Inapproximability Results for the Weight Problems of Subgroup Permutation Codes

Min-Zheng Shieh and Shi-Chun Tsai, *Member, IEEE*

***Abstract*—A subgroup permutation code is a set of permutations on n symbols with the property that its elements are closed under the operation of composition. In this paper, we give inapproximability results for the minimum and maximum weight problems of subgroup permutation codes under several well-known metrics. Based on previous works, we prove that under Hamming, Lee, Cayley, Kendall's tau, Ulam's, and ℓ_p distance metrics, 1) there is no polynomial-time $2^{\log^{1-\epsilon} n}$-approximation algorithm for the minimum weight problem for any constant $\epsilon > 0$ unless $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{\mathrm{polylog}(n)})$ (quasi-polynomial time), and 2) there is no polynomial-time r-approximation algorithm for the minimum weight problem for any constant $r > 1$ unless $\mathrm{P} = \mathrm{NP}$. Under ℓ_∞-metric, we prove that it is NP-hard to approximate the minimum weight problem within factor $2 - \epsilon$ for any constant $\epsilon > 0$. We also prove that for any constant $\epsilon > 0$, it is NP-hard to approximate the maximum weight within $\sqrt[p]{\frac{3}{2}} - \epsilon$ under ℓ_p distance metric, and within $\frac{3}{2} - \epsilon$ under Hamming, Lee, Cayley, Kendall's tau, and Ulam's distance metrics.**

***Index Terms*—Approximation algorithms, coding theory, computational complexity, subgroup code.**

I. Introduction

In this paper, we investigate the complexity of determining the minimum weight and maximum weight of subgroup codes under various distance metrics. Related problems have been studied for linear codes for some time. Here, we focus on inapproximability results for subgroup permutation codes.

A permutation code of length n is a subset of all permutations over $\{1, \ldots, n\}$. We say a permutation code C has minimum distance d under some metric $\delta(\cdot, \cdot)$ if for any pair of distinct permutations π and ρ in C, $\delta(\pi, \rho) \geq d$. We call the set of such permutations an (n, d)-PA. Recently, permutation codes have been found to be useful in several applications, such as power line communication (see [16] and [20]–[22]), multilevel flash memories (see, e.g., [9], [10], and [18]), and cryptography (see [12], [13], and [15]). For these applications, researchers mainly focus on creating permutation codes within certain distance d under Hamming, Kendall's tau, Chebyshev distance metrics, etc.

Manuscript received March 14, 2011; revised December 11, 2011 and May 09, 2012; accepted July 09, 2012. Date of publication July 26, 2012; date of current version October 16, 2012. This work was supported in part by the National Science Council of Taiwan under Contracts NSC-97–2221-E-009–064-MY3 and NSC-98–2221-E-009–078-MY3. This paper was presented in part at the 2010 IEEE International Symposium on Information Theory.

M.-Z. Shieh is with the Intelligent Information and Communications Research Center, National Chiao Tung University, Hsinchu 30050, Taiwan, R.O.C. (e-mail: mzshieh@nctu.edu.tw).

S.-C. Tsai is with the Department of Computer Science, National Chiao Tung University, Hsinchu 30050, Taiwan, R.O.C. (e-mail: sctsai@cs.nctu.edu.tw).

Communicated by E. Arikan, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2012.2208618

We use S_n to represent all of the permutations over $\{1, \ldots, n\}$. S_n is also called the symmetric group in algebra. In this paper, we study permutation codes, which also form a subgroup of S_n. We call them subgroup codes. A subgroup code C is often defined by a generator set $\{\pi_1, \ldots, \pi_k\}$ and all permutations in C can be written in a sequence of compositions from elements in the generator set.

It is natural to ask how to determine the minimum distance of a code and how to compute the closest codeword for a certain received string. Both problems have analogous versions for linear codes and lattices. For linear code, it is to determine the minimum distance when given the generator matrix of the code [2]. This problem under Hamming distance had been proved to be NP-complete by Vardy [19]. The analogous problem for lattice under Hamming distance is also NP-hard by Arora *et al.* [1]. The corresponding problems for lattices are the shortest lattice vector problem (SVP) and the closest vector problem. SVP under ℓ_p-norm is NP-hard, even for approximating within $p^{1-\epsilon}$ for any constant $\epsilon > 0$[11]. SVP under Chebyshev distance is also NP-hard, even for approximating within $n^{1/\log\log n}$ factor [6]. For the subgroup permutation code version, both problems are proved to be NP-complete under many distance metrics, such as Hamming, ℓ_p-norm, Kendall's tau, etc. [3], [4].

For right-invariant metrics, the minimum distance problem of subgroup permutation codes is equivalent to finding the minimum weight permutation π, where the weight of π is defined as the distance between π and the identity. Based on Vardy's work[19], Cameron and Wu [4] proved that under Hamming, Lee, Cayley, Kendall's tau, Ulam's distance metrics, and ℓ_p-metric for $1 \leq p < \infty$, the minimum weight problem of subgroup permutation codes is NP-hard. Moreover, based on previous results[4], [5], [7], for the minimum weight problem of subgroup permutation codes over $\{1, \ldots, 2n\}$ under the aforementioned distance metrics, we prove that 1) for any constant $\epsilon > 0$, the existence of a polynomial-time $2^{\log^{1-\epsilon} n}$-approximation algorithm implies $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{\mathrm{polylog}(n)})$; and 2) for any $r > 1$, the existence of a polynomial-time r-approximation algorithm implies $\mathrm{P} = \mathrm{NP}$.

For ℓ_∞-metric, Cameron and Wu [4] proved that the minimum weight problem is NP-complete. However, their reduction overlooked some details. We give a correct version to prove the NP-hardness and further prove that it is NP-hard to approximate the minimum weight within $2 - \epsilon$ for any constant $\epsilon > 0$.

As for the maximum weight problem, with Håstad's result [8], we prove that for any constant $\epsilon > 0$, it is NP-hard to approximate the maximum weight of a subgroup permutation code within $\frac{3}{2} - \epsilon$ under Hamming, Lee, Cayley, Kendall's tau, and Ulam's distance metrics. Under ℓ_p-metric for $1 \leq p < \infty$, we prove that it is NP-hard to approximate the maximum weight

of a subgroup permutation code within $\sqrt[p]{\frac{3}{2}} - \epsilon$. However, for the ℓ_∞-metric, the maximum weight problem has a polynomial time algorithm[4].

The rest of the paper is organized as follows. We define some notations in Section II. We show the inapproximability results of the minimum weight problems in Section III. We give the inapproximability result of the maximum weight problem in Section IV. Finally, Section V concludes the paper.

II. Preliminary

We use $[n]$ to indicate the set $\{1, \ldots, n\}$. A permutation π over $[n]$ is a bijective function from $[n]$ to $[n]$. There are several representations for a permutation. Here we use a truth table to denote a permutation $\pi = [\pi(1), \ldots, \pi(n)]$, which can be written as the product of cycles. A cycle $(p_0, \ldots, p_{k-1})$ represents a permutation with $\pi(p_i) = p_{i+1}$ for $i \in \mathbb{Z}_k$. Any permutation can be written in a product form of disjoint cycles. For example, $\pi = [2, 3, 1, 4, 6, 5] = (1, 2, 3)(4)(5, 6)$. Usually, we ignore the cycle with only one element; therefore $[2, 3, 1, 4, 6, 5] = (1, 2, 3)(5, 6)$.

Let S_n denote the set of all permutations over $[n]$. It is well known that S_n is a group with the composition operation. We define the product of permutations f and $g \in S_n$ as $fg = [f(g(1)), \ldots, f(g(n))]$. The identity permutation in S_n is $\boldsymbol{e} = [1, \ldots, n]$. f^m indicates the mth power of permutation f, and we define $f^0 = \boldsymbol{e}$ and $f^m = ff^{m-1}$ for $m > 0$. We say that $\{\pi_1, \ldots, \pi_k\}$ is a generator set for a subgroup $H \subseteq S_n$, if every permutation $\pi \in H$ can be written as a product of a sequence of compositions from elements in the generator set.

We say δ is a right-invariant metric if $\delta(\pi, \rho) = \delta(\pi\tau, \rho\tau)$ for all permutations π, ρ, and τ. In this paper, we adopt the following right-invariant metrics.

1) Hamming distance: $d_H(\pi, \rho) = |\{i \in [n] : \pi(i) \neq \rho(i)\}|$.
2) ℓ_p-metric: $\ell_p(\pi, \rho) = \sqrt[p]{\sum_{i\in[n]} |\pi(i) - \rho(i)|^p}$.
3) ℓ_∞-metric: $\ell_\infty(\pi, \rho) = \max_{i\in[n]} |\pi(i) - \rho(i)|$.
4) Lee distance: $d_L(\pi, \rho) = \sum_{i\in[n]} \min(|\pi(i) - \rho(i)|, n - |\pi(i) - \rho(i)|)$.
5) Cayley distance: $d_T(\pi, \rho)$ is the minimum number of transpositions required to obtain π from ρ.
6) Kendall's tau: $d_I(\pi, \rho)$ is the minimum number of adjacent transpositions required to obtain π from ρ.
7) Ulam's distance: $d_U(\pi, \rho) = n - k$ where the longest increasing subsequence in $\pi\rho^{-1}(1), \ldots, \pi\rho^{-1}(n)$ has length k.

We say that a permutation π has weight $w_\delta(\pi) \triangleq \delta(\boldsymbol{e}, \pi)$ under a right-invariant metric δ. We define the maximum and minimum weight problems for subgroup permutation code under a right-invariant metric δ as follows. We call it $\mathrm{MAXWSPA}_\delta$ and $\mathrm{MINWSPA}_\delta$ for short, respectively. Both of them are in NP, since computing the weight of a permutation π and verifying if π is in the subgroup by Schreier–Sims algorithm [17] are in P.

Definition 1 ($\mathrm{MAXWSPA}_\delta$)*:* Given a generator set $\{g_1, \ldots, g_k\}$ for a subgroup H of S_n and a positive integer B, determine if there exists a permutation $\pi \in H$ such that $w_\delta(\pi) \geq B$.

It is already known that $\mathrm{MAXWSPA}_{\ell_\infty}$ has a polynomial time algorithm[4].

Definition 2 ($\mathrm{MINWSPA}_\delta$)*:* Given a generator set $\{g_1, \ldots, g_k\}$ for a subgroup H of S_n and a positive integer B, determine if there exists a permutation $\pi \in H$ such that $0 < w_\delta(\pi) \leq B$.

We briefly review some handy tools from works by Cameron and Wu[4]. They gave a mapping ψ from binary strings of length n to permutations over $[2n]$ as $\psi(x) = \prod_{i:x_i=1}(2i-1, 2i)$. For example, $\psi(101) = (1, 2)(5, 6)$. There are close connections between the weight of $\psi(x)$ under various metrics and the Hamming weight of x, i.e., the number of 1 in x. We use $w_\delta(y)$ to indicate the weight of y under metric δ.

Fact 1 (see [4]): For every binary string x of Hamming weight W, we have the following relations.

1) $w_{d_T}(\psi(x)) = w_{d_I}(\psi(x)) = w_{d_U}(\psi(x)) = W$.
2) $w_{d_H}(\psi(x)) = w_{d_L}(\psi(x)) = 2W$.
3) $w_{\ell_p}(\psi(x)) = \sqrt[p]{2W}$ for $1 \leq p < \infty$.

For a binary code C, let $\Psi(C) = \{\psi(x) : x \in C\}$. Cameron and Wu[4] also showed a connection between binary linear codes and subgroup permutation codes.

Fact 2 (see [4]): Suppose a binary linear code C has a basis $\{b_1, \ldots, b_n\}$; then $\Psi(C)$ is a subgroup permutation code and $\{\psi(b_1), \ldots, \psi(b_n)\}$ generates $\Psi(C)$. Moreover, if the maximum and minimum Hamming weights of C are $W_{\max}$ and $W_{\min}$, respectively, then the following statements are true.

1) Under Cayley distance, Kendall's tau distance and Ulam's distance, the maximum and minimum weights of $\Psi(C)$ are $W_{\max}$ and $W_{\min}$, respectively.
2) Under Hamming distance and Lee distance, the maximum and minimum weights of $\Psi(C)$ are $2W_{\max}$ and $2W_{\min}$, respectively.
3) Under ℓ_p-metric for $1 \leq p < \infty$, the maximum and minimum weights of $\Psi(C)$ are $\sqrt[p]{2W_{\max}}$ and $\sqrt[p]{2W_{\min}}$, respectively.

They used these facts to prove the NP-hardness of the minimum weight problems of subgroup permutation codes under Hamming, Lee, Cayley, Kendall's tau, Ulam's, and ℓ_p distance metrics, by reductions from the minimum weight problem of binary linear codes under Hamming distance, which was proved to be NP-hard by Vardy[19].

Theorem 1 (see [19]): The minimum weight problem for binary linear codes under Hamming distance is NP-hard.

Corollary 1 (see [4]): Under Hamming distance, Lee distance, Cayley distance, Kendall's tau distance, Ulam's distance, and ℓ_p-metric for $1 \leq p < \infty$, the minimum weight problem for subgroup permutation codes is NP-hard.

Definition 3: For $r > 1$, we say that an algorithm A is an r-approximation algorithm for a minimization (maximization) problem if A always outputs a feasible solution whose cost is no more (less) than r ($\frac{1}{r}$) times of the minimum (maximum) cost on any input, respectively.

Note that A cannot output an answer whose cost is less (more) than the minimum (maximum) cost, respectively, since it is not a feasible solution.

III. Minimum Weight Problems

In this section, we first give inapproximability results for the minimum weight problems under various metrics mentioned in Section II except the ℓ_∞-metric in Section III. Then, we clarify Cameron and Wu's reduction in [4] for $\mathrm{MINWSPA}_{\ell_\infty}$. At last, we show the inapproximability result for $\mathrm{MINWSPA}_{\ell_\infty}$ by modifying the reduction in [4].

A. Under Non-ℓ_∞-Metric

We can easily obtain inapproximability results from Fact 2, which actually provides a gap-preserving reduction and can be used for proving inapproximability results. Dumer *et al.* [7] gave some inapproximability results on the minimum weight problem for linear codes via randomized reductions, and these reductions were derandomized by Cheng and Wan [5].

Theorem 2 (see [5]): For any constant $\epsilon > 0$, the existence of a polynomial-time $2^{\log^{1-\epsilon} n}$-approximation algorithm for the minimum Hamming weight problem of linear codes over any finite field implies $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{\mathrm{polylog}(n)})$. Moreover, for arbitrary constant $c > 1$, the existence of a polynomial-time c-approximation algorithm for the minimum Hamming weight problem of linear codes over any finite field implies $\mathrm{P} = \mathrm{NP}$.

Corollary 2: For any constant $\epsilon > 0$, the existence of a polynomial-time $2^{\log^{1-\epsilon} n}$-approximation algorithm for the minimum weight problem of subgroup permutation codes over $[2n]$, under Hamming, Lee, Cayley, Kendall's tau, Ulam's distance metrics, and ℓ_p-metric for $1 \leq p < \infty$, implies $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{\mathrm{polylog}(n)})$.

Proof: Assume A is a polynomial-time $2^{\log^{1-\epsilon} n}$-approximation algorithm for $\mathrm{MINWSPA}_{d_H}$. Then, we can approximate the minimum Hamming weight problem for binary linear codes by the following procedure.

1) On input basis $\{b_1, \ldots, b_n\}$, construct the generator set $\{\psi(b_1), \ldots, \psi(b_n)\}$.
2) Output $s/2$, where s is the result of A on input $\{\psi(b_1), \ldots, \psi(b_n)\}$.

Let the $W_{\min}$ be the minimum Hamming weight of the binary linear code generated by $\{b_1, \ldots, b_n\}$. By Fact 2, we know the minimum weight of the subgroup permutation code generated by $\{\psi(b_1), \ldots, \psi(b_n)\}$ is $2W_{\min}$. Since A is a $2^{\log^{1-\epsilon} n}$-approximation algorithm, the output is at most $\frac{1}{2} \cdot 2^{\log^{1-\epsilon} n} \cdot 2W_{\min} = 2^{\log^{1-\epsilon} n} W_{\min}$. This implies the procedure is a polynomial-time $2^{\log^{1-\epsilon} n}$-approximation algorithm for the minimum Hamming weight problem of binary linear codes. Therefore, we have $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{\mathrm{polylog}(n)})$ by Theorem 2. For the rest metrics, the claim follows similarly. ■

Moreover, we have the following corollary.

Corollary 3: For arbitrary constant $c > 1$, the existence of a polynomial-time c-approximation algorithm for the minimum weight problem of subgroup permutation codes over $[2n]$, under Hamming, Lee, Cayley, Kendall's tau, Ulam's distance metrics, and ℓ_p-metric for $1 \leq p < \infty$, implies $\mathrm{P} = \mathrm{NP}$.

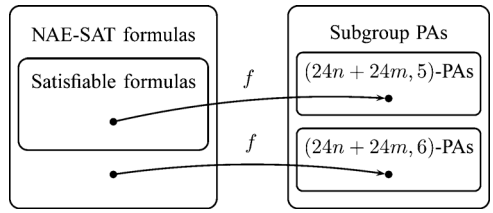

Fig. 1. Sketch of the reduction in [4].

B. Cameron and Wu's Reduction for $\mathrm{MINWSPA}_{\ell_\infty}$

In the previous section, we obtain the inapproximability results of the minimum weight problem under various metrics based on Fact 1. The weight of $\psi(x)$ under those metrics increases monotonically with the Hamming weight W of the binary string x. Observe that $w_{\ell_\infty}(\psi(x)) = 1$ for every x with nonzero Hamming weight. Thus, $w_{\ell_\infty}(\psi(x))$ does not increase monotonically with the Hamming weight of x. This indicates that the reduction in Section III-A does not work under ℓ_∞-metric. Therefore, we need a different approach to deal with the minimum weight problem under ℓ_∞-metric.

Cameron and Wu [4] gave another reduction from Not-All-Equal-SAT (NAESAT) to $\mathrm{MINWSPA}_{\ell_\infty}$. NAESAT is a classical NP-complete problem (e.g., see [14, p. 187]), thus $\mathrm{MINWSPA}_{\ell_\infty}$ is also NP-hard. However, the reduction in Cameron and Wu's work[4] contains a flaw, which couldn't prove the correctness of their claim. In this section, we fix the flaw in their proof and clarify their reduction.

We give the formal definition of Not-All-Equal-SAT problem as follows.

Definition 4 (NAESAT): Given a boolean formula ϕ in conjunctive normal form, which consists of m exact-3-literal clauses $c_1, \ldots, c_m$ over n variables $x_1, \ldots, x_n$, decide whether there exists an assignment σ such that for every clause c, not all literals in c are assigned to the same truth value.

If a formula ϕ has such an assignment σ, then ϕ is satisfiable, and we say σ is a satisfying assignment. Otherwise, ϕ is unsatisfiable. Cameron and Wu's reduction [4] maps an n-variable-m-clause formula ϕ into a generator set $f(\phi) = \{g_1, g_1', \ldots, g_n, g_n', g^*\}$, where $g_1, g_1', \ldots, g_n, g_n', g^*$ permute elements in $\{1, 2, \ldots, 2n + 24m\}$, which actually should be $\{1, 2, \ldots, 24n + 24m\}$.

To illustrate the permutations, we define two handy operations as follows.

Definition 5: Let $(p_1, \ldots, p_\ell)$ be a cycle of a permutation.

1) shift operation: $s_k(p_1, \ldots, p_\ell) := (p_1 + k, \ldots, p_\ell + k)$.
2) stretch operation: $a_k(p_1, \ldots, p_\ell) := (k * p_1, \ldots, k * p_\ell)$.

Both operations can be applied to permutations, i.e., working on each cycle of the permutations. For example, let $\pi = (1, 2)(3, 4)$, we have $s_4(\pi) = (1+4, 2+4)(3+4, 4+4) = (5, 6)(7, 8)$ and $a_2(\pi) = (1 \cdot 2, 2 \cdot 2)(3 \cdot 2, 4 \cdot 2) = (2, 4)(6, 8)$. Note that shift operation does not change the weight under ℓ_∞-metric since the differences between entries are preserved and for stretch operation a_k the weight is amplified by k times.

TABLE I
OPERATIONS OF K_3'

$\circ$	$\boldsymbol{e}$	h_1	h_2	h_3	g	gh_1	gh_2	gh_3
$\boldsymbol{e}$	$\boldsymbol{e}$	h_1	h_2	h_3	g	gh_1	gh_2	gh_3
h_1	h_1	$\boldsymbol{e}$	h_3	h_2	gh_1	g	gh_3	gh_2
h_2	h_2	h_3	$\boldsymbol{e}$	h_1	gh_2	gh_3	g	gh_1
h_3	h_3	h_2	h_1	$\boldsymbol{e}$	gh_3	gh_2	gh_1	g
g	g	gh_1	gh_2	gh_3	$\boldsymbol{e}$	h_1	h_2	h_3
gh_1	gh_1	g	gh_3	gh_2	h_1	$\boldsymbol{e}$	h_3	h_2
gh_2	gh_2	gh_3	g	gh_1	h_2	h_3	$\boldsymbol{e}$	h_1
gh_3	gh_3	gh_2	gh_1	g	h_3	h_2	h_1	$\boldsymbol{e}$

Observe that both operations preserve the algebraic structure, i.e., $s_k(\pi)s_k(\rho) = s_k(\pi\rho)$ and $a_k(\pi)a_k(\rho) = a_k(\pi\rho)$ for arbitrary permutations π, ρ and any integer k. We will use the *shift* and *stretch* operations for constructing permutations.

Before we go through Cameron and Wu's reduction [4] in detail, we define some useful notations for construction. Given a binary string $b_1 \cdots b_n$, we recursively define a permutation $\kappa_{b_1 \cdots b_n}$ and the set of such permutations as follows.

Definition 6: Let $\kappa_0 = \boldsymbol{e}$, $\kappa_1 = (1,2)$.

1) $\kappa_{0^{n-1}1} = (1,2)(3,4)\cdots(2^n - 1, 2^n)$
2) $\kappa_{b_1 b_2 \cdots b_{n-1} 0} = a_2(\kappa_{b_1 \cdots b_{n-1}}) s_{-1}(a_2(\kappa_{b_1 \cdots b_{n-1}}))$.
3) $\kappa_{b_1 b_2 \cdots b_{n-1} 1} = \kappa_{b_1 b_2 \cdots b_{n-1} 0} \kappa_{0^{n-1}1}$ for $b_1 \cdots b_{n-1} \neq 0^{n-1}$.
4) $K_n = \{\kappa_{b_1 \cdots b_n} : b_1, \ldots, b_n \in \{0,1\}\}$.

Examples:

$$\begin{aligned}
\kappa_{00} &= a_2(\kappa_0)s_{-1}(a_2(\kappa_0)) = (1)(2)(3)(4) = \boldsymbol{e},\\
\kappa_{10} &= a_2((1,2))s_{-1}(a_2((1,2))) = (1,3)(2,4),\\
\kappa_{11} &= \kappa_{10}(1,2)(3,4) = (1,4)(2,3),\\
\kappa_{110} &= a_2(\kappa_{11})s_{-1}(a_2(\kappa_{11})) = (1,7)(2,8)(3,5)(4,6),\\
\kappa_{111} &= \kappa_{110}(1,2)(3,4)(5,6)(7,8) = (1,8)(2,7)(3,6)(4,5),\\
K_2 &= \{\boldsymbol{e}, (1,2)(3,4), (1,3)(2,4), (1,4)(2,3)\}.
\end{aligned}$$

In Cameron and Wu's work [4], they used permutations in Klein four-group, which is exactly K_2, as the main construction blocks to obtain the NP-hardness results of the maximum weight problem under various metrics. They also modified this reduction to obtain the NP-hardness result for $\mathrm{MINWSPA}_{\ell_\infty}$. We will show that their modified gadgets can actually be constructed by permutations in K_3. These families of permutations have many good properties. First of all, the weight of any permutation κ_β in K_n can be identified with its subscript β. Another good property is that the algebraic structure of K_n under composition is isomorphic to the group of binary strings of length n under the bitwise-exclusive-OR operation. The proofs for Proposition 1 and 2 can be found in the Appendix.

Proposition 1: Let $(\beta)_2$ denote the integer represented by the binary string β. $w_{\ell_\infty}(\kappa_{b_1 \cdots b_n}) = (b_1 \cdots b_n)_2$.

Proposition 2: For $i = 1, \ldots, n$, $b_i, b_i' \in \{0,1\}$, we have that $\kappa_{b_1 \cdots b_n}\kappa_{b_1' \cdots b_n'} = \kappa_{b_1'' \cdots b_n''}$ where $b_i'' = b_i \oplus b_i'$ for $i \in [n]$.

By Proposition 2, K_n is also commutative, and every element in K_n is the inverse permutation of itself.

Proposition 3: For $b_1, \ldots, b_n, b_1', \ldots, b_n' \in \{0,1\}$, we have that $\kappa_{b_1 \cdots b_n}\kappa_{b_1' \cdots b_n'} = \kappa_{b_1' \cdots b_n'}\kappa_{b_1 \cdots b_n}$.

Proof: By Proposition 2, $\kappa_{b_1 \cdots b_n}\kappa_{b_1' \cdots b_n'} = \kappa_{b_1'' \cdots b_n''}$ for $b_i'' = b_i \oplus b_i'$ for $i \in [n]$. Similarly, $\kappa_{b_1' \cdots b_n'}\kappa_{b_1 \cdots b_n}$ also equals $\kappa_{b_1'' \cdots b_n''}$. Therefore, the proposition is true. ■

Proposition 4: $\kappa_{b_1 \cdots b_n} = \kappa_{b_1 \cdots b_n}^{-1}$.

Proof: Since $\kappa_{b_1 \cdots b_n}\kappa_{b_1 \cdots b_n} = \kappa_{0^n} = \boldsymbol{e}$, we have that $\kappa_{b_1 \cdots b_n} = \kappa_{b_1 \cdots b_n}^{-1}$. ■

Now, we introduce the building blocks given by Cameron and Wu [4]. Let

$$\begin{aligned}
h_1 &= (1,3)(2,4)(5,7)(6,8)(9,13)(10,14)\\
&\quad (11,15)(12,16)(17,23)(18,24)(19,21)(20,22)\\
&= \kappa_{010}s_8(\kappa_{100})s_{16}(\kappa_{110})\\
h_2 &= (1,5)(2,6)(3,7)(4,8)(9,15)(10,16)\\
&\quad (11,13)(12,14)(17,19)(18,20)(21,23)(22,24)\\
&= \kappa_{100}s_8(\kappa_{110})s_{16}(\kappa_{010})\\
h_3 &= (1,7)(2,8)(3,5)(4,6)(9,11)(10,12)\\
&\quad (13,15)(14,16)(17,21)(18,22)(19,23)(20,24)\\
&= \kappa_{110}s_8(\kappa_{010})s_{16}(\kappa_{100})\\
g &= (1,8)(2,7)(3,6)(4,5)(9,16)(10,15)\\
&\quad (11,14)(12,13)(17,24)(18,23)(19,22)(20,21)\\
&= \kappa_{111}s_8(\kappa_{111})s_{16}(\kappa_{111}).
\end{aligned}$$

By Proposition 2, we can derive that $h_u h_v = h_w$ for $\{u,v,w\} = \{1,2,3\}$ and $h_1h_2h_3 = \boldsymbol{e}$. Moreover, $\{h_1, h_2, g\}$ generates a commutative group K_3', and it is easy to verify that every element in K_3' is the inverse of itself (see Table I).

By Proposition 1, we can easily characterize the weight of every element in K_3'

$$\begin{aligned}
w_{\ell_\infty}(h_1) &= \max\{w_{\ell_\infty}(\kappa_{010}), w_{\ell_\infty}(\kappa_{100}), w_{\ell_\infty}(\kappa_{110})\} = 6\\
w_{\ell_\infty}(h_2) &= \max\{w_{\ell_\infty}(\kappa_{100}), w_{\ell_\infty}(\kappa_{110}), w_{\ell_\infty}(\kappa_{010})\} = 6\\
w_{\ell_\infty}(h_3) &= \max\{w_{\ell_\infty}(\kappa_{110}), w_{\ell_\infty}(\kappa_{010}), w_{\ell_\infty}(\kappa_{100})\} = 6\\
w_{\ell_\infty}(g) &= \max\{w_{\ell_\infty}(\kappa_{111}), w_{\ell_\infty}(\kappa_{111}), w_{\ell_\infty}(\kappa_{111})\} = 7\\
w_{\ell_\infty}(gh_1) &= \max\{w_{\ell_\infty}(\kappa_{101}), w_{\ell_\infty}(\kappa_{011}), w_{\ell_\infty}(\kappa_{001})\} = 5\\
w_{\ell_\infty}(gh_2) &= \max\{w_{\ell_\infty}(\kappa_{011}), w_{\ell_\infty}(\kappa_{001}), w_{\ell_\infty}(\kappa_{101})\} = 5\\
w_{\ell_\infty}(gh_3) &= \max\{w_{\ell_\infty}(\kappa_{001}), w_{\ell_\infty}(\kappa_{101}), w_{\ell_\infty}(\kappa_{011})\} = 5.
\end{aligned}$$

The variable gadget v_i for the ith variable is $s_{24(i-1)}(h_1)$. The clause gadget $h_{j,k}$ for the kth literal in the jth clause is defined as $s_{24(n+j-1)}(h_k)$. Let $P = \{(i,j,k) : x_i \text{ is the } k-\text{th literal in } c_j\}$, and $Q = \{(i,j,k) : \overline{x}_i \text{ is the } k\text{th literal in } c_j\}$. For an n-variable-m-clause E3CNF-formula (exact-3-literal-conjunctive-normal-form-formula) $\phi = c_1 \wedge \cdots \wedge c_m$ over variables $x_1, \ldots, x_n$, Cameron and Wu constructed a generator set $f(\phi) = \{g_1, \ldots, g_n, g_1', \ldots, g_n', g^*\}$, where the generators are defined as

$$\begin{aligned}
g_i &= v_i \prod_{(i,j,k)\in P} h_{j,k} = s_{24(i-1)}(h_1) \prod_{(i,j,k)\in P} s_{24(n+j-1)}(h_k)\\
g_i' &= v_i \prod_{(i,j,k)\in Q} h_{j,k} = s_{24(i-1)}(h_1) \prod_{(i,j,k)\in Q} s_{24(n+j-1)}(h_k)
\end{aligned}$$

for every $i \in [n]$ and

$$g^* = \prod_{j\in[n+m]} s_{24(j-1)}(g).$$

It is clear that the construction of $f(\phi)$ can be done in polynomial time. Here, we give a simple example as follows. Let

$$\phi=(x_1\vee x_2\vee x_3)\wedge(\overline{x}_1\vee x_2\vee x_3)\wedge(x_1\vee\overline{x}_2\vee x_3)\wedge(\overline{x}_1\vee\overline{x}_2\vee x_3).$$

By their construction, the generator set consists of

$$\begin{aligned}
g_1 &= v_1h_{1,1}h_{3,1} = h_1s_{72}(h_1)s_{120}(h_1)\\
g_1' &= v_1h_{2,1}h_{4,1} = h_1s_{96}(h_1)s_{144}(h_1)\\
g_2 &= v_2h_{1,2}h_{2,2} = s_{24}(h_1)s_{72}(h_2)s_{96}(h_2)\\
g_2' &= v_2h_{3,2}h_{4,2} = s_{24}(h_1)s_{120}(h_2)s_{144}(h_2)\\
g_3 &= v_3h_{1,3}h_{2,3}h_{3,3}h_{4,3}\\
&= s_{48}(h_1)s_{72}(h_3)s_{96}(h_3)s_{120}(h_3)s_{144}(h_3)\\
g_3' &= v_3 = s_{48}(h_1)\\
g^* &= gs_{24}(g)s_{48}(g)s_{72}(g)s_{96}(g)s_{120}(g)s_{144}(g).
\end{aligned}$$

Note that the variable and the clause gadgets use only h_1, h_2, h_3, and g with proper shift operations. Therefore, the generators inherit the commutative property and the self-inverse property from h_1, h_2, h_3, and g. This fact enables us to write every permutation π in the subgroup generated by $f(\phi)$ as $\pi = g^{*z^*}g_1^{z_1}{g_1'}^{z_1'}\cdots g_n^{z_n}{g_n'}^{z_n'}$ where $z^*, z_1, z_1', \ldots, z_n, z_n' \in \{0,1\}$ are determined by π.

Cameron and Wu proved the minimum weight of the permutation code G generated by $f(\phi)$ is 5 if and only if ϕ is a satisfiable NAESAT instance. However, their proof, as mentioned earlier, has a flaw. Here, we give a correct proof.

Theorem 3: For an NAESAT instance ϕ, let G be the group generated by $f(\phi)$. Under ℓ_∞-metric, if ϕ is satisfiable, then the minimum weight of G is 5, else it is 6.

Proof: Every nonidentity permutation π in G must have the minimum ℓ_∞ weight 5, since for every $i \in [n+m]$, elements in $\{24i-23,\ldots,24i\}$ are permuted by $s_{24(i-1)}(\kappa)$ where $\kappa \in K_3'$. Thus, the minimum weight of G is 5 under ℓ_∞-metric.

Assume ϕ is satisfiable. There exists a satisfying assignment σ. Let $\pi = g^*g_1^{z_1}{g_1'}^{1-z_1}\cdots g_n^{z_n}{g_n'}^{1-z_n}$ where $z_i = 1$ if $\sigma(x_i) = \mathrm{T}$ and $z_i = 0$ if $\sigma(x_i) = \mathrm{F}$ for $i \in [n]$. For every $i \in [n]$, π must permute elements in $\{24i-23,\ldots,24i\}$ by $s_{24(i-1)}(gh_1)$, since g_i, g_i', g^* are the only generators permuting $\{24i-23,\ldots,24i\}$ and π uses g^* and exact one of g_i and g_i'. Assume the kth literal of the jth clause is assigned true by σ. If the literal is x_i or $\overline{x}_i$, then π permutes $\{24(n+j)-23,\ldots,24(n+j)\}$ with g_i or g_i', respectively. Since σ is satisfying, there are one or two literals assigned true in each clause. Recall that $h_uh_v = h_w$ for $\{u,v,w\} = \{1,2,3\}$. So we have that π permutes $\{24(n+j)-23,\ldots,24(n+j)\}$ by one of $s_{24(n+j-1)}(gh_1), s_{24(n+j-1)}(gh_2), s_{24(n+j-1)}(gh_3)$ for $j \in [m]$. Hence, $w_{\ell_\infty}(\pi) = \max_{i\in[3]} w_{\ell_\infty}(gh_i) = 5$.

Now we turn to unsatisfiable ϕ. Let π be an nonidentity permutation such that $\pi = g^{*z^*}g_1^{z_1}{g_1'}^{z_1'}\cdots g_n^{z_n}{g_n'}^{z_n'} \in G$.

The weight of π depends on $z^*, z_1, z_1', \ldots, z_n, z_n'$. There are three possible cases.

TABLE II
VALUES OF $\chi_\alpha(x)$

x	1	2	3	4
$\chi_\alpha(x)$	1	2	$\alpha+2$	$\alpha+3$
x	5	6	7	8
$\chi_\alpha(x)$	$\alpha+4$	$\alpha+5$	$2\alpha+5$	$2\alpha+6$

1) $z^* = 0$: For $j \in [n+m]$, $\{24j-23,\ldots,24j\}$ can only be permuted by $s_{24(j-1)}(h_1), s_{24(j-1)}(h_2), s_{24(j-1)}(h_3)$ and their products, since π does not use g^*. We have $w_{\ell_\infty}(\pi) \geq \min_{i\in[3]} w_{\ell_\infty}(h_i) = 6$.
2) $z^* = 1$ and $z_i + z_i' \neq 1$ for some $i \in [n]$: Since $z_i, z_i' \in \{0,1\}$, we have $z_i = z_i'$, i.e., π uses either both of g_i and g_i' or none of them. Recall that $s_{24(i-1)}(h_1)s_{24(i-1)}(h_1) = \boldsymbol{e}$. g_i will cancel out the effect of g_i' on the positions in $\{24i-23,\ldots,24i\}$. Hence, π must permute $\{24i-23,\ldots,24i\}$ by $s_{24(i-1)}(g)$ and $w_{\ell_\infty}(\pi) \geq w_{\ell_\infty}(g) = 7$.
3) $z^* = 1$ and $z_i + z_i' = 1$ for every $i \in [n]$: Define an assignment σ for ϕ by setting $\sigma(x_i) = \mathrm{T}$ if and only if $z_i = 1$. Since ϕ is unsatisfiable, there exists $j \in [m]$ such that the jth clause is not satisfied by σ, i.e., all literals in the jth clause are all assigned to the same value. Note that π has a factor $s_{24(n+j-1)}(h_k)$ if and only if the kth literal of the jth clause is assigned true. Recall that $h_1h_2h_3 = \boldsymbol{e}$, we have that π must permute $\{24(n+j)-23,\ldots,24(n+j)\}$ by $s_{24(n+j-1)}(g)$ and $w_{\ell_\infty}(\pi) \geq w_{\ell_\infty}(g) = 7$.

Since $w_{\ell_\infty}(g_1) = 6$, we conclude that G has minimum weight 6 under ℓ_∞-metric when G is generated by $f(\phi)$ for unsatisfiable ϕ. ■

C. Inapproximability of $MINWSPA_{\ell_\infty}$

Theorem 3 actually implies that it is NP-hard to approximate $\mathrm{MINWSPA}_{\ell_\infty}$ within $1.2-\epsilon$ for any constant $\epsilon > 0$. We improve the factor to $2-\epsilon$ by relabeling the elements in the cycles of the building blocks h_1, h_2, h_3, and g. *Relabling* a cycle $(p_1,\ldots,p_k)$ with a function χ is defined as $(\chi(p_1),\chi(p_2),\ldots,\chi(p_k))$. We define $\chi_\alpha : [8] \to [2\alpha+6]$ by Table II.

For every binary string β of length 3, let $\kappa_\beta^{(\alpha)}$ denote the result of relabeling the cycles of κ_β by χ_α. For example

$$\begin{aligned}
\kappa_{010}^{(\alpha)} &= (\chi_\alpha(1),\chi_\alpha(3))(\chi_\alpha(2),\chi_\alpha(4))\\
&\quad(\chi_\alpha(5),\chi_\alpha(7))(\chi_\alpha(6),\chi_\alpha(8))\\
&= (1,\alpha{+}2)(2,\alpha{+}3)(\alpha{+}4,2\alpha{+}5)(\alpha{+}5,2\alpha{+}6)\\
\kappa_{100}^{(\alpha)} &= (1,\alpha{+}4)(2,\alpha{+}5)(\alpha{+}2,2\alpha{+}5)(\alpha{+}3,2\alpha{+}6)\\
\kappa_{110}^{(\alpha)} &= (1,2\alpha{+}5)(2,2\alpha{+}6)(\alpha{+}2,\alpha{+}4)(\alpha{+}3,\alpha{+}5)\\
\kappa_{111}^{(\alpha)} &= (1,2\alpha{+}6)(2,2\alpha{+}5)(\alpha{+}2,\alpha{+}5)(\alpha{+}3,\alpha{+}4).
\end{aligned}$$

It is clear that $K_3^{(\alpha)} = \{\kappa_{b_1b_2b_3}^{(\alpha)} : b_1,b_2,b_3 \in \{0,1\}\}$ is a group, and the weight distribution of $K_3^{(\alpha)}$ is listed in Table III.

Instead of K_3, we use $K_3^{(\alpha)}$ to construct the building blocks h_1, h_2, h_3 and g as follows:

$$\begin{aligned}
h_1 &= \kappa_{010}^{(\alpha)}s_{2\alpha+6}(\kappa_{100}^{(\alpha)})s_{4\alpha+12}(\kappa_{110}^{(\alpha)})\\
h_2 &= \kappa_{100}^{(\alpha)}s_{2\alpha+6}(\kappa_{110}^{(\alpha)})s_{4\alpha+12}(\kappa_{010}^{(\alpha)})\\
h_3 &= \kappa_{110}^{(\alpha)}s_{2\alpha+6}(\kappa_{010}^{(\alpha)})s_{4\alpha+12}(\kappa_{100}^{(\alpha)})\\
g &= \kappa_{111}^{(\alpha)}s_{2\alpha+6}(\kappa_{111}^{(\alpha)})s_{4\alpha+12}(\kappa_{111}^{(\alpha)}).
\end{aligned}$$

TABLE III
WEIGHT DISTRIBUTION OF $K_3^{\langle\alpha\rangle}$

β	000	001	010	011
$w_{\ell_\infty}(\kappa_\beta^{(\alpha)})$	0	1	$\alpha+1$	$\alpha+2$
β	100	101	110	111
$w_{\ell_\infty}(\kappa_\beta^{(\alpha)})$	$\alpha+3$	$\alpha+4$	$2\alpha+4$	$2\alpha+5$

Note that all of h_1, h_2, h_3, and g act on permutations over $\{1,\dots,6\alpha+18\}$ now. Let $\alpha' = 6\alpha+18$. We can construct a generator set $f_\alpha(\phi) = \{g_1, g_1', \dots, g_n, g_n', g^*\}$ from a formula $\phi = c_1 \wedge \cdots \wedge c_m$ over variables $x_1,\dots,x_n$ by setting

$$g_i = s_{\alpha'(i-1)}(h_1) \prod_{(i,j,k)\in P} s_{\alpha'(n+j-1)}(h_k),$$
$$g_i' = s_{\alpha'(i-1)}(h_1) \prod_{(i,j,k)\in Q} s_{\alpha'(n+j-1)}(h_k),$$

for every $i \in [n]$ and

$$g^* = \prod_{j\in[n+m]} s_{\alpha'(j-1)}(g)$$

where P and Q are as defined earlier. Similar to the argument in Section III-B, we have the following theorem.

Theorem 4: For an NAESAT instance ϕ, let G be the group generated by $f_\alpha(\phi)$. If ϕ is satisfiable, then the minimum weight of G is $\alpha+4$ under ℓ_∞-metric, else the minimum weight is $2\alpha+4$.

By combining the NP-completeness of NAESAT and Theorem 4, we obtain an inapproximability result for $\mathrm{MINWSPA}_{\ell_\infty}$.

Corollary 4: For any constant $\epsilon > 0$, there does not exist a polynomial-time $(2-\epsilon)$-approximation algorithm for computing the minimum weight of a subgroup code generated by a set of permutations under ℓ_∞-metric unless $\mathrm{P} = \mathrm{NP}$.

Proof: Assume A is a polynomial-time $(2-\epsilon)$-approximation algorithm. Then, we can construct a polynomial-time algorithm for NAESAT with A.

1) Set $\alpha > \frac{4}{\epsilon}$.
2) For formula ϕ, construct the generator set $f_\alpha(\phi)$ and run $A(f_\alpha(\phi))$.
3) If $A(f_\alpha(\phi))$ outputs a number no more than $(2-\epsilon)(\alpha+4)$, then accept ϕ; reject otherwise.

Let H be the subgroup code generated by $f_\alpha(\phi)$. For a satisfiable formula ϕ, the minimum weight of H is $\alpha+4$ and $A(f_\alpha(\phi)) \le (2-\epsilon)(\alpha+4)$, by assumption. Hence, ϕ would be accepted by the above algorithm. For unsatisfiable ϕ, the minimum weight of H is $2\alpha+4$. Since $\alpha\epsilon > 4$ and an approximation algorithm cannot give an answer less than the minimum solution, we have

$$A(f_\alpha(\phi)) \ge 2\alpha+4 > 2\alpha+8-\alpha\epsilon-4\epsilon = (2-\epsilon)(\alpha+4).$$

This implies that NAESAT is in P if such A exists. Since NAESAT is NP-complete, the claim is true. ■

IV. MAXIMUM WEIGHT PROBLEMS

The results of the minimum weight problems basically follow from Cameron and Wu's work[4]. Instead of using the approach in Section III-A, Cameron and Wu [4] gave another reduction from NAESAT to the maximum weight problem to obtain NP-hardness results for the maximum weight problems under various metrics. However, it is not clear how to obtain inapproxinable results via their reduction.

In this section, we provide a simple reduction from MAX-E3-LIN-2, which does not admit any polynomial-time $(2-\epsilon)$-approximation algorithm unless $\mathrm{P} = \mathrm{NP}$[8], to the maximum Hamming weight problem for binary linear codes. With this reduction, we can apply Fact 2 to obtain inapproximability results for $\mathrm{MAXWSPA}_\delta$ where δ can be ℓ_p-metric for $1 \le p \le \infty$, Hamming, Lee, Cayley, Kendall's tau, or Ulam's distance metrics. First, we give a formal definition of MAX-E3-LIN-2.

Definition 7 (MAX-E3-LIN-2): Given a system L of linear equations over Z_2 with exactly 3 variables in each equation, determine the maximum number of equations which can be satisfied simultaneously.

An instance of MAX-E3-LIN-2 is in the following form:

$$\begin{aligned} x_{a_{1,1}} + x_{a_{1,2}} + x_{a_{1,3}} &= y_1 \\ x_{a_{2,1}} + x_{a_{2,2}} + x_{a_{2,3}} &= y_2 \\ \vdots \qquad\qquad\quad &= \vdots \\ x_{a_{m,1}} + x_{a_{m,2}} + x_{a_{m,3}} &= y_m. \end{aligned}$$

Håstad [8] proved that it is NP-hard to approximate MAX-E3-LIN-2 within $2-\epsilon$ for any constant $\epsilon > 0$ with the following theorem.

Theorem 5 (see [8]): Given an instance L of MAX-E3-LIN-2 of n variables and m equations. For any constant $\epsilon > 0$, it is NP-hard to distinguish the following two conditions.

1) There are at least $(1-\epsilon)m$ equations in L that can be satisfied simultaneously.
2) There are at most $(\frac{1}{2}+\epsilon)m$ equations in L that can be satisfied simultaneously.

We prove the following theorem by a gap-preserving reduction from MAX-E3-LIN-2 to the maximum Hamming weight problem for binary linear codes.

Theorem 6: For any constant $\epsilon > 0$ and any binary linear codes of length $(\frac{3}{2}-\epsilon)m$, it is NP-hard to distinguish the following two conditions.

1) The maximum Hamming distance is at least $(\frac{3}{2}-2\epsilon)m$.
2) The maximum Hamming distance is at most m.

Proof: First construct a basis $\{b^0, b^1, b^2, \dots, b^n\}$ from a system L of MAX-E3-LIN-2 with n variables and m equations

$$\begin{aligned} x_{a_{1,1}} + x_{a_{1,2}} + x_{a_{1,3}} &= y_1 \\ x_{a_{2,1}} + x_{a_{2,2}} + x_{a_{2,3}} &= y_2 \\ \vdots \qquad\qquad\quad &= \vdots \\ x_{a_{m,1}} + x_{a_{m,2}} + x_{a_{m,3}} &= y_m. \end{aligned}$$

The basis is defined as follows:

$$b_j^0 = \begin{cases} 1, & \text{if } m < j \le (3/2-\epsilon)m \\ 1-y_j, & \text{if } j \le m \end{cases}$$

and for $i > 0$

$$b_j^i = \begin{cases} 0, & \text{if } j > m \\ 1, & \text{if } a_{j,1} = i \text{ or } a_{j,2} = i \text{ or } a_{j,3} = i \\ 0, & \text{otherwise.} \end{cases}$$

Let x be a vector that maximizes the number of satisfied equations in L. Set $b = b^0 + \sum_{i:x_i=1} b^i$. If there are at least $(1-\epsilon)m$ equations in L satisfied with x, then b has at least $(\frac{3}{2}-2\epsilon)m$ 1's. This is due to the following.

1) The last $(\frac{1}{2}-\epsilon)m$ bits must be 1's from b^0,
2) If $x_{a_{j,1}} + x_{a_{j,2}} + x_{a_{j,3}} = y_j$, then we have $b_j = 1$.

For the second item, by the definition of b, we have

$$\begin{aligned} b_j &= b_j^0 + \sum_{i:x_i=1} b_j^i = b_j^0 + \sum_{i=1}^{n} b_j^i x_i \\ &= b_j^0 + x_{a_{j,1}} + x_{a_{j,2}} + x_{a_{j,3}} = b_j^0 + y_j = 1. \end{aligned}$$

On the other hand, let b' be the codeword of maximum Hamming weight generated by $\{b^0, b^1, b^2, \ldots, b^n\}$. For convenience, let $P \subseteq \{b^0, \ldots, b^n\}$ be the set of elements that generate b', i.e., b' is the sum of elements in P. Suppose b' has the weight greater than m. Then, $b^0 \in P$, since $b^1, \ldots, b^n$ have 1's only at the first m bits. Define x' by setting $x'_i = 1$ if and only if $b^i \in P - \{b^0\}$. By the definition of the jth bit of b', we have

$$\begin{aligned} b'_j &= b_j^0 + \sum_{b^i \in P} b_j^i = b_j^0 + \sum_{i=1}^{n} b_j^i x'_i \\ &= b_j^0 + x'_{a_{j,1}} + x'_{a_{j,2}} + x'_{a_{j,3}} \\ &= 1 - y_j + x'_{a_{j,1}} + x'_{a_{j,2}} + x'_{a_{j,3}}. \end{aligned}$$

It is easy to verify that the jth equation $x_{a_{j,1}} + x_{a_{j,2}} + x_{a_{j,3}} = y_j$ is satisfied by x' if and only if $b'_j = 1$, so x' satisfies more than $m - (\frac{1}{2}-\epsilon)m = (\frac{1}{2}+\epsilon)m$ equations in L. That is, if there are at most $(\frac{1}{2}+\epsilon)m$ equations in L satisfied simultaneously, then the maximum Hamming weight of the binary linear code generated by $\{b^0, b^1, b^2, \ldots, b^n\}$ is at most m.

The reduction above can be done in polynomial time. This implies we can distinguish the two conditions in Theorem 5 in polynomial time if we can distinguish these two conditions for binary linear codes. Therefore, the theorem holds. ■

By Theorem 6, we have the following corollaries immediately.

Corollary 5: For any constant $\epsilon > 0$, it is NP-hard to approximate the maximum Hamming weight of a binary linear code within $\frac{3}{2} - \epsilon$.

Proof: Suppose there is a polynomial time r-approximation algorithm for this problem. By Theorem 6, $m > (\frac{3}{2} - 2\epsilon')m/r$ for any constant $\epsilon' > 0$; otherwise, it would contradict Theorem 6. By choosing $\epsilon = 2\epsilon'$, we have $r > \frac{3}{2} - \epsilon$. ■

By Fact 2, we have the following results.

Corollary 6: For any constant $\epsilon > 0$, it is NP-hard to approximate the maximum weight of a subgroup permutation code within $\frac{3}{2} - \epsilon$ under Hamming, Lee, Cayley, Kendall's tau and Ulam's distance metrics.

Corollary 7: For any constant $\epsilon > 0$, it is NP-hard to approximate the maximum weight of a subgroup permutation code within $\sqrt[p]{\frac{3}{2}} - \epsilon$ under ℓ_p-metric for $1 \le p < \infty$.

V. Conclusion

We prove that for every $\epsilon > 0$ and $r > 1$, there does not exist polynomial time approximation algorithm for $\mathrm{MINWSPA}_{d_H}$ and $\mathrm{MINWSPA}_{\ell_p}$ for $1 \le p < \infty$ within $2^{\log^{1-\epsilon} n}$ unless $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{\mathrm{polylog}(n)})$ and within r unless $\mathrm{P} = \mathrm{NP}$, respectively. These results also apply to other distance metrics, such as Lee, Cayley, Kendall's tau, and Ulam's distance metrics. These can be further refined by any improvement on the inapproximabilty of the minimum Hamming weight problem for binary linear codes.

For ℓ_∞ metric, we give a clarification of the NP-hardness proof for $\mathrm{MINWSPA}_{\ell_\infty}$ in Cameron and Wu's work [4]. Moreover, we prove that $\mathrm{MINWSPA}_{\ell_\infty}$ does not admit any polynomial-time $(2-\epsilon)$-approximation algorithm for any constant $\epsilon > 0$ unless $\mathrm{P} = \mathrm{NP}$.

We show that it is NP-hard to approximate $\mathrm{MAXWSPA}_{\ell_p}$ within any constant factor less than $\sqrt[p]{\frac{3}{2}}$ for $1 \le p < \infty$. It is also NP-hard to approximate $\mathrm{MAXWSPA}_\delta$ within any constant factor less than 3/2, where δ can be Hamming, Lee, Cayley, Kendall's tau, or Ulam's distance metric.

Appendix

B. Proof of Proposition 1

We prove this by induction on n. For $n = 1$, κ_0 and κ_1 are the only two permutations. Since $w_{\ell_\infty}(\kappa_0) = w_{\ell_\infty}(\boldsymbol{e}) = 0$ and $w_{\ell_\infty}(\kappa_1) = \max\{|1-2|, |2-1|\} = 1$, the proposition is true for $n = 1$. Assume the proposition is true up to $n-1$. There are three cases for n.

1) $b_1 \cdots b_n = 0^{n-1}1$: $w_{\ell_\infty}(\kappa_{0^{n-1}1}) = 1 = (0^{n-1}1)_2$, because $\kappa_{0^{n-1}1}$ only swaps adjacent positions.
2) $b_n = 0$: Note that $s_{-1}(a_2(\kappa_{b_1\cdots b_{n-1}}))$ permutes only odd positions and $a_2(\kappa_{b_1\cdots b_{n-1}})$ permutes even ones only. It is clear that $s_{-1}(a_2(\kappa_{b_1\cdots b_{n-1}}))$ and $a_2(\kappa_{b_1\cdots b_{n-1}})$ permute disjoint positions. By the induction hypothesis, we have $w_{\ell_\infty}(\kappa_{b_1\cdots b_n}) = \max\{w_{\ell_\infty}(a_2(\kappa_{b_1\cdots b_{n-1}})), w_{\ell_\infty}(s_{-1}(a_2(\kappa_{b_1\cdots b_{n-1}})))\} = (b_1\cdots b_{n-1})_2 \times 2 = (b_1\cdots b_n)_2$.
3) $b_n = 1$ and $b_1\cdots b_{n-1} \ne 0^{n-1}$: Let $\kappa = \kappa_{b_1\cdots b_{n-1}}$, i.e., $[\kappa(1), \ldots, \kappa(2^{n-1})]$, and $\kappa' = \kappa_{b_1\cdots b_{n-1}0}$. By definition, we have

$$\kappa' = [2\kappa(1)-1, 2\kappa(1), \ldots, 2\kappa(2^{n-1})-1, 2\kappa(2^{n-1})]$$

$\kappa_{b_1\cdots b_n}(2i-1) = \kappa'(2i) = 2\kappa(i)$ and $\kappa_{b_1\cdots b_n}(2i) = \kappa'(2i-1) = 2\kappa(i) - 1$ for every $i \in [2^{n-1}]$. Let $j = \arg\max_{i\in[2^{n-1}]} |i - \kappa(i)|$. Since $b_1\cdots b_{n-1} \ne 0^{n-1}$ implies $\kappa \ne \boldsymbol{e}$, we have $j - \kappa(j) \ne 0$. If $j < \kappa(j)$, then the

difference between $\boldsymbol{e}$ and $\kappa_{b_1\cdots b_n}$ at the $(2j-1)$th position is $|2j-1-\kappa_{b_1\cdots b_n}(2j-1)| = 2\kappa(j)-2j+1 = 2w_{\ell_\infty}(\kappa)+1$. If $j > \kappa(j)$, then the difference between $\boldsymbol{e}$ and $\kappa_{b_1\cdots b_n}$ at the $2j$th position is $|2j-\kappa_{b_1\cdots b_n}(2j)| = 2j-2\kappa(j)+1 = 2w_{\ell_\infty}(\kappa)+1$. We have $w_{\ell_\infty}(\kappa_{b_1\cdots b_n}) = 2w_{\ell_\infty}(\kappa)+1 = 2\times(b_1\cdots b_{n-1})_2+1 = (b_1\cdots b_n)_2$.

Hence, we conclude that $\kappa_{b_1\cdots b_n} = (b_1\cdots b_n)_2$ by induction.

C. Proof of Proposition 2

We prove it by induction on n. For $n=1$, we have

$$\begin{aligned}\kappa_0\kappa_0 &= \boldsymbol{e}\boldsymbol{e} = \boldsymbol{e} = \kappa_0\\ \kappa_0\kappa_1 &= \boldsymbol{e}(1,2) = (1,2) = \kappa_1\\ \kappa_1\kappa_0 &= (1,2)\boldsymbol{e} = (1,2) = \kappa_1\\ \kappa_1\kappa_1 &= (1,2)(1,2) = \boldsymbol{e} = \kappa_0.\end{aligned}$$

The proposition is true for $n=1$. Assume the proposition is true for $n<k$. For $n=k>1$, let $\kappa = \kappa_{b_1\cdots b_{k-1}}$, $\kappa' = \kappa_{b_1'\cdots b_{k-1}'}$ and $\kappa'' = \kappa_{b_1''\cdots b_{k-1}''}$. By the induction hypothesis, we have $\kappa\kappa' = \kappa''$. There are four cases.

1) $b_k = b_k' = 0$: By definition, we have

$$\begin{aligned}\kappa_{b_1\cdots b_k}\kappa_{b_1'\cdots b_k'} &= a_2(\kappa)s_{-1}(a_2(\kappa))a_2(\kappa')s_{-1}(a_2(\kappa'))\\ &= a_2(\kappa)a_2(\kappa')s_{-1}(a_2(\kappa))s_{-1}(a_2(\kappa'))\\ &= a_2(\kappa'')s_{-1}(a_2(\kappa'')) = \kappa_{b_1''\cdots b_{k-1}''0}\\ &= \kappa_{b_1''\cdots b_k''}.\end{aligned}$$

Since $s_{-1}(a_2(\kappa))$ and $a_2(\kappa')$ permute disjoint positions, we can exchange them. Thus, the second equality holds. The third equality is due to that shift and stretch operations do not change the algebraic structures.

2) $b_k = 0$ and $b_k' = 1$: By applying the result of the first case, we have

$$\begin{aligned}\kappa_{b_1\cdots b_k}\kappa_{b_1'\cdots b_k'} &= \kappa_{b_1\cdots b_{k-1}0}\kappa_{b_1'\cdots b_{k-1}'0}\kappa_{0^{k-1}1}\\ &= \kappa_{b_1''\cdots b_{k-1}''0}\kappa_{0^{k-1}1}\\ &= \kappa_{b_1''\cdots b_k''}.\end{aligned}$$

3) $b_k = 1$ and $b_k' = 0$: Let $\pi = \kappa_{0^{k-1}1}\kappa_{b_1'\cdots b_{k-1}'0}$, i.e., π is to swap $2i-1$ and $2i$ for $i\in[2^{k-1}]$ after applying $\kappa_{b_1'\cdots b_{k-1}'0}$. Observe that

$$\pi = [2\kappa'(1), 2\kappa'(1)-1, \ldots, 2\kappa'(2^{k-1}), 2\kappa'(2^{k-1})-1].$$

In other words, π is also equal to $\kappa_{b_1'\cdots b_{k-1}'1}$. With this fact and the result in the second case, we have

$$\begin{aligned}\kappa_{b_1\cdots b_k}\kappa_{b_1'\cdots b_k'} &= \kappa_{b_1\cdots b_{k-1}0}\kappa_{0^{k-1}1}\kappa_{b_1'\cdots b_{k-1}'0}\\ &= \kappa_{b_1\cdots b_{k-1}0}\kappa_{b_1'\cdots b_{k-1}'1}\\ &= \kappa_{b_1''\cdots b_k''}.\end{aligned}$$

4) $b_k = b_k' = 1$: By definition, we have

$$\begin{aligned}\kappa_{b_1\cdots b_k}\kappa_{b_1'\cdots b_k'} &= \kappa_{b_1\cdots b_{k-1}0}\kappa_{0^{k-1}1}\kappa_{b_1'\cdots b_{k-1}'0}\kappa_{0^{k-1}1}\\ &= \kappa_{b_1\cdots b_{k-1}0}\kappa_{b_1'\cdots b_{k-1}'0}\kappa_{0^{k-1}1}\kappa_{0^{k-1}1}\\ &= \kappa_{b_1''\cdots b_{k-1}''0}\\ &= \kappa_{b_1''\cdots b_k''}.\end{aligned}$$

The second equality is true, since $\kappa_{0^{k-1}1}\kappa_{b_1'\cdots b_{k-1}'0} = \kappa_{b_1'\cdots b_{k-1}'1} = \kappa_{b_1'\cdots b_{k-1}'0}\kappa_{0^{k-1}1}$. The third equation holds, because swapping the same pair twice is equivalent to identity, i.e., $\kappa_{0^{k-1}1}\kappa_{0^{k-1}1} = \boldsymbol{e}$.

It is clear that $\kappa_{b_1\cdots b_k}\kappa_{b_1'\cdots b_k'} = \kappa_{b_1''\cdots b_k''}$ in all cases. The proposition is true.

References

[1] S. Arora, L. Babai, J. Stern, and Z. Sweedyk, "The hardness of approximate optima in lattices, codes, and systems of linear equations," *J. Comput. Syst. Sci.*, vol. 54, pp. 317–331, 1997.

[2] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, "On the inherent intractibility of certain coding problems," *IEEE Trans. Inf. Theory*, vol. IT-24, no. 3, pp. 384–386, May 1978.

[3] C. Buchheim, P. J. Cameron, and T. Wu, "On the subgroup distance problem," *Discrete Math.*, vol. 309, pp. 962–968, 2009.

[4] P. J. Cameron and T. Wu, "The complexity of the weight problem for permutation and matrix groups," *Discrete Math.*, vol. 310, pp. 408–416, 2010.

[5] Q. Cheng and D. Wan, "A deterministic reduction for the gap minimum distance problem," in *Proc. Symp. Theory Comput.*, 2009, pp. 33–38.

[6] I. Dinur, "Approximating SVP_∞ to within almost polynomial factors is NP-hard," *Combinatorica*, vol. 23, pp. 205–243, 2003.

[7] I. Dumer, D. Micciancio, and M. Sudan, "Hardness of approximating the minimum distance of a linear code," *IEEE Trans. Inf. Theory*, vol. 49, no. 1, pp. 22–37, Jan. 2003.

[8] J. Håstad, "Some optimal inapproximability results," *J. ACM*, vol. 48, pp. 798–859, 2001.

[9] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, "Rank modulation for flash memories," in *Proc. IEEE Int. Symp. Inf. Theory*, 2008, pp. 1731–1735.

[10] A. Jiang, M. Schwartz, and J. Bruck, "Error-correcting codes for rank modulation," in *Proc. IEEE Int. Symp. Inf. Theory*, 2008, pp. 1736–1740.

[11] S. Khot, "Hardness of approximating the shortest vector problem in lattices," *J. ACM*, vol. 52, no. 5, pp. 789–808, 2005.

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

[13] T.-T. Lin, S.-C. Tsai, and W.-G. Tzeng, "Efficient encoding and decoding with permutation arrays," in *Proc. IEEE Int. Symp. Inf. Theory*, 2008, pp. 211–214.

[14] C. Papadimitriou, *Computational Complexity*. Reading, MA: Addison-Wesley, 1995.

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

[16] K. W. Shum, "Permutation coding and MFSK modulation for frequency selective channel," in *Proc. IEEE Int. Symp. Pers., Indoor Mobile Radio Commun.*, Sep. 2002, vol. 13, pp. 2063–2066.

[17] C. Sims, "Computational methods in the study of permutation groups," in *Computational Problems in Abstract Algebra*. Pergamon, Oxford, U.K.: , 1970, pp. 169–183.

[18] 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, Jun. 2010.

[19] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Trans. Inf. Theory*, vol. 43, no. 6, pp. 1757–01766, Nov. 1997.

[20] A. J. H. Vinck, "Coded modulation for powerline communications," *Proc. Int. J. Electron. Commun*, vol. 54, pp. 45–49, 2000.

[21] A. J. H. Vinck and J. Häring, "Coding and modulation for power-line communications," presented at the Int. Symp. Power Line Commun., Limerick, Ireland, Apr. 2000.

[22] A. J. H. Vinck, J. Häring, and T. Wadayama, "Coded M-FSK for power line communications," in *Proc. IEEE Int. Symp. Inf. Theory*, 2000, p. 137.

Min-Zheng Shieh received the B.S. and M.S. degrees in Computer Science and Information Engineering and the Ph.D. degree in Computer Science and Engineering, all from National Chiao Tung University, Taiwan, in 2003, 2004 and 2011, respectively.

Since 2012, He has been an assistant research fellow of the Information and Communication Technology Laboratories, National Chiao Tung University. His main research interests include computational complexity, algorithms, coding theory and discrete mathematics.

Shi-Chun Tsai (M'06) received his BS and MS degrees in Computer Science and Information Engineering from National Taiwan University, Taiwan, in 1984 and 1988, respectively; and Ph.D. degree in Computer Science from the University of Chicago, USA, in 1996.

During 1993–1996, he served as a Lecturer in Computer Science Department of the University of Chicago. During 1996–2001, he was an Associate Professor of Information Management Department and Computer Science and Information Engineering Department of National Chi Nan University, Taiwan. In 2001, he joined the Department of Computer Science of National Chiao Tung University, Taiwan. He was promoted to full Professor in 2007. Since 2010, he has served as the Director of Information Technology Service Center, National Chiao Tung University. His research interests include Computational Complexity, Algorithms, Coding theory, Combinatorics and Design of Service Oriented Systems.