Contents

Introduction

Metrics on permutations are often used to address statistical problems associated with partially ranked data. Such problems in their simplest form arise in situations such as the following: suppose there are t persons and a set of n items. Each person ranks the first k items of his choice for $k < n$. We might need to address statistical questions that involves measuring the degree of association between the partial rankings of two persons. Or we might need to figure out if this partial ranking data points to a significant statistical difference between two different subpopulations of rankers.

There are several other applications of metrics on permutation groups in problems concerning partially ranked data. In particular, permutation groups can be used as an error correcting code with respect to some general metric on permutation ([6]). This question leads to several interesting problems about permutation groups.

Some of the basic calculations involving metrics on permutations, while being essential in multiple applications, were shown to be NP-complete. In the simplest form they can be solved by enumerating all elements of a permutation, which os $O(n!)$. To address these, more effective probabilistic algorithms can be applied.

In this paper we give necessary definitions and an overview of the most commonly used metrics on permutations. We also focus on two common problems for metrics on permutations: Subgroup Distance Problem (SDP) and Minimum Weight Problem (MWP). They have multiple applications, in particular in statistics and big data.

For some metrics we refer to existing algorithms with respect to these metrics, for others we describe algorithms that allow to improve the running time. Also, we analyze the complexity classes for the algorithms mentioned with respect to different metrics.

Chapter 1

Definitions and Overview of Problems

1.1 Symmetric Groups and Permutation Groups

The key concept of the paper is the *symmetric group* and the *permutation group.*

Definition 1.1.1. *The symmetric group S_n, or $Sym(X)$ on a finite set X is the group whose elements are all bijective functions from X to X and whose group operation is that of function composition:*

$$S_n : X \leftrightarrow X.$$

In other words, elements of a symmetric group are different permutations of elements of some finite set. There are $n!$ different permutations on a set of n elements, thus the order of a symmetric group S_n is $n!$.

For the purpose of convenience elements of a finite set are often mapped to natural numbers. For that reason it is common to address *symmetric groups of degree n.*

Definition 1.1.2. *The symmetric group of degree n is the symmetric group on the set $X = \{1, 2, 3, \ldots, n\}$.*

Symmetric groups despite having interesting properties usually appear along with their subgroups. A subgroup of a symmetric group G is called a *permutation group.*

Definition 1.1.3. *A permutation group is a group G whose elements are permutations of a given set X and whose group operation is composition of permutations in G.*

The way in which the elements of a permutation group permute the elements of the set is called its *group action*.

All that is necessary for an arbitrary set of permutations on a set X to be a permutation group is to satisfy the group axioms. That is, a subset of a symmetric group is a permutation group if it contains the identity permutation, the inverse permutation of each permutation in it, and is closed under composition of its permutations.

Important properties of a permutation group are its *degree* and *order*.

Definition 1.1.4. *The degree of the permutation group G on a finite set X is the number of elements in the set.*

$$deg(G) = |X|.$$

Definition 1.1.5. *The order of the permutation group G on a finite set X is the number of elements in the group.*

$$ord(G) = |G|.$$

By Lagrange's theorem, the order of a finite permutation group of degree n must divide the order of a corresponding symmetric group $n!$.

Another important characteristic of a permutation group is its *generating set*

Definition 1.1.6. *A subset S of a group G is a generating set for group G if the smallest subgroup of G containing S is the G itself.*

It means that every element of a group G can be expressed a product of elements of its generating set S.

Theorem 1.1.1. *Every finite group G has a generating set of size at most* $\log_2 |G|$.

Proof. If G is a finite group, and $g_1, \ldots, g_m$ is a minimal set of generators, let $G_n = \langle g_1, \ldots, g_n \rangle$. Then by minimality $g_k \neq e \ \forall k$, and $g_{n+1} \notin G_n \ \forall n$. Then G_{n+1} contains at least the two disjoint cosets $eG_n = G_n$ and $g_{n+1}G_n$ of G_n, so $|G_{n+1}| \geq 2|G_n|$. By induction $|G| = |G_m| \geq 2^m$, so $m \leq log_2|G|$. □

As a consequence, every subgroup of S_n has a generating set of size $O(n \log n)$.

Let $[n]$ denote the set $\{1, 2, \ldots, n\}$.

Definition 1.1.7. *Let G be a subgroup contained in S_n. $\forall i \in [n]$, G has the subgroup $G_i = \{g \in G \mid g(i) = i\}$ consisting of all elements of G that fix i. For any subset $I \subseteq [n]$ we define the subgroup that pointwise stabilizes I:*

$$G_I = \{g \in G \mid g(j) = j \ \forall j \in I\}.$$

Let $G^{(i)} = G_{[i-1]}$ for $1 \leq i \leq n$. Then we have a chain of stabilizers

$$G = G^{(0)} \geq G^{(1)} \geq G^{(2)} \geq \cdots \geq G^{(n)} = e.$$

They have the following property:

$$\frac{|G^{(i)}|}{|G^{(i+1)}|} \leq n - i.$$

By Lagrange's theorem $G^{(i)}$ is a disjoint union of at most $n - i$ right cosets of $G^{(i+1)}$. Thus,

$$G^{(i)} = G^{(i+1)}\sigma_1 \cup \cdots \cup G^{(i+1)}\sigma_k, \ k \leq n - i.$$

The permutations σ_j are called coset representatives.

Definition 1.1.8. *A strong generating set for G is a set S of permutations such that for each $i = 1, 2, \ldots, n-1$, $(S \cap G^{(i)}) \setminus G^{(i+1)}$ is a complete set of distinct coset representatives for $G^{(i)}$.*

The *Schreier-Sims* algorithm allows to compute a strong generating set of size $O(n^2)$ for given permutation group G in polynomial time. As a result, the following tasks can be computed in polynomial time as well.

1. Computing order of a permutation group.
2. Testing membership in a given permutation group.
3. Uniformly sampling an element from a given permutation group.
4. Given a permutation group G acting on Ω, computing orbit of any point $k \in \Omega$.
   (Orbit of point k is $\{i \in \Omega \mid \exists g \in G(k) = i\}$.)

1.2 Metrics on Permutations

The following is the classical definition of a metric for the case of symmetric groups.

Definition 1.2.1. *A function $d : S_n \times S_n \longmapsto \mathbb{R}$ is a metric on permutation group S_n if:*

1. $\forall x, y \in S_n\ d(x,y) \geq 0$, *and* $d(x,y) = 0 \iff x = y$ *(non-negativity).*
2. $\forall x, y \in S_n\ d(x,y) = d(y,x)$ *(symmetry).*
3. $\forall x, y, z \in S_n\ d(x,y) \leq d(x,z) + d(z,y)$ *(triangle inequality).*

Let e denote the identity permutation on S_n.

Definition 1.2.2. *For $x \in S_n$ $d(x,e)$ is the norm of x for metric d and is denoted $\|x\|$.*

Definition 1.2.3. *Metric d on S_n is right-invariant if it satisfies $d(x,y) = d(xz,yz)$, and left-invariant, if it satisfies $d(x,y) = d(zx,zy)$ $\forall x, y, z \in S_n$. The metric is bi-invariant, if it is both left- and right-invariant.*

There are numerous metrics on permutations with known applications. Some of the most commonly used are:

1. **Hamming Distance:** $d(x,y) = |\{1 \leq i \leq n | x(i) \neq y(i)\}|$. Hamming distance is simply the number of different bits in two permutations. It is commonly used in algorithms for search problems, where states can be represented as vectors. Hamming metric is often used as a heuristic function in graph search algorithms for a variety of problems, including artificial intelligence. In particular, it is used in cases where Hamming distance satisfies the constraint of admissible heuristic with respect to actual distance.

2. l_p **Distance:** $(p \geq 1)$: $d(x,y) = (\sum_{i=1}^{n} |x(i) - y(i)|^p)^{1/p}$. l_p is a commonly used generalization of Euclidian metric applied to symmetric groups. In case of $i = 1$ it is a sum of absolute distances coordinate-wise, which is

similar to Hamming distance but takes into account the absolute values of the per-coordinate differences. In case of $i = 2$ it is an Euclidian distance if permutations are considered points in n-dimensional space.

3. l_∞ **Distance, or Chebyshev Distance:** $d(x, y) = \max_{1 \leq i \leq n} |x(i) - y(i)|$. l_∞ is the corner case of l_p, when $p \to \infty$.

4. **Cayley Distance:** $d(x, y)$ = minimum number of transpositions taking permutation x to permutation y. Cayley distance is the most 'natural' permutation with respect to operations on permutations, however, it is harder to compute compared to other metrics and involves a couple of known properties of permutations. Given any two permutations, the Cayley distance between them is equivalent to the distance between the composition of the first with the inverse of the second and the identity permutation, i.e. the norm of the composition. The norm is equal to the length of permutation minus the number of cycles in it ([7]).

All given metrics are right invariant, Hamming and Cayley metrics are also left invariant.

We also need to define distance between a permutation and permutation group in a symmetric group.

Definition 1.2.4. *For $S \subseteq S_n$ and $x \in S_n$ the distance between x and S is*

$$d(x, S) = \min_{y \in S} d(x, y).$$

Definition 1.2.5. *Let $x \in S_n$, $r \in \mathbb{R}^+$.*

$$B_n(x, r, d) = \{y \in S_n | d(x, y) \leq r\}$$

is the ball of radius r centered at x for a metric d.

Let the volume $Vol(S)$ of a subset $S \subseteq S_n$ be the number of permutations in the subset $|S|$.

For right invariant metric d, $\forall x \in S_n, r \geq 0\, Vol(B_n(e, r, d)) = Vol(B_n(x, r, d))$.

We are now ready to define the Subgroup Distance Problem and Minimum Weight Problem for a metric d.

Definition 1.2.6. ***Subgroup Distance Problem (SDP):*** *Input instances are* (G, x, k)*, where* $G \subseteq S_n$ *is given by a generating set,* $x \in S_n$*, and* $k \geq 0$*. Is* $d(x, G) \leq k$*?*

Definition 1.2.7. ***Minimum Weight Problem (MWP):*** *Input instances are* (G, k)*, where* $G \subseteq S_n$ *is given by a generating set and* $k \geq 0$*. Is there a* $x \in G \backslash \{e\}$ *with* $\|x\| \leq k$*?*

Approximate solutions to MWP and SDP are also interesting, especially taking into consideration that classical MWP and SDP are NP-complete ([2]). For MWP, given $y > 1$ the problem is to find an element $x \in G, x \neq e$ such that $\|x\|$ is bounded by y times the optimal value. Likewise for SDP. We can define promise decision versions of SDP and MWP that capture this notion of approximation.

For any permutation metric d, the promise problem $GapSDP_y$ where y is a function of n, is defined as follows: inputs are the SDP inputs (G, x, k). An instance (G, x, k) is a YES instance if there $\exists z \in G$ such that $d(z, x) \leq k$. (G, x, k) is a NO instance if $\forall z \in G\ d(z, x) \geq yk$. The problem $GapMWP_y$ is defined in the same way.

We say that an algorithm solves the promise problem if it decides correctly on the YES and NO instances.

Note that if we can compute y-approximate solution for MWP (or SDP), we can solve corresponding promise problem $GapMWP_y$ ($GapSDP_y$ resp.). Suppose we can compute y-approximate solution for MWP, which means we can compute $x \in G$ such that $\|x\| \leq yt$, where t is norm of shortest non-identity permutation in G. To solve an instance (G, k) of $GapMWP_y$ we simply check if $\|x\| > yk$, if so then we have $yt > yk$, thus $t > k$. This implies (G, k) is not a YES instance of $GapMWP_y$. In other case when $\|x\| \leq yk$, which implies (G, k) is not a NO instance of $GapMWP_y$. Similarly if we can compute y approximate solution for SDP we can solve the promise version of SDP.

Our goal is to study the complexity of MWP with respect to Hamming and l_∞ metrics. MWP is known to be NP-hard with respect to both of these metrics even for abelian permutation groups ([1]). A naive brute-force search algorithm for MWP (which enumerates all the permutations and finds a permutation in G with shortest nonzero norm) takes $O(n!)$ steps since $G \subseteq S_n$ can have up to $n!$ elements. It follows that if $G \subseteq S_n$ is an abelian group then $|G| \leq O(2^n)$, so with the classical Schrier-Sims algorithm we can enumerate all permutations in G and

find the one with the smallest nonzero norm. This gives $O(2^n)$ algorithm to solve MWP for abelian groups.

For Hamming metric there is a deterministic $O(2^n)$ time algorithm based on the classical Schrier-Sims algorithm ([1]). However, the problem for l_∞ metric does not appear amenable to a permutation group-theoretic approach. We analyze a $O(2^n)$ time randomized algorithm for the problem. This algorithm adapts some ideas from the Ajtai-Kumar-Sivakumar algorithm for the shortest vector problem for integer lattices. The basic idea of AKS algorithm is to pick a large number of lattice points randomly and perturb them with a certain distribution, then apply the sieving procedure on these perturbed lattice points successively to get shorter lattice points. This algorithm uses similar procedure.

Chapter 2

Minimum Weight Problem

2.1 Minimum Weight Problem Over l_∞ Metric

Consider the search version of MWP: given $G \subseteq S_n$, the goal is to find a permutation $x \in G \setminus \{e\}$ with minimum norm with respect to a metric d. We refer to such a $x \in G$ as a shortest permutation in G with respect to the metric d. It is known that a decision version of MWP is NP-hard for various metrics including l_∞, Cayley and Hamming metrics.

For l_∞ metric there is a known $O(2^n)$ time randomized algorithm for finding a shortest permutation for $G \subseteq S_n$ given by generating set ([1]). The algorithm uses the framework for the shortest vector problem for integer lattices.

The basic idea is to pick N elements of G independently and uniformly at random, $N = 2^{cn}$, where the constant c will be appropriately chosen. Each of these elements is then multiplied by a random permutation of a smaller norm to get a new set of N elements. A sieving procedure is applied several times on this new set of permutations. The main property of the sieving is that after each stage for remaining permutations the maximum norm is halved and in the process at most $2^{\dot{c}n}$ elements are sieved out for a small constant $\dot{c}$.

Repeated sieving reduces the maximum norm until it becomes a constant multiple of norm of shortest permutation of G. In the end, for some x, y from the final set of permutations, xy^{-1} will be a shortest permutation with high probability.

The following theorem sums up the algorithm for MWP over l_∞ metric.

Theorem 2.1.1. *Given a permutation group $G \subseteq S_n$ as input, there is a random-*

ized $2^{O(n)}$ *time algorithm which finds a permutation in* $G \setminus \{e\}$ *with the smallest possible norm with respect to* l_∞ *metric with probability at least* $2^{-\Omega(n)}$.

The theorem and the algorithm of finding a permutation with the smallest norm is detailed in [1]. The algorithm can be used in applications that do not necessarily require exact solution, and significantly improves the running time by avoiding enumeration of the whole group.

2.2 Minimum Weight Problem Over Hamming Metric

In general case the Minimum Weight Problem (MWP) with respect to Hamming metric is NP-hard ([1], [4]). Given a permutation group $G \subseteq S_n$, Maximum Weight Problem is to find $x \in G$ of largest possible norm. In is also known that the decision version of the Maximum Weight Problem with respect to Hamming metric is also NP-hard. We will analyze $O(2^n)$ algorithm to solve both of these weight problems for Hamming metric.

First we give an easy $2^{O(n)}$ time deterministic algorithm to find $x \in G \setminus \{e\}$ with the least possible Hamming norm. It turns out a well-known algorithm from permutation groups can be used for this purpose. Suppose $G \subseteq S_n$ is given by a generator set. The problem is to find a shortest permutation in G for the Hamming metric. For every $S \subseteq [n]$ consider the point-wise stabilizer subgroup $G_S \subseteq G$ defined as $G_S = \{g \in G | \forall i \in S : g(i) = i\}$.

Using the Schrier-Sims algorithm, we can compute a generating set for G_S in polynomial time. Thus, in $O(2^n)$ time we can compute G_S for all $S \subseteq [n]$ and find the largest $t < n$ for which there is $S \subseteq [n]$ such that $|S| = t$ and G_S is a nontrivial subgroup. Clearly, any $x \neq e \in G_S$ is a shortest permutation with respect to Hamming metric.

Maximum Weight Problem: First we consider a special case of Maximum Weight Problem. Given permutation group $G \subseteq S_n$, our goal is to check whether G has a fixed-point free permutation, i.e. a permutation with Hamming norm n, and if G has a fixed-point free permutation output one. Using Inclusion-Exclusion Principle we give a $O(2^n)$ time deterministic algorithm for the search version of the problem.

As before, let G_S be the subgroup of G that point-wise fixes $S \subseteq [n]$. Let $F \subseteq G$ denote the set of fix-point free elements. Clearly, $F \cap G_S \neq \emptyset$ for each nonempty S. Also, $F \cap \bigcup_{S \neq \emptyset} G_S = G$. In $O(2^n)$ time we can compute generating sets for all G_S.

Consider the cosets of $G_{[1]}$ inside G. Clearly if G has a fixed-point free permutation $x, x \notin G_{[1]}$ and x lies on the cosets of $G_{[1]}$. Basic idea of the algorithm is to search a fixed-point free permutation for these cosets individually. We know that for a set of permutations in the same coset of $G_{[1]}$, 1 is mapped to a fixed

element. This enables us to fix images of $1, \ldots, k$ and work with smaller cosets.

So, inductively assume that we have already computed a coset H_{k-1} of $G_{[k-1]}$ in G, where for all $x \in H_{k-1}$ $x(i) = \alpha_i, \alpha_i \in [n], \alpha_i \neq 1 for 1 \leq i \leq k-1$ and H_{k-1} contains a fix-point free permutation if G does.

We now show how to compute a point $\alpha_k \in [n]$ which will fix the coset H_k of $G[k]$ in $2^{O(n)}$ time such that $\forall x \in H_k$ $x(i) = \alpha_i$, $\alpha_i \neq i$ for $i = 1..k$ and H_k contains a one. It is easy to see that by repeating this successively we can find a fix-point free permutation in G.

First, from the orbit of k under action of G we pick a candidate point α_k distinct from $\alpha_1, \ldots, \alpha_{k-1}$ and k. Let $H_k = \{x \in G | x(i) = \alpha_i, 1 \leq i \leq k\}$.

Let $A_i = H_k \cap G_{\{i\}}$ for $i = k+1$ to n. It is clear that H_k contains a fixed point free permutation iff $A_{k+1} \cup A_{k+2} \cup \cdots \cup A_n \subset H_k$ iff $|A_{k+1} \cup A_{k+2} \cup \cdots \cup A_n| < |H_k|$.

So if we can compute $|A_{k+1} \cup A_{k+2} \cup \cdots \cup A_n|$ we can then simply compare with $|H_k|$ and if it is strictly less than $|H_k|$ we know that there is a fixed-point free permutation in H_k. So we have found α_k such that $\forall x \in H_k$, $x(i) = \alpha_i$, $\alpha_i \neq i$ for $i = 1 \ldots k$ and H_k contains a fix-point free element if G contains a one. If $|H_k| = |A_{k+1} \cup \cdots \cup A_n|$, then there is no fixed point free permutation in H_k for the current choice of α_k, so we pick another candidate value for α_k in the orbit of k and proceed similarly.

The question boils down to computing cardinality of $A_{k+1} \cup \cdots \cup A_n$, for which we are going to use inclusion-exclusion principle. By inclusion-exclusion principle we know that

$$|A_{k+1} \cup \cdots \cup A_n| = \sum_{S \subseteq \{k+1, \ldots, n\}} (-1)^{|S|+1} | \cap_{j \in S} A_j |.$$

To compute right hand side of the above equation we need to know $| \cap_{j \in S} A_j |$ for all subsets $S \subseteq \{k+1, \ldots, n\}$. From the definition of A_i we can imply that $| \cap_{j \in S} A_j | = H_k \cap G_S$ for any set $S \subseteq \{k+1, \ldots, n\}$. We can compute a generating set for G_S in polynomial time for any $S \subseteq \{k+1, \ldots, n\}$ using the Schrier-Sims algorithm. Furthermore, it is known the coset intersection problem $H_k \cap G_S$ can also be solved in $2^{O(n)}$. Thus, in time $2^{O(n)}$ we can compute $|\cap_{i \in S} A_i|$ for all subsets $S \subseteq \{k+1, \ldots, n\}$. In $2^{O(n)}$ further steps, by using the Inclusion-Exclusion formula, we can compute $|A_{k+1} \cup \cdots \cup A_n|$.

This gives $2^{O(n)}$ time algorithm to find a fix point free permutation.

The algorithm for Maximum Weight Problem is similar only with some minor changes to the algorithm for computing fixed-point free permutation. Let $G \subseteq S_n$ is a given group and we want to compute $x \in G$ with maximum possible Hamming norm. Consider pointwise stabilizer groups G_S for all $S \subseteq [n]$. For each S we compute a fixed-point free permutation in G_S (if one exists) and output a largest Hamming norm permutation among these. Correctness of the algorithm is almost immediate. We summarize results in this section in the following theorem.

Theorem 2.2.1. *Given a permutation group $G \subseteq S_n$ by a generating set, in $O(2^n)$ time we can find $x \in G \setminus \{e\}$ with the smallest possible norm and $y \in G$ with the largest possible norm with respect to Hamming metric.*

Chapter 3

Complexity Limits

Since $GapSDP_y$ is NP-hard for $y \leq (\log n)^c$, it is natural to explore its complexity for larger gaps. For the GapCVP (Closest Vector Problem) on lattices, Goldreich and Goldwasser have shown a constant round IP protocol for $O(\sqrt{n/\log n})$ gap in the case of l_2 norm ([1]). Consequently, for this gap GapCVP is not NP-hard unless polynomial hierarchy collapses. We adapt similar ideas to the permutation group setting. For the Hamming and Cayley metric we give a constant round IP protocol for the complement problem of $GapSDP_y$ for $y \geq n/\log n$, such that the protocol rejects "YES" instances of $GapSDP_y$ with probability at least $n^{-\log n}$, and always accepts the "NO" instances. Note that there is no specific reason for choosing Hamming or Cayley metrics. Actually the protocol is fairly generic, it needs certain volume bounds on metric balls, right invariance of the metric and uniform sampling procedure from metric balls. So it might work for other metrics as well, we have chosen these metrics only as a representative.

For designing the IP protocols we require uniform random sampling procedures from metric balls for the Hamming and Cayley metrics.

We first consider the Cayley metric. Recall that the Cayley distance between x and y is the least number of transpositions required to take x to y. Let k be the number of cycles in x. Each transposition multiplied to x increases or decreases the number of cycles by 1. Since x is transformed to y with the fewest transpositions if we always multiply by a transposition that increments the number of cycles, we have $d(x, y) = n - k$. Thus, a Cayley metric ball of radius r contains

$x \in S_n$ such that x has at least $n-r$ cycles. The number $c(n,k)$ of permutation in S_n with exactly k cycles is the so called Stirling number of the first kind and it satisfies the recurrence relation $c(n,k) = (n-1)c(n-1,k) + c(n-1,k-1)$. We can compute $c(m,l)$, $0 \leq m \leq n$, $0 \leq l \leq k$ using the recurrence for $c(n,k)$.

Theorem 3.0.1. *Let $S \subseteq S_n$ be the set of permutations with k cycles. Let $N = |S| = c(n,k)$. Then there exists a polynomial in n time computable bijective function $f_{n,k} : [N] \to S$.*

Proof. If $n = k = 1$, clearly such function exists, $f_{1,1}(1)$ is simply defined as identity element of S_1. We use induction on $n+k$. Assume that such functions exist for $n+k \leq t$. Now consider n, k such that $n+k = t+1$. We define the function $f_{n,k}(i)$, for $1 \leq i \leq N$:

1. If $i > (n-1)c(n-1,k)$, let $x = f_{n-1,k-1}(i-(n-1)c(n-1,k))$ and y be obtained by appending a 1-cycle (n) to x. Define $f_{n,k}(i) = y$.

2. If $i \leq (n-1)c(n-1,k)$ then find j such that $(j-1)c(n-1,k) < i \leq jc(n-1,k)$. Let $x = f_{n-1,k}(i-(j-1)c(n-1,k))$, write x as product of disjoint cycles. Let $y \in S_n$ be obtained by inserting n in the jth position of the cyclic decomposition of x. Define $f_{n,k}(i) = y$.

Clearly, $f_{n,k}$ is polynomial time computable. We show $f_{n,k}$ is bijective by induction. Suppose $f_{n-1.k-1}$ and $f_{n-1,k}$ are bijective. Each $y \in S_n$ with k cycles can be uniquely obtained either by inserting element n in cyclic decomposition of a $x \in S_{n-1}$ with k cycles (which can be done in $n-1$ ways) or by attaching a 1-cycle with element n to some $x \in S_{n-1}$ with $k-1$ cycles. It follows that $f_{n,k}$ is bijective. □

To uniformly sample $y \in S_n$ with k cycles, we pick $m \in \{1,2,\ldots,c(n,k)\}$ uniformly at random and let $y = f_{n,k}(m)$.

Theorem 3.0.2. *There is a randomized procedure which runs in time poly(n) and samples from $B_n(e,r,d)$ uniformly, where d denotes Cayley metric.*

Now consider the Hamming Metric. The Hamming ball of radius r contains all $y \in S_n$ such that $y(i) \neq i$ for at most r points. Hence, $Vol(B - n(e,r,d)) = \sum_{i=1}^{r} \binom{n}{i} D_i$, where D_i denotes the number of derangements on i points. We can easily enumerate all i-element subsets of $[n]$. The number D_i of derangements on i points satisfies the recurrence $D_i = (i-1)(D_{i-1} + D_{i-2})$. With similar ideas

as used for sampling for Cayley metric balls we can do uniform random sampling from Hamming metric balls in polynomial time.

Theorem 3.0.3. *For $r > 0$, there exists a randomized procedure which runs in time poly(n) and samples uniformly at random from the Hamming balls of radius r around $e(B_n(e, r, d))$.*

We now describe the simple 2-round IP protocol for the Hamming metric. Let (G, τ, r) be input instance of $GapSDP_y$ for $y \geq n/\log n$, and d is the Hamming metric.

1. Verifier: picks $\sigma \in \{0, 1\}$ $\psi \in G$, $\beta \in B_n(e, yr/2, d)$ uniformly at random. The verifier sends to the prover the permutation $\pi = \beta\psi$ if $\sigma = 0$, and $\pi = \beta\tau\psi$ if $\sigma = 1$.

2. Prover: The prover sends $b = 0$ if $d(\pi, G) < d(\pi, \tau G)$ and $b = 1$ otherwise.

3. Verifier: Accepts if $b = \sigma$.

For the protocol we need polynomial time random sampling from a permutation group which is known. We also need uniform sampling from Hamming metric balls which is given by the previous theorem. To prove correctness of the protocol, we need the following theorem.

Theorem 3.0.4. *If $d(\tau, G) > yr$ then for all $\psi_1, \psi_2 \in G$, $B_n(\psi_1, yr/2, d) \cap B_n(\tau\psi_2, yr/2, d) = \emptyset$.*

Proof. Suppose $\pi \in B_n(\psi_1, yr/2, d) \cap B_n(\tau\psi_2, yr/2, d)$. We have $d(\psi_1, \pi) \leq yr/2$ and $d(\pi, \tau\psi_2) \leq yr/2$. By triangle inequality, $d(\psi_1, \tau\psi_2) \leq yr$. This implies $d(\psi_1{\psi_2}^{-1}, \tau) \leq yr$. But $d(\psi_1{\psi_2}^{-1}) \geq d(\tau, G) > yr$, a contradiction. □

Theorem 3.0.5. *The verifier always accepts if (G, τ, r) is "NO" instance of $GapSDP_y$. Furthermore, the verifier rejects with probability at least $n^{-\log n}$ if G, τ, r is a "YES" instance of $GapSDP_y$.*

For the Cayley metric too a similar IP protocol can be designed. As an immediate consequence we have the following theorem.

Theorem 3.0.6. *For the Hamming and Cayley metrics, $GapSDP_y$ for $y \geq n/\log n$ is not NP-hard unless coNP has constant round interactive protocols with constant error probability with the verifier allowed $O(n^{\log n})$ running time.*

Recall that $GapMWP_y$ is Turing reducible to $GapSDP_y$ for solvable groups and the Turing reduction makes queries with the same gap. Hence, by the above theorem it follows that $GapMWP_y$ with respect to solvable groups for $y \geq n/\log n$ is also unlikely to be NP-hard for Hamming and Cayley metrics.

Conclusion

Our goal was to study the Minimum Weight Problem (MWP) and the Subgroup Distance Problem (SDP) with respect to different metrics on permutation groups. For some we can adapt upper and lower bound results from the analogous problems in case of integer lattices.

We analyzed the algorithmic complexity of MWP with respect to Hamming and l_∞ metrics. It is known that MWP is NP-complete for several natural permutation metrics, including Hamming and l_∞ metric, even if the concerned permutation group is abelian. If the given group is an abelian permutation group then its size is bounded by $O(2^n)$. So both the problems MWP and SDP can be solved in $O(2^n)$ time for abelian permutation groups by enumerating the elements of given group. More non-trivial case of non-abelian permutation groups needs further analysis and no effective procedures are yet known.

We analyzed a $O(2^n)$ time algorithm for MWP in case of Hamming metric. The algorithm is group theoretic and is based on the classical Schrier-Sims algorithm. MWP with respect to l_∞ metric does not appear amenable to a permutation group-theoretic approach.

It is known that SDP is NP-hard and it easily follows that SDP is hard to approximate within a factor of $log^{O(1)}n$ unless $P = NP$ ([1]). On the contrary, SDP for approximation factor more than $n/\log n$ is not NP-hard unless there is an unlikely containment of complexity classes. For several permutation metrics the minimum weight problem is polynomial-time reducible to the subgroup distance problem for solvable permutation groups. These results adapts ideas from the analogous results in the case of integer lattices.

The algorithms analyzed are still not poly-time in the worst case, however they are a noticeable improvement over the brute-force $O(n!)$ time algorithms.

Considering that the problem is NP-complete, it is unlikely that a deterministic algorithm that is guaranteed to run in polynomial time exists. Most statistical applications allow approximate solutions, thus can use the algorithms described.

Contents

[1] V. Arvind, Pushkar S. Soglekar. Algorithmic Problems on Permutation Groups. SOFSEM 2008: Theory and Practice of Computer Science. 34th Conference on Current Threads in Theory and Practice of Computer Science, Novy Smokovec, Slovakia, January 19-25, 2008. Proceedings.

[2] Thaynara Arielly de Lima, Mauricio Ayala-Rincon. Complexity of Cayley Distance and other General Metrics on Permutation Groups.

[3] Christoph Buchheim, Peter J. Cameron, Taoyang Wu. On the subgroup distance problem. Discrete Mathematics, 2009.

[4] Peter J. Cameron, Taoyang Wu. The complexity of the weight problem for permutation groups. Electronic Notes in Discrete Mathematics, 2007.

[5] Viswanath Nagarajan. Approximation Algorithms for Sequencing Problems. Tepper School of Business, Carnegie Mellon University, Pittsburgh, 2009.

[6] Tommaso Schiavinotto, Thomas Stutzle. A Review of Metrics on Permutations for Search Landscape Analysis. Darmstadt University of Technology Computer Science Department, Intellectics Group.

[7] A. Cayley. Note on the theory of permutations. Philosophical Magazine, 1849.