graph $G(n,w)$ consists of vertices which correspond to the set of all binary words of length n and weight w. An edge exists between two vertices, i.e., two words, if they are not confusable. The algorithm `MaxCliqueDyn` [13] is then used to determine the maximum size of the clique in these graphs $G(n,w)$. Since two words with different weights are not confusable, the set of all words in these maximum cliques form an $(n,1)_2$-CPC code. Hence, we determine $A_2(n,1)$ for $n \leqslant 15$.

For $n = 16$, we apply Construction 1 to find $A_2(16,1) \geqslant 1644$. We summarize the results in Table I and highlight the optimal values in bold.

n	Upper Bound	Lower Bound from [10]	New Lower Bound
3	4	4	**4**
4	8	8	**8**
5	12	12	**12**
6	16	16	**16**
7	30	28	**30**
8	46	42	**46**
9	64	64	**64**
10	116	104	**116**
11	178	157	**178**
12	256	246	**256**
13	450	388	**450**
14	696	594	**696**
15	1024	930	**1024**
16	1750	1454	1644

TABLE I: Upper and Lower Bounds on $A_2(n,1)$

IV. Code Construction for General q and r

In this section, we use the string concatentation method to construct two families of q-ary codes in r-bounded CPC. The first construction yields a family of codes with sizes larger than those constructed in Theorem 1, while the second construction yields a family of codes with good local properties.

A. General Code Constructions

For convenience, let $q_1 = \lfloor q/2 \rfloor$, $q_2 = \lceil q/2 \rceil$, $\Sigma_1 = [0, q_1 - 1]$ and $\Sigma_2 = [q_1, q-1]$.

Construction 2. *Let* $\mathcal{D}_i$ *be a* $(2r-1, r)_{q_i}$*-CPC code over* Σ_i *for* $i = 1, 2$. *Let*

$$\mathcal{E}_n = \underbrace{\mathcal{D}_1 | \mathcal{D}_2 | \mathcal{D}_1 | \mathcal{D}_2 | \cdots | \mathcal{D}_j}_{n \text{ times}},$$

where $j = 1$ *if* n *is odd,* $j = 2$*, otherwise.*

Theorem 4. *For all integers* n *and* q*, the code* $\mathcal{E}_n$ *from Construction 2 is an* $(n(2r-1), r)_q$*-CPC code. The decoder for* $\mathcal{E}_n$ *is given by Algorithm 2.*

Proof Outline. To show that $\mathcal{E}_n$ is an $(n(2r-1), r)_q$-CPC code, we demonstrate the correctness of Algorithm 2. In particular, we suppose that a codeword $\boldsymbol{x} = \boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_n$ belongs to $\mathcal{E}_n$. Hence, $\boldsymbol{x}_i$ belongs to $\mathcal{D}_1$ if i is odd and $\mathcal{D}_2$, otherwise. Let $\boldsymbol{y} = \boldsymbol{y}_1|\boldsymbol{y}_2|\cdots|\boldsymbol{y}_n \in B_r(\boldsymbol{x})$ and we prove that the output of Algorithm 2 is indeed $\boldsymbol{x}$, or, $\texttt{decode}_2(\boldsymbol{y}) = \boldsymbol{x}$.

Using the notation in Algorithm 2, we prove the following properties for all $i \in [1, n]$,

(i) $\boldsymbol{y}'_i|\boldsymbol{y}_{i+1}|\boldsymbol{y}_{i+2}|\cdots|\boldsymbol{y}_n \in B_r(\boldsymbol{x}_i|\boldsymbol{x}_{i+1}|\boldsymbol{x}_{i+2}|\cdots|\boldsymbol{x}_n)$;
(ii) $\boldsymbol{z}_i \in B_r(\boldsymbol{x}_i)$.

The proof is by induction and due to space constraints, we omit the details of the proof. The theorem is then immediate from the above properties. ■

Algorithm 2 Decoder for $\mathcal{E}_n$

Input: $\boldsymbol{y} = \boldsymbol{y}_1|\boldsymbol{y}_2|\cdots|\boldsymbol{y}_n \in B_r(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{E}_n$
Output: $\texttt{decode}_2(\boldsymbol{y})$ such that $\texttt{decode}_2(\boldsymbol{y}) = \boldsymbol{x}$
$\boldsymbol{y}'_1 \leftarrow \boldsymbol{y}_1$
for $i \in [1, n]$ **do**
  $j \leftarrow 1$ if i is odd, $j \leftarrow 2$, otherwise
  $\boldsymbol{z}_i \leftarrow$ subsequence of $\boldsymbol{y}'_i|\boldsymbol{y}_{i+1}$ formed by the first $(2r-1)$ entries in Σ_j
  $\boldsymbol{y}'_{i+1} \leftarrow$ subsequence that remains after removing $\boldsymbol{z}_i$ from $\boldsymbol{y}'_i|\boldsymbol{y}_{i+1}$
  $\boldsymbol{x}_i \leftarrow$ the word decoded from $\boldsymbol{z}_i$ using $\mathcal{D}_j$
return $\boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_n$

Example 4. Consider $r = 1$ in Construction 2. It is trivial that an optimal $(1,1)_{q_i}$-CPC code of size q_i exists for $i = 1, 2$. Therefore, by Theorem 4, there exists an $(n,1)_q$-CPC code of size $(q_1)^{\lceil n/2 \rceil}(q_2)^{\lfloor n/2 \rfloor} \approx (q/2)^n$. In other words, this family of codes has rate $1 - \log_q 2$.

The previous known construction in Theorem 1 yields a family of $(n,1)_q$-CPC codes of size $(q/3)^n$, or rate $1 - \log_q 3$. Therefore, Construction 2 improves the previous known lower bound by a factor of $(1.5)^n$ for all n.

Example 5. Generalizing Example 4, we consider $r \geqslant 1$ in Construction 2 and $\mathcal{D}_i$ to be the set of all possible nondecreasing sequences of length $2r - 1$ over Σ_i for $i = 1, 2$. Hence, for $i = 1, 2$, $\mathcal{D}_i$ is a $(2r-1, r)_{q_i}$-CPC code and in particular, $\mathcal{D}_i$ has size $\binom{q_i+2r-2}{2r-1}$. Applying Theorem 4, we obtain an $(n(2r-1), r)_q$-CPC code $\mathcal{E}_n$ of size $\binom{q_1+2r-2}{2r-1}^{\lceil n/2 \rceil}\binom{q_2+2r-2}{2r-1}^{\lfloor n/2 \rfloor} \approx \binom{q/2+2r-2}{2r-1}^n$.

Again, the previous known lower bound in Theorem 1 yields codes of size $(q/(2r+1))^{n(2r-1)}$. Therefore, via Stirling's approximation, Construction 2 improves the construction by a factor of $(1.5)^n$ for all n.

Furthermore, Algorithm 2 may be simplified and have its running time reduced as shown in Algorithm 2$*$.

Algorithm 2$*$ Decoder for $\mathcal{E}_n$ in Example 5

Input: $\boldsymbol{y} = \boldsymbol{y}_1|\boldsymbol{y}_2|\cdots|\boldsymbol{y}_n \in B_r(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{E}_n$
Output: $\texttt{decode}^*_2(\boldsymbol{y})$ such that $\texttt{decode}^*_2(\boldsymbol{y}) = \boldsymbol{x}$
$\boldsymbol{y}'_1 \leftarrow \boldsymbol{y}_1$
for $i \in [1, n]$ **do**
  $j \leftarrow 1$ if i is odd, $j \leftarrow 2$, otherwise
  $\boldsymbol{z}_i \leftarrow$ subsequence of $\boldsymbol{y}'_i|\boldsymbol{y}_{i+1}$ formed by the first $(2r-1)$ entries in Σ_j
  $\boldsymbol{y}'_{i+1} \leftarrow$ subsequence that remains after removing $\boldsymbol{z}_i$ from $\boldsymbol{y}'_i|\boldsymbol{y}_{i+1}$
  $\boldsymbol{x}_i \leftarrow$ sorted $\boldsymbol{z}_i$ in nondecreasing order
return $\boldsymbol{x}_1|\boldsymbol{x}_2|\cdots|\boldsymbol{x}_n$

B. Codes with Local Properties

The string concatenation method can be used to construct CPC codes with local properties. In this setting, a message vector $\boldsymbol{x}$ is transmitted over the r-bounded CPC and the received vector $\boldsymbol{y}$ is stored in memory without any error-correction. At a later time, a user would like to retrieve certain *parts* of the message $\boldsymbol{x}$ by accessing only a *limited portion* of the stored vector $\boldsymbol{y}$. Our coding objective is to ensure the correct retrieval of information, while minimizing the number of coordinates accessed. Our construction is as follows.

Construction 3. *Let $\ell \geqslant 2$. Let $\mathbf{1}$ denote the all-ones vector of length $2r+1$ and $\mathcal{F}_0 = \{\lambda\mathbf{1} : \lambda \in \Sigma\}$. Suppose that $\mathcal{F}_i$ is an $(n_i, r)_q$-CPC code for $i \in [1, \ell]$. Let*

$$\mathcal{G}_\ell = \underbrace{\mathcal{F}_1|\mathcal{F}_0|\mathcal{F}_2|\mathcal{F}_0|\cdots|\mathcal{F}_0|\mathcal{F}_\ell}_{(2\ell-1)\text{ blocks}}.$$

Using similar idea as in the proof of Theorem 4, we state the following result without proof.

Theorem 5. *The code $\mathcal{G}_\ell$ from Construction 3 is an $(n,r)_q$-CPC code with $n = (2r+1)(\ell-1) + \sum_{i=1}^{\ell} n_i$. Furthermore, suppose that $\boldsymbol{x} \in \mathcal{G}_\ell$ is transmitted and $\boldsymbol{y} \in B_r(\boldsymbol{x})$ is received. Using Algorithm 3, any coordinate of $\boldsymbol{x}$ can be correctly decoded by accessing at most $\max_{i\in[1,\ell]} n_i + 2r + 2$ coordinates of $\boldsymbol{y}$.*

We adopt the following notation in Algorithm 3. A typical code in $\mathcal{G}_\ell$ is written as

$$\boldsymbol{x} = \boldsymbol{x}_1|\boldsymbol{x}_{(0,1)}|\boldsymbol{x}_2|\boldsymbol{x}_{(0,2)}|\cdots|\boldsymbol{x}_{(0,\ell-1)}|\boldsymbol{x}_\ell. \tag{2}$$

Let $\boldsymbol{y} \in B_r(\boldsymbol{x})$ and we rewrite $\boldsymbol{y}$ as:

$$\boldsymbol{y} = \boldsymbol{y}_1|\boldsymbol{y}_{(0,1)}|\boldsymbol{y}_2|\boldsymbol{y}_{(0,2)}|\cdots|\boldsymbol{y}_{(0,\ell-1)}|\boldsymbol{y}_\ell. \tag{3}$$

Algorithm 3 Local decoder for $\mathcal{G}_\ell$

Input: $\boldsymbol{y}$ as defined by (2) and (3)
index $i \in [1,\ell] \cup (\{0\} \times [1, \ell-1])$
Output: $\texttt{decode}_3(\boldsymbol{y}, i)$ such that $\texttt{decode}_3(\boldsymbol{y}, i) = \boldsymbol{x}_i$
if $i = (0, i')$ **then**
  $\lambda \leftarrow (r+1)$st coordinate of $\boldsymbol{y}_{(0,i')}$
  return $\lambda\mathbf{1}$
else
  $\lambda_L \leftarrow (r+1)$st coordinate[2] of $\boldsymbol{y}_{(0,i-1)}$
  $\lambda_R \leftarrow (r+1)$st coordinate of $\boldsymbol{y}_{(0,i)}$
  $\boldsymbol{y}' \leftarrow$ substring of length $n_i + 2r$ by concatenating the last r coordinates of $\boldsymbol{y}_{(0,i-1)}$, $\boldsymbol{y}_i$ and the first r coordinates of $\boldsymbol{y}_{(0,i)}$
  $\boldsymbol{z} \leftarrow$ subsequence that remains after removing r λ_L's and r λ_R's from $\boldsymbol{y}'$
  $\boldsymbol{x}_i \leftarrow$ the word decoded from $\boldsymbol{z}$ using $\mathcal{F}_i$
  return $\boldsymbol{x}_i$

Example 6. Consider $q = 2$ and $r = 1$. While Construction 3 is similar to Construction B in [10], the rate obtained from the former is higher. Consider the same seed code $\mathcal{F}$, a $(24, 1)_2$-CPC code of size 50220 (see [10, Example 17]). Apply Construction 3 by setting $\mathcal{F}_i = \mathcal{F}$ for all $i \in [1, \ell]$ and we obtain the code $\mathcal{G}_\ell$ to be a $(27\ell - 3, 1)_2$-CPC code of size $2^{\ell-1} \cdot 50220^\ell$. Then the asymptotic rate of this code family is 0.615. On the other hand, using the same seed code, Construction B yields a family with rate 0.609.

While the rate of this code family is not as high as the family of codes constructed by Construction 1, the codes constructed in this example enjoy certain local properties. Specifically, suppose that the message $\boldsymbol{x} \in \mathcal{G}_\ell$ is transmitted and $\boldsymbol{y}$ is received. Using Algorithm 3, we are able to retrieve any bit of $\boldsymbol{x}$ by reading at most 28 consecutive bits in $\boldsymbol{y}$ for all values of ℓ.

[2]Due to space constraints, we omit the detailed steps in the algorithm when the index i assumes boundary values 1 and ℓ.

V. Conclusion

Using string concatenation, we have constructed codes capable of correcting errors in the Chebyshev permutation channel. In addition to higher rates, our codes have linear time decoding algorithms and a particular class of codes possesses good local properties.

For the special case $q = 2$, $r = 1$, we have verified the conjecture that $A_2(n, 3) = \lfloor 2^{2n/3} \rfloor$ for $n \leqslant 15$ and constructed a family of codes with rates 0.643. Unfortunately, the conjecture remains open.

Acknowledgement

This work was done while X. Zhang was a research fellow at Nanyang Technological University.

The authors thank the anonymous reviewers and the TPC member for their constructive comments.

References

[1] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data networks*. Prentice-Hall International New Jersey, 1992, vol. 2.

[2] M. Kovacevic and D. Vukobratovic, "Multiset codes for permutation channels," *arXiv preprint arXiv:1301.7564*, 2013.

[3] M. Kovacevic and P. Popovski, "Zero-error capacity of a class of timing channels," *IEEE Trans. Inform. Theory*, vol. 60, no. 11, pp. 6796–6800, 2014.

[4] V. Anantharam and S. Verdu, "Bits through queues," *IEEE Trans. Inform. Theory*, vol. 42, no. 1, pp. 4–18, 1996.

[5] S. Kadloor, R. S. Adve, and A. W. Eckford, "Molecular communication using brownian motion with drift," *IEEE Trans. NanoBioscience*, vol. 11, no. 2, pp. 89–99, 2012.

[6] J. M. Walsh and S. Weber, "Capacity region of the permutation channel," in *46th Ann. Allerton Conf. Commun. Control Comput.*. 2008, pp. 646–652.

[7] J. M. Walsh, S. Weber, and C. W. Maina, "Optimal rate–delay tradeoffs and delay mitigating codes for multipath routed and network coded networks," *IEEE Trans. Inform. Theory*, vol. 55, no. 12, pp. 5491–5510, 2009.

[8] S. Shamai and E. Zehavi, "Bounds on the capacity of the bit-shift magnetic recording channel," *IEEE Trans. Inform. Theory*, vol. 37, no. 3, pp. 863–872, 1991.

[9] V. Y. Krachkovsky, "Bounds on the zero-error capacity of the input-constrained bit-shift channel," *IEEE Trans. Inform. Theory*, vol. 40, no. 4, pp. 1240–1244, 1994.

[10] M. Langberg, M. Schwartz, and E. Yaakobi, "Coding for the ℓ_∞-limited permutation channel," in *Proc. Int. Symp. Inform. Theory*. 2015, pp. 1936–1940.

[11] M. Deza, and T. Huang, "Metrics on permutations, a survey," in *J. Combinatorics Inform. Syst. Sciences*. Citeseer, 1998.

[12] D. E. Knuth, R. L. Graham, and O. Patashnik, *Concrete mathematics*. Adison Wesley, 1989.

[13] J. Konc, and D. Janežič, "An improved branch and bound algorithm for the maximum clique problem," *MATCH Commun. Math. Comput. Chem.*, 2007, vol. 58, pp. 569–590, http://www.sicmm.org/~konc/maxclique.