Combinatorial bandit prediction with relaxation-based approximation algorithm

Ryohei Nagaura[1,a)] Kohei Hatano[1,2,b)] Eiji Takimoto[1,c)]

Abstract: We consider combinatorial online prediction problems with bandit feedbacks. In particular, we take an approach of converting an offline approximation algorithm to an online one. Under the natural assumption that the offline combinatorial problem has a relaxation-based approximation scheme, we show an efficient and robust conversion method.

1. Introduction

Combinatorial online decision making problems often arise in various situations such as scheduling, ranking, routing, and so on, where decisions are combinatorial objects, e.g., permutations, spanning trees, and paths. More precisely, the combinatorial online decision making problem we consider is formulated as a repeated game between the algorithm and the adversary. The algorithm is given a fixed decision set $C \subset \mathbb{R}^n_+$ of decisions. The protocol of the game is defined for each trial $t = 1, \ldots, T$:

(1) The algorithm chooses a decision vector $\boldsymbol{c}_t \in C$ from the decision set C.
(2) The adversary chooses a loss vector $\boldsymbol{s}_t \in \mathbb{R}^n_+$.
(3) The adversary returns a feedback to the algorithm:
- (Full information setting) The feedback is the loss vector $\boldsymbol{s}_t$.
- (Bandit setting) The feedback is the values of the loss $l_t = \boldsymbol{c}_t^\top \boldsymbol{s}_t$

(4) The algorithm incurs the loss l_t.

In particular, we focus on the bandit setting of the game, where the algorithm can observe only the loss at each trial and it does not know the loss vector. Further assumptions are that the algorithm is possibly randomized and that the adversary possibly know how the algorithm works, but does not know the results of random variables used by the algorithm. The goal of the algorithm is to minimize the α-regret for sufficiently small $\alpha \geq 1$,

$$\alpha\text{-regret} = \mathbb{E}\left[\sum_{t=1}^{T} \boldsymbol{c}_t^\top \boldsymbol{s}_t\right] - \alpha \min_{\boldsymbol{c} \in C} \sum_{t=1}^{T} \boldsymbol{c}^\top \boldsymbol{s}_t,$$

where the expectation is taken with respect to the randomness of the algorithm.

There are two approaches to the problem. The first approach is to design algorithms for specific classes of decision sets such as permutations, k-sets, paths, spanning trees and so on. This approach, however, needs careful elaboration for each specific classes. The other approach is to "convert" an offline approximation algorithm to a corresponding online algorithm. This approach is advantageous to the first one in that it is generic; we can apply many existing offline approximation algorithms without designing specific online algorithms. In this paper, we consider the second approach to the online combinatorial bandit problem.

Previous work assumes that for the decision set C, there exists an α-approximation algorithm. More precisely, their assumption is that there exists an algorithm, with input $\boldsymbol{s} \in \mathbb{R}^n_+$, outputs $\boldsymbol{c} \in C$ such that $\boldsymbol{c}^\top \boldsymbol{s} \leq \alpha \min_{\boldsymbol{c}^* \in C} \boldsymbol{c}^{*\top} \boldsymbol{s}$. For simplicity, the running time of the approximation algorithm is constant. Kakade et al. [5] proposed a strategy which achieves α-$regret = O(\alpha T^{2/3})$ with $O(poly(n)T)$ time at each trial. Garber [4] proposed a more efficient algorithm which runs in time $O(poly(n) \log T)$ with the same regret bound.

In this paper, we assume a stronger but natural condition for the offline approximation algorithm. We assume the approximation algorithm is based on a relaxation scheme. That is, it first solves a continuous linear optimization problem over a relaxed set $relax(C)$, where $relax(C) \supset C$ is a convex polytope defined by polynomially many linear constraints. Then it obtains a decision $\boldsymbol{c}$ from the relaxed solution satisfying $\boldsymbol{c}^\top \boldsymbol{s} \leq \alpha \min_{\boldsymbol{x}^* \in relax(C)} \boldsymbol{x}^{*\top} \boldsymbol{s}$. Under this assumption, we obtain a bandit online algorithm which achieves $(\alpha + \epsilon)$-regret $O((\alpha + \epsilon)T^{2/3})$ with $O(poly(n, 1/\epsilon))$ computation time per trial.

Our result is obtained by extending the work of Fujita et al. [3] for the full information setting to the bandit setting. The key technique of Fujita et al. is to construct a "metarounding" algorithm from the relaxation-based approximation algorithm. An α-metarounding algorithm for C, when given $\boldsymbol{x} \in relax(C)$ as input, outputs $\boldsymbol{c} \in C$ such that for any $\boldsymbol{s} \in \mathbb{R}^n_+$, $\boldsymbol{c}^\top \boldsymbol{s} \leq \alpha \boldsymbol{x}^\top \boldsymbol{s}$. In particular, they show a construction method of $(\alpha + \epsilon)$-metarounding algorithm whose running time is $O(poly(n, 1/\epsilon))$. We will use the metarounding algorithm while omitting the details. We summa-

[1] Kyushu University, Japan
[2] RIKEN AIP, Japan
[a)] ryohei.nagaura@inf.kyushu-u.ac.jp
[b)] hatano@inf.kyushu-u.ac.jp
[c)] eiji@inf.kyushu-u.ac.jp

rize the previous results and ours in Table 1.

Table 1 Summary of results.

	α-regret (($\alpha+\epsilon$) for ours)	time per trial
Kakade et al. [5]	$O(\alpha T^{\frac{3}{8}})$	$O(poly(n)T^{\frac{3}{8}})$
Garber [4]	$O(\alpha T^{\frac{3}{8}})$	$O(poly(n)\log T)$
ours	$O((\alpha+\epsilon)T^{\frac{3}{8}})$	$O(poly(n,1/\epsilon))$

Other related work

There are several previous results for the conversion approach in the full information setting. The first result is the Follow the Perturbed Leader (FPL) proposed by Kalai and Vempala [6]. Kakade et al. [5] proposed a strategy which achieves α-*regret* = $O(\alpha\sqrt{T})$ with $O(poly(n)T)$ time at each trial. Garber [4] game a better algorithm whose running time is $O(poly(n)\log T)$ with the same regret bound. In the bandit setting, Bubeck et al. [1] proposed the OSMD algorithm for convex decision sets. This algorithm achieves optimal regret bounds for many classes of decision sets.

2. Preliminaries

In this section, we define remark notations, definitions and problem setting.

2.1 Notations

We denote scalars with lower case letters and vectors with bold case letters, matrix with uppercase letters. The ith element of vector $\boldsymbol{x}$ is denoted by $x(i)$. Vectors are column in the standard, so inner product between vector $\boldsymbol{x}$ and $\boldsymbol{y}$ is denoted by $\boldsymbol{x}^\top\boldsymbol{y}$ with transpose symbol $\top$. Let $\mathbb{R}$ be a whole set of real number and $\mathbb{R}_+$ be a non-negative set of $\mathbb{R}$. We denote a hypercube whose vertices are in $\{-a,a\}^n$ as $\mathcal{B}(a)$,i.e.,$\mathcal{B}(a)=[-a,a]^n$. The expectation operator of a distribution D is denoted by $\mathbb{E}_D[\cdot]$. Let $[n]$ be a set of all positive integers less than $n+1$ (i.e.$[n]=\{1,2,\ldots,n\}$).

2.2 Definitions

Definition 2.1 (Bregman Divergence)**.** *Let $F:\Gamma\to\mathbb{R}$ be a strictly convex function. We denote a Bregman divergence D_F with respect to F and define below.*

$$D_F(\boldsymbol{x},\boldsymbol{y})=F(\boldsymbol{x})-F(\boldsymbol{y})-\nabla F(\boldsymbol{y})^\top(\boldsymbol{x}-\boldsymbol{y})$$

Definition 2.2 (Legendre-Fenchel dual)**.** *We define the Legendre-Fenchel dual of a function F as*

$$F^\star(\boldsymbol{y})=\min_{\boldsymbol{x}}\left(\boldsymbol{x}^\top\boldsymbol{y}-F(\boldsymbol{x})\right).$$

Note that $(\nabla F^\star)^{-1}=\nabla F$.

Definition 2.3 (β-Barycentric Spanner (β-BS))**.** *A set $\{\boldsymbol{q}_1,\boldsymbol{q}_2,\ldots,\boldsymbol{q}_n\}\subset C$ is a β-Barycentric Spanner (β-BS) for C if*

$$\forall\boldsymbol{c}\in C,\ \exists\lambda_1,\ldots,\lambda_n\in[-\beta,\beta],\ \boldsymbol{c}=\sum_{i\in[n]}\lambda_i\boldsymbol{q}_i$$

3. Algorithm

In this section, we propose a bandit algorithm described in Algorithm 1. As a preprocessing, the algorithm calculates β-BS for a real decision set C and make a mapping matrix $Q(i)(j)=q_j(i)$ for every $i,j\in[n]$. We consider a mapped set $\hat{C}=Q^{-1}C$ because it becomes easy to analyze regret thanks to some propositions described in appendix

We define $\hat{\mathcal{P}}$ a mapped space of $\mathcal{P}$ by Q, i.e. $\hat{\mathcal{P}}=Q^{-1}\mathcal{P}$. $\hat{\mathcal{P}}$ also consists of $poly(n)$ linear constraint. Due to proposition in appendix, it is clearly that $\hat{C}$ is contained by a n-dimensional hypercube $\mathcal{B}(\beta)$. Intersection between $\hat{\mathcal{P}}$ and $\mathcal{B}(\beta)$ is denoted by $\hat{\mathcal{B}}$. Note that $\hat{\mathcal{B}}$ has $poly(n)$ linear constraint since $\hat{\mathcal{P}}$ and $\mathcal{B}(\beta)$ also have. When we update online combinatorial prediction problems over pseudo decision space, then algorithm solve the problem over real decision set. Framework of our algorithm is as follows: In each iteration $t=1,\ldots,T$, at first, the algorithm decides to do either exploration or exploitation with probability $\gamma,1-\gamma$ respectively . The algorithm choose i_t uniformly at random and outputs $\boldsymbol{e}_{i_t}$ in exploration term. Else, algorithm solves α-metarouding problem with input $\hat{\boldsymbol{x}}_t$ and play $\hat{\boldsymbol{c}}$ distributed by some metarouding algorithm. After outputting decision vector, the algorithm receives incur its loss l_t. Next, algorithm estimates a pseudo loss vector. If it explored in first step, estimate loss vector is calculated as $\tilde{\boldsymbol{s}}_t=l_t\frac{n}{\gamma}\boldsymbol{e}_{i_t}$. Else $\tilde{\boldsymbol{s}}_t=\boldsymbol{0}$. This method makes $\tilde{\boldsymbol{s}}_t$ unbiased estimator of $\hat{\boldsymbol{s}}_t$ described in lemma3.1 in detail. Then the algorithm updates $\hat{\boldsymbol{x}}_t$ OSMD-like with regarding estimate vector $\tilde{\boldsymbol{s}}_t$ as a true pseudo loss vector $\hat{\boldsymbol{s}}_t$ over $\hat{\mathcal{P}}$. Last, algorithm finds the most closest point in $\hat{\mathcal{B}}$ with respect to Bregman divergence. We remark pseudo code of our algorithm below.

Algorithm 1 Proposed algorithm

(1) Parameters: learning ratio $\eta>0$, exploration parameter $0<\gamma<1$
(2) Calculate β-BS(C) = $\{\boldsymbol{q}_1,\ldots,\boldsymbol{q}_n\}$ and a mapping matrix Q, $\hat{\mathcal{B}}=\hat{\mathcal{P}}\cap\mathcal{B}(\beta)$.
(3) Set $\boldsymbol{x}_1\in\hat{\mathcal{P}}$ be any initial point.
(4) For $t=1,\ldots,T$
  (a) $b_t=\begin{cases}0 & \text{(with probability }\gamma)\\ 1 & \text{(with probability }1-\gamma)\end{cases}$
  (b) if $b_t=0$(exploration) then
    (i) The algorithm chooses $i_t\in[n]$ uniformly at random and play $\hat{\boldsymbol{c}}_t=\boldsymbol{e}_{i_t}$.
    (ii) Incur loss l_t and calculate $\tilde{\boldsymbol{s}}_t=\frac{n}{\gamma}l_t\boldsymbol{e}_{i_t}$.
  (c) else(exploitation)
    (i) Solves α-metarouding problem with input $\hat{\boldsymbol{x}}_t$.
    (ii) Incur loss l_t and calculate $\tilde{\boldsymbol{s}}_t=\boldsymbol{0}$.
  (d) update $\hat{\boldsymbol{x}}_{t+\frac{1}{2}}=\nabla F^\star(\nabla F(\hat{\boldsymbol{x}}_t)-\eta\tilde{\boldsymbol{s}}_t)$.
  (e) Project $\hat{\boldsymbol{x}}_{t+1/2}$ onto $\hat{\mathcal{B}}$. That is $\hat{\boldsymbol{x}}_{t+1}=\arg\min_{\hat{\boldsymbol{x}}\in\hat{\mathcal{B}}}D_F(\hat{\boldsymbol{x}},\hat{\boldsymbol{x}}_{t+1/2})$.

We use a Euclidean norm squared regularizer $F(\boldsymbol{x})=\sum_{i\in[n]}x(i)^2$ in OSMD-like update, so

$$D_F(\boldsymbol{x},\boldsymbol{y})=\|\boldsymbol{x}-\boldsymbol{y}\|_2^2$$

In our algorithm, it holds the following lemmas.

Lemma 3.1. *$\tilde{\boldsymbol{s}}_t$ is a unbiased estimater of $\hat{\boldsymbol{s}}_t$, i.e.,*

$$\mathbb{E}\left[\tilde{\boldsymbol{s}}_t\right]=\hat{\boldsymbol{s}}_t$$

Proof. Since $\tilde{\boldsymbol{s}}_t$ is independent of randomness of exploitation,

$$\begin{aligned}
\mathbb{E}[\tilde{\boldsymbol{s}}_t] &= \mathbb{E}_{b_t,i_t}[\tilde{\boldsymbol{s}}_t] \\
&= \gamma \mathbb{E}_{i_t}[\tilde{\boldsymbol{s}}_t | b_t = 0] + (1-\gamma)\mathbb{E}[\tilde{\boldsymbol{s}}_t | b_t = 1] \\
&= \gamma \frac{1}{n} \sum_{k\in[n]} \frac{n}{\gamma} l_t \boldsymbol{e}_k + \boldsymbol{0} \\
&= \sum_{k\in[n]} \boldsymbol{e}_k l_t \\
&= \sum_{k\in[n]} \boldsymbol{e}_k \boldsymbol{e}_k^\top \hat{\boldsymbol{s}}_t \\
&= \hat{\boldsymbol{s}}_t
\end{aligned}$$

□

Lemma 3.2. *Euclidean norm squared of* $\tilde{\boldsymbol{s}}_t$

$$\mathbb{E}\left[\|\tilde{\boldsymbol{s}}_t\|_2^2\right] \le \frac{n^2 D^2}{\gamma}$$

Proof.

$$\begin{aligned}
\mathbb{E}[\|\tilde{\boldsymbol{s}}_t\|_2^2] &= \mathbb{E}[\tilde{\boldsymbol{s}}_t^\top \tilde{\boldsymbol{s}}_t] \\
&= \gamma \frac{1}{n} \sum_{k\in[n]} \left(\frac{n}{\gamma} l_t \boldsymbol{e}_k\right)^\top \left(\frac{n}{\gamma} l_t \boldsymbol{e}_k\right) + (1-\gamma)\boldsymbol{0} \\
&= \frac{n}{\gamma} l_t^2 n \\
&\le \frac{n^2 D^2}{\gamma} \quad (\because l_t \le D)
\end{aligned}$$

□

4. Main Results

Theorem 4.1. *Using MBB algorithm [3] in line (4)-(c)-(i) in algorithm 3 and setting* $\eta = O(\beta D^{-1} T^{-2/3}), \gamma = O(nT^{-1/3})$*, our algorithm runs at each trial* $poly(n, 1/\epsilon)$ *time and achieves the expected regret below.*

$$(\alpha+\epsilon) - regret = O((\alpha+\epsilon)\beta n D T^{\frac{2}{3}})$$

Proof. For any $\hat{\boldsymbol{x}} \in \hat{\mathcal{P}}$,

$$\begin{aligned}
&\mathbb{E}\left[\textstyle\sum_{t=1}^T \boldsymbol{c}_t^\top \boldsymbol{s}_t\right] - (\alpha+\epsilon)\textstyle\sum_{t=1}^T \boldsymbol{x}^\top \boldsymbol{s}_t \\
&= \mathbb{E}\left[\sum_{t=1}^T \hat{\boldsymbol{c}}_t^\top \hat{\boldsymbol{s}}_t\right] - (\alpha+\epsilon)\sum_{t=1}^T \hat{\boldsymbol{x}}^\top \hat{\boldsymbol{s}}_t \\
&= \mathbb{E}\left[\sum_{t=1}^T \hat{\boldsymbol{c}}_t^\top \hat{\boldsymbol{s}}_t - (\alpha+\epsilon)\sum_{t=1}^T \hat{\boldsymbol{x}}_t^\top \hat{\boldsymbol{s}}_t + (\alpha+\epsilon)\sum_{t=1}^T \hat{\boldsymbol{x}}_t^\top \hat{\boldsymbol{s}}_t - (\alpha+\epsilon)\sum_{t=1}^T \hat{\boldsymbol{x}}^\top \hat{\boldsymbol{s}}_t\right] \\
&= \mathbb{E}\left[\sum_{t=1}^T \hat{\boldsymbol{c}}_t^\top \hat{\boldsymbol{s}}_t - (\alpha+\epsilon)\sum_{t=1}^T \hat{\boldsymbol{x}}_t^\top \hat{\boldsymbol{s}}_t\right] \\
&\quad + \mathbb{E}\left[(\alpha+\epsilon)\sum_{t=1}^T \hat{\boldsymbol{x}}_t^\top \hat{\boldsymbol{s}}_t - (\alpha+\epsilon)\sum_{t=1}^T \hat{\boldsymbol{x}}^\top \hat{\boldsymbol{s}}_t\right] \\
&\le TD\gamma + (\alpha+\epsilon)\mathbb{E}\left[\sum_{t=1}^T \hat{\boldsymbol{x}}_t^\top \hat{\boldsymbol{s}}_t - \sum_{t=1}^T \hat{\boldsymbol{x}}^\top \hat{\boldsymbol{s}}_t\right] \qquad (1)
\end{aligned}$$

Since $\mathbb{E}[\hat{\boldsymbol{c}}_t] = \gamma \frac{1}{n}\sum_{k\in[n]} \boldsymbol{e}_k + (1-\gamma)\mathbb{E}_{MBB}[\hat{\boldsymbol{c}}]$, so we obtain

$$\begin{aligned}
&\mathbb{E}\left[(\hat{\boldsymbol{c}}_t - (\alpha+\epsilon)\hat{\boldsymbol{x}}_t)^\top \boldsymbol{s}_t\right] \\
&= \gamma \frac{1}{n}\sum_{k\in[n]} \boldsymbol{e}_k^\top \boldsymbol{s}_t + \mathbb{E}_{MBB}[\hat{\boldsymbol{c}}^\top \boldsymbol{s}_t] - \gamma \mathbb{E}_{MBB}[\hat{\boldsymbol{c}}^\top \boldsymbol{s}_t] - (\alpha+\epsilon)\hat{\boldsymbol{x}}^\top \boldsymbol{s}_t \\
&\le \gamma \frac{1}{n}\sum_{k\in[n]} \boldsymbol{e}_k^\top \boldsymbol{s}_t + (\alpha+\epsilon)\hat{\boldsymbol{x}}^\top \boldsymbol{s}_t - \gamma \mathbb{E}_{MBB}[\hat{\boldsymbol{c}}^\top \boldsymbol{s}_t] - (\alpha+\epsilon)\hat{\boldsymbol{x}}^\top \boldsymbol{s}_t \\
&\le \gamma \frac{1}{n}\sum_{k\in[n]} D \quad (\because l_t \le D) \\
&= D\gamma
\end{aligned}$$

Then the first term of (1) is bounded by $TD\gamma$. Thanks to lemma3.1, the second term of (2) is

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^T \hat{\boldsymbol{x}}_t^\top \hat{\boldsymbol{s}}_t - \sum_{t=1}^T \hat{\boldsymbol{x}}^\top \hat{\boldsymbol{s}}_t\right] &= \mathbb{E}\left[\sum_{t=1}^T \hat{\boldsymbol{x}}_t^\top \mathbb{E}[\tilde{\boldsymbol{s}}_t] - \sum_{t=1}^T \hat{\boldsymbol{x}}^\top \mathbb{E}[\tilde{\boldsymbol{s}}_t]\right] \\
&= \mathbb{E}\left[\sum_{t=1}^T \hat{\boldsymbol{x}}_t^\top \tilde{\boldsymbol{s}}_t - \sum_{t=1}^T \hat{\boldsymbol{x}}^\top \tilde{\boldsymbol{s}}_t\right]
\end{aligned}$$

From Generalized Pythagorean Theorem (see e.g. [2]),

$$D_F(\hat{\boldsymbol{x}}_{t+1}, \hat{\boldsymbol{x}}_{t+1/2}) + D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_{t+1}) \le D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_{t+1/2})$$

Then,

$$\begin{aligned}
\eta \tilde{\boldsymbol{s}}_t^\top (\hat{\boldsymbol{x}}_t - \hat{\boldsymbol{x}}) &= (\hat{\boldsymbol{x}}_t - \hat{\boldsymbol{x}})^\top (\nabla F(\hat{\boldsymbol{x}}_{t+1/2}) - \nabla F(\hat{\boldsymbol{x}}_t)) \\
&\quad (\because \nabla F(\hat{\boldsymbol{x}}_{t+1/2}) = \nabla F(\hat{\boldsymbol{x}}_t) - \eta \tilde{\boldsymbol{s}}_t) \\
&= D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_t) + D_F(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{x}}_{t+1/2}) - D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_{t+1/2}) \\
&\le D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_t) + D_F(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{x}}_{t+1/2}) \\
&\quad - D_F(\hat{\boldsymbol{x}}_{t+1}, \hat{\boldsymbol{x}}_{t+1/2}) - D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_{t+1}) \\
&\le D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_t) + D_F(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{x}}_{t+1/2}) - D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_{t+1}) \\
&\quad (\because D_F(\hat{\boldsymbol{x}}_{t+1}, \hat{\boldsymbol{x}}_{t+1/2}) \ge 0)
\end{aligned}$$

Summing up this inequality for $t = 1, \ldots, T$, we get

$$\begin{aligned}
\sum_{t=1}^T \eta \tilde{\boldsymbol{s}}_t^\top (\hat{\boldsymbol{x}}_t - \hat{\boldsymbol{x}}) &\le D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_1) - D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_{T+1}) + \sum_{t=1}^T D_F(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{x}}_{t+1/2}) \\
&\le D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_1) + \sum_{t=1}^T D_F(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{x}}_{t+1/2}) \quad (\because D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_{T+1}) \ge 0)
\end{aligned}$$

Here,

$$\begin{aligned}
D_F(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{x}}_{t+1/2}) &= \|\hat{\boldsymbol{x}}_t - \hat{\boldsymbol{x}}_{t+1/2}\|_2^2 \\
&= \|\eta \tilde{\boldsymbol{s}}_t\|_2^2 \\
&\le \eta^2 \frac{n^2 D^2}{\gamma} \quad (\because lemma3.2)
\end{aligned}$$

(1)And,

$$D_F(\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_1) \le 4n\beta^2$$

Since $\hat{\mathcal{C}} \subset \hat{\mathcal{P}}$, we have $(\alpha+\epsilon)-$regret upper bounded

(2) $$TD\gamma + (\alpha+\epsilon)\frac{4n\beta^2}{\eta} + (\alpha+\epsilon)\frac{T\eta n^2 D^2}{\gamma}$$

By tuning parameters properly, we complete proof. □

5. Concluding Remarks

In this paper, we propose a bandit algorithm using an approximation algorithm based on LP relaxation. Our algorithm is superior to previous work in the two points. One is that our algorithm's computation time at each iteration doesn't depend on T. The other is that the player does not need to know the approximation ratio α. One of the future works is to improve the regret bound to $(\alpha+\epsilon)$-*regret* $= O((\alpha+\epsilon)\sqrt{T})$.

References

[1] Sébastien Bubeck, Nicolò Cesa-Bianchi, and Sham M. Kakade. Towards minimax policies for online linear optimization with bandit feedback. In *COLT 2012 - The 25th Annual Conference on Learning Theory, June 25-27, 2012, Edinburgh, Scotland*, pages 41.1–41.14, 2012.

[2] Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

[3] Takahiro Fujita, Kohei Hatano, and Eiji Takimoto. Combinatorial online prediction via metarounding. In *Algorithmic Learning Theory - 24th International Conference, ALT 2013, Singapore, October 6-9, 2013. Proceedings*, pages 68–82, 2013.

[4] Dan Garber. Efficient online linear optimization with approximation algorithms. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 627–635, 2017.

[5] Sham M. Kakade, Adam Tauman Kalai, and Katrina Ligett. Playing games with approximation algorithms. *SIAM J. Comput.*, 39(3):1088–1106, 2009.

[6] Adam Tauman Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *J. Comput. Syst. Sci.*, 71(3):291–307, 2005.

Appendix

We define a mapping matrix $Q \in C^n$ which have ith $\beta-$BS $\boldsymbol{q}_i$ in its ith column i.e. $Q(i)(j) = q_j(i)$. We consider a mapped concept $\hat{C}$ of C, which is $\hat{C} = \{Q\boldsymbol{c} | \boldsymbol{c} \in C\}$. Then the following propositions holds.

Proposition 1. *$\hat{C}$ includes standard basis $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$.*

Proof. By definition of Q,

$$\forall i \in [n]. \boldsymbol{q}_i = Q\boldsymbol{e}_i$$

.

Q is inverticle for almost natural discrete concepts, so

$$\forall i \in [n]. \ \boldsymbol{e}_i = Q^{-1}\boldsymbol{q}_i \in \hat{C}$$

□

Proposition 2. *Suppose that a discrete set C has β-BS $\{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n\}$, then $\{\boldsymbol{e}_1, \ldots \boldsymbol{.e}_n\}$ are β-BS of $\hat{C}$.*

Proof. By definition of β-BS for C,

$$\forall \boldsymbol{c} \in C. \ \boldsymbol{c} = \sum_{i \in [n]} \lambda_i \boldsymbol{q}_i$$

Multiplying the matrix Q^{-1} to both sides from left side of the above equation, we obtain

$$\forall \hat{\boldsymbol{c}} \in \hat{C}. \ \hat{\boldsymbol{c}} = \sum_{i \in [n]} \lambda_i \boldsymbol{e}_i$$

□

Considering pseudo adversary which outputs pseudo loss vector $\hat{\boldsymbol{s}} = Q^\top \boldsymbol{s}$, the following proposition holds.

Proposition 3. *A loss between decision vector $\boldsymbol{c}$ and loss vector $\boldsymbol{s}$ equals to a loss between corresponding pseudo decision and loss vectors.*

Proof.

$$\begin{aligned}\hat{\boldsymbol{c}}^\top \hat{\boldsymbol{s}} &= (Q^{-1}\boldsymbol{c})^\top Q^\top \boldsymbol{s} \\ &= \boldsymbol{c} Q^{-\top} Q^\top \boldsymbol{s} \\ &= \boldsymbol{c}^\top \boldsymbol{s}\end{aligned}$$

□