Online Rank Aggregation

Shota Yasutake, Kohei Hatano, Eiji Takimoto, and Masayuki Takeda
Department of Informatics, Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan
shouta.yasutake, hatano, eiji, takeda]@inf.kyushu-u.ac.jp

Abstract

We consider an online learning framework where the task is to predict a permutation which represents a ranking of \(n\) fixed objects. At each trial, the learner incurs a loss defined as Kendall tau distance between the predicted permutation and the true permutation given by the adversary. This setting is quite natural in many situations such as information retrieval and recommendation tasks. We prove a lower bound of the cumulative loss and hardness results. Then, we propose an algorithm for this problem and prove its relative loss bound which shows our algorithm is close to optimal.

1 Introduction

The rank aggregation problem is, given \(m\) permutations of \(n\) fixed elements, to find a permutation that minimizes the sum of “distances” between itself and each given permutation. Here, each permutation represents a ranking over \(n\) elements. These days, the rank aggregation problem also arises in information retrieval tasks such as combining several search results given by different search engines. In particular, the optimal ranking is called Kemeny optimal when the distance is defined as Kendall tau distance (which we will define later). From now on, we only consider Kendall tau distance as our distance measure.

It is known that the rank aggregation problem is NP-hard even when \(m \geq 4\) [3]. Some approximation algorithms are known as well. For example, Ailon et al. proposed a 11/7-approximation algorithm [2]. Further, Kenyon-Mathieu and Schudy proposed a PTAS (polynomial time approximation scheme) which runs in doubly exponential in the precision parameter \(\varepsilon > 0\) [5].

In this paper, we consider an online version of the ranking aggregation problem, which we call “online rank aggregation”. This problem is about online prediction of permutations. Let \(S_n\) be the set of all permutations of \(n\) fixed elements. Then the online rank aggregation problem consists of the following protocol for each trial \(t\): (i) The learner predicts a permutation \(\hat{\sigma}_t \in S_n\). (ii) The adversary gives the learner the true permutation \(\sigma_t \in S_n\). (iii) The learner receives the loss \(d(\sigma_t, \hat{\sigma}_t)\), Kendall tau distance between \(\sigma_t\) and \(\hat{\sigma}_t\). The goal of the learner is to minimize the cumulative loss bound \(\sum_{t=1}^{T} d(\sigma_t, \hat{\sigma}_t)\).

First of all, we derive a lower bound of the cumulative loss of any learning algorithm for online rank aggregation. More precisely, we show that there exists a probabilistic adversary such that for any learning algorithm for online rank aggregation, the cumulative loss is at least \(\min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma_t, \sigma) + \Omega(n^2 \sqrt{T})\).

Then we prove hardness results. In particular, we prove that no randomized polynomial time algorithm whose cumulative loss bound matches the lower bound, under the common assumption that NP \(\not\subset\) BPP. Further, we show that, under the same assumption, there exists no fully randomized polynomial approximation scheme (FPRAS) with cumulative loss bound \((1 + \varepsilon) \min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma_t, \sigma) + \Omega(n^2 \sqrt{T})\).
Table 1: The cumulative loss bounds $a \min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma_t, \sigma) + O(n^2 \sqrt{T})$.

<table>
<thead>
<tr>
<th>factor $a$</th>
<th>time complexity per iteration</th>
<th>our result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (optimal) $(1 + \varepsilon)$ poly time implies $NP = BPP$</td>
<td>combination of [4] and [5]</td>
<td></td>
</tr>
<tr>
<td>3/2</td>
<td>$poly(n, T)$</td>
<td>our result</td>
</tr>
<tr>
<td>4</td>
<td>$poly(n)$</td>
<td>our result</td>
</tr>
<tr>
<td></td>
<td>$O(n^2)$</td>
<td>our result</td>
</tr>
</tbody>
</table>

$O(n^2 \sqrt{T})$, where FPRAS is a polynomial time algorithm runs in $1/\varepsilon$. Therefore, the cumulative loss bound of our algorithm is close to the best one achieved by polynomial time algorithms.

On the other hand, by using Kakade et al’s offline-online converter [4] and the PTAS algorithm for rank aggregation [5], it can be shown that there exists an algorithm, for any $\varepsilon > 0$, its cumulative loss bound is $(1 + \varepsilon) \min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma_t, \sigma) + O(n^2 \sqrt{T})$, with its running time is $poly(T)n^{O(1/\varepsilon)^6}$.

Finally, we propose an efficient algorithm for online rank aggregation. For the algorithm which we call PermRank, we prove its expected cumulative loss bound is at most

$$3 \min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma_t, \sigma) + O\left(n^2 \sqrt{T}\right).$$

The running time is that for solving a convex optimization problem with $O(n^2)$ variables and $O(n^3)$ linear constraints, which does not depends on $T$. In addition, a version of our algorithm runs in time $O(n^2)$ with a weaker loss bound that has factor 4 instead of 3/2 (omitted). We summarize the cumulative loss bounds in Table 1.

## 2 Preliminaries

Let $n$ be a fixed integer s.t. $n \geq 1$, and we denote $[n] = \{1, \ldots, n\}$. Let $N = n(n-1)/2$. Let $S_n$ be the set of permutations on $[n]$. The Kendall tau distance $d(\sigma_1, \sigma_2)$ between permutations $\sigma_1, \sigma_2 \in S_n$ is defined as $d(\sigma_1, \sigma_2) = \sum_{i,j=1}^{n} I(\sigma_1(i) > \sigma_1(j) \land \sigma_2(i) < \sigma_2(j))$, where $I(\cdot)$ is the indicator function, i.e., $I(true) = 1$ and $I(false) = 0$. That is, Kendall tau distance between two permutations is the total number of pairs of elements for which the orders in two permutations disagree. By definition, it holds that $0 \leq d(\sigma_1, \sigma_2) \leq N$, and it is known that Kendall tau distance satisfies the conditions of metric.

A comparison vector $q$ is a vector in $\{0,1\}^N$. We define the following mapping $\phi: S_n \to [0,1]^N$ which maps a permutation to a comparison vector: For $i, j \in [n]$ s.t. $i \neq j$, $\phi(\sigma)_{ij} = 1$ if $\sigma(i) < \sigma(j)$, and $\phi(\sigma)_{ij} = 0$, otherwise. Then note that the Kendall tau distance between two permutations is represented as 1-norm distance between corresponding comparison vectors, $d(\sigma_1, \sigma_2) = \|\phi(\sigma_1) - \phi(\sigma_2)\|_1$, where 1-norm $\|x\|_1 = \sum_{i=1}^{N} |x_i|$. For example, for a permutation $\sigma = (1, 3, 2)$, the corresponding comparison vector is given as $\phi(\sigma) = (1, 1, 0)$. In general, for some comparison vectors, there is no corresponding permutation. For example, the comparison vector $(1,0,1)$ represents that $\sigma(1) > \sigma(2), \sigma(2) > \sigma(3), \sigma(3) > \sigma(1)$, for which no permutation $\sigma$ exists. In particular, if a comparison vector $q \in \{0,1\}^N$ has a corresponding permutation, we say that $q$ is consistent. We denote $\phi(S_n)$ as the set of consistent comparison vectors in $\{0,1\}^N$.

## 3 Lower bound

In this section, we derive a $\Omega(n^2 \sqrt{T})$ lower bound of the cumulative loss for online rank aggregation. In particular, our lower bound is obtained when the adversary is probabilistic.

**Theorem 1.** For any online prediction algorithms of permutations and any integer $T \geq 1$, there exists a sequence $\sigma_1, \ldots, \sigma_T$ such that

$$\sum_{t=1}^{T} d(\sigma_t, \tilde{\sigma}_t) \geq \min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma_t, \sigma) + \Omega(n^2 \sqrt{T}).$$  (1)
4 Hardness

In this section, we discuss the hardness of online prediction with the optimal cumulative loss bound which matches the lower bound (1). We will show that existence of a randomized polynomial time prediction algorithm with the optimal cumulative loss bound implies a randomized polynomial algorithm for the rank aggregation, which is NP-hard [3]. A formal statement is given as follows:

**Theorem 2.** Under the assumption that \( \text{NP} \neq \text{BPP} \), there is no randomized polynomial time algorithm whose cumulative loss bound matching the lower bound (1) for the online aggregation problem.

Now we consider the possibility of fully polynomial time randomized approximation schemes (FPRAS) with cumulative loss bound \((1+\varepsilon)\min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma, \sigma_t) + O(n^2 \sqrt{T})\), whose running time is polynomial in \( n, T \) and \( 1/\varepsilon \). We say that such a FPRAS has \((1+\varepsilon)\)-approximate optimal cumulative loss bound. Then, note that if we set \( \varepsilon = 1/\sqrt{T} \), then its cumulative loss bound becomes indeed optimal.

**Corollary 3.** Under the assumption that \( \text{NP} \neq \text{BPP} \), there is no FPRAS with \((1+\varepsilon)\)-approximate optimal cumulative loss bound for the online aggregation problem.

Therefore, it is hard to improve the factor \( 1 + \varepsilon \) for arbitrary given \( \varepsilon > 0 \).

5 Our algorithm

In this section we propose our algorithm PermRank. Our idea behind PermRank consists of two parts. The first idea is that we regard a permutation as a \( N(=n(n-1)/2) \) dimensional comparison vector and deal with a problem of predicting comparison vectors. More precisely, we consider a Bernoulli trial model for each component \( i,j \) of a comparison vector. In other words, for each component \( i,j \), we assume a biased coin for which head appears with probability \( p_{ij} \), and we estimate each parameter \( p_{ij} \) in an online fashion.

The second idea is how we generate a permutation from the estimated comparison vector. As we mentioned earlier, for a given comparison vector, there might not exist a corresponding permutation. To overcome this situation, we use a variant of KWIKSORT algorithm proposed by Ailon et al. [2], LPKWIKSORT \( h \) [1]. Originally, KWIKSORT is used to solve the rank aggregation problem. The basic idea of KWIKSORT is to sort elements in a brute-force way by looking at local pairwise order only. We will show later that by using LPKWIKSORT \( h \) we can obtain a permutation whose corresponding comparison vector is close enough to the estimated comparison vector.

The algorithm uses LPKWIKSORT \( h \) and projection techniques which are now standards in online learning researches. More precisely, after the update (and before applying LPKWIKSORT \( h \)), PermRank projects the updated vector onto the set of probability vectors satisfying triangle inequalities: \( p_{ij} \leq p_{ik} + p_{kj} \) for any \( i,j,k \in [n] \), where \( p_{ij} = 1 - p_{ji} \). Note that any consistent comparison vector satisfies these triangle inequalities. We show the detail of PermRank in Appendix.

5.1 Our Analysis

In this subsection we show our relative loss bound of PermRank.

For LPKWIKSORT \( h \), the following property is proved \(^1\).

**Lemma 1** (Ailon [1]). Suppose that permutation \( \hat{\sigma}_t \) is output of LPKWIKSORT \( h \) at trial \( t \). Then, for each trial \( t \),

\[
E[d(\sigma_t, \hat{\sigma}_t)] \leq \frac{3}{2} ||p_t - y_t||_1,
\]

where the expectation is about the randomization in LPKWIKSORT \( h \).

\(^1\)Originally, Lemma 1 is proved for the case where the solution of a LP relaxation of the (partial) rank aggregation problem is given as input. But, in fact, the lemma holds for any probability vectors satisfying triangle inequality.
Algorithm 1 PermRank

1. Let $p_1 = (\frac{1}{2}, \ldots, \frac{1}{2}) \in [0,1]^N$.
2. For $t = 1, \ldots, T$
   (a) Predict a permutation $\hat{\sigma}_t = \text{LPWIKSORT}_t(p_t)$.
   (b) Get a true permutation $\sigma_t$ and let $y_t = \phi(\sigma_t)$.
   (c) Update $p_{t+\frac{1}{2}}$ as
      \[
      p_{t+\frac{1}{2}} = \frac{p_{t,ij} e^{-\eta(1-y_{t,ij})}}{(1-p_{t,ij}) e^{-\eta y_{t,ij}} + p_{t,ij} e^{-\eta(1-y_{t,ij})}}.
      \]
   (d) Let $p_{t+1}$ be the projection of $p_{t+\frac{1}{2}}$ onto the set of points satisfying triangle inequalities. That is,
      
      \[
      p_{t+1} = \arg \inf_{p} \Delta_2(p, p_{t+\frac{1}{2}})
      \]
      sub. to:
      \[
      p_{ik} \leq p_{ij} + p_{jk}, \text{ for } i, j, k \in [n],
      \]
      \[
      p_{ij} \geq 0, \text{ for } i, j \in [n].
      \]

By using Lemma 1, we obtain the cumulative loss bound of PermRank as follows:

**Theorem 4.** For $\eta = \ln(1 + \sqrt{1/T})$, the expected cumulative loss of Permrank is at most

\[
E \left[ \sum_{t=1}^{T} d(\sigma_t, \hat{\sigma}_t) \right] \leq \frac{3}{2} \min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma_t, \sigma) + O \left( n^2 \sqrt{T} \right).
\]

**References**


6 Appendix

In Appendix, we show the details of LPKWIKSORT$_h$ and proofs of the theorems.

6.1 LPKWIKSORT

We show the details of LPKWIKSORT$_h$ in Algorithm 6.1.

The algorithm LPKWIKSORT$_h$ uses the following function $h$:

$$h(x) = \begin{cases} 0 & 0 \leq x \leq \frac{1}{6} \\ \frac{3}{2}x - \frac{1}{4} & \frac{1}{6} < x \leq \frac{5}{6} \\ 1 & \frac{5}{6} < x \leq 1 \end{cases}$$

Note that $h$ is symmetric in the sense that $h(1 - x) = 1 - h(x)$.

A description of LPKWIKSORT$_h$ is given in Algorithm 2.

Algorithm 2 LPKWIKSORT$_h$ (Ailon et al. [1])

**Input:** a $N$-dimensional vector $p \in [0, 1]^N$

**Output:** a permutation

1. Let $S_L$ and $S_R$ be empty sets, respectively.
2. Pick an integer $i$ from $\{1, \ldots, n\}$ randomly.
3. For each $j \in \{1, \ldots, n\}$ such that $j \neq i$,
   (a) With probability $h(p_{ij})$, put $j$ in $S_L$.
   (b) Otherwise, put $j$ in $S_R$.
4. Let $p_L, p_R$ be the comparison vector induced by $S_L$ and $S_R$, respectively.
5. Output $(\text{LPKWIKSORT}_h(p_L), i, \text{LPKWIKSORT}_h(p_R))$.

6.2 Proof of Theorem 1

**Proof.** The proof partly follows a well known technique. We consider the following strategy of the adversary: At each trial $t$, give the learning algorithm either the permutation $\hat{\sigma}_t = \sigma^0 = (1, \ldots, n)$ or $\hat{\sigma}_t = \sigma^1 = (n, n-1, \ldots, 1)$ randomly with probability half. Note that the corresponding comparison vectors are $\phi(\sigma^0) = (0, \ldots, 0)$ and $\phi(\sigma^1) = (1, \ldots, 1)$, respectively.

Then, for any $t \geq 1$ and any permutation $\hat{\sigma}_t$, $E[d(\hat{\sigma}_t - \hat{\sigma}_t)] = \binom{n}{2}/2$. This implies that the expected cumulative loss of any learning algorithm is exactly $\frac{\binom{n}{2}}{2} T$, because of the linearity of the expectation.

Next, we consider the expected cumulative loss of the best of $\sigma^0$ and $\sigma^1$, that is, $E\left[\min_{i=0,1} \sum_{t=1}^{T} d(\sigma_t - \sigma^i)\right]$. By our construction of the adversary, this expectation is reduced to

$$E \left[ \min_{p=0,1} \sum_{t=1}^{T} d(\sigma_t - \sigma^p) \right] = \left(\frac{n}{2}\right) E_{y_1, \ldots, y_T} \left[ \min_{p=0,1} \sum_{t=1}^{T} |p - y_t| \right],$$

where $y_1, \ldots, y_T$ are independent random $\{0, 1\}$-variables. The above expectation can be further written as

$$\left(\frac{n}{2}\right) E_{y_1, \ldots, y_T} \left[ \min_{p=0,1} \sum_{t=1}^{T} |p - y_t| \right] = \frac{\binom{n}{2}}{2} - \frac{\binom{n}{2}}{2} E_{y_1, \ldots, y_T} \left[ (\# \text{ of } 0s) - (\# \text{ of } 1s) \right].$$

Then the second term in the last equality is bounded as $-\binom{n}{2} \Omega(\sqrt{T})$. 


Thus, we have $E \left[ \sum_{t=1}^{T} d(\sigma_t, \tilde{\sigma}_t) - \min_{p=0,1} \sum_{t=1}^{T} d(\sigma_t, \sigma^p) \right] \geq \Omega(n^2 \sqrt{T})$. So, there exists a sequence $\sigma_1, \ldots, \sigma_T$ such that
\[
\sum_{t=1}^{T} d(\sigma_t, \tilde{\sigma}_t) \geq \min_{p=0,1} \sum_{t=1}^{T} d(\sigma_t, \sigma^p) + \Omega(n^2 \sqrt{T}) \geq \min_{\sigma \in S_n} \sum_{t=1}^{T} d(\sigma_t, \sigma) + \Omega(n^2 \sqrt{T}).
\]
\[
\square
\]

6.3 Proof of Theorem 2

Proof. We assume a randomized polynomial time online algorithm $A$ with the optimal cumulative loss bound. Given $m$ fixed permutations, we choose a permutation uniformly randomly among them and we run $A$ on the chosen permutation. We repeat this procedure for $T = cm^2 n^4$ times repeatedly, where $c$ is some constant. For a sufficiently large $c > 0$, the average expected loss of $A$ w.r.t. $m$ permutation is that of the best permutation plus $1/(4m)$. Then, we pick up a permutation randomly among predicted permutations $\tilde{\sigma}_1, \ldots, \tilde{\sigma}_T$. We call this permutation as the representative permutation. Note that expected average loss of the representative permutation is exactly that of the best permutation plus $1/(4m)$.

Now, we repeat this procedure for $k$ times and get $k = O(n^4 m^2)$ representative permutations. By using Hoeffding’s bound, with probability at least, say, $2/3$, the best among $k$ representative ones has the best average loss plus $1/(2m)$. Note that, since Kendall tau distance takes integers in $[0, n(n - 1)/2]$, the average loss of the best representative takes values in $[0, 1/m, 2/m, \ldots, n(n - 1)/T/2]$. So, the average loss of the best representative is the same with that of the best permutation. Therefore, we can find the best permutation in polynomial time in $n$ and $m$ with probability at least $2/3$. Since rank aggregation is NP-hard, this implies that $NP \subseteq BPP$.

\[
\square
\]

6.4 Proof of Theorem 4

Before we prove Theorem 4, we need additional lemmas. For $p, q \in [0, 1]$, the binary relative entropy $\Delta_2(p, q)$ between $p$ and $q$ is defined as $\Delta_2(p, q) = p \ln \frac{p}{q} + (1 - p) \ln \frac{1 - p}{1 - q}$. Further, we extend the definition of the binary relative entropy for vectors in $[0, 1]^N$. That is, for any $p, q \in [0, 1]^N$, the binary relative entropy is given as $\Delta_2(p, q) = \sum_{i=1}^{N} \Delta_2(p_i, q_i)$.

Lemma 2. For each $t = 1, \ldots, T$ and any comparison vector $q$,
\[
\Delta_2(q, p_t) - \Delta_2(q, p_{t+1}) \geq -\eta\|y_t - q\|_1 + (1 - e^{-\eta})\|y_t - p_t\|_1.
\]

Proof. By applying Generalized Pythagorean Theorem, we obtain
\[
\Delta_2(q, p_t) - \Delta_2(q, p_{t+1}) \geq \Delta_2(q, p_t) - \Delta_2(q, p_{t+1}) + \Delta_2(p_{t+1}, p_{t+\frac{1}{2}}) \\
\geq \Delta_2(q, p_t) - \Delta_2(q, p_{t+\frac{1}{2}}).
\]

Further, by a standard calculation,
\[
\Delta_2(q, p_t) - \Delta_2(q, p_{t+\frac{1}{2}}) \geq -\eta\|y_t - q\|_1 + (1 - e^{-\eta})\|y_t - p_t\|_1,
\]
which completes the proof.

\[
\square
\]

Lemma 3. For any comparison vector $q \in \{0, 1\}^N$,
\[
\sum_{t=1}^{T} \|y_t - p_t\|_1 \leq \eta \sum_{t=1}^{T} \|y_t - q\|_1 + \frac{n(n-1) \ln 2}{1 - e^{-\eta}}.
\]

Proof. By summing up the inequality in Lemma 2 for $t = 1, \ldots, T$, we get that
\[
\eta \sum_{t=1}^{T} \|y_t - q\|_1 - \Delta_2(q, p_{T+1}) + \Delta_2(q, p_1) \\
1 - e^{-\eta}
\]
Since $\Delta_2(q, p_{T+1}) \geq 0$ and $\Delta_2(q, p) \leq \ln 2$, we complete the proof.

\[
\square
\]

6
Proof of Theorem 4. If we set $\eta = 2 \ln(1 + 1/\sqrt{T})$, by the fact that $\eta \leq e^{\frac{2}{\sqrt{T}}} - e^{-\frac{2}{\sqrt{T}}}$, we get

$$\frac{\eta}{1 - e^{-\eta}} \leq e^{\frac{\eta}{\sqrt{T}}} = 1 + 1/\sqrt{T},$$

and

$$\frac{1}{1 - e^{-\eta}} = \frac{(1 + 1/\sqrt{T})^2}{1/T + 2/\sqrt{T}} \leq 1 + \sqrt{T}/2,$$

respectively. So, by Lemma 3 and Lemma 1, we complete the proof.