Vol.2011-AL-134 No.12
2011/3/7

Online Prediction over Permutahedron

Shota Yasutake,[†1] Kohei Hatano,[†1] Shuji Kijima,[†1]
Eiji Takimoto[†1] and Masayuki Takeda[†1]

We consider an online prediction problem where the player is supposed to predict a permutation of n fixed objects at each trial. This problem is motivated by a scheduling problem whose objective is to minimize the sum of waiting times of n sequential tasks. We propose an online prediction algorithm which predicts almost as well as the best fixed permutation in hindsight.

1. Introduction

Permutation is one of fundamental concepts in discrete mathematics and computer science. Permutations can naturally represent ranking or allocation of fixed objects. So, they have been applied to ranking in machine learning and information retrieval, recommendation tasks or scheduling tasks.

More formally, a permutation σ over the set $\{1,\ldots,n\}$ of n fixed objects is a bijective function from $\{1,\ldots,n\}$ to $\{1,\ldots,n\}$. Another popular way of representing a permutation σ over the set $\{1,\ldots,n\}$ is to describe it as the n-dimensional vector in $\{1,\ldots,n\}^n$, defined as $\boldsymbol{\sigma} = (\sigma(1),\ldots,\sigma(n))$. For example, $(3,4,2,1)$ is a representation of permutation for $n=4$. Let S_n be the set of all permutations over $\{1,\ldots,n\}$, i.e., $S_n = \{\boldsymbol{\sigma} \in \{1,\ldots,n\}^n |\ \boldsymbol{\sigma}$ is a permutation over $\{1,\ldots,n\}\}$.

We consider the following online prediction problem. For each trial $t = 1,\ldots,T$,

(1) The player predicts a permutation $\boldsymbol{\sigma}_t \in S_n$.
(2) The adversary returns a loss vector $\boldsymbol{\ell}_t \in [0,1]^n$.
(3) The player incur loss $\boldsymbol{\sigma}_t \cdot \boldsymbol{\ell}_t$.

†1 Department of Informatics, Kyushu University.
E-mail:{shouta.yasutake, hatano, kijima, eiji, takeda}@inf.kyushu-u.ac.jp.

The goal of the player is to minimize the regret:

$$\sum_{t=1}^{T} \boldsymbol{\sigma}_t \cdot \boldsymbol{\ell}_t - \min_{\boldsymbol{\sigma} \in S_n} \sum_{t=1}^{T} \boldsymbol{\sigma} \cdot \boldsymbol{\ell}_t.$$

This problem is motivated by the following online job scheduling problem. Suppose that we have n jobs to be processed sequentially. Every day, we determine a schedule represented by a permutation in S_n a priori. Then, at the end of the day, we are given processing time $\ell_i \in [0,1]$ of each job i (assume that the waiting time is normalized up to 1). A typical goal would be to minimize total sum of processing time $\sum_{i=1}^{n} \ell_i$. But, instead, we might want to minimize *the sum of waiting time* over all jobs.

For example, suppose that we process jobs according to a permutation $\boldsymbol{\sigma} = (3,2,1,4)$ and each processing time is given as $\boldsymbol{\ell} = (\ell_1,\ell_2,\ell_3,\ell_4)$. Here, we interpret $\boldsymbol{\sigma}$ so that job i is processed with priority $\boldsymbol{\sigma}(i)$. In other words, jobs with higher priority are processed earlier. So, we process jobs $4,1,2$, and 3 sequentially. Waiting times of jobs $i = 4,1,2,3$ are ℓ_4, $\ell_4+\ell_1$, $\ell_4+\ell_1+\ell_2$, and $\ell_4+\ell_1+\ell_2+\ell_3$, respectively. So, the sum of waiting time is exactly $\boldsymbol{\sigma} \cdot \boldsymbol{\ell}$.

Originally, such job scheduling problem was considered in the offline setting. In the offline setting, we are given n jobs whose each processing times are known a priori. At the same time, we are also given a partial order over n jobs, since some jobs have to be processed earlier than other jobs. Then, the goal of the offline problem is to find a schedule (or permutation) which is consistent with the given partial order, for which the sum of waiting time of jobs are minimized. It is known that this problem is NP-hard in general[5),6)]. However, if the partial order forms a series parallel digraph, the problem is solved in $O(n \log n)$[5)]. Further researches of this problem are studied under generalized settings [7)–9)].

In this paper, we propose a randomized prediction algorithm whose expected regret is at most $O(n^2\sqrt{\log n}\sqrt{T})$. For each trial, our algorithm runs in time $O(n^2)$ using $O(n)$ space. Further, we show that the lower bound of the regret is at least $\Omega(n^2\sqrt{\log n}\sqrt{T})$. Therefore our algorithm is optimal.

There is a previous algorithm, PermELearn[4)] proposed by Helmbold and Warmuth, that is applicable to our problem though the algorithm was developed for a different setting. It can be shown that, PermELearn has the same regret bound of ours, so this algorithm is optimal as well. However, PermELearn needs $O(n^2)$

space and the running time is $\tilde{O}(n^6)$.

2. Preliminaries

For any fixed positive integer n, we denote $[n]$ as the set $\{1,\ldots,n\}$. *Permutahedron* P_n be the set of points $\boldsymbol{p} \in \mathbb{R}^n_+$ satisfying

$$\sum_{i\in S} p_i \le \sum_{i\in S}(n+1-i), \text{ for any } S \subset [n], \text{and}$$
$$\sum_{i=1}^{n} p_i = \frac{n(n+1)}{2}.$$

The *unnormalized relative entropy* $\Delta(\boldsymbol{p},\boldsymbol{q})$ from $\boldsymbol{q} \in \mathbb{R}^n_+$ to $\boldsymbol{p} \in \mathbb{R}^n_+$ is defined as

$$\Delta(\boldsymbol{p},\boldsymbol{q}) = \sum_{i=1}^{n} p_i \ln \frac{p_i}{q_i} + \sum_{i=1}^{n} q_i - \sum_{i=1}^{n} p_i.$$

It is known that $\Delta(\boldsymbol{p},\boldsymbol{q}) \ge 0$ and $\Delta(\boldsymbol{p},\boldsymbol{q}) = 0$ if and only if $\boldsymbol{p} = \boldsymbol{q}$. Unnormalized relative entropy is not symmetric in general, i.e., $\Delta(\boldsymbol{p},\boldsymbol{q}) \neq \Delta(\boldsymbol{q},\boldsymbol{p})$ for some $\boldsymbol{p},\boldsymbol{q} \in \mathbb{R}^n_+$. Also, Unnormalized relative entropy is a special case of Bregman divergence[1),3)], which generalizes Euclid distance or natural distance measures.

We will use a geometric property of Bregman divergence which is known as Generalized Pythagorean Theorem. We show a version of the theorem adapted for unnormalized relative entropy.

Theorem 1 (Bregman[1),3)]). *Let $C \subset \mathbb{R}^n_+$ be any convex set. Let $\boldsymbol{q}$ be any point in $\mathbb{R}^n_+$*

$$\boldsymbol{p} = \inf_{\boldsymbol{p}'\in C} \Delta(\boldsymbol{p}',\boldsymbol{q}).$$

Then, it holds for any $\boldsymbol{r} \in C$ that

$$\Delta(\boldsymbol{r},\boldsymbol{q}) \ge \Delta(\boldsymbol{r},\boldsymbol{p}) + \Delta(\boldsymbol{p},\boldsymbol{q}).$$

Further, this inequality becomes an equality if C is an affine set.

3. Algorithm

In this section, we propose our algorithm PermutahedLearn and prove its regret bound.

3.1 Main Structure

The main structure of PermutahedLearn is shown in Fig. 1. The algorithm maintains a weight vector $\boldsymbol{p}_t$ in $\mathbb{R}^n_+$, which represents a mixture of permutations in S_n. At each trial t, it decomposes $\boldsymbol{p}_t$ into permutations, chooses a permutation $\boldsymbol{\sigma}_t$ randomly according to its coefficient, and predicts the permutation $\boldsymbol{\sigma}_t$. After the loss $\boldsymbol{\ell}_t$ is assigned, PermutahedLearn updates the weight vector $\boldsymbol{p}_t$ in a multiplicative way and projects it onto the permutahedron P_n.

Fig. 1 PermutahedLearn

PermutahedLearn

(1) Let $\boldsymbol{p}_1 = (\frac{n+1}{2},\ldots,\frac{n+1}{2}) \in [0,n]^n$.
(2) For $t = 1,\ldots,T$
  (a) Run **Decomposition**$(\boldsymbol{p}_t)$ and decompose $\boldsymbol{p}_t$ as
  $$\boldsymbol{p}_t = \sum_{i=1}^{k} \lambda_i \boldsymbol{\sigma}^{(i)},$$
  where $k \le n$, each $\lambda_i \ge 0$, $\sum_i \lambda_i = 1$, and each $\boldsymbol{\sigma}^{(i)}$ is in S_n.
  (b) Choose $\boldsymbol{\sigma}_t$ randomly from $\{\boldsymbol{\sigma}^{(1)},\ldots,\boldsymbol{\sigma}^{(k)}\}$ according to the distribution $\boldsymbol{\lambda}$.
  (c) Incur a loss $\boldsymbol{\sigma}_t \cdot \boldsymbol{\ell}_t$.
  (d) Update $\boldsymbol{p}_{t+\frac{1}{2}}$ as
  $$p_{t+\frac{1}{2},i} = \frac{p_{t,i}e^{-\eta\ell_{t,i}}}{\sum_{j=1}^{n} p_{t,j}e^{-\eta\ell_{t,j}}} \cdot \frac{n(n+1)}{2}.$$
  (e) Run **Projection**$(\boldsymbol{p}_{t+\frac{1}{2}})$ and get $\boldsymbol{p}_{t+1}$, the projection of $\boldsymbol{p}_{t+\frac{1}{2}}$ onto the permutahedron P_n. That is,
  $$\boldsymbol{p}_{t+1} = \arg\inf_{p\in P_n} \Delta(\boldsymbol{p},\boldsymbol{p}_{t+\frac{1}{2}}).$$

The main structure of our algorithm itself is built on a standard technique in online learning literature (see, e.g.,[4)]). Yet, our technical contribution is to develop efficient projection and decomposition techniques specifically designed for the permutahedron.

We begin our analysis of PermutahedLearn with the following lemma.

Lemma 1. *For any $\boldsymbol{q} \in P_n$ and for any $t \ge 1$,*

$$\Delta(\boldsymbol{q},\boldsymbol{p}_t) - \Delta(\boldsymbol{q},\boldsymbol{p}_{t+1}) \ge -\eta\boldsymbol{q}\cdot\boldsymbol{\ell}_t + (1-e^{-\eta})\boldsymbol{p}_t\cdot\boldsymbol{\ell}.$$

Proof. By using Generalized Pythagorean Theorem,

$$\Delta(\boldsymbol{q}, \boldsymbol{p}_{t+\frac{1}{2}}) \geq \Delta(\boldsymbol{q}, \boldsymbol{p}_{t+1}) + \Delta(\boldsymbol{p}_{t+1}, \boldsymbol{p}_{t+\frac{1}{2}}).$$

Since unnormalized relative entropy is non-negative,

$$\Delta(\boldsymbol{q}, \boldsymbol{p}_{t+\frac{1}{2}}) \geq \Delta(\boldsymbol{q}, \boldsymbol{p}_{t+1}). \tag{1}$$

So, by using inequality (1),

$$\Delta(\boldsymbol{q}, \boldsymbol{p}_t) - \Delta(\boldsymbol{q}, \boldsymbol{p}_{t+1}) \geq \Delta(\boldsymbol{q}, \boldsymbol{p}_t) - \Delta(\boldsymbol{q}, \boldsymbol{p}_{t+\frac{1}{2}}). \tag{2}$$

Then, by using the fact that $\sum_i p_{t+\frac{1}{2},i} = \sum_i p_{t+1,i} = n(n+1)/2$, the right hand side of inequality (2) is

$$\begin{aligned}
\Delta(\boldsymbol{q}, \boldsymbol{p}_t) - \Delta(\boldsymbol{q}, \boldsymbol{p}_{t+\frac{1}{2}}) &= \sum_i q_i \ln \frac{p_{t+1,i}}{p_{t,i}} \\
&= \sum_i q_i \ln \frac{e^{-\eta \ell_{t,i}}}{\sum_i p_{t,i} e^{-\eta \ell_{t,i}} / \frac{n(n+1)}{2}} \\
&= -\eta \boldsymbol{q} \cdot \boldsymbol{\ell}_t - \frac{n(n+1)}{2} \ln \sum_i p_{t,i} / \frac{n(n+1) 2^{-\eta \ell_{t,i}}}{e}. \qquad (3)
\end{aligned}$$

By convexity of $f(x) = e^{-\eta x}$, we have $e^{-\eta \ell_{t,i}} \leq 1 - (1 - e^{-\eta}) \ell_{t,i}$. This implies that

$$\begin{aligned}
\ln \sum_i \frac{p_{t,i}}{n(n+1)/2} e^{-\eta \ell_{t,i}} &\leq \ln \left(1 - \boldsymbol{p}_t \cdot \boldsymbol{\ell}_t / \frac{n(n+1)}{2} \right) \\
&\leq -\boldsymbol{p}_t \cdot \boldsymbol{\ell}_t / \frac{n(n+1)}{2}, \qquad (4)
\end{aligned}$$

where the last inequality follows from the fact that $1 - x \leq e^{-x}$. Finally, by combining (3) and (4), we complete the proof. □

Then we prove a cumulative regret bound of weight vectors $\boldsymbol{p}_t$.

Lemma 2. *For any $T \geq 1$ it holds that*

$$\sum_{t=1}^{T} \boldsymbol{p}_t \cdot \boldsymbol{\ell}_t \leq \frac{\eta \inf_{\boldsymbol{p} \in P_n} \sum_{t=1}^{T} \boldsymbol{p} \cdot \boldsymbol{\ell}_t + \ln n}{1 - e^{-\eta}}.$$

Proof. By summing up the inequality in Lemma 1 for $t = 1, \ldots, T$, for any $\boldsymbol{q} \in P_n$, we get

$$\begin{aligned}
(1 - e^{-\eta}) \sum_{t=1}^{T} \boldsymbol{p}_t \cdot \boldsymbol{\ell}_t &\leq \eta \sum_{t=1}^{T} \boldsymbol{q} \cdot \boldsymbol{\ell}_t + \Delta(\boldsymbol{q}, \boldsymbol{p}_1) - \Delta(\boldsymbol{q}, \boldsymbol{p}_{T+1}) \\
&\leq \eta \sum_{t=1}^{T} \boldsymbol{q} \cdot \boldsymbol{\ell}_t + \Delta(\boldsymbol{q}, \boldsymbol{p}_1), \qquad (5)
\end{aligned}$$

where the last inequality holds since $\Delta(\boldsymbol{q}, \boldsymbol{p}_{T+1})$ is non-negative. Note that, by setting $N = n(n+1)/2$, we have

$$\begin{aligned}
\Delta(\boldsymbol{q}, \boldsymbol{p}_1) &= N \sum_{i=1}^{n} \frac{q_i}{N} \ln \frac{q_i / N}{p_{1,i} / N} \\
&\leq N \ln n.
\end{aligned}$$

Then, by rearranging the inequality (5), we prove the inequality as claimed. □

To complete our analysis for our algorithm, we specify the subroutines Projection and Decomposition, respectively, in the following subsections.

3.2 Projection

We propose an efficient algorithm Projection for computing the projection onto the permutahedron P_n. Formally, the problem is stated as follows:

$$\begin{aligned}
&\inf_{\boldsymbol{p}} \Delta(\boldsymbol{p}, \boldsymbol{q}) \\
&\text{sub. to} \\
&\sum_{j \in S} p_j \leq \sum_{j \in S} (n + 1 - j), \text{ for any } S \subset [n], \\
&\sum_{j=1}^{n} p_j = \frac{n(n+1)}{2}. \qquad (6)
\end{aligned}$$

Here we omit the positivity constraints $\boldsymbol{p} \geq \boldsymbol{0}$ since relative entropy projection always preserves positivity.

Apparently, this problem does not seem to be tractable as it has exponentially many constraints. But, we show that relevant constraints are only linearly many.

For simplicity, we assume that elements in $\boldsymbol{q}$ are sorted in descending order, i.e., $q_1 \geq q_2 \geq \cdots \geq q_n$. This can be achieved in time $O(n \log n)$ by sorting $\boldsymbol{q}$. First, we show that, by this projection, the order in $\boldsymbol{q}$ is preserved.

Lemma 3. *Let $\boldsymbol{p}^*$ be the projection of $\boldsymbol{q}$. Then we have $p_1 \geq p_2 \geq \ldots p_n$.*

Proof. Assume that the claim is false. Then, there exists $i \leq j$ such that $p_i^* < p_j$ and $q_i \geq q_j$. Let $\boldsymbol{r}$ be the vector obtained by exchanging p_i^* and p_j^* in $\boldsymbol{p}^*$. Then,

$$\begin{aligned}
\Delta(\boldsymbol{p}^*, \boldsymbol{q}) - \Delta(\boldsymbol{r}, \boldsymbol{q}) &= p_i^* \ln \frac{p_i^*}{q_i} + p_j^* \ln \frac{p_j^*}{q_j} - p_j^* \ln \frac{p_j^*}{q_i} - p_i^* \ln \frac{p_i^*}{q_j} \\
&= p_i^* \ln \frac{q_j}{q_i} + p_j^* \ln \frac{q_i}{q_j} = (p_j^* - p_i^*) \ln \frac{q_i}{q_j} \geq 0,
\end{aligned}$$

where, the last inequality holds since $p_j^* \geq p_i^*$ and $q_i \geq q_j$. This contradicts the

assumption that $\boldsymbol{p}^*$ is the projection. □

By Lemma 3, observe that once the conditions

$$\sum_{j=1}^{i} p_j \le \sum_{j=1}^{i}(n+1-j),\ j=1,\dots,n-1$$

are satisfied, other inequality constraints are satisfied as well since for any $S \subset [n]$ such that $|S| = i$,

$$\sum_{j\in S} p_j \le \sum_{j=1}^{i} p_i.$$

Therefore, the problem (6) is reduced to the following one.

$$\begin{aligned}
&\inf_{\boldsymbol{p}} \Delta(\boldsymbol{p},\boldsymbol{q})\\
&\text{sub. to}\\
&\sum_{j=1}^{i} p_j \le \sum_{j}^{i}(n+1-j),\ \text{for } i=1,\dots,n-1,\\
&\sum_{j=1}^{n} p_j = \frac{n(n+1)}{2}. \qquad (7)
\end{aligned}$$

The KKT conditions imply that $\boldsymbol{p}^*$ is the projection if and only if $\boldsymbol{p}$ satisfies the following conditions.

$$\begin{aligned}
&p_i^* = \frac{q_i \exp\left(-\sum_{j=i}^{n-1}\alpha_j\right)}{Z},\ (i=1,\dots,n-1)\\
&p_n^* = \frac{q_n}{Z},\\
&\sum_{j=1}^{i} p_j^* \le \sum_{j=1}^{i}(n+1-j),\ i=1,\dots,n-1\\
&\sum_{i=1}^{n} p_i = \frac{n(n+1)}{2},\\
&\alpha_i\Big(\sum_{j=1}^{i} p_j - \sum_{j=1}^{i}(n+1-j)\Big) = 0,\ \text{for } i=1,\dots,n-1,\\
&\alpha_i \ge 0,\ \text{for } i=1,\dots,n, \qquad (8)\\
& \qquad (9)
\end{aligned}$$

where Z is the normalization constant so that $\sum_i p_i = n(n+1)/2$.

Fig. 2 Projection

Projection

Input: $\boldsymbol{q} \in \mathbb{R}^n_+$ satisfying that $q_1 \ge q_2 \ge \cdot \ge q_n$.

Output: projection $\boldsymbol{p}$ of $\boldsymbol{q}$ onto P_n.

(1) Let $i_0 = 0$.

(2) **For** $t = 1,\dots,$

(a) Let

$$C_i^t = \frac{\sum_{j=1}^{i}(n+1-j) - \sum_{j=1}^{i_{t-1}} p_j}{\sum_{j=i_{t-1}+1}^{i} q_j}$$

and

$$i_t = \arg\min_{i:i_{t-1}<i\le n} C_i^t.$$

If there are multiple minimizers, choose the largest one as i_t.

(b) Set $p_{i_{t-1}+1} = q_{i_{t-1}+1}C_{i_t}^t, \dots, p_{i_t} = q_{i_t}C_{i_t}^t$.

(c) **If** $i_t = n$, **then** break.

(3) **Output** $\boldsymbol{p}$.

Now we describe the detail of the projection algorithm in Fig. 2.

Lemma 4. *(1) Given $\boldsymbol{q}$, the algorithm Projection outputs the projection of $\boldsymbol{q}$ onto the permutahedron P_n.*

(2) The time complexity of Projection is $O(n^2)$.

Proof. We show that there exists $\alpha_1,\dots,\alpha_{n-1}$ and Z such that the output $\boldsymbol{p}$ satisfies the optimality conditions (9), which completes the proof of the first statement.

First of all, we show that $C_{i_{t-1}}^{t-1} \le C_{i_t}^t$ for each iteration t. Because of the definition of $C_{i_{t-1}}^{t-1}$, we have $C_{i_{t-1}}^{t-1} < C_{i_t}^{t-1}$. So, it suffices to prove that $C_{i_t}^{t-1} < C_{i_t}^t$. To see this, observe that

$$\sum_{i=1}^{i_{t-2}} p_j + C_{i_t}^{t-1} \sum_{j=i_{t-2}+1}^{i_t} q_j = \sum_{j=1}^{i_t}(n+1-j),$$

and

$$\begin{aligned}\sum_{j=1}^{i_t}(n+1-j) &= \sum_{i=1}^{i_{t-1}} p_j + C_{i_t}^t \sum_{j=i_{t-1}+1}^{i_t} q_j \\ &= \sum_{i=1}^{i_{t-2}} p_j + C_{i_{t-1}}^{t-1} \sum_{j=i_{t-2}+1}^{i_{t-1}} q_j + C_{i_t}^t \sum_{j=i_{t-1}+1}^{i_t} q_j \\ &< \sum_{i=1}^{i_{t-2}} p_j + C_{i_t}^{t-1} \sum_{j=i_{t-2}+1}^{i_{t-1}} q_j + C_{i_t}^t \sum_{j=i_{t-1}+1}^{i_t} q_j,\end{aligned}$$

where the last inequality holds since $C_{i_{t-1}}^{t-1} < C_{i_t}^{t-1}$.

By rearranging the inequalities above, we have

$$C_{i_t}^{t-1} \sum_{j=i_{t-1}+1}^{i_t} q_j < C_{i_t}^t \sum_{j=i_{t-1}+1}^{i_t} q_j,$$

which implies $C_{i_t}^{t-1} < C_{i_t}^t$.

Now we fix each α_{i_t} so that $e^{-\alpha_{i_t}} C_{i_{t+1}}^{t+1} = C_{i_t}^t$, i.e., $\alpha_{i_t} = \ln(C_{i_{t+1}}^{t+1}/C_{i_t}^t)$ and fix Z to be $Z = C_n^T$, where T satisfies $i_T = n$. Note that since $C_{i_{t+1}}^{t+1} > C_{i_t}^t$), each α_{i_t} is strictly positive. For other $i \notin \{i_1, \ldots, i_T\}$, we set $\alpha_i = 0$. Then, each p_{i_t} can be expressed as

$$\begin{aligned}p_{i_t} &= q_{i_t} C_{i_t}^t \\ &= q_i \exp\left(-\alpha_{i_t} - \alpha_{i_{t+1}} - \cdots - \alpha_{i_T}\right)/Z \\ &= q_i \exp\left(-\alpha_{i_t} - \alpha_{i_t+1} - \cdots - \alpha_{n-1}\right)/Z.\end{aligned}$$

Similarly, for other i such that $i_{t-1} < i < i_t$, we have

$$\begin{aligned}p_i &= q_i C_{i_t}^t \\ &= q_i \exp\left(-\alpha_{i_t} - \alpha_{i_t+1} - \cdots - \alpha_{n-1}\right)/Z \\ &= q_i \exp\left(-\alpha_i - \alpha_{i+1} - \cdots - \alpha_{n-1}\right)/Z.\end{aligned}$$

To see if the specified α_is and Z satisfies the optimality conditions (9), observe that (i) for each i_t,

$$\sum_{j=1}^{i_t} p_j = \sum_{j=1}^{i_{t-1}} p_j + \sum_{j=i_{t-1}+1}^{i_t} q_j C_{i_t}^t = \sum_j (n+1-j)$$

and $\alpha_{i_t} > 0$, and (ii) for each i such that $i_{t-1} < i < i_t$,

$$\sum_{j=1}^{i} p_j = \sum_{j=1}^{i_{t-1}} p_j + \sum_{j=i_{t-1}+1}^{i_t} q_j C_{i_t}^t \quad \le \sum_{j=1}^{i_{t-1}} p_j + \sum_{j=i_{t-1}+1}^{i_t} q_j C_i^t = \sum_j (n+1-j)$$

and $\alpha_i = 0$.

Finally, the algorithm terminates in time $O(n^2)$ since the number of iteration is at most n and each iteration takes $O(n)$ time, which completes the second statement of the lemma. □

3.3 Decomposition

In this subsection, we describe how to represent a point $\boldsymbol{p} \in P_n$ by a convex combination of permutations. For simplicity assume that $p_1 \ge \cdots \ge p_n$. To begin with, we define special points in the permutahedron, which we call *permutations with ties*. Suppose $\boldsymbol{q} \in P_n$ satisfies that $q_1 \ge q_2 \ge \cdots \ge q_n$. A permutation with ties $\boldsymbol{q} \in P_n$ satisfies that if $q_i > q_{i+1}$ then $\sum_{j=1}^{i} q_j = \sum_{j=1}^{i}(n+1-i)$ hold for any $i \in [n]$. For example, if $\boldsymbol{q} \in P_5$ satisfies $q_1 = q_2 > q_3 = q_4 = q_5$, then $\boldsymbol{q}$ is uniquely determined as $\boldsymbol{q} = (4.5, 4.5, 2, 2, 2)$. Note that every permutation with ties $\boldsymbol{q}$ satisfying that $q_1 \ge q_2 \ge \cdots \ge q_n$ is represented by a convex combination of (at most) two permutations, namely $(\boldsymbol{\sigma} + \boldsymbol{\sigma'})/2$ where $\sigma = (n, n-1, \ldots, 1)$ and $\boldsymbol{\sigma'}$ is a "partially reversed" permutation satisfying that

$$\begin{aligned}\sigma'(i) &> \sigma'(j) \quad \text{if} \quad q_i > q_j \text{ and,} \\ \sigma'(i) &< \sigma'(i+1) \quad \text{if} \quad q_i = q_{i+1}.\end{aligned}$$

Note that $\boldsymbol{\sigma'}$ is uniquely determined by $\boldsymbol{q} \in R$ and $\boldsymbol{\sigma}$. For example, let $\boldsymbol{r} = (4.5, 4.5, 2, 2, 2)$ and $\boldsymbol{\sigma} = (5, 4, 3, 2, 1)$, then its partially reversed permutation $\boldsymbol{\sigma'}$ is $(4, 5, 1, 2, 3)$.

Now we describe our algorithm to represent $p \in P_n$ with a convex combination of permutations. Note that $\boldsymbol{\sigma}^1 = \boldsymbol{\sigma}^0$ holds if the input $\boldsymbol{p}$ satisfies that $p_i > p_{i+1}$ for any $i \in [n-1]$.

We will prove the following lemma on Decomposition.

Lemma 5. *Decomposition provides a convex combination of permutations representing an arbitrarily given $p \in P_n$. Its running time is $O(n^2)$.*

To show Lemma 5, we show the following Lemmas.

Lemma 6. *At any iteration t in Decomposition, $\boldsymbol{p}^t$ satisfies that $p_i^t \ge p_{i+1}^t$ for any $i \in [n-1]$.*

Proof. We give an inductive proof with respect to t. In case of $t = 1$, it is clear. In case of $t > 1$, we assume $p_i^{t-1} \ge p_{i+1}^{t-1}$ holds for any $i \in [n-1]$. If $p_i^{t-1} = p_{i+1}^{t-1}$, then $q_i^{t-1} = q_{i+1}^{t-1}$ holds, from the definition of $\boldsymbol{q}^{t-1}$. Thus

Fig. 3 Decomposition

Decomposition
Input: $\boldsymbol{p} \in P_n$ satisfying that $p_1 \geq p_2 \geq \cdots \geq p_n$.
Output: Permutations $\boldsymbol{\sigma}^0, \ldots, \boldsymbol{\sigma}^T$ and $\lambda_0, \ldots, \lambda_T \in \mathbb{R}_{>0}$ s.t. $\sum_{i=0}^{T} \lambda_i \boldsymbol{\sigma}^i = \boldsymbol{p}$, $\sum_{i=0}^{T} \lambda_i = 1$.
(1) Let $\boldsymbol{\sigma}^0 = (n, n-1, \ldots, 1)$, $\boldsymbol{p}^1 = \boldsymbol{p}$ and $\lambda = 1$.
(2) **For** $t = 1, \ldots,$
  (a) Find a partially reversed permutation $\boldsymbol{\sigma}^t$ with respect to $\boldsymbol{p}^t$ and $\boldsymbol{\sigma}^0$. Let $\boldsymbol{q}^t = (\boldsymbol{\sigma}^0 + \boldsymbol{\sigma}^t)/2$.
  (b) Let
$$\lambda_t = \min_{i \in [n]} \left\{ \frac{p_i^t - p_{i+1}^t}{q_i^t - q_{i+1}^t} \mid q_i^t \neq q_{i+1}^t \right\},$$
  where we define $p_{n+1}^t = q_{n+1}^t = 0$, for convenience.
  (c) Let $\boldsymbol{p}^{t+1} = \boldsymbol{p}^t - \lambda_t \boldsymbol{q}^t$ and let $\lambda = \lambda - \lambda_t$.
  (d) **If** λ=0 **then** let $T = t$ and break.
(3) Set $\lambda_0 = 1/2$ and $\lambda_t = \lambda_t/2$ for $t \in [T]$.
  Output permutations $\boldsymbol{\sigma}^0, \ldots \boldsymbol{\sigma}^t$ and $\lambda_0, \ldots, \lambda_T$.

$$p_{\sigma(i)}^t = p_i^{t-1} - \lambda_{t-1} q_i^{t-1} = p_{i+1}^{t-1} - \lambda_{t-1} q_{i+1}^{t-1} = p_{i+1}^t$$

and we obtain the claim. If $p_i^{t-1} > p_{i+1}^{t-1}$, then $q_i^{t-1} > q_{i+1}^{t-1}$ holds, and

$$\begin{aligned} p_i^{t+1} - p_{i+1}^{t+1} &= p_i^t - \lambda_t q_i^t - (p_{i+1}^t - \lambda_t q_{i+1}^t) \\ &= p_i^t - p_{i+1}^t - \lambda_t (q_i^t - q_{i+1}^t) \\ &= (q_i^t - q_{i+1}^t) \left(\frac{p_i^t - p_{i+1}^t}{q_i^t - q_{i+1}^t} - \lambda_t \right) \geq 0 \end{aligned}$$

where the last inequality becomes from the definition of

$$\lambda_t = \min_{i \in [n]} \left\{ \left(p_{i+1}^t - p_i^t\right) / \left(q_{i+1}^t - q_i^t\right) \mid q_{i+1}^t \neq q_i^t \right\}.$$

□

Lemma 7. *In Decomposition,* $\boldsymbol{p}^{T+1}$ $(= \boldsymbol{p}^T - \lambda^T \boldsymbol{q}^T) = 0$ *holds.*

Proof. Without loss of generality, we may assume that $p_1 \geq p_2 \geq \cdots \geq p_n$, for simplicity of notations. First we show $\boldsymbol{p}^{T+1} \geq 0$. Since Lemma 6, if there exists $j \in [n]$ satisfying that $p_j^{T+1} < 0$, then $p_n^{T+1} < 0$ holds. Thus it is enough to show $p_n^{T+1} \geq 0$. Let $i^* = \min\{j \in [n] \mid p_j^T = p_n^T\}$. Then we have $p_{i^*}^T = p_{i^*+1}^T = \cdots = p_n^T$ and $q_{i^*}^T = q_{i^*+1}^T = \cdots = q_n^T$. Hence, we get $p_{i^*}^{T+1} = p_{i^*+1}^{T+1} = \cdots = p_n^{T+1}$. In case of $i^* \geq 2$, $p_{i^*-1}^t > p_{i^*}^t$ holds for any $t \in [T]$, meaning that $q_{i^*-1}^t > q_{i^*}^t$ holds for any $t \in [T]$. Thus we can see that $\sum_{j=i^*}^{n} q_j^t = \sum_{j=i^*}^{n} (n+1-j)$ holds for any $t \in [T]$, from the definition of $\boldsymbol{q}^t$. Then we obtain

$$\begin{aligned} \sum_{j=i^*}^{n} \sum_{t=1}^{T} \lambda_t q_j^t &= \sum_{t=1}^{T} \lambda_t \sum_{j=i^*}^{n} q_j^t = \sum_{t=1}^{T} \lambda_t \sum_{j=i^*}^{n} (n+1-j) \\ &= \sum_{j=i^*}^{n} (n+1-j) \leq \sum_{j=i^*}^{n} p_j \end{aligned}$$

where the last inequality is due to constraints of the permutahedron $\sum_{j=1}^{i^*-1} p_j \leq \sum_{j=1}^{i^*-1} (n+1-j)$ and $\sum_{j=1}^{n} p_j = \sum_{j=1}^{n} (n+1-j)$. Thus we obtain that

$$\sum_{j=i^*}^{n} p_j^{T+1} = \sum_{j=i^*}^{n} \left(p_j - \sum_{t=1}^{T} \lambda_t q_j^t \right) \geq 0.$$

As discussed above, $p_{i^*}^{T+1} = p_{i^*+1}^{T+1} = \cdots = p_n^{T+1}$ holds, and we obtain $p_n^{T+1} \geq 0$. In case of $i^* = 1$, the proof is done in a similar way.

Now we show $\boldsymbol{p}^{T+1} = 0$. Since $\boldsymbol{p} \in P_n$, $\sum_{j=1}^{n} p_j^{T+1} = \sum_{j=1}^{n} (n+1-j)$ holds. In a similar way as the proof of $\boldsymbol{p}^{T+1} \geq 0$,

$$\sum_{j=1}^{n} \sum_{t=1}^{T} \lambda_t q_j^t = \sum_{t=1}^{T} \lambda_t \sum_{j=1}^{n} q_j^t = \sum_{t=1}^{T} \lambda_t \sum_{j=1}^{n} (n+1-j) = \sum_{j=1}^{n} (n+1-j).$$

Since $\boldsymbol{p}^{T+1} \geq 0$, $\boldsymbol{p}^{T+1} = \boldsymbol{p} - \sum_{t=1}^{T} \lambda_t \boldsymbol{q}^t = 0$. □

Lemma 8. *The number of iterations* T *is at most* n.

Proof. From the definition of λ_t, there is at least one $i \in [n]$ satisfying that $p_i^t > p_{i+1}^t$ and $p_i^{t+1} = p_{i+1}^{t+1}$. If $p_i^t = p_{i+1}^t$, then $p_i^{t+1} = p_{i+1}^{t+1}$ as discussed in the proof of Lemma 6. Now the claim is clear. □

Proof of Lemma 5. Since Lemma 7, it is clear that the output $\sum_{t=0}^{T} \lambda_t \boldsymbol{\sigma}^t$ by Decomposition is equal to an arbitrarily given $\boldsymbol{p} \in P_n$.

It is not difficult to see that every lines in Decomposition is done in $O(n)$. Hence, we obtain that the running time is $O(n^2)$, since Lemma 8. □

We remark that $|\{\boldsymbol{\sigma}^0, \boldsymbol{\sigma}^1, \ldots, \boldsymbol{\sigma}^T\}| \leq n$ holds, the existence of such representation is suggested by well-known Caratheodory's theorem.

Memory-Efficient Implementation of Decomposition Now we discuss an algorithm with $O(n)$ space and in $O(n^2)$ time to obtain a random permutation $\boldsymbol{\pi} \in \{\boldsymbol{\sigma}^0, \boldsymbol{\sigma}^1, \ldots, \boldsymbol{\sigma}^T\}$ according to the probability λ_t, using a modified version of Decomposition. Firstly notice that we do not need to memorize $\boldsymbol{\sigma}^t$ in Decomposition to compute λ_s and $\boldsymbol{\sigma}^s$ for $s > t$. Thus two-paths algorithm is easily obtained; we memorize only λ_t for $t \in \{1, \ldots, T\}$ in the first run of Decomposition, and we choose $\boldsymbol{\sigma}^t$ with the probability λ_t in the second run of Decomposition, then we obtain a desired $\boldsymbol{\pi}$. In fact, it is easily improved to one-path algorithm, consisting of the second run only.

3.4 Main Result

Now we are ready to prove the main result. Note that by using the algorithm Decomposition,

$$E\left[\boldsymbol{\sigma}_t \cdot \boldsymbol{\ell}_t\right] = \boldsymbol{p}_t.$$

So, by Lemma 2, we get the following theorem immediately.

Theorem 2.

$$E\left[\sum_{t=1}^{T} \boldsymbol{\sigma}_t \cdot \boldsymbol{\ell}_t\right] \leq \frac{\eta \min_{\boldsymbol{\sigma} \in S_n} \sum_{t=1}^{T} \boldsymbol{\sigma} \cdot \boldsymbol{\ell}_t + \frac{n(n+1)}{2} \ln n}{1 - e^{-\eta}}.$$

In particular, if we set $\eta = 2\ln(1 + 1/\sqrt{T})$, since, $\eta \leq e^{\frac{\eta}{2}} - e^{-\frac{\eta}{2}}$ for $\eta \geq 0$, we have $\frac{\eta}{1-e^{-\eta}} \leq e^{\frac{\eta}{2}} = (1 + 1/\sqrt{T})$, and $\frac{1}{1-e^{-\eta}} = \frac{(1+1/\sqrt{T})^2}{1/T+2/\sqrt{T}} \leq 1 + \sqrt{T}/2$. Further, by using the fact that $\boldsymbol{\sigma} \cdot \boldsymbol{\ell}_t = O(n^2)$, we get the following corollary.

Corollary 3. *For* $\eta = 2\ln(1 + 1/\sqrt{T})$,

$$E\left[\sum_{t=1}^{T} \boldsymbol{\sigma}_t \cdot \boldsymbol{\ell}_t\right] \leq \min_{\boldsymbol{\sigma} \in S_n} \sum_{t=1}^{T} \boldsymbol{\sigma} \cdot \boldsymbol{\ell}_t + O(n^2\sqrt{T \ln n}).$$

4. Lower Bound

In this section, we derive a lower bound of the regret for our online prediction problem over the permutahedron P_n. To do so, we develop a reduction from the n-expert problem defined as follows.

n-expert problem

For each trial $t = 1, \ldots, T$

(1) The player chooses an expert i_t.

(2) The adversary assigns each expert i loss $\ell_{t,i} \in [0, 1]$.

(3) The player incurs loss $\ell_t = \ell_{t,i_t}$.

For the n-expert problem, lower bound of the regret is known:

Theorem 4 (Cesa-bianchi et al.[2])**.** *For n-expert problem, for sufficiently large T, the regret is*

$$\Omega\left(\sqrt{T \ln n}\right)$$

Now we show a reduction from n-expert problem to our prediction problem over the permutahedron P_n. Assume that we are given an algorithm which predicts permutations. For each trial t, first, we get a permutation $\boldsymbol{\sigma}_t$ and choose the expert $\sigma_t(1)$. Then, the adversary assigns each expert i loss $\ell_{t,i}$. Now we construct a loss $\bar{\boldsymbol{\ell}}_t$ for the permutation $\boldsymbol{\sigma}_t$ as follows: $\bar{\boldsymbol{\ell}}_t = \ell_{t,\sigma_t(1)}(1, \ldots, 1)$.

Then observe that

$$\boldsymbol{\sigma}_t \cdot \boldsymbol{\ell}_t = \frac{n(n+1)}{2} \ell_{t,\sigma_t(1)},$$

which implies the lower bound of our problem.

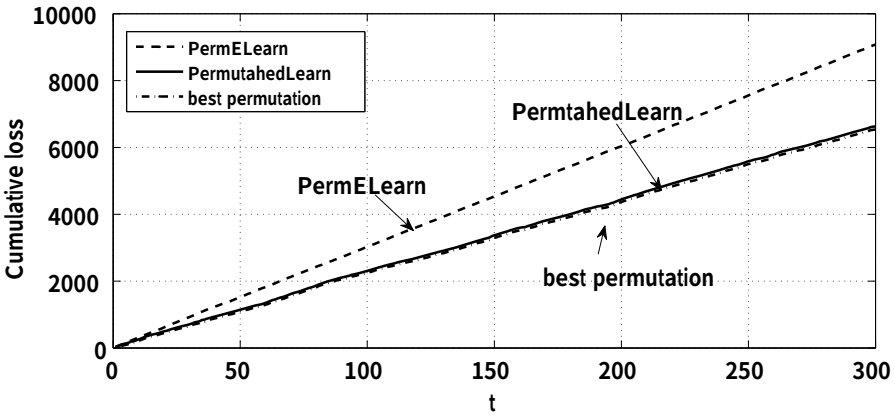
Theorem 5. *For our prediction problem over permutahedron, for sufficiently large T, the regret is*

$$\Omega\left(n^2\sqrt{T \ln n}\right).$$

5. Experimental Results

In this section, we show our initial experiments of our algorithms for artificial

Fig. 4 Cumulative losses of PermtahedLearn, PermELearn, and the best permutation.



data. For our artificial data, we fix $n = 10$. To generate a loss vector at each trial t, we specify each i-the element $\ell_{t,i}$ of the loss vector $\boldsymbol{\ell}_t$ independently randomly as follows:

$$\ell_{t,i} = \begin{cases} 1 & \text{, with probability } r_i, \text{ and} \\ 0 & \text{, otherwise,} \end{cases}$$

where we set $r_i = i/n$. We generate $T = 300$ random loss vectors.

The algorithms we compare are PermtahedLearn, PermELearn and the best permutation in hindsight. As the parameter η, we consider $\eta \in \{0.025, 0.05, 0.1, 0.2\}$. For each setting of η, we run algorithms for 3 times and choose the one attaining the lowest average cumulative losses as the best parameters for each of them. As a result, we specify $\eta = 0.2$ for PermtahedLearn and $\eta = 0.1$ for PermELearn, respectively.

We plot the cumulative losses of algorithms with their best parameters in Fig. 4. As can be seen, the cumulative loss of PermutahedLearn is much smaller than that of PermELearn and competitive with that of the best fixed permutation in hindsight.

6. Conclusion

In this paper, we propose an efficient prediction algorithm for an online prediction problem over the n-dimensional permutahedron. The upper bound of the regret of our algorithm matches the lower bound, so our algorithm is optimal. Further, our algorithm runs in time $O(n^2)$ and uses $O(n)$ space at each trial.

An interesting future work would be investigating the case where permutations to predict have to meet some partial-order constraints, as studied in previous works[5] for the offline setting.

Acknowledgments This work is supported in part by MEXT Grand-in-Aid for Young Scientists (B) 21700171.

References

1) L.M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Physics*, 7:200–217, 1967.
2) N.Cesa-Bianchi, Y.Freund, D.Haussler, D.P. Helmbold, R.E. Schapire, and M.K. Warmuth. How to use expert advice. *Journal of the Association for Computing Machinery*, 44(3):427–485, 1997.
3) N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.
4) D. P. Helmbold and M. K. Warmuth. Learning permutations with exponential weights. *Journal of Machine Learning Research*, 10:1705–1736, 2009.
5) E.L. Lawler. On sequencing jobs to minimize weighted completion time subject to precedence constraints. *Annals of Discrete Mathematics 2*, 2:75–90, 1978.
6) J.Lenstra and A.R. Kan. Complexity of scheduling under precedence constraints. *Operations Research*, 26:22–35, 1978.
7) M.Queyranne and Y.Wang. Single-machine scheduling polyhedra with precedence constraints. *Mathematics of Operations Research*, 16(1):1–20, 1991.
8) A. von Arnim, U. Faigle, and R. Schrader. The permutahedron of series-parallel posets. *Discrete Applied Mathematics*, 28(1):3–9, 1990.
9) A.von Arnim and A.S. Schulz. Facets of the generalized permutahedron of a poset. *Discrete Applied Mathematics*, 72:179–192, 1997.