

Improved Algorithms for Online Load Balancing

Liu, Yaxiong

Department of Informatics, Kyushu University

Hatano, Kohei

Faculty of Arts and Science, Kyushu University

Takimoto, Eiji

Department of Informatics, Kyushu University

<https://hdl.handle.net/2324/4495597>

出版情報 : SOFSEM 2021 : theory and practice of computer science : 47th International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2021, Bolzano-Bozen, Italy, January 25-29, 2021 : proceedings, pp.203-217, 2021-01-11. Springer-Verlag

バージョン :

権利関係 :



Improved algorithms for online load balancing^{*}

Yaxiong Liu^{1,3}[0000–0001–8588–4543], Kohei Hatano^{2,3}[0000–0002–1536–1269], and
Eiji Takimoto¹[0000–0001–9524–2553]
{yaxiong.liu, hatano, eiji}@inf.kyushu-u.ac.jp¹²³

¹ Department of Informatics, Kyushu University, Japan

² Faculty of Arts and Science, Kyushu University, Japan

³ RIKEN AIP, Japan

Abstract. We consider an online load balancing problem and its extensions in the framework of repeated games. On each round, the player chooses a distribution (task allocation) over K servers, and then the environment reveals the load of each server, which determines the computation time of each server for processing the task assigned. After all rounds, the cost of the player is measured by some norm of the cumulative computation-time vector. The cost is the makespan if the norm is L_∞ -norm. The goal is to minimize the regret, i.e., minimizing the player’s cost relative to the cost of the best fixed distribution in hindsight. We propose algorithms for general norms and prove their regret bounds. In particular, for L_∞ -norm, our regret bound matches the best known bound and the proposed algorithm runs in polynomial time per trial involving linear programming and second order programming, whereas no polynomial time algorithm was previously known to achieve the bound.

Keywords: online learning · blackwell approachability · online load balancing · makespan · second order cone programming.

1 Introduction

Online load balancing problem is an active research topic since last century. Instead of the traditional measurement of algorithm performance, competitive ratio (e.g., [2] [3] [5] [12]), we utilize another well-known measurement as “Regret”, involved by [6].

In this paper we define online load balancing problem as follows. There are K parallel servers and the protocol is defined as a game between the player and the environment. On each round $t = 1, \dots, T$, (i) the player selects a distribution α_t over K servers, which can be viewed as an allocation of data, (ii) then the environment assigns a loaded condition $l_{t,i}$ for each server i and the loss of server i is given as $\alpha_{t,i} l_{t,i}$. The goal of the player is to minimize the makespan of the cumulative loss vector of all servers after T rounds, i.e., $\max_{i=1, \dots, K} \sum_{t=1}^T \alpha_{t,i} l_{t,i}$, compared relatively to the makespan obtained by the optimal static allocation

^{*} This work was supported by JSPS KAKENHI Grant Numbers JP19H04174 and JP19H04067, respectively.

α^* in hindsight. More precisely, the goal is to minimize the regret, the difference between the player's makespan and the static optimal makespan. The makespan cost can be viewed as L_∞ -norm of the vector of cumulative loss of each server (we will give a formal definition of the problem in the next section).

Even-Dar et al. [6] gave an algorithm based on the regret minimum framework by involving an extra concept, the Blackwell approachability [4] with respect to L_2 -norm, to a target set, which is defined in the following section. This algorithm achieves the regret bound as $O(\sqrt{KT})$. Simultaneously another algorithm, DIFF, achieves the regret upper bound as $O((\ln K)\sqrt{T})$. Rahklin et al. [13] gave a theoretical result for the online load balancing problem, that the upper bound to regret can achieve $O(\sqrt{(\ln K)T})$, rather than $O((\ln K)\sqrt{T})$. However there is no efficient algorithm given in this paper to obtain this regret.

Then, there were some explorations about the equivalence between the Blackwell approachability and online linear optimization(OLO) [1], in addition and online convex optimization(OCO) by involving a support function [15].

These work [1] [15] implied that the Blackwell approachability with respect to general norm can be guaranteed by sub-linearity of regret from OCO problem reduced by Blackwell approaching game. Moreover due to this result we give an efficient algorithm to online load balancing problem, achieving the best known regret.

More specifically speaking, we propose algorithms for online load balancing for arbitrary norms under a natural assumption. This algorithm is composed by three reductions. (i) First reduction is from load balancing problem to Blackwell approaching game in a general metric. In this reduction we extend the L_2 -norm of load balancing problem in [6] to any general norm with a reasonable assumption. In this reduced Blackwell approaching game the metric is induced by the norm of load balancing problem. This reduction implies that the regret of load balancing problem can be bounded by the convergence rate of a corresponding Blackwell approaching game. (ii) Second reduction directly follows the existing work. Due to [15], we give a reduction from Blackwell approaching game to an OCO problem, by showing the existence of such reduction. Thus we can bound the regret of online load balancing with the regret of corresponding OCO problem. (iii) The last reduction is from OCO problem to two OLO problems, so that we can predict with FTRL.

Conclusively, we can predict the allocation of serves on each round in online load balancing according to the prediction of corresponding two OLO problems. Simultaneously we give the regret bound of online load balancing problem with this OLO regret.

Thus our technical contributions are the following:

- We propose a new reduction technique from online load balancing to a Blackwell approaching game. This reduction enables us to use more general norms, induced by online load balancing, in Blackwell approaching game rather than L_2 -norm used in the previous work [6]. Based on this reduction we can reduce online load balancing with general norm to OLO problem, by using the reduction technique of Shimkin [15] from Blackwell games to OCO problem,

further to OLO problems. In conclusion, online load balancing problem can be reduced to two OLO problems according to our reduction route. Therefore the regret bound to online load balancing problem can be optimized by the corresponding OLO regret bound.

- Especially, according to above reduction route, we give an efficient algorithm for online load balancing w.r.t. L_∞ -norm, achieving the best known $O(\sqrt{T \ln K})$ regret. The algorithm involves linear programming and the second order cone programming and runs in polynomial time per trial. This is the first polynomial time algorithm achieving $O(\sqrt{T \ln K})$ regret.

2 Preliminaries

First we give some notations. We use $\|\cdot\|$ to denote a norm of a vector. Moreover, for a norm $\|\cdot\|$, $\|\mathbf{x}\|_*$ denotes the dual norm of $\|\mathbf{x}\|$. A norm $\|\cdot\|$ over \mathbb{R}^d is monotone if $\|\mathbf{x}\| \leq \|\mathbf{y}\|$ whenever $|x_i| \leq |y_i|$ for every $1 \leq i \leq d$.

2.1 Online load balancing

Firstly we begin with a standard (offline) load balancing problem. Suppose that we have K servers to do a simple task with a large amount of data. The task can be easily parallelized in such a way that we can break down the data into K pieces and assign them to the servers, and then each server processes the subtask in time proportional to the size of data assigned. An example is to find blacklisted IP addresses in an access log data. Each server is associated with loaded condition, expressed in terms of “the computation time per unit data”. The goal is to find a data assignment to the servers so as to equalize the computation time for all servers. In other words, we want to minimize the *makespan*, defined as the maximum of the computation time over all servers.

Formally, the problem is described as follows: The input is a K -dimensional vector $\mathbf{l} = (l_1, l_2, \dots, l_K) \in \mathbb{R}_+^K$, where each l_i represents the loaded condition of the i -th server. The output is a K -dimensional probability vector $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_K) \in \Delta(K) = \{\boldsymbol{\alpha} \in [0, 1]^K \mid \sum_{i=1}^K \alpha_i = 1\}$, where each α_i represents the fraction of data assigned to the i -th server. The goal is to minimize the makespan $\|\boldsymbol{\alpha} \odot \mathbf{l}\|_\infty$, where $\boldsymbol{\alpha} \odot \mathbf{l} = (\alpha_1 l_1, \alpha_2 l_2, \dots, \alpha_K l_K)$. Note that it is clear that the optimal solution is given by $\alpha_i = l_i^{-1} / \sum_{j=1}^K l_j^{-1}$, which equalizes the computation time of every server as $C_\infty^*(\mathbf{l}) \stackrel{\text{def}}{=} \min_{\boldsymbol{\alpha} \in \Delta(K)} \|\boldsymbol{\alpha} \odot \mathbf{l}\|_\infty = \frac{1}{\sum_{j=1}^K 1/l_j}$.

Note also that the objective is generalized to the L_p -norm for any p in the literature.

In this paper, we consider a more general objective $\|\boldsymbol{\alpha} \odot \mathbf{l}\|$ for an arbitrary norm that satisfies certain assumptions stated below. In the general case, the optimal value is denoted by $C^*(\mathbf{l}) \stackrel{\text{def}}{=} \min_{\boldsymbol{\alpha} \in \Delta(K)} \|\boldsymbol{\alpha} \odot \mathbf{l}\|$.

Assumption 1 *Throughout the paper, we put the following assumptions on the norm. (i) The norm is monotone, and (ii) the function C^* is concave.*

Note that the first assumption is natural for load balancing and the both assumptions are satisfied by L_p -norm for $p > 1$.

Now we proceed to the online load balancing problem with respect to a norm $\|\cdot\|$ that satisfies Assumption 1. The problem is described as a repeated game between the learner and the environment who may behave adversarially. In each round $t = 1, 2, \dots, T$, the learner chooses an assignment vector $\alpha_t \in \Delta(K)$, and then receives from the environment a loaded condition vector $\mathbf{l}_t \in [0, 1]^K$, which may vary from round to round. After the final round is over, the performance of the learner is naturally measured by $\left\| \sum_{t=1}^T \alpha_t \odot \mathbf{l}_t \right\|$. We want to make the learner perform nearly as well as the performance of the best fixed assignment in hindsight (offline optimal solution), which is given by $C^*(\sum_{t=1}^T \mathbf{l}_t)$. To be more specific, the goal is to minimize the following *regret*:

$$\text{Regret}(T) = \left\| \sum_{t=1}^T \alpha_t \odot \mathbf{l}_t \right\| - C^* \left(\sum_{t=1}^T \mathbf{l}_t \right).$$

2.2 Repeated game with vector payoffs and approachability

We briefly review the notion of Blackwell's approachability, which is defined for a repeated game with vector payoffs. The game is specified by a tuple $(A, B, r, S, \text{dist})$, where A and B are convex and compact sets, $r : A \times B \rightarrow \mathbb{R}^d$ is a vector-valued payoff function, $S \subseteq \mathbb{R}^d$ is a convex and closed set called the *target set*, and $\text{dist} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_+$ is a metric. The protocol proceeds in trials: In each round $t = 1, 2, \dots, T$, the learner chooses a vector $\mathbf{a}_t \in A$, the environment chooses a vector $\mathbf{b}_t \in B$, and then the learner obtains a vector payoff $\mathbf{r}_t \in \mathbb{R}^d$, given by $\mathbf{r}_t = r(\mathbf{a}_t, \mathbf{b}_t)$. The goal of the learner is to make the average payoff vector arbitrarily close to the target set S .

Definition 1 (Approachability). *For a game $(A, B, r, S, \text{dist})$, the target set S is approachable with convergence rate $\gamma(T)$ if there exists an algorithm for the learner such that the average payoff $\bar{\mathbf{r}}_T = (1/T) \sum_{t=1}^T \mathbf{r}_t$ satisfies*

$$\text{dist}(\bar{\mathbf{r}}_T, S) \stackrel{\text{def}}{=} \min_{\mathbf{s} \in S} \text{dist}(\bar{\mathbf{r}}_T, \mathbf{s}) \leq \gamma(T)$$

against any environment. In particular, we simply say that S is approachable if it is approachable with convergence rate $o(T)$.

Blackwell characterizes the approachability in terms of the support function as stated in the proposition below.

Definition 2. *For a set $S \subseteq \mathbb{R}^d$, the support function $h_S : \mathbb{R}^d \rightarrow \mathbb{R} \cup \{\infty\}$ is defined as*

$$h_S(\mathbf{w}) = \sup_{\mathbf{s} \in S} \langle \mathbf{s}, \mathbf{w} \rangle.$$

It is clear from definition that h_S is convex whenever S is convex.

Definition 3 (Blackwell [4]). A game $(A, B, r, S, \text{dist})$ satisfies Blackwell Condition, if and only if

$$\forall \mathbf{w} \in \mathbb{R}^d \left(\min_{\mathbf{a} \in A} \min_{\mathbf{b} \in B} \langle \mathbf{w}, r(\mathbf{a}, \mathbf{b}) \rangle \leq h_S(\mathbf{w}) \right). \quad (1)$$

Remark 1. In [4], Blackwell characterized the approachability of a target set for L_2 -norm metric in terms of the Blackwell condition.

In what follows, we only consider a norm metric, i.e, $\text{dist}(\mathbf{r}, \mathbf{s}) = \|\mathbf{r} - \mathbf{s}\|$ for some norm $\|\cdot\|$ over \mathbb{R}^d . The following proposition is useful.

Proposition 1. For any $\mathbf{w} \in \mathbb{R}^d$, $\mathbf{s}^* = \arg \max_{\mathbf{s} \in S} \langle \mathbf{s}, \mathbf{w} \rangle$ is a sub-gradient of $h_S(\mathbf{w})$ at \mathbf{w} .

Proof. For any $\mathbf{w}, \mathbf{u} \in \mathbb{R}^d$, let $\mathbf{s}^* = \arg \max_{\mathbf{s} \in S} \langle \mathbf{s}, \mathbf{w} \rangle$ and $\mathbf{s}^u = \arg \max_{\mathbf{s} \in S} \langle \mathbf{s}, \mathbf{u} \rangle$. Since $\langle \mathbf{s}^*, \mathbf{u} \rangle \leq \langle \mathbf{s}^u, \mathbf{u} \rangle$, we have

$$\begin{aligned} h_S(\mathbf{w}) - h_S(\mathbf{u}) &= \sup_{\mathbf{s} \in S} \langle \mathbf{s}, \mathbf{w} \rangle - \sup_{\mathbf{s} \in S} \langle \mathbf{s}, \mathbf{u} \rangle = \langle \mathbf{s}^*, \mathbf{w} \rangle - \langle \mathbf{s}^u, \mathbf{u} \rangle \\ &\leq \langle \mathbf{s}^*, \mathbf{w} - \mathbf{u} \rangle, \end{aligned}$$

which implies the proposition. \square

2.3 Online convex optimization

In this subsection we briefly review online convex optimization with some known results. See, e.g., [14, 7] for more details.

An online convex optimization (OCO) problem is specified by (W, F) , where $W \subseteq \mathbb{R}^d$ is a compact convex set called the decision set and $F \subseteq \{f : W \rightarrow \mathbb{R}\}$ is a set of convex functions over W called the loss function set. The OCO problem (W, F) is described by the following protocol between the learner and the adversarial environment. For each round $t = 1, 2, \dots, T$, the learner chooses a decision vector $\mathbf{w}_t \in W$ and then receives from the environment a loss function $f_t \in F$. In this round, the learner incurs the loss given by $f_t(\mathbf{w}_t)$. The goal is to make the cumulative loss of the learner nearly as small as the cumulative loss of the best fixed decision. To be more specific, the goal is to minimize the following regret:

$$\text{Regret}_{(W, F)}(T) = \sum_{t=1}^T f_t(\mathbf{w}_t) - \min_{\mathbf{w} \in W} \sum_{t=1}^T f_t(\mathbf{w}).$$

Here we add the subscript (W, F) to distinguish from the regret for online load balancing.

Any OCO problem can be reduced to an online linear optimization (OLO) problem, which is an OCO problem with linear loss functions. More precisely, an OLO problem is specified by (W, G) , where $G \subseteq \mathbb{R}^d$ is the set of cost vectors

such that the loss function at round t is $\langle \mathbf{g}_t, \cdot \rangle$ for some cost vector $\mathbf{g}_t \in G$. For the OLO problem (W, G) , the regret of the learner is thus given by

$$\text{Regret}_{(W,G)}(T) = \sum_{t=1}^T \langle \mathbf{g}_t, \mathbf{w}_t \rangle - \min_{\mathbf{w} \in W} \sum_{t=1}^T \langle \mathbf{g}_t, \mathbf{w} \rangle.$$

The reduction from OCO to OLO is simple. Run any algorithm for OLO (W, G) with $\mathbf{g}_t \in \partial f_t(\mathbf{w}_t)$, and then it achieves $\text{Regret}_{(W,F)}(T) \leq \text{Regret}_{(W,G)}(T)$, provided that G is large enough, i.e., $G \supseteq \bigcup_{f \in F, \mathbf{w} \in W} \partial f(\mathbf{w})$.

A standard FTRL (follow-the-regularized-leader) strategy for the OLO problem (W, G) is to choose \mathbf{w}_t as

$$\mathbf{w}_t = \arg \min_{\mathbf{w} \in W} \left(\sum_{s=1}^{t-1} \langle \mathbf{g}_s, \mathbf{w} \rangle + \eta_t R(\mathbf{w}) \right), \quad (2)$$

where $R : W \rightarrow \mathbb{R}$ is a strongly convex function called the regularizer and $\eta_t \in \mathbb{R}_+$ is a parameter. Using the strategy (2) the following regret bound is known.

Proposition 2 ([14]). *Suppose that the regularizer $R : W \rightarrow \mathbb{R}$ is σ -strongly convex w.r.t. some norm $\|\cdot\|$, i.e., for any $\mathbf{w}, \mathbf{u} \in W$, for any $\mathbf{z} \in \partial R(\mathbf{w})$, $R(\mathbf{u}) \geq R(\mathbf{w}) + \langle \mathbf{z}, \mathbf{u} - \mathbf{w} \rangle + \frac{\sigma}{2} \|\mathbf{u} - \mathbf{w}\|^2$. Then, for the OLO problem (W, G) , the regret of the strategy (2) satisfies*

$$\text{Regret}_{(W,G)}(T) = O(D_R L_G \sqrt{T/\sigma}),$$

where $D_R = \sqrt{\max_{\mathbf{w} \in W} R(\mathbf{w})}$, $L_G = \max_{\mathbf{g} \in G} \|\mathbf{g}\|_*$ and $\eta_t = (L_G/D_R) \sqrt{T/\sigma}$.

Note however that the strategy does not consider the computational feasibility at all. For efficient reduction, we need an efficient algorithm that computes a sub-gradient $\mathbf{g} \in \partial f(\mathbf{w})$ when given (a representation of) $f \in F$ and $\mathbf{w} \in W$, and an efficient algorithm for solving the convex optimization problem (2).

For a particular OLO problem (W, G) with L_1 ball decision set $W = \{\mathbf{w} \in \mathbb{R}^d \mid \|\mathbf{w}\|_1 \leq 1\}$, an algorithm called EG^\pm [8] finds in linear time the optimal solution of (2) with an entropic regularizer and achieves the following regret.

Theorem 2 ([9]). *For the OLO problem (W, G) with $W = \{\mathbf{w} \in \mathbb{R}^d \mid \|\mathbf{w}\|_1 \leq 1\}$ and $G = \{\mathbf{g} \in \mathbb{R}^d \mid \|\mathbf{g}\|_\infty \leq M\}$, EG^\pm achieves*

$$\text{Regret}_{(W,G)}(T) \leq M \sqrt{2T \ln(2d)}.$$

3 Main result

In this section, we propose a meta-algorithm for online load balancing, which is obtained by combining a reduction to two independent OLO problems and an OLO algorithm (as an oracle) for the reduced problems. Note that the reduced

OLO problems depend on the choice of norm for online load balancing, and the OLO problems are further reduced to some optimization problems defined in terms of the norm. For efficient implementation, we assume that the optimization problems are efficiently solved.

Now we consider the online load balancing problem on K servers with respect to a norm $\|\cdot\|$ defined over \mathbb{R}^K that satisfies Assumption 1. The reduction we show consists of three reductions, the first reduction is to a repeated game with vector payoffs, the second one is to an OCO problem, and the last one is to two OLO problems. In the subsequent subsections, we give these reductions, respectively.

3.1 Reduction to a vector payoff game

We will show that the online load balancing problem can be reduced to the following repeated game with vector payoffs, denoted by $P = (A, B, r, S, \text{dist})$, where

- $A = \Delta(K)$, $B = [0, 1]^K$,
- $r : A \times B \rightarrow \mathbb{R}^K \times \mathbb{R}^K$ is the payoff function defined as $r(\alpha, \mathbf{l}) = (\alpha \odot \mathbf{l}, \mathbf{l})$,
- $S = \{(\mathbf{x}, \mathbf{y}) \in [0, 1]^K \times [0, 1]^K \mid \|\mathbf{x}\| \leq C^*(\mathbf{y})\}$, and
- dist is the metric over $\mathbb{R}^K \times \mathbb{R}^K$ defined as $\text{dist}(\mathbf{r}, \mathbf{s}) = \|\mathbf{r} - \mathbf{s}\|^+$, where $\|\cdot\|^+$ is the norm over $\mathbb{R}^K \times \mathbb{R}^K$ defined as

$$\|(\mathbf{x}, \mathbf{y})\|^+ = \|\mathbf{x}\| + \|\mathbf{y}\|.$$

Here we use the convention that $\mathbb{R}^{2K} = \mathbb{R}^K \times \mathbb{R}^K$. Note that the target set S is convex since $\|\cdot\|$ is convex and C^* is concave by our assumption. Note also that it is easy to verify that $\|\cdot\|^+$ is a norm whenever $\|\cdot\|$ is a norm, and its dual is

$$\|(\mathbf{x}, \mathbf{y})\|_*^+ = \max\{\|\mathbf{x}\|_*, \|\mathbf{y}\|_*\}. \quad (3)$$

The reduction is similar to that in [6], but they consider a fixed norm $\|\cdot\|_2$ to define the metric, no matter what norm is used for online load balancing.

Proposition 3. *Assume that we have an algorithm for the repeated game P that achieves convergence rate $\gamma(T)$. Then, the algorithm, when directly applied to the online load balancing problem, achieves*

$$\text{Regret}(T) \leq T\gamma(T).$$

Proof. Let \mathcal{A} denote an algorithm for the repeated game P with convergence rate $\gamma(T)$. Assume that when running \mathcal{A} against the environment of online load balancing, we observe, in each round t , $\alpha_t \in \Delta(K)$ output from \mathcal{A} and $\mathbf{l}_t \in [0, 1]^K$ output from the environment.

Let $(\mathbf{x}, \mathbf{y}) = \arg \min_{(\mathbf{x}, \mathbf{y}) \in S} \|\bar{\mathbf{r}}_T - (\mathbf{x}, \mathbf{y})\|^+$, where $\bar{\mathbf{r}}_T = (1/T) \sum_{t=1}^T r(\alpha_t, \mathbf{l}_t)$ is the average payoff. Note that by the assumption of \mathcal{A} , we have $\|\bar{\mathbf{r}}_T - (\mathbf{x}, \mathbf{y})\|^+ \leq \gamma(T)$. For simplicity, let $L_T^A = (1/T) \sum_{t=1}^T \alpha_t \odot \mathbf{l}_t$ and $L_T = (1/T) \sum_{t=1}^T \mathbf{l}_t$.

Then, we have

$$\begin{aligned}
(1/T)\text{Regret}(T) &= \|L_T^A\| - C^*(L_T) \\
&= [\|\mathbf{x}\| - C^*(\mathbf{y})] + [\|L_T^A\| - \|\mathbf{x}\|] + [C^*(\mathbf{y}) - C^*(L_T)] \\
&\leq \|L_T^A - \mathbf{x}\| + \left[\min_{\alpha \in \Delta(K)} \|\alpha \odot \mathbf{y}\| - \min_{\alpha \in \Delta(K)} \|\alpha \odot L_T\| \right] \\
&\leq \|L_T^A - \mathbf{x}\| + \max_{\alpha \in \Delta(K)} [\|\alpha \odot \mathbf{y}\| - \|\alpha \odot L_T\|] \\
&\leq \|L_T^A - \mathbf{x}\| + \max_{\alpha \in \Delta(K)} \|\alpha \odot (\mathbf{y} - L_T)\| \\
&\leq \|L_T^A - \mathbf{x}\| + \|\mathbf{y} - L_T\| \\
&= \|(L_T^A, L_T) - (\mathbf{x}, \mathbf{y})\|^+ = \|\bar{r}_T - (\mathbf{x}, \mathbf{y})\|^+ \leq \gamma(T),
\end{aligned}$$

where the first inequality is from the definition of S and the triangle inequality, the third inequality is from the triangle inequality, and the fourth inequality is from the monotonicity of the norm. \square

3.2 Reduction to an OCO problem

Next we give the second sub-reduction from the repeated game P to an OCO problem. We just follow a general reduction technique of Shimkin [15] as given in the next theorem.

Theorem 3 ([15]). *Let $(A, B, r, S, \text{dist})$ be a repeated game with vector payoffs, where $\text{dist}(\mathbf{r}, \mathbf{s}) = \|\mathbf{r} - \mathbf{s}\|$ for some norm $\|\cdot\|$ over \mathbb{R}^d . Assume that we have an algorithm \mathcal{A} that witnesses the Blackwell condition, i.e., when given $\mathbf{w} \in \mathbb{R}^d$, \mathcal{A} finds $\mathbf{a} \in A$ such that $\langle \mathbf{w}, r(\mathbf{a}, \mathbf{b}) \rangle \leq h_S(\mathbf{w})$ for any $\mathbf{b} \in B$. Assume further that we have an algorithm \mathcal{B} for the OCO problem (W, F) , where $W = \{\mathbf{w} \in \mathbb{R}^d \mid \|\mathbf{w}\|_* \leq 1\}$ and $F = \{f : \mathbf{w} \mapsto \langle -r(\mathbf{a}, \mathbf{b}), \mathbf{w} \rangle + h_S(\mathbf{w}) \mid \mathbf{a} \in A, \mathbf{b} \in B\}$. Then, we can construct an algorithm for the repeated game such that its convergence rate $\gamma(T)$ satisfies*

$$\gamma(T) \leq \frac{\text{Regret}_{(W, F)}(T)}{T}.$$

Moreover, the algorithm runs in polynomial time (per round) if \mathcal{A} and \mathcal{B} are polynomial time algorithms.

For completeness, the reduction algorithm (Algorithm 1) is as follow.

The rest to show in this subsection is to ensure the existence of algorithm \mathcal{A} required for the reduction as stated in the theorem above. In other words, we show that the Blackwell condition holds for our game $P = (\Delta(K), [0, 1]^K, r, S, \text{dist})$, where $r(\alpha, \mathbf{l}) = (\alpha \odot, \mathbf{l}, \mathbf{l}) \in \mathbb{R}^K \times \mathbb{R}^K$, $S = \{(\mathbf{x}, \mathbf{y}) \in [0, 1]^K \times [0, 1]^K \mid \|\mathbf{x}\| \leq C^*(\mathbf{y})\}$, and $\text{dist}(\mathbf{r}, \mathbf{s}) = \|\mathbf{r} - \mathbf{s}\|^+$.

Lemma 1. *The Blackwell condition holds for game P . That is, for any $\mathbf{w} \in \mathbb{R}^K \times \mathbb{R}^K$, we have*

$$\min_{\alpha \in \Delta(K)} \max_{\mathbf{l} \in [0, 1]^K} \langle \mathbf{w}, r(\alpha, \mathbf{l}) \rangle \leq h_S(\mathbf{w}).$$

Algorithm 1 Reduction from game $(A, B, r, S, \text{dist})$ with $\text{dist}(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\|$ to OCO [15]

Require: An algorithm \mathcal{A} that, when given \mathbf{w} , finds $\mathbf{a} \in A$ such that $\langle \mathbf{w}, r(\mathbf{a}, \mathbf{b}) \rangle \leq h_S(\mathbf{w})$ for any $\mathbf{b} \in B$.

Require: An algorithm \mathcal{B} for the OCO problem (W, F) , where $W = \{\mathbf{w} \mid \|\mathbf{w}\|_* \leq 1\}$ and $F = \{f : \mathbf{w} \mapsto \langle -r(\mathbf{a}, \mathbf{b}), \mathbf{w} \rangle + h_S(\mathbf{w}) \mid \mathbf{a} \in A, \mathbf{b} \in B\}$.

for $t = 1, 2, \dots, T$ **do**

1. Obtain $\mathbf{w}_t \in W$ from \mathcal{B} .

2. Run $\mathcal{A}(\mathbf{w}_t)$ and obtain $\mathbf{a}_t \in A$.

3. Output $\mathbf{a}_t \in A$ and observe $\mathbf{b}_t \in B$.

4. Construct the loss function $f_t : \mathbf{w} \mapsto \langle -r(\mathbf{a}_t, \mathbf{b}_t), \mathbf{w} \rangle + h_S(\mathbf{w})$ and feed it to \mathcal{B} .

end for

Proof (Proof of Lemma 1). Let $\mathbf{w} = (\mathbf{w}_1, \mathbf{w}_2) \in \mathbb{R}^K \times \mathbb{R}^K$. By the definition of r , the inner product in the Blackwell condition can be rewritten as a bilinear function

$$f(\boldsymbol{\alpha}, \mathbf{l}) = \langle \mathbf{w}, r(\boldsymbol{\alpha}, \mathbf{l}) \rangle = \sum_{i=1}^K w_{1,i} \alpha_i l_i + \sum_{i=1}^K w_{2,i} l_i$$

over $\Delta(K) \times [0, 1]^K$. Therefore, f meets the condition of Minimax Theorem of von Neumann. and we have

$$\min_{\boldsymbol{\alpha} \in \Delta(K)} \max_{\mathbf{l} \in [0, 1]^K} f(\boldsymbol{\alpha}, \mathbf{l}) = \max_{\mathbf{l} \in [0, 1]^K} \min_{\boldsymbol{\alpha} \in \Delta(K)} f(\boldsymbol{\alpha}, \mathbf{l}).$$

Let $\mathbf{l}^* = \arg \max_{\mathbf{l} \in [0, 1]^K} \min_{\boldsymbol{\alpha} \in \Delta(K)} f(\boldsymbol{\alpha}, \mathbf{l})$ and $\boldsymbol{\alpha}^* = \arg \min_{\boldsymbol{\alpha} \in \Delta(K)} \|\boldsymbol{\alpha} \odot \mathbf{l}^*\|$. Note that by the definition of S , we have $(\boldsymbol{\alpha}^* \odot \mathbf{l}^*, \mathbf{l}^*) \in S$. Hence we get

$$\begin{aligned} \min_{\boldsymbol{\alpha} \in \Delta(K)} \max_{\mathbf{l} \in [0, 1]^K} f(\boldsymbol{\alpha}, \mathbf{l}) &= \max_{\mathbf{l} \in [0, 1]^K} \min_{\boldsymbol{\alpha} \in \Delta(K)} f(\boldsymbol{\alpha}, \mathbf{l}) \\ &= f(\boldsymbol{\alpha}^*, \mathbf{l}^*) \\ &= \langle \mathbf{w}, ((\boldsymbol{\alpha}^* \odot \mathbf{l}^*), \mathbf{l}^*) \rangle \\ &\leq \sup_{\mathbf{s} \in S} \langle \mathbf{w}, \mathbf{s} \rangle \\ &= h_S(\mathbf{w}), \end{aligned}$$

which completes the lemma. \square

The lemma ensures the existence of algorithm \mathcal{A} . On the other hand, for an algorithm \mathcal{B} we need to consider the OCO problem (W, F) , where the decision set is

$$W = \{\mathbf{w} \in \mathbb{R}^K \times \mathbb{R}^K \mid \|\mathbf{w}\|_*^+ \leq 1\}, \quad (4)$$

and the loss function set is

$$F = \{f : \mathbf{w} \mapsto \langle -r(\boldsymbol{\alpha}, \mathbf{l}), \mathbf{w} \rangle + h_S(\mathbf{w}) \mid \boldsymbol{\alpha} \in \Delta(K), \mathbf{l} \in [0, 1]^K\}. \quad (5)$$

Since W is a compact and convex set and F consists of convex functions, we could apply a number of existing OCO algorithms to obtain $\text{Regret}_{(W, F)}(T) = O(\sqrt{T})$.

In the next subsection, we show that the problem can be simplified to two OLO problems.

3.3 Reduction to two OLO problems

Consider the OCO problem (W, F) given by (4) and (5). Following the standard reduction technique from OCO to OLO stated in Section 2.3, we obtain an OLO problem (W, G) to cope with, where $G \subseteq \mathbb{R}^K \times \mathbb{R}^K$ is any set of cost vectors that satisfies

$$G \supseteq \bigcup_{\substack{f \in F \\ \mathbf{w} \in W}} \partial f(\mathbf{w}) = \left\{ -r(\boldsymbol{\alpha}, \mathbf{l}) + \mathbf{s} \mid \boldsymbol{\alpha} \in \Delta(K), \mathbf{l} \in [0, 1]^K, \mathbf{s} \in \bigcup_{\mathbf{w} \in W} \partial h_S(\mathbf{w}) \right\}. \quad (6)$$

By (3), the decision set W can be rewritten as $W = B_*(K) \times B_*(K)$ where $B_*(K) = \{\mathbf{w} \in \mathbb{R}^K \mid \|\mathbf{w}\|_* \leq 1\}$ is the K -dimensional unit ball with respect to the dual norm $\|\cdot\|_*$. By Proposition 1, any $\mathbf{s} \in \partial h_S(\mathbf{w})$ is in the target set S , which is a subset of $[0, 1]^K \times [0, 1]^K$. Moreover, $r(\boldsymbol{\alpha}, \mathbf{l}) = (\boldsymbol{\alpha} \odot \mathbf{l}, \mathbf{l}) \in [0, 1]^K \times [0, 1]^K$ for any $\boldsymbol{\alpha} \in \Delta(K)$ and $\mathbf{l} \in [0, 1]^K$. Therefore, $G = [-1, 1]^K \times [-1, 1]^K$ satisfies (6).

Thus, $(B_*(K) \times B_*(K), [-1, 1]^K \times [-1, 1]^K)$ is a suitable OLO problem reduced from the OCO problem (W, F) . Furthermore, we can break the OLO problem into two independent OLO problems $(B_*(K), [-1, 1]^K)$ in the straightforward way: Make two copies of an OLO algorithm \mathcal{C} for $(B_*(K), [-1, 1]^K)$, denoted by \mathcal{C}_1 and \mathcal{C}_2 , and use them for predicting the first half and second half decision vectors, respectively. More precisely, for each trial t , (1) receive predictions $\mathbf{w}_{t,1} \in B_*(K)$ and $\mathbf{w}_{t,2} \in B_*(K)$ from \mathcal{C}_1 and \mathcal{C}_2 , respectively, (2) output their concatenation $\mathbf{w}_t = (\mathbf{w}_{t,1}, \mathbf{w}_{t,2}) \in W$, (3) receive a cost vector $\mathbf{g}_t = (\mathbf{g}_{t,1}, \mathbf{g}_{t,2}) \in [0, 1]^K \times [0, 1]^K$ from the environment, (4) feed $\mathbf{g}_{t,1}$ and $\mathbf{g}_{t,2}$ to \mathcal{C}_1 and \mathcal{C}_2 , respectively, to make them proceed.

It is clear that the procedure above ensures the following lemma.

Lemma 2. *The OCO problem (W, F) defined as (4) and (5) can be reduced to the OLO problem $(B_*(K), [0, 1]^K)$, and*

$$\text{Regret}_{(W, F)}(T) \leq 2\text{Regret}_{(B_*(K), [0, 1]^K)}(T).$$

3.4 Putting all the pieces together

Combining all reductions stated in the previous subsections, we get an all-in-one algorithm as described in Algorithm 2.

It is clear that combining Proposition 3, Theorem 3 and Lemma 2, we get the following regret bound of Algorithm 2.

Theorem 4. *Algorithm 2 achieves*

$$\text{Regret}(T) \leq 2\text{Regret}_{(B_*(K), [-1, 1]^K)}(T),$$

Algorithm 2 An OLO-based online load balancing algorithm

Require: An algorithm \mathcal{A} that, when given \mathbf{w} , finds $\boldsymbol{\alpha} = \arg \min_{\boldsymbol{\alpha} \in \Delta(K)} \max_{\mathbf{l} \in [0,1]^K} \langle \mathbf{w}, (\boldsymbol{\alpha} \odot \mathbf{l}, \mathbf{l}) \rangle$.

Require: An algorithm \mathcal{B} that, when given \mathbf{w} , finds $\mathbf{s} \in \partial h_S(\mathbf{w})$.

Require: Two copies of an algorithm, \mathcal{C}_1 and \mathcal{C}_2 , for the OLO problem $(B_*(K), [-1, 1]^K)$.

for $t = 1, 2, \dots, T$ **do**

1. Obtain $\mathbf{w}_{t,1}$ and $\mathbf{w}_{t,2}$ from \mathcal{C}_1 and \mathcal{C}_2 , respectively, and let $\mathbf{w}_t = (\mathbf{w}_{t,1}, \mathbf{w}_{t,2})$.
2. Run $\mathcal{A}(\mathbf{w}_t)$ and obtain $\boldsymbol{\alpha}_t \in \Delta(K)$.
3. Output $\boldsymbol{\alpha}_t$ and observe $\mathbf{l}_t \in [0, 1]^K$.
4. Run $\mathcal{B}(\mathbf{w}_t)$ and obtain $\mathbf{s}_t = (\mathbf{s}_{t,1}, \mathbf{s}_{t,2})$.
5. Let $\mathbf{g}_{t,1} = -\boldsymbol{\alpha}_t \odot \mathbf{l}_t + \mathbf{s}_{t,1}$ and $\mathbf{g}_{t,2} = -\mathbf{l}_t + \mathbf{s}_{t,2}$.
6. Feed $\mathbf{g}_{t,1}$ and $\mathbf{g}_{t,2}$ to \mathcal{C}_1 and \mathcal{C}_2 , respectively.

end for

where the regret in the right hand side is the regret of algorithm \mathcal{C}_1 (and \mathcal{C}_2 as well). Moreover, if \mathcal{A} , \mathcal{B} and \mathcal{C}_1 runs in polynomial time, then Algorithm 2 runs in polynomial time (per round).

By applying the FTRL as in (2) to the OLO problem $(B_*(K), [-1, 1]^K)$ with a strongly convex regularizer R , Proposition 2 implies the following regret bound.

Corollary 1. Assume that there exists a regularizer $R : B_*(K) \rightarrow \mathbb{R}$ that is σ -strongly convex w.r.t. L_1 -norm. Then, there exists an algorithm for the online load balancing problem that achieves

$$\text{Regret}(T) = O(D_R \sqrt{T/\sigma}),$$

where $D_R = \sqrt{\max_{\mathbf{w} \in B_*(K)} R(\mathbf{w})}$.

In particular, for the OLO problem $(B_1(K), [-1, 1]^K)$, algorithm EG^\pm achieves $\sqrt{2T \ln 4K}$ regret bound as shown in Theorem 2. Thus we have $O(\sqrt{T \ln K})$ regret bound for online load balancing with respect to L_∞ -norm (i.e., w.r.t. makespan), which improves the bound of [6] by a factor of $\sqrt{\ln K}$. Moreover, for L_∞ -norm, it turns out that we have polynomial time algorithms for \mathcal{A} and \mathcal{B} , which we will give in the next section. We thus obtain the following corollary.

Corollary 2. There exists a polynomial time (per round) algorithm for the online load balancing problem with respect to L_∞ -norm that achieves

$$\text{Regret}(T) \leq 2\sqrt{2T \ln 4K}.$$

4 Algorithmic details for L_∞ -norm

In this section we give details of Algorithm 2 for the makespan problem, i.e., for L_∞ -norm.

4.1 Computing α_t

First, we give details of implementation of \mathcal{A} in Algorithm 2. Specifically, on the round t , we need to choose α_t , which is the optimal solution of the problem in Lemma 1. That is,

$$\min_{\alpha \in \Delta(K)} \max_{l \in [0,1]^K} \langle w_1, (\alpha \odot l) \rangle + \langle w_2, l \rangle, \quad (7)$$

where we set that $w = (w_1, w_2)$ and w_1 and w_2 are K -dimensional vectors, respectively. We see that the optimization of this objective function is defined by $l_i = 0$ if $w_{1,i} \cdot \alpha_i + w_{2,i} \leq 0$, otherwise we let $l_i = 1$. Hence we can convert our problem to choose α as

$$\min_{\alpha \in \Delta(K)} \max_{l \in [0,1]^K} \langle w_1, (\alpha \odot l) \rangle + \langle w_2, l \rangle = \min_{\alpha \in \Delta(K)} \sum_{i=1}^K \max \{0, \alpha_i w_{1,i} + w_{2,i}\},$$

which is equivalent to

$$\min_{\alpha \in \Delta(K), \beta \geq 0} \sum_{i=1}^K \beta_i \quad \text{s.t.} \quad \beta_i \geq w_{1,i} \alpha_i + w_{2,i} \quad \forall i = 1, \dots, K.$$

The above problem is a linear program with $O(K)$ variables and $O(K)$ linear constraints. Thus, computing α_t in the problem (7) can be solved in polynomial time.

4.2 Computing subgradients g_t for the ∞ -norm

The second component of Algorithm 2 is the algorithm \mathcal{B} , which computes subgradients $s_t \in \partial h_S(w_t)$. By Proposition 1, we have $s_t = \arg \max_{s \in S} \langle s, w_t \rangle$. Recall that $S = \{(x, y) \in [0, 1]^K \times [0, 1]^K \mid \|x\|_\infty \leq C_\infty^*(y)\}$. In particular, the condition that $\|x\|_\infty \leq C_\infty^*(y)$ can be represented as

$$\max_i x_i \leq \min_{\alpha \in \Delta(K)} \|\alpha \odot y\|_\infty \iff x_i \leq \frac{1}{\sum_{j=1}^K \frac{1}{y_j}}, \forall i.$$

Therefore, the computation of the subgradient s_t is formulated as

$$\max_{x, y \in [0,1]^K} \langle w_1, x \rangle + \langle w_2, y \rangle \quad \text{s.t.} \quad x_i \leq \frac{1}{\sum_{j=1}^K \frac{1}{y_j}}, \quad \forall i = 1, \dots, K. \quad (8)$$

Now we show that there exists an equivalent second order cone programming (SOCP) formulation (e.g., [11]) for this problem.

First we give the definition of the second order cone programming, and then we give a proposition, which states that our optimization problem is equivalent to the second order cone programming.

Definition 4. *The standard form for the second order conic programming (SOCP) model is as follows:*

$$\min_{\mathbf{x}} \langle \mathbf{c}, \mathbf{x} \rangle \text{ s.t. } A\mathbf{x} = \mathbf{b}, \|C_i\mathbf{x} + \mathbf{d}_i\|_2 \leq \mathbf{e}_i^\top \mathbf{x} + \mathbf{f}_i \text{ for } i = 1, \dots, m,$$

where the problem parameters are $\mathbf{c} \in \mathbb{R}^n$, $C_i \in \mathbb{R}^{n_i \times n}$, $\mathbf{d}_i \in \mathbb{R}^{n_i}$, $\mathbf{e} \in \mathbb{R}^n$, $\mathbf{f}_i \in \mathbb{R}$, $A \in \mathbb{R}^{p \times n}$, and $\mathbf{b} \in \mathbb{R}^p$. $\mathbf{x} \in \mathbb{R}^n$ is the optimization variable.

Then we obtain the following proposition.

Proposition 4. $\sum_{i=1}^K \frac{x^2}{y_i} \leq x$, $x \geq 0$ and $y_i \geq 0$ is equivalent to $x^2 \leq y_i z_i$, where $y_i, z_i \geq 0$ and $\sum_{i=1}^K z_i = x$.

Proof. On the direction “ \Rightarrow ”

From $\sum_{i=1}^K \frac{x^2}{y_i} \leq x$ we obtain that $\sum_{i=1}^K \frac{x}{y_i} \leq 1$. By setting

$$z_i = x \cdot \frac{\frac{1}{y_i}}{\sum_i \frac{1}{y_i}},$$

we can have that $x^2 \leq y_i z_i$, and $\sum_{i=1}^K z_i = x$.

On the other direction “ \Leftarrow ” Due to $x^2 \leq y_i z_i$, we have $\frac{x^2}{y_i} \leq z_i$. So we have that

$$\sum_{i=1}^K \frac{x^2}{y_i} \leq \sum_{i=1}^K z_i = x.$$

□

Again in our case we need find to the optimal vector $\mathbf{s} \in S$, which satisfies that $\mathbf{s}_t = \arg \max_{\mathbf{s} \in S} \langle \mathbf{w}_t, \mathbf{s} \rangle$. Then we can reduce our problem in following theorem.

Theorem 5. *The optimization problem (8) can be solved by the second order cone programming.*

Proof. To prove this theorem we only need to represent the original problem (8) as a standard form of the SOCP problem. Note that we only consider the case that $y_i \neq 0$ for all $i = 1, \dots, K$. The case where $y_i = 0$ for some i is trivial. To see this, by definition of S , we know that for all i , $x_i = 0$. Then, the resulting problem is a linear program, which is a special case of the SOCP. Now we assume that $y_i \neq 0$ for $i = 1, \dots, K$. For $x_i \leq \frac{1}{\sum_j \frac{1}{y_j}}$, we multiply x_i on both sides and rearrange the inequality:

$$\sum_{j=1}^K \frac{x_i^2}{y_j} \leq x_i.$$

By Proposition 4, this is equivalent with

$$y_j z_{i,j} \geq x_i^2, \quad y_j, z_{i,j} \geq 0, \quad \sum_{j=1}^K z_{i,j} = x_i.$$

By [11], we may rewrite it as follows: For each i ,

$$x_i^2 \leq y_j z_{i,j}; \quad y_j, z_{i,j} \geq 0 \iff \|(2x_i, y_j - z_{i,j})\|_2 \leq y_j + z_{i,j} \quad \forall j = 1, \dots, K. \quad (9)$$

The above equivalence is trivial. On the other hand, since $x_i \leq \frac{1}{\sum_j \frac{1}{y_j}}$, and $y_i \in [0, 1]$, naturally we have $x_i \in [0, 1]$. So we need only constrain that $y_i \in [0, 1]$. We can apply the fact that if y_i is positive so $|y_i| = y_i$, and if $y_i \leq 1$, so $|y_i| \leq 1$. Therefore we may give a $(K^2 + 2K) \times (K^2 + 2K)$ -matrix C_i in SOCP, and the variable vector is composed as follows:

$$\tilde{\mathbf{x}} = (x_1, \dots, x_K, y_1, \dots, y_K, z_{1,1}, \dots, z_{1,K}, \dots, z_{K,1}, \dots, z_{K,K}), \quad (10)$$

where for $z_{i,j}$, i is corresponding to x_i .

Now we may give the second order cone programming of our target problem as follows:

$$\begin{aligned} & \min_{\tilde{\mathbf{x}}} \langle -(\mathbf{w}_1, \mathbf{w}_2, 0, \dots, 0), \tilde{\mathbf{x}} \rangle \\ & \text{s.t. } \|C_i \tilde{\mathbf{x}}\|_2 \leq \mathbf{e}_i^\top \tilde{\mathbf{x}} + \mathbf{d}_i \quad \forall i = 1, \dots, K^2 + 2K, \\ & A\tilde{\mathbf{x}} = \mathbf{b}. \end{aligned} \quad (11)$$

where C_i , \mathbf{e}_i , A and \mathbf{b} are defined as follows:

Firstly the matrix C for hyperbolic constraints are given as: For a fixed $s \in [K]$, where $[K] = \{1, \dots, K\}$ in matrix C_i , where $i \in [(s-1)K, sK]$ we let $(C_i)_{1,s} = 2$, $(C_i)_{K+i,K+i} = 1$, $(C_i)_{2K+(s-1)K+i, 2K+(s-1)K+i} = -1$, and others are 0. \mathbf{e}_i is defined as $(\mathbf{e}_i)_{K+i} = 1$ and $(\mathbf{e}_i)_{2K+(s-1)K+i} = 1$, others are 0.

Next we need to constrain that y_i is less than 1. For $i \in [K^2, K^2 + K]$ we let that $(C_i)_{K+i,K+i} = 1$ and others are 0. And we let that \mathbf{e}_i is a zero vector and $\mathbf{d}_i = 1$. It means that $\|y_i\| \leq 1$. For $i \in [K^2 + K, K^2 + 2K]$, we set $(C_i)_{K+i,K+i} = 1$, $\mathbf{e}_{K+i} = 1$, and $\mathbf{d}_i = 0$.

At last we need to constrain that $\sum_{j=1}^K z_j = x_i$ in equation 9: Let $A \in \mathbb{R}^{K \times (3K+K^2)}$ for each row vector A_j , where $j \in [K]$, we have that $(A_j)_j = 1$ and $(A_j)_{2K+(j-1)K+m} = -1$, for all $m = 1, \dots, K$. Now the matrix A is composed by the row vectors A_j . and \mathbf{b} is a zero vector. \square

5 Conclusion

In this paper we give a framework for online load balancing problem by reducing it to two OLO problems. Moreover, for online load balancing problem with respect to L_∞ -norm we achieve the best known regret bound in polynomial time. Firstly, we reduce online load balancing with $\|\cdot\|$ norm to a vector payoff game measured by combination norm $\|\cdot\|^\dagger$. Next due to [15] this vector payoff game is reduced to an OCO problem. At last, we can reduce this OCO problem to two independent OLO problems. Especially, for makespan, we give an efficient algorithm, which achieves the best known regret bound $O(\sqrt{T \ln K})$, by processing linear programming and second order cone programming in each trial. Recently

Kwon [10] proposed a similar reduction with other type of induced norm instead of our combination norm.

There are some open problems left in this topic. For instance, an efficient algorithm for online load balancing with respect to general norm or p -norm is still an open problem. Kwon's reduction [10] might be helpful to this discussion. Furthermore, the lower bound of online load balancing is still unknown.

References

1. Abernethy, J., Bartlett, P.L., Hazan, E.: Blackwell approachability and no-regret learning are equivalent. In: Proceedings of the 24th Annual Conference on Learning Theory. pp. 27–46 (2011)
2. Azar, Y.: On-line load balancing. In: Fiat, A., Woeginger, G.J. (eds.) Online Algorithms: The State of the Art, pp. 178–195. Springer Berlin Heidelberg, Berlin, Heidelberg (1998). <https://doi.org/10.1007/BFb0029569>, <https://doi.org/10.1007/BFb0029569>
3. Azar, Y., Kalyanasundaram, B., Plotkin, S., Pruhs, K.R., Waarts, O.: Online load balancing of temporary tasks. In: Workshop on algorithms and data structures. pp. 119–130. Springer (1993)
4. Blackwell, D., et al.: An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics* **6**(1), 1–8 (1956)
5. Crescenzi, P., Gambosi, G., Nicosia, G., Penna, P., Unger, W.: On-line load balancing made simple: Greedy strikes back. *Journal of Discrete Algorithms* **5**(1), 162–175 (2007)
6. Even-Dar, E., Kleinberg, R., Mannor, S., Mansour, Y.: Online learning for global cost functions. In: COLT (2009)
7. Hazan, E.: Introduction to Online Convex Optimization. *Foundations and Trends in Optimization* **2**(3-4), 157–325 (2016), <http://ocobool.cs.princeton.edu/>
8. Hoeven, D., Erven, T., Kotłowski, W.: The many faces of exponential weights in online learning. In: Conference On Learning Theory. pp. 2067–2092 (2018)
9. Kivinen, J., Warmuth, M.K.: Exponentiated gradient versus gradient descent for linear predictors. *information and computation* **132**(1), 1–63 (1997)
10. Kwon, J.: Refined approachability algorithms and application to regret minimization with global costs. arXiv preprint arXiv:2009.03831 (2020)
11. Lobo, M.S., Vandenberghe, L., Boyd, S., Lebret, H.: Applications of second-order cone programming. *Linear algebra and its applications* **284**(1-3), 193–228 (1998)
12. Molinaro, M.: Online and random-order load balancing simultaneously. In: Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 1638–1650. Society for Industrial and Applied Mathematics (2017)
13. Rakhlin, A., Sridharan, K., Tewari, A.: Online learning: Beyond regret. In: Proceedings of the 24th Annual Conference on Learning Theory. pp. 559–594 (2011)
14. Shalev-Shwartz, S.: Online learning and online convex optimization. *Foundations and Trends® in Machine Learning* **4**(2), 107–194 (2012)
15. Shimkin, N.: An online convex optimization approach to blackwell's approachability. *The Journal of Machine Learning Research* **17**(1), 4434–4456 (2016)