Online Learning for Global Cost Functions

Eyal Even-Dar
Google Research.
evendar@google.com

Robert Kleinberg
Cornell University
rdk@cs.cornell.edu

Shie Mannor
Technion and
McGill University
shiemannor@gmail.com

Yishay Mansour
Google Research and
Tel-Aviv University.
mansour@tau.ac.il

Abstract

We consider an online learning setting where at each time step the decision maker has to choose how to distribute the future loss between k alternatives, and then observes the loss of each alternative. Motivated by load balancing and job scheduling, we consider a global cost function (over the losses incurred by each alternative), rather than a summation of the instantaneous losses as done traditionally in online learning. Such global cost functions include the makespan (the maximum over the alternatives) and the L_d norm (over the alternatives). Based on approachability theory, we design an algorithm that guarantees vanishing regret for this setting, where the regret is measured with respect to the best static decision that selects the same distribution over alternatives at every time step. For the special case of makespan cost we devise a simple and efficient algorithm. In contrast, we show that for concave global cost functions, such as L_d norms for $d < 1$, the worst-case average regret does not vanish.

1 Introduction

We consider online learning with a global cost function. Namely, a decision maker who is facing an arbitrary environment is trying to minimize a cost function that depends on the entire history of his choices (hence we term it a global cost function). To motivate the discussion, consider a job scheduling problem with a job that can be distributed to multiple machines and each job incurs a different loss per machine. At each stage the decision maker first selects a distribution over machines and only then observes the losses (loads) of each machine. The loss per machine adds up weighted by the distribution the decision-maker selected. The objective of the decision maker is to have his makespan (i.e., the loss of the worst machine, or the infinity norm of the losses) as *small* as possible. The comparison class the decision maker considers is that of all static allocations and the decision maker wants to minimize the difference between the cost of the loss vector and the cost of the best static allocation.

The online learning (or regret minimization) framework have had many successful applications in machine learning, game theory, and beyond; see [9]. In spite of the advantages of this approach it suffers from some inherent limitations. First, the size of the comparison class affects the magnitude of the regret (although only in a logarithmic way). Second, the approach usually assumes that there is essentially no *state* of the system and that the decision problem is the same at every stage (see [4, 5, 6, 15, 10] for deviations from this standard approach). Finally, there is a tacit assumption that the objective function is additive in the losses over the time steps. This last assumption does not hold in many natural problems such as job scheduling and load balancing settings, for example.

While there are similarities in the motivation between online competitive algorithms [8] and the regret minimization setup, the models have some significant differences. First, the comparison class is substantially different. While competitive analysis allows the optimal scheduler to be "arbitrary," regret minimization limits the size of the comparison class. On the other hand, the competitive analysis approach bounds only the *ratio* of the performances, while the regret minimization bounds the *difference* between them. Finally, there is also a very important difference in the information model. In regret minimization the decision maker first selects an alternative (or a distribution over alternatives) and only then observes the losses, while in the standard online setup the algorithm first observes the loads (losses) and only then selects a machine (alternative).

Our model assumes a finite number of alternatives (machines). The information model is similar to the regret minimization: the decision maker first selects a distribution over machines (alternatives) and only then observes the losses (loads). We have a static comparison class, which includes the algorithms that use the same distribution over the machines in every time step. Our objective function is much in the spirit of the job scheduling, including objective functions such as makespan or L_2-norm. We bound the difference between the performance of our online algorithm and the performance of the best static distribution with respect to a *global* cost function (say makespan). Note that as opposed to many regret minimization works, we consider not only the best alternative, but rather the best distribution over alternatives, so the comparison class in our setup is infinite.[1]

[1] It is instructive to discuss the difference between minimizing an additive loss (as in the best experts) and minimizing the makespan (the alternative with the maximum loss). Consider two alternatives

Our contributions include the following:

- We present the online learning model for a global cost. This model is natural in the online learning setup and it has not been studied to the best of our knowledge.

- We present an algorithm for global cost functions (such as makespan, or any L_d-norm, for $d > 1$) that minimizes the difference between the cost of our online algorithm and the best static distribution in a rate of $O(\sqrt{k/T})$, where T is the number of time steps. Our algorithm only requires that the objective function is convex and that the optimal cost is a concave function (with some additional assumptions about their derivatives). Our general algorithm guaranteeing a vanishing average regret is based on approachability theory. In spite of its good rates, the explicit computation of the regret minimizing strategy does not seem to be easy.

- For the important special case of makespan cost function we provide a specialized simple deterministic online algorithm, which is both computationally efficient and has a regret bound of $O(\log(k)/\sqrt{T})$, where k is the number of different alternatives. Our algorithm is based on a recursive construction where we first solve the case of two alternatives and then provide a recursive construction for any number of alternatives. Our recursive construction maintains the computational efficiency of the two alternative algorithm, and its regret depends only logarithmically on the number of alternatives.

- We complement our algorithms with an impossibility result. We show that one cannot hope to have a similar result for *any* global cost function. Specifically, we show for a wide class of concave function (including any L_d-norm, for $0 < d < 1$) that any online algorithm would have $\Omega(1)$ regret. More specifically, we show that the ratio of the online cost to the best distribution cost is bounded away from one.

Related Work: Unfortunately, we can not give a comprehensive review of the large body of research on regret minimization algorithm, and we refer the interested reader to [9]. In the following we highlight a few more relevant research directions.

There has been an ongoing interest in extending the basic comparison class for the regret minimization algorithm, for example by introducing *shifting experts* [11] *time selection functions* [7] and *wide range regret* [14]. Still, all those works assume that the loss is additive between time steps.

A different research direction has been to improve the computational complexity of the regret minimization algorithms, especially in the case that the comparison class is exponential in size. General computationally efficient transformation where given by [13], where the cost function is linear and the optimization oracle can be computed in polynomial time, and extended by [12], to the case where we are given only an approximate-optimization oracle.

There has been a sequence of works establishing the connection between online competitive algorithms [8] and online learning algorithm [9]. One issue is that online learning algorithms are stateless, while many of the problems address in the competitive analysis literature have a state (see, [4]). For many problems one can use the online learning algorithms and guarantee a near-optimal static solution, however a straightforward application requires both exponential time and space. Computationally efficient solutions have been given to specific problems including, paging [5], data-structures [6], and routing [2, 1].

In contrast to our work, all the above works concentrate on the case where the global cost function is additive between time steps.

2 Model

We consider an online learning setup where a decision maker has a finite set $K = \{1, \ldots, k\}$ of k alternatives (machines) to choose from. At each time step t, the algorithm A selects a distribution $\alpha_t^A \in \Delta(K)$ over the set of alternatives K, where $\Delta(K)$ is the set of distributions over K. Following that the adversary selects a vector of losses (loads) $\ell_t \in [0,1]^k$. The average loss of alternative i is denoted by $L_T(i) = \frac{1}{T}\sum_{t=1}^T \ell_t(i)$. The average loss of the online algorithm A on alternative i is $L_T^A(i) = \frac{1}{T}\sum_{t=1}^T \alpha_t^A(i)\ell_t(i)$ and its average loss vector is $L_T^A = (L_T^A(1), \ldots, L_T^A(k))$. Let the loss vector of a static allocation $\alpha \in \Delta(K)$ be $L_T^\alpha = \alpha \odot L_T$ where $x \odot y = (x(1)y(1), \ldots, x(k)y(k))$.

The objective of the online algorithm A is to minimize a *global cost function* $C(L_T^A)$. In traditional online learning the global cost C is assumed to be an additive function, i.e., $C(L_T^A) = \sum_{i=1}^k L_T^A(i)$. The main focus of this work is to consider alternative cost functions which arise in many natural problems such as job scheduling. More specifically we consider the following norms: the *makespan*, $C_\infty(L_T^A) = \max_{i=1}^k L_T^A(i)$, and the d-norm cost, $C_d(L_T^A) = (\sum_{i=1}^k (L_T^A(i))^d)^{1/d}$ for $d > 0$. Note that for $d > 1$ the d-norm cost is convex while for $d < 1$ it is concave.

Our comparison class is the class of static allocations for $\alpha \in \Delta(K)$. We define the *optimal cost function* $C^*(L_T)$ as the minimum over $\alpha \in \Delta(K)$ of $C(L_T^\alpha)$. We denote by $\alpha_C^*(L_T)$ a minimizing $\alpha \in \Delta(K)$ and called the *optimal static allocation*,[2] i.e.,

$$\begin{aligned} C^*(L_T) &= \min_{\alpha\in\Delta(K)} C(L_T^\alpha) = \min_{\alpha\in\Delta(K)} C(\alpha \odot L_T) \\ &= C(\alpha_C^*(L_T) \odot L_T). \end{aligned}$$

We denote by C_d^* and C_∞^* the global optimal cost function C^* of the d-norm cost and the makespan, respectively.

The regret of algorithm A at time T is defined as,

$$R_T(A) = C(L_T^A) - C^*(L_T).$$

Given a subset of the time steps $\mathcal{B}$ we denote by $L_\mathcal{B} = \frac{1}{T}\sum_{t\in\mathcal{B}} \ell_t$ the average losses during the time steps in $\mathcal{B}$, and

and a sequence of T identical losses of $(2, 1)$. Minimizing the additive loss would always select alternative 2, and have cost T. Minimizing the makespan would select a distribution $(1/3, 2/3)$ and would have cost $(2/3)T$.

[2] When the minimum is not unique we assume it can be selected according to some predefined order.

for an algorithm A we denote by $L^A_{\mathcal{B}}(i) = \frac{1}{T}\sum_{t\in\mathcal{B}} \alpha^A_t(i)\ell_t(i)$ its average loss during $\mathcal{B}$ on alternative i, and by $L^A_{\mathcal{B}} = (L^A_{\mathcal{B}}(1), \ldots, L^A_{\mathcal{B}}(k))$ the average loss of A during $\mathcal{B}$.

3 Preliminaries

For the most part of the paper we will assume that the cost function, C, is a convex function while C^* is a concave function. The following is immediate from the definition of the norms.

Lemma 1 *The d-norm global cost function C_d, for $d > 1$, and the makespan global cost function C_∞ are convex functions.*

The challenging part is to show that the optimal cost function C^* is a concave function. In Appendix A we prove this for d-norm (Lemma 21) and the makespan (Lemma 22).

Lemma 2 *The d-norm global optimal cost function C^*_d, for $d > 1$, and the makespan global optimal cost function C^*_∞ are concave functions.*

In Appendix A we derive for both the d-norm and the makespan their optimal cost function and optimal static allocation (Lemmas 20 and 23).

Lemma 3 *For the d-norm global optimal cost function we have $\alpha^*_{C_d} = \left(\frac{1/L_T(i)^{\frac{d}{d-1}}}{\sum_{j=1}^k 1/L_T(j)^{\frac{d}{d-1}}}\right)_{i\in K}$ and $C^*_d(L) = \left(\frac{1}{\sum_{j=1}^k 1/L_j^{\frac{d}{d-1}}}\right)^{\frac{d-1}{d}}$, and for the makespan global optimal cost function we have $\alpha^*_{C_\infty} = \left(\frac{1/L_T(i)}{\sum_{i=1}^k 1/L_T(j)}\right)_{i\in K}$ and $C^*_\infty(\ell) = \left(\frac{1}{\sum_{j=1}^k 1/L_T(j)}\right)$.*

Denote by D_C be the Lipschitz constant of the global cost function C, by $D^i_{\alpha,C}$ the Lipschitz constant of $x(i)a(i)$, where $a = \alpha^*_C(x)$, and $D_{\alpha,C} = \max_{i\in K} D^i_{\alpha,C}$. (All the Lipschitz constants are under the 1-norm.) In Appendix A we bound those Lipschitz constants by 1 for the d-norm and the makespan (Lemma 4).

Lemma 4 *For the d-norm global optimal cost function C_d, for $d > 1$, the Lipschitz constants $D_{C_d} = 1$ and $D_{\alpha,C_d} = 1$, and for the makespan global optimal cost function C_∞ the Lipschitz constants $D_{C_\infty} = 1$ and $D_{\alpha,C_\infty} = 1$.*

We now prove that given any partition of the losses to subsets, if we bound the regret in each subset, then the sum of the subsets' regrets will bound the regret of the online algorithm.

Lemma 5 *Suppose that C is convex and C^* is concave. Let $\mathcal{T}_i$ be a partition of the T time steps to m sets, and let ON be an online algorithm such that for each $\mathcal{T}_i$ we have $C(L^{ON}_{\mathcal{T}_i}) - C^*(L_{\mathcal{T}_i}) \le R_i$. Then*

$$C(L^{ON}_T) - C^*(L_T) \le \sum_{i=1}^m R_i$$

Proof: From the assumption in the theorem we have that $\sum_{i=1}^m C(L^{ON}_{\mathcal{T}_i}) - \sum_{i=1}^m C^*(L_{\mathcal{T}_i}) \le \sum_{i=1}^m R_i$ Since $\mathcal{T}_i$ is a partition, $L_T = \sum_{i=1}^m L_{\mathcal{T}_i}$ and $L^{ON}_T = \sum_{i=1}^m L^{ON}_{\mathcal{T}_i}$. Since C^* is concave, $C^*(L_T) \ge \sum_{i=1}^m C^*(L_{\mathcal{T}_i})$, and since C is convex, $C(L^{ON}_T) \le \sum_{i=1}^m C(L^{ON}_{\mathcal{T}_i})$. Combining the three inequalities derive the theorem. ∎

The importance of Lemma 5 is the following. If we are able to partition the time steps, in retrospect, and bound the regret in each part of the partition, then the above theorem states that the overall regret is bounded by the sum of the local regret.

Remark: Our model is essentially fractional, where the decision maker selects a distribution, and each alternative receives exactly its fraction according to the distribution. However, in a randomized model we can simulate our fractional model, and use its distributions to run a randomized algorithm. Using standard concentration bounds, the additional loss per action, due to the randomization, is at most $O(\sqrt{(\log k)/T})$, with high probability. Together with the fact that C is Lipschitz we would obtain similar results.

4 Low regret approachability-based algorithm

In this section we use Blackwell's approachability theory to construct a strategy that has zero asymptotic regret. Recall that according to approachability, one considers a vector-valued repeated game. The main question that is considered is if the decision maker can guarantee that the average (vector-valued) reward can be made asymptotically close (in norm sense) to some desired target set. We provide the essential background and relevant results from approachability theory in Appendix B. This section starts with construction of the game followed by a definition of the target set and the application of approachability theory to prove that the set is approachable. We finally describe explicitly a regret minimizing strategy based on approachability theory.

Consider a repeated game against Nature where the decision maker chooses at every stage an action $\alpha \in \Delta(K)$ and Nature chooses an action $\ell \in [0,1]^k$. The vector valued reward obtained by the decision maker is a $2k$ dimensional vector whose first k coordinates are $\alpha \odot \ell$ and the last k coordinates are ℓ. Let us denote this immediate vector-valued reward by $m_t = (\alpha_t \odot \ell_t, \ell_t)$. It easily follows that the average reward $\hat{m}_T = (\sum_{t=1}^T m_t)/T = (L^A_T, L_T)$. We are now ready to define the target set in this game.

$$S = \{(x,y) \in \mathbb{R}^k \times \mathbb{R}^k : x, y \ge 0; C(x) \le C^*(y)\}. \quad (1)$$

Note that S is a set in $\mathbb{R}^{2k}$. We will show below that S is approachable (see Appendix B). We first claim that S is convex.

Lemma 6 *Suppose that C is convex and C^* is concave. Then S is convex.*

Proof: Suppose that $(x^i, y^i) \in S$ for $i = 1,2$ and fix $\beta \in [0,1]$ (we let x^i and y^i denote vectors in $\mathbb{R}^k$). Since C is convex we have that $C(\beta x^1 + (1-\beta)x^2) \le \beta C(x^1) + (1-\beta)C(x^2)$. Since $(x^i, y^i) \in S$ for $i = 1,2$ we have that $C(x^i) \le C^*(y^i)$ for $i = 1,2$. So that $\beta C(x^1) + (1-$

$\beta)C(x^2) \leq \beta C^*(y^1) + (1-\beta)C^*(y^2)$. Using the fact that C^* is concave we obtain that $\beta C^*(y^1) + (1-\beta)C^*(y^2) \leq C^*(\beta y^1 + (1-\beta)y^2)$. Combining the above inequalities we conclude that $C(\beta x^1 + (1-\beta)x^2) \leq C^*(\beta y^1 + (1-\beta)y^2)$, completing the proof. ■

We are going to use Blackwell's approachability theory. We note that in standard approachability it is usually assumed that the set of available actions is finite. In our game, both players can choose from compact action spaces. Namely, the decision maker's action is $\alpha \in \Delta(K)$ and Nature's action is $\ell \in [0,1]^k$. Let $p \in \Delta(\Delta(K))$ denote a probability distribution over α and let $q \in \Delta([0,1]^k)$ denote a probability distribution over ℓ. (See Appendix B for further discussion.) We can define the expected vector-valued reward as:

$$m(p,q) = \int_{\alpha \in \Delta(K)} \int_{\ell \in [0,1]^k} p_\alpha q_\ell (\alpha \odot \ell, \ell).$$

Theorem 7 *The set S is approachable.*

Proof: According to Blackwell's theorem for convex sets (Theorem 26 in Appendix B) it suffices to prove that for every $q \in \Delta([0,1]^k)$ there exists a $p \in \Delta(\Delta(K))$ such that $m(p,q) \in S$. Indeed, consider some $q \in \Delta([0,1]^k)$ and let $\overline{L}(q)$ denote the mean of ℓ under q (that is $\overline{L}(q) = \int q_\ell \ell$), and let p be a singleton whose mass is centered at $\alpha_c^*(\overline{L}(q))$. Then $m(p,q) = (\alpha_C^*(\overline{L}(q)) \odot \overline{L}(q), \overline{L}(q))$ and is in S by definition. ■

Since S is approachable let us explain how to approach it in a constructive way. We follow Blackwell's first approachability theorem specialized for convex sets (Theorem 25). Given the current vector valued reward $\hat{m}_t = (L_t^A, L_t)$ we first compute the direction into the set S. For that, we need to solve the following convex optimization problem:

$$\begin{aligned} &\min_{x,y \in \mathbb{R}^k} \|(L_t^A, L_t) - (x,y)\|_2 \qquad (2)\\ &s.t.\ C(x) \leq C^*(y)\\ &\qquad x \geq 0. \end{aligned}$$

According to Blackwell's approaching strategy (see Appendix B), if the solution to (2) is 0, i.e., $\hat{m}_t \in S$ the decision maker can act arbitrarily at time t. If not, then let $u = (x,y) - \hat{m}_t$ where (x,y) are the minimizers of (2), and let us write $u = (u_\alpha, u_\ell)$ to denote the two k dimensional components of u. We are looking for a mixed action $p \in \Delta(\Delta(K))$ that guarantees that the expected reward at time $t+1$ is on the other side of the supporting hyperplane at (x,y). We know that such a mixed action exists since S is approachable and convex (Theorem 25). The optimization problem we try to solve is therefore:

$$\sup_{p \in \Delta(\Delta(K))} \inf_{q \in \Delta([0,1]^k)} \int_\alpha \int_\ell p_\alpha q_\ell \left(u_\alpha^\top (\alpha \odot \ell) + u_\ell^\top \ell\right) \quad (3)$$

$$= \sup_{p \in \Delta(\Delta(K))} \min_{\ell \in [0,1]^k} \int_\alpha p_\alpha u_\alpha^\top (\alpha \odot \ell) + u_\ell^\top \ell \quad (4)$$

$$= \sup_{\alpha \in \Delta(K)} \min_{\ell \in [0,1]^k} u_\alpha^\top (\alpha \odot \ell) + u_\ell^\top \ell \quad (5)$$

$$= \max_{\alpha \in \Delta(K)} \min_{\ell \in [0,1]^k} u_\alpha^\top (\alpha \odot \ell) + u_\ell^\top \ell, \quad (6)$$

The DIFF Algorithm:
At time $t = 1$ we have $p_1(1) = p_1(2) = 1/2$,
at time $t \geq 2$ we have,
$p_{t+1}(1) = p_t(1) + \frac{p_t(2)\ell_t(2) - p_t(1)\ell_t(1)}{\sqrt{T}}$,
and $p_t(2) = 1 - p_t(1)$.

Figure 1: Algorithm for two alternatives.

where (3) is the problem that we are trying to solve (in the lifted space); Eq. (4) holds since the dependence in ℓ is linear in q so that for any p there is a singleton q which obtains the minimum; Eq. (5) holds since p only affects Eq. (4) linearly through α so that any measure over $\Delta(\Delta(K))$ can be replaced with its mean; and Eq. (6) holds because of continuity and the compactness of the sets we optimize on. Now, the optimization problem of Eq. (6), is a maximization of a convex function or alternatively, the solution of a zero-sum game with k actions for one player and 2^k for the other player and can be efficiently solved (for a fixed k). We summarize the above results in the following Theorem.

Theorem 8 *The following strategy guarantees that the regret goes to 0.*

1. *At every stage t solve (2).*
2. *If the solution is 0, play an arbitrary action.*
3. *If the solution is more than 0, compute $u = (x,y) - \hat{m}_t$ where (x,y) are the minimizers of (2) and solve (6). Play a maximizing α.*

Furthermore, the rate of convergence of the expected regret is $O(\sqrt{k/T})$.

Proof: This is an approaching strategy to S. The only missing part is to argue that approaching S implies minimizing the regret. This is true by construction and since C and C^* are Lipschitz. As for the rate of convergence for approachability: see the discussion in Appendix B. ■.

5 Algorithms for Makespan

In this section we present algorithms for the makespan global cost function. The main benefit of the algorithms is that they are simple to implement, computationally efficient, and have an improved regret bound. For k alternatives the average regret vanishes at the rate of $O((\log k)T^{-1/2})$.

The construction of the algorithms is done in two parts. The first part is to show an algorithm for the basic case of two alternatives (Section 5.1). The second part is a recursive construction, that builds an algorithm for 2^r alternatives from two algorithms for 2^{r-1} alternatives, and one algorithm for two alternatives. The recursive construction essentially builds a complete binary tree, the main issue in the construction is to define the losses that each algorithm observes (Section 5.2).

5.1 Two Alternatives

The DIFF Algorithm: We denote the distribution of DIFF at time t by $\alpha_t^D = (p_t(1), p_t(2))$. Algorithm DIFF starts with

$p_1(1) = p_1(2) = 1/2$, and the update of the distribution of DIFF is done as follows,

$$p_{t+1}(1) = p_t(1) + \frac{p_t(2)\ell_t(2) - p_t(1)\ell_t(1)}{\sqrt{T}},$$

and by definition $p_t(2) = 1 - p_t(1)$. Note that we are guaranteed that $p_t(1) \in [0,1]$, since $p_t(1) \geq p_{t-1}(1)(1 - \frac{1}{\sqrt{T}})$.

Before we dive into the proof, we would like to give some high level intuition. Assume that in a certain subset of the time steps $\mathcal{T}$ the algorithm uses approximately the same probabilities, say $(\rho, 1-\rho)$and that the load on the two alternative increased by exactly the same amount. This will imply that $\rho L_{\mathcal{T}}(1) = (1-\rho)L_{\mathcal{T}}(2)$, so it has to be the case that $\rho = \frac{L_{\mathcal{T}}(2)}{L_{\mathcal{T}}(1)+L_{\mathcal{T}}(2)}$. Note that this is the optimal static allocation for time steps $\mathcal{T}$. Now, using Lemma 5, if we partition the time steps to such $\mathcal{T}$'s, we will have low regret (originating mainly from the fact that we are using approximately the same distribution ρ).

The proof is done through a sequence of modification of the input sequence, where the objective is to guarantee a partition of the time steps to $\mathcal{T}$'s with the desired properties. First, we will add $O(\sqrt{T})$ losses to get the DIFF algorithm to end in its initial state, where the difference between the alternative is 0 (See Claim 9.) Second, we consider a partition $[0,1]$ to intervals of size $\Theta(T^{-1/2})$ each. We modify the sequence of losses to make sure that for each update, DIFF distribution before and after the update are in one interval, possibly by splitting a single update to two updates. (See Claim 11.) We show that the optimal action for each sub-sequence is close to the actions performed in the interval (recall that any two actions in the intervals can differ by at most $\Theta(T^{-1/2})$. (See Claim 12.) Combining the steps, we can partition the time steps to sub-sequences $\mathcal{T}$'s, according to $p_t(1)$, and during each $\mathcal{T}$ the increase in the online load on the alternatives is identical, and therefore near optimal. We complete the proof by summing over the subsequences. (See Claim 13 and Theorem 14.)

We would abuse notation and given a sequence of losses ℓ we let $L^D_t(\ell, i) = \sum_{\tau=1}^t p_\tau(i)\ell_\tau(i)$ be the aggregate loss of alternative i, after t steps using the $DIFF$ algorithm, and $L^D_t(\ell)$ be the aggregate loss vector. Let $\Lambda_t = L^D_t(\ell,2) - L^D_t(\ell,1)$. Equivalently, algorithm DIFF sets $p_t(1) = (1/2) + \Lambda_t/T^{1/2}$. The proof will be done by a sequence of transformation of the sequences of losses ℓ, where for each transformation we would bound the effect on the online $DIFF$ algorithm and the optimal static assignment $\alpha^*_{C_\infty}(\ell)$.

Given the sequence of losses ℓ we extend the sequence by m additional losses to a loss sequence $\hat{\ell}$ such that the $DIFF$ algorithm would end with two perfectly balanced losses, i.e., $\Lambda_{T+m} = 0$. Specifically, assume that $\Lambda_T > 0$, then we add losses of the form $(1,0)$. After each such loss we have in the $DIFF$ algorithm that $p_{t+1}(1) = p_t(1)(1 - \frac{1}{T^{1/2}})$. This implies that for some $m \leq 2T^{1/2}$ we have $p_{T+m}(1) = 1/2$, where the last loss might be of the form $(\eta, 0)$ for some $\eta \in [0,1]$. The case of $\Lambda_T < 0$ is similar, and we add losses of the form $(0,1)$.

Claim 9 *The following holds for $\hat{\ell}$:*

1. $L^D_{T+k}(\hat{\ell},1) = L^D_{T+k}(\hat{\ell},2)$,
2. $C_\infty(L^D_{T+m}(\hat{\ell})) = C_\infty(L^D_T(\ell))$, *and*
3. $C^*_\infty(\hat{\ell}) \leq C^*_\infty(\ell) + m$ *where* $m \leq 2T^{1/2}$.

As a by product of our proof, we have established the following claim, that the loss on the two alternatives is similar.

Claim 10 $\Lambda_T \leq 2\sqrt{T}$.

Consider a grid $\mathcal{G}$ which includes $0.1\,T^{1/2}$ points $z_i = 10T^{-1/2}$. We say that the update crosses the grid at point z_i at time t if either $p_{t-1}(1) < z_i < p_t(1)$ or $p_{t-1}(1) > z_i > p_t(1)$.

We define bin BIN_i to be $[z_i, z_{i+1}]$. Let $\mathcal{T}_i$ be the set of time steps that either start or end in bin BIN_i, i.e., $\mathcal{T}_i = \{t : p_t(1) \in BIN_i$ or $p_{t-1}(1) \in BIN_i\}$. (Each time step can be in at most two consecutive bins, since $|p_t(1) - p_{t-1}(1)| \leq T^{-1/2}$.) We will modify the loss sequence $\hat{\ell}$ such that in each bin, all the updates start and end inside the bin. Let $\hat{b}$ be the following sequence of losses and probabilities. If at time t we have that both $p_t(1)$ and $p_{t-1}(1)$ are contained in one bin, then we set $\hat{b}_{2t-1} = (\ell_t, p_t)$ and $\hat{b}_{2t} = ((0,0), p_t)$. Otherwise, $p_t(1)$ and $p_{t-1}(1)$ are contained in two consecutive bins, say BIN_i and BIN_{i+1}. We add to $\hat{b}$ two elements, $\hat{b}_{2t-1} = (\lambda\ell_t, p_t)$ and $\hat{b}_{2t} = ((1-\lambda)\ell_t, p_t)$, such that $p_{t-1}(1) + \lambda(p_t(2)\ell_t(2) - p_t(1)\ell_t(1))/T^{1/2} = z_{i+1}$. Since we replace each entry in $\hat{\ell}$ by two entries in $\hat{b}$, this implies that the length of $\hat{b}$ is at most $2(T + T^{1/2}) \leq 4T$.

Given a subset S of the indices of the sequence $\hat{b} = \{(w_t, q_t)\}$ we define $L^D_S(\hat{b}, a) = \sum_{t\in S} w_t(a)q_t(a)$, $L^D_S(\hat{b})$ be the aggregate loss vector, $L_S(\hat{b}, a) = \sum_{t\in S} w_t(a)$, for $a \in \{1,2\}$. Let $\mathcal{A}$ include all the indices in the sequence $\hat{b}$. Clearly, $L^D_{\mathcal{A}}(\hat{b}, a) = L^D_{T+m}(\hat{\ell}, a)$ and $L_{\mathcal{A}}(\hat{b}, a) = \sum_{t=1}^{T+m} \hat{\ell}_t(a) = L_{T+m}(\hat{\ell}, a)$.

Claim 11 $C_\infty(L^D_{\mathcal{A}}(\hat{b})) = C_\infty(L^D_{\mathcal{A}}(\hat{\ell}))$ *and* $C^*_\infty(\hat{b}) = C^*_\infty(\hat{\ell})$.

Now we can define a partition of the time steps in $\hat{b}$. Let $\delta_t = (w_t(2)q_t(2) - w_t(1)q_t(1))/\sqrt{T}$. (Note that for updates which do not cut a boundary we have $\delta_{2t-1} = p_{t+1}(1) - p_t(1)$, while for updates that cut the boundary we have $\delta_{2t-1} + \delta_{2t} = p_{t+1}(1) - p_t(1)$. Let $v_t = 1/2 + \sum_{i=1}^t \delta_t$. We say that update t in $\hat{b}$ belongs to bin BIN_i if $v_t, v_{t-1} \in [z_{i-1}, z_i]$. (Note that by construction we have that every update belongs to a unique bin.) Let $\mathcal{T}_i$ be all the time points in $\hat{b}$ that belong to BIN_i. (Note that $\sum_{t\in\mathcal{T}_i} \delta_t = 0$, i.e. the increased load for the two alternatives in DIFF is the same.) Since $\{\mathcal{T}_i\}$ is a partition of the indexes of $\hat{b}$, by Lemma 5 we can lower bound the optimal cost by

$$C_\infty(L^D_{2(T+m)}(\hat{b})) - C^*_\infty(\hat{b}) \leq \sum_{i=1}^{0.1T^{1/2}} C_\infty(L^D_{\mathcal{T}_i}(\hat{b})) - C^*_\infty(\hat{b}_{\mathcal{T}_i})$$

Our main goal is to bound the difference

$$C_\infty(L^D_{\mathcal{T}_i}(\hat{b})) - C^*_\infty(\hat{b}_{\mathcal{T}_i}).$$

Given the losses at time steps $\mathcal{T}_i$ of $\hat{b}$, we denote static optimal allocation, $\alpha^*_{C_\infty}(\hat{b}_{\mathcal{T}_i})$, by $(q^*_i, 1-q^*_i)$.

Lemma 12 $|q_i^* - z_i| \leq 30T^{-1/2}$.

Proof: For the updates in $\mathcal{T}_i$ we have that $\sum_{t\in\mathcal{T}_i} \delta_t = 0$. Consider the function,

$$f(x) = \sum_{t\in\mathcal{T}_i} x\, w_t(1) - \sum_{t\in\mathcal{T}_i}(1-x)w_t(2) = x(L_{\mathcal{T}_i}(1) + L_{\mathcal{T}_i}(2)) - L_{\mathcal{T}_i}(2).$$

First, for q_i^* we have that $f(q_i^*) = 0$. Second, since for any $t \in \mathcal{T}_i$ we have that $p_t(1) \in [z_{i-1}, z_{i+2}]$ then $f(z_{i-1}) \leq \sum_{t\in\mathcal{T}_i} \delta^t = 0 \leq f(z_{i+2})$. Therefore $q_i^* \in [z_{i-1}, z_{i+2}]$. Since, $z_{i+2} - z_{i-1} = 30T^{-1/2}$, we have that $|z_i - q_i^*| \leq 30T^{-1/2}$. ■

Lemma 13

$$\sum_{i=1}^{0.1T^{1/2}} C_\infty(L^D_{\mathcal{T}_i}(\hat{b})) - C^*_\infty(\hat{b}_{\mathcal{T}_i}) \leq 240T^{1/2}$$

Proof: By Lemma 12 for any $t \in \mathcal{T}_i$ we have that $|p_t(1) - q_i^*| \leq 30T^{-1/2}$. This implies that,

$$C_\infty(L^D_{\mathcal{T}_i}(\hat{b})) - C^*_\infty(\hat{b}_{\mathcal{T}_i}) \leq 30T^{-1/2}\max\{L_{\mathcal{T}_i}(\hat{b},1), L_{\mathcal{T}_i}(\hat{b},2)\}$$

and therefore

$$\sum_{i=1}^{0.1T^{1/2}} C_\infty(L^D_{\mathcal{T}_i}(\hat{b})) - C^*_\infty(\hat{b}_{\mathcal{T}_i}) \leq$$

$$30T^{-1/2}\sum_{i=1}^{0.1T^{1/2}} \max\{L_{\mathcal{T}_i}(\hat{b},1), L_{\mathcal{T}_i}(\hat{b},2)\} \leq$$

$$30T^{-1/2}\cdot 2(4T) = 240T^{1/2}.$$

■

Theorem 14 *For any loss sequence ℓ, we have that* $C_\infty(L^D(\ell)) - C^*_\infty(\ell) \leq 241T^{1/2}$.

Proof: Given the loss sequence ℓ, we define the loss sequence $\hat{\ell}$. By Claim 9 we have that $C_\infty(L^D_{T+m}(\hat{\ell})) \geq C_\infty(L^D_T(\ell))$ and $C^*_\infty(\hat{\ell}) \leq C^*_\infty(\ell) + T^{1/2}$. By Claim 11 we have that $C^*_\infty(\hat{\ell}) = C^*_\infty(\hat{b})$, and $C_\infty(L^D_{T+m}(\hat{\ell})) = C_\infty(L^D_{2(T+m)}(\hat{b}))$. From Lemma 13 we have that $\sum_{i=1}^{0.1T^{1/2}} C_\infty(L^D_{\mathcal{T}_i}(\hat{b})) - C^*_\infty(\mathcal{T}_i) \leq 240T^{1/2}$. By Lemma 5 and Lemma 23, we can bound the sum by the global function, hence we have that $C_\infty(L^D_{2(T+m)}(\hat{b})) - C^*_\infty(\hat{b}) \leq 240T^{1/2}$. Therefore, $C_\infty(L^D_T(\ell)) - C^*_\infty(\ell) \leq 241T^{1/2}$. ■

5.2 Multiple alternatives

In this section we show how to transform a low regret algorithm for two alternatives to one that can handle multiple alternatives. For the base case we assume that we are given an online algorithm A_2, such that for any loss sequence $\ell_1,\ldots,\ell_T$ over two alternatives, guarantees that its average regret is at most $\alpha/\sqrt{T}$. Using A_2 we will build a sequence of algorithms A_{2^r}, such that for any loss sequence $\ell_1,\ldots,\ell_T$ over 2^r alternatives, A_{2^r} guarantees that its average regret is at most $O(\alpha\, r/\sqrt{T})$.

> **The MULTI Algorithm $A^K_{2^r}$:**
> *Alternative set:* K of size 2^r partitioned to I and J, $|I| = |J| = 2^{r-1}$.
> *Procedures:* $A_2, A^I_{2^{r-1}}, A^J_{2^{r-1}}$.
> *Action selection:*
> Algorithm $A^I_{2^{r-1}}$ gives $x_t \in \Delta(I)$.
> Algorithm $A^J_{2^{r-1}}$ gives $x_t \in \Delta(J)$.
> Algorithm A_2 gives $(z_t, 1-z_t)$.
> Output $\alpha_t \in \Delta(I\cup J)$ where,
> for $i \in I$ set $\alpha_t(i) = x_t(i)z_t$, and
> for $j \in J$ set $\alpha_t(j) = y_t(j)(1-z_t)$.
> *Loss distribution:*
> Given the loss ℓ_t.
> Algorithm $A^I_{2^{r-1}}$ receives $\ell_t(I)$.
> Algorithm $A^J_{2^{r-1}}$ receives $\ell_t(J)$.
> Algorithm A_2 receives $(\omega_t(I), \omega_t(J))$,
> where $\omega_t(I) = 2^{-r+1}\sum_{i\in I}\ell_t(i)x_t(i)$,
> and $\omega_t(J) = 2^{-r+1}\sum_{j\in J}\ell_t(j)y_t(j)$.

Figure 2: Algorithm for multiple alternatives.

Our basic step in the construction is to build an algorithm A_{2^r} using two instances of $A_{2^{r-1}}$ and one instance of A_2. A_{2^r} would work as follows. We partition the set of actions K to two subsets of size 2^{r-1}, denoted by I and J; for each subset we create a copy of $A_{2^{r-1}}$ which we denote by $A^I_{2^{r-1}}$ and $A^J_{2^{r-1}}$. The third instance A^M_2 would receive as an input the average loss of $A^I_{2^{r-1}}$ and $A^J_{2^{r-1}}$. Let us be more precise about the construction of A_{2^r}.

At each time step t, A_{2^r} receives a distribution $x_t \in \Delta(I)$ from $A^I_{2^{r-1}}$, a distribution $y_t \in \Delta(J)$ from $A^J_{2^{r-1}}$ and probability $z_t \in [0,1]$ from A^M_2. The distribution of A_{2^r} at time t, $\alpha_t \in \Delta(I\cup J)$, is defined as follows. For actions $i \in I$ we set $\alpha_t(i) = x_t(i)z_t$, and for actions $j \in J$ we set $\alpha_t(j) = y_t(j)(1-z_t)$. Given the action α_t, A_{2^r} observes a loss vector ℓ_t. It then provides $A^I_{2^{r-1}}$ with the loss vector $(\ell_t(i))_{i\in I}$, $A^J_{2^{r-1}}$ with the loss vector $(\ell_t(j))_{j\in J}$, and A^M_2 with the loss vector $(\omega_t(I), \omega_t(J))$, where $\omega_t(I) = 2^{-r+1}\sum_{i\in I}\ell_t(i)x_t(i)$ and $\omega_t(J) = 2^{-r+1}\sum_{j\in J}\ell_t(j)y_t(j)$.

The tricky part in the analysis is to relate the sequences x^t, y^t and z^t to the actual aggregate loss of the individual alternatives. We will do this in two steps. First we will bound the difference between the average loss on the alternatives, and the optimal solution. Then we will show that the losses on the various alternatives is approximately balanced, bounding the difference between the maximum and the minimum loss.

The input to algorithm A^M_2 on its first alternative (generated by $A^I_{2^{r-1}}$) has an average loss

$$W(I) = \frac{1}{T}\sum_t \omega_t(I) = \frac{1}{T}\left[\sum_t 2^{-r+1}\sum_{i\in I} x_t(i)\ell_t(i)\right],$$

similarly, on the second alternative it has an average loss

$$W(J) \quad = \quad \frac{1}{T}\sum_t \omega_t(J) = \frac{1}{T}\left[\sum_t 2^{-r+1}\sum_{j\in J} y_t(j)\ell_t(j)\right]$$

First, we derive the following technical lemma (its proof is in Appendix C).

Lemma 15 *Assume that A_2^M is at hight r (part of $A_{2^r}^{I\cup J}$). Then*

$$W(I\cup J) \le \frac{1}{\frac{1}{W(I)}+\frac{1}{W(J)}} + \frac{\alpha}{\sqrt{T}} \le \frac{1}{\sum_{a\in I\cup J}\frac{1}{L_T(a)}} + \frac{r\alpha}{\sqrt{T}}.$$

The following claim, based on Claim 10, bounds the difference between the average of all alternatives and the loss of a specific alternative.

Claim 16 *For any alternative $i \in K$ we have, $|W(K) - L_T^{A_{2^r}}(i)| \le \frac{r}{\sqrt{T}}$.*

Proof: We can view the algorithm A_{2^r} as generating a complete binary tree. Consider a path from the root to a node that represents alternative $i \in K$. Let the sets of alternatives on the path be $I_0, \dots, I_r$, where $I_0 = K$ and $I_r = \{i\}$. We are interested in bounding $|W(I_0) - W(I_r)|$. Clearly,

$$|W(I_0) - W(I_r)| \le \sum_{i=1}^{r} |W(I_{i-1}) - W(I_i)|.$$

Since $W(I_{i-1}) = \frac{1}{2}(W(I_i) + W(I_{i-1} - I_i))$, we have that $|W(I_{i-1}) - W(I_i)| = \frac{1}{2}|W(I_i) - W(I_{i-1} - I_i)|$. Since the alternative sets I_i and $I_{i-1} - I_i$ are the alternatives of an A_2^M algorithm (at depth i), by Claim 10 we have that $|W(I_i) - W(I_{i-1} - I_i)| \le \frac{2}{\sqrt{T}}$. This implies that $|W(I_{i-1}) - W(I_i)| \le \frac{1}{\sqrt{T}}$. Therefore, $|W(I_0) - W(I_r)| \le \frac{r}{\sqrt{T}}$. ■

Now we can combine the two claims to the following theorem.

Theorem 17 *Suppose the global cost function is makespan. For any set K of 2^r alternatives, and for any loss sequence $\ell_1, \dots, \ell_T$, algorithm A_{2^r} will have regret at most $O(\frac{\log|K|}{\sqrt{T}})$, i.e., $C_\infty(L_T^{A_{2^r}}) - C_\infty^*(\ell) \le 242\frac{r}{\sqrt{T}}$.*

Proof: By Lemma 15 we have that $W(K) - C_\infty^*(\ell) \le \frac{r\alpha}{\sqrt{T}}$. By Claim 16 we have that for any alternative $i \in K$, $L_T^{A_{2^r}}(i) - W(K) \le \frac{r}{\sqrt{T}}$. Therefore, $C_\infty(L_T^{A_{2^r}}) - C_\infty^*(\ell) \le 242\frac{r}{\sqrt{T}}$, since $\alpha \le 241$ by Theorem 14. ■

6 Lower Bound for Non-Convex functions

In this section we show a lower bound that holds for a large range of non-convex global cost functions, defined as follows.

Definition 18 *A function f is γ_f-strictly concave if for every $x > 0$: (1) $\frac{f(x/3)+f(2x/3)}{f(x)} \ge \gamma_f > 1$, (2) $\lim_{x\to 0} xf'(x) = 0$. (3) f is non-negative concave and increasing (4) $\lim_{x\to 0} f(x) = 0$*

Note that any function $f(x) = x^\beta$ for $\beta < 1$, is γ_f-strictly concave with $\gamma_f = \left(\frac{2}{3}\right)^\beta + \left(\frac{1}{3}\right)^\beta$.

Theorem 19 *Let $C(L_T^A) = \sum_{i=1}^k f(L_T^A(i))$, where f is (γ_f)-strictly concave. For any online algorithm A there is an input sequence such that $C(L_T^A)/C^*(L_T) > \gamma_f$.*

The proof can be found in Appendix D.

7 Open Problems

In this work we define the setting of an online learning with a global cost function C. For the case that C is convex and C^* is concave, we showed that there are online algorithms with vanishing regret. On the other hand, we showed that for certain concave functions C the regret is not vanishing. Giving a complete characterization when does a cost function C enable a vanishing regret is very challenging open problem. In this section we outline some possible research directions to address this problem.

First, if C is such that for some monotone function g, $C' = g(C)$ satisfy the same properties as d-norm: C' is convex, $C'^*(L) = \min_\alpha C'(L \odot \alpha)$ is concave, and $\alpha^*(L)$ is Lipschitz, the algorithm of Section 4 for C' would still lead to vanishing regret. For example, $C(L) = \sum_{i=1}^k (L_T(i))^d$ can be solved with that approach.

A more interesting case is when the properties are violated. Obviously, in light of the lower bound of Section 6 attaining C^* in all cases is impossible. The question is if there is some more relaxed goal that can be attained. As in [15], one can take the convex hull of the target set in Eq. (1) and still consider an approaching strategy. It can be shown that by using the approachability strategy to the convex hull, the decision maker minimizes the regret obtained with respect to the function C^C defined below (i.e., $R_T^C(A) = C(L_T^A) - C^C(L_T)$). Specifically:

$$C^C(L) = \sup_{\substack{L^1, L^2, \dots, L^m,\ \beta^1, \dots \beta^m,\ \beta_j \ge 0: \\ \sum \beta_j = 1,\ \sum_{j=1}^m \beta^j L^j = L}} C\left(\sum_{j=1}^m \beta^j \alpha_C^*(L^j) \odot L^j\right).$$

It follows that $C^*(L) \le C^C(L)$ and $C^*(L) = C^C(L)$ if C is convex and C^* concave (since in that case the convex hull of S equals S). In general, however, C^C may be strictly larger than C^*. The question if C^C is the lowest cost that can be attained against any loss sequence is left for future research.

A different open problem is a computational one. The approachability-based scheme requires projecting a point to a convex set. Even if we settle for ϵ projection (which would lead to a vanishing regret up to an ϵ), the computational complexity does not seem to scale better than $(1/\epsilon)^k$. It would be interesting to understand if the complexity minimizing the regret for cost functions other than the makespan can have better dependence on ϵ and k. In particular, it would be interesting to devise regret minimization algorithms for d-norm cost functions.

References

[1] B. Awerbuch and R. Kleinberg. Online linear optimization and adaptive routing. *J. Comput. Syst. Sci.*, 74(1):97–114, 2008.

[2] B. Awerbuch and Y. Mansour. Adapting to a reliable network path. In *PODC*, pages 360–367, 2003.

[3] D. Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific J. Math.*, 6(1):1–8, 1956.

[4] A. Blum and C. Burch. On-line learning and the metrical task system problem. In *COLT*, pages 45–53, 1997.

[5] A. Blum, C. Burch, and A. Kalai. Finely-competitive paging. In *FOCS*, pages 450–458, 1999.

[6] A. Blum, S. Chawla, and A. Kalai. Static optimality and dynamic search-optimality in lists and trees. *Algorithmica*, 36(3):249–260, 2003.

[7] A. Blum and Y. Mansour. From external to internal regret. *Journal of Machine Learning Research*, 8:1307–1324, 2007.

[8] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[9] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, 2006.

[10] D. P. de Farias and N. Megiddo. Combining expert advice in reactive environments. *Journal of the ACM*, 53(5):762–799, 2006.

[11] Y. Freund, R. Schapire, Y. Singer, and M. Warmuth. Using and combining predictors that specialize. In *STOC*, pages 334–343, 1997.

[12] S. M. Kakade, A. Tauman-Kalai, and K. Ligett. Playing games with approximation algorithms. In *STOC*, pages 546–555, 2007.

[13] A. Kalai and S. Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291 – 307, 2005. An earlier version appeared in COLT 2003.

[14] E. Lehrer. A wide range no-regret theorem. *Games and Economic Behavior*, 42:101–115, 2003.

[15] S. Mannor and N. Shimkin. The empirical Bayes envelope and regret minimization in competitive Markov decision processes. *MOR*, 28(2):327–345, 2003.

A Properties of Norms

We are now ready to extend the results to any norm. Let us start by computing what is the optimal assignment and what is its cost.

Lemma 20 *Consider the d-norm global cost function and a sequence of losses $\ell_1, \ldots, \ell_T$. The optimal stationary distribution is given by $\alpha^*_{C_d}(\ell) = (\frac{1/L_T(i)^{\frac{d}{d-1}}}{\sum_{j=1}^k 1/L_T(j)^{\frac{d}{d-1}}})_{i\in K}$ and the optimal global cost function is given by*

$$C^*_d(\ell) = \frac{1}{T}\left(\frac{1}{\sum_{j=1}^k 1/L_T(j)^{\frac{d}{d-1}}}\right)^{\frac{d-1}{d}}.$$

Proof: Recall that $L_T(1), \ldots, L_T(k)$ are the average losses of the alternatives. To compute the optimal d norm static distribution, we need to optimize the following cost function $C_d(\ell) = \left(\sum_{i=1}^k(\alpha(i)L_T(i))^d\right)^{1/d}$ subject to the constraint that $\sum_{i=1}^k \alpha(i) = 1$. Using Lagrange multipliers we obtain that the optimum is obtained for the following values

$$(\alpha^*_{C_d}(\ell))(i) = \alpha(i) = \frac{1/L_T(i)^{\frac{d}{d-1}}}{\sum_{j=1}^k 1/L_T(j)^{\frac{d}{d-1}}}.$$

This implies that the cost of the optimal distribution is given by

$$\begin{aligned} C^*_d(L) &= \left(\sum_{i=1}^k(\alpha(i)L_T(i))^d\right)^{1/d} \\ &= \left(\sum_{i=1}^k\left(\frac{1/L_T(i)^{\frac{d}{d-1}}}{\sum_{j=1}^k L_T(j)^{\frac{d}{d-1}}}1/L_T(i)\right)^d\right)^{1/d} \\ &= \left(\sum_{i=1}^k\left(\frac{\frac{1}{L_T(i)^{\frac{1}{d-1}}}}{\sum_{j=1}^k 1/L_T(j)^{\frac{d}{d-1}}}\right)^d\right)^{1/d} \\ &= \frac{1}{\sum_{j=1}^k 1/L_T(j)^{\frac{d}{d-1}}}\left(\sum_{i=1}^k \frac{1}{L_T(i)^{\frac{d}{d-1}}}\right)^{1/d} \\ &= \left(\frac{1}{\sum_{j=1}^k 1/L_T(j)^{\frac{d}{d-1}}}\right)^{\frac{d-1}{d}}. \quad \blacksquare \end{aligned}$$

After computing C^*_d we are ready to prove its concavity.

Lemma 21 *The function $C^*_d(\ell)$ is concave.*

Proof: In order to show that C^*_d is concave we once again will prove that its Hessian is semidefinite negative. We first compute the derivative and the partial derivatives of order 2, we use for notation $\beta = \frac{d}{d-1} > 1$ and $\gamma^*(x) = \frac{1}{\sum_{j=1}^k 1/L_j^\beta}$ thus $C^*_d(x) = \gamma^*(x)^{1/\beta}$

$$\begin{aligned} \frac{\partial C^*_d}{\partial x(i)} &= \frac{1}{\beta}(\gamma^*(x))^{1/\beta-1}\frac{\beta x(i)^{-(\beta+1)}}{\left(\sum_{m=1}^k \frac{1}{x(m)^\beta}\right)^2} \\ &= x(i)^{-(\beta+1)}\gamma^*(x)^{\frac{1}{\beta}+1} \end{aligned}$$

$$\begin{aligned} \frac{\partial^2 C^*_d}{\partial x(i)\partial x(j)} &= x(i)^{-(\beta+1)}(\frac{1}{\beta}+1)(\gamma^*(x))^{1/\beta}\frac{\beta x(j)^{-(\beta+1)}}{\left(\sum_{m=1}^k \frac{1}{x(m)^\beta}\right)^2} \\ &= (\beta+1)\gamma^*(x)^{\frac{1}{\beta}+2}x(i)^{-(\beta+1)}x(j)^{-(\beta+1)} \end{aligned}$$

$$\begin{aligned} \frac{\partial^2 C^*_d}{\partial^2 x(i)} &= -(\beta+1)x(i)^{-(\beta+2)}\gamma^*(x)^{\frac{1}{\beta}+1} \\ &\quad +x(i)^{-(\beta+1)}(\frac{1}{\beta}+1)\gamma^*(x)^{\frac{1}{\beta}}\frac{\beta x(i)^{-(\beta+1)}}{\left(\sum_{m=1}^k \frac{1}{x(m)^\beta}\right)^2} \\ &= -(\beta+1))x(i)^{-(\beta+2)}\gamma^*(x)^{\frac{1}{\beta}+1} \\ &\quad +(\beta+1)x(i)^{-2(\beta+1)}\gamma^*(x)^{\frac{1}{\beta}+2} \end{aligned}$$

$$\begin{aligned} &= (\beta+1)\gamma^*(x)^{\frac{1}{\beta}+2} \\ &\left(-x(i)^{-(\beta+2)}\sum_{m=1}^{k}x(m)^{-\beta}+x(i)^{-2(\beta+1)}\right) \\ &= (\beta+1)\gamma^*(x)^{\frac{1}{\beta}+2} \\ &\left(-x(i)^{-(\beta+2)}\sum_{m=1,k\neq i}^{k}x(m)^{-\beta}\right). \end{aligned}$$

Since all entries of the Hessian are factor of $(\beta+1)\gamma^*(x)^{\frac{1}{\beta}+2}$ which is positive we can eliminate it without effecting the semidefinte negative property.

$$\begin{aligned} aHa' &= \sum_{i=}^{k}\sum_{j=i+1}^{k}2x(i)^{-(\beta+1)}x(j)^{-(\beta+1)}a_ia_j \\ &+\sum_{i=1}^{k}a_i^2\left(-x(i)^{-(\beta+2)}\sum_{m=1,k\neq i}^{k}x(m)^{-\beta}\right) \\ &= \sum_{i=1}^{k}\sum_{j=i+1}^{k}\Big(-a_i^2x(i)^{-(\beta+2)}x(j)^{-\beta} \\ &-a_j^2x(j)^{-(\beta+2)}x(i)^{-\beta} \\ &+2x(i)^{-(\beta+1)}x(j)^{-(\beta+1)}a_ia_j\Big) \\ &= \sum_{i=1}^{k}\sum_{j=i+1}^{k}x(i)^{-\beta}x(j)^{-\beta} \\ &\left(-\frac{a_i^2}{x(i)^2}-\frac{a_j^2}{x(j)^2}+\frac{2a_ia_j}{x(i)x(j)}\right) \\ &= \sum_{i=1}^{k}\sum_{j=i+1}^{k}-x(i)^{-\beta}x(j)^{-\beta}(\frac{a_i}{x(i)}-\frac{a_j}{x(j)})^2\leq 0.\blacksquare \end{aligned}$$

Lemma 22 *The function C^*_∞ is concave.*

Proof: In order to do so we will show that the Hessian of C^* is semi definite positive. We start by computing the partial derivatives of C^*.

$$\begin{aligned} \frac{\partial C^*}{\partial x(i)} &= \frac{1/x(i)^2}{(\sum_{j=1}^{k}1/x(j))^2}\ \frac{\partial^2 C^*}{\partial x(i)\partial x(j)} \\ &= \frac{2/(x(i)^2x(j)^2)}{(\sum_{j=1}^{k}1/x(j))^3},\quad \frac{\partial^2 C^*}{\partial x(i)^2} \\ &= \frac{-2\sum_{j\neq i}\frac{1}{x(i)^3x(j)}}{(\sum_{j=1}^{k}1/x(j))^3}. \end{aligned}$$

Next we show that $aHa' < 0$ where is the Hessian and a is any vector. To simplify the computation we eliminate the common factor $2/(\sum_{j=1}^{k}1/x(j))^3$ which is positive from all entries.

$$\begin{aligned} aHa' &= \sum_{i=1}^{k}\sum_{j=i+1}^{k}\frac{2a_ia_j}{x(i)^2x(j)^2}-\sum_{i=1}^{k}\sum_{j\neq i}\frac{a_i^2}{x(i)^3x(j)} \\ &= \sum_{i=1}^{k}\sum_{j=i+1}^{k}\frac{1}{x(i)x(j)} \\ &\left(-\frac{a_i^2}{x(i)^2}+\frac{2a_ia_j}{x(i)x(j)}--\frac{a_j^2}{x(j)^2}\right) \\ &= \sum_{i=1}^{k}\sum_{j=i+1}^{k}\frac{-1}{x(i)x(j)}\left(\frac{a_i}{x(i)}-\frac{a_j}{x(j)}\right)^2\leq 0. \end{aligned}$$

Therefore, we showed that C^* is concave and we conclude the proof. ■

Consider the the makespan metric which is the standard metric for load balancing and job scheduling tasks.

Lemma 23 *Consider the makespan global cost function and a sequence of losses $\ell_1,\ldots,\ell_T$. The optimal stationary distribution is given by $\alpha^*_{C_\infty}(\ell)=(\frac{1/L_T(i)}{\sum_{j=1}^{k}1/L_T(j)})_{i\in K}$ and the optimal global cost function is given by*

$$C^*_d(\ell)=\frac{1}{T}\left(\frac{1}{\sum_{j=1}^{k}1/L_T(j)}\right).$$

Lemma 24 *Let $f_i(x)=x(i)a_i$, where $a_i=(\alpha^*_{C_d}(x))(i)$ then for the d norm metric and for makespan we have that $|\frac{\partial f_i}{\partial x(i)}|\leq 1$ for every i.*

Proof: Let $Y=\sum_{j\neq i}x(j)^{-\beta}$, where for d norm $\beta=\frac{d}{d-1}$ and for the makespan β is 1.

$$\begin{aligned} \frac{\partial f}{\partial x(i)} &= (1-\beta)\frac{x(i)^{-\beta}}{Y+x(i)^{-\beta}}+\beta\frac{x(i)^{-2\beta}}{(Y+x(i)^{-\beta})^2} \\ &= (1-\beta)\frac{x(i)^{-\beta}}{Y+x(i)^{-\beta}}+\beta\left(\frac{x_1^{-\beta}}{Y+x(i)^{-\beta}}\right)^2\leq 1.\blacksquare \end{aligned}$$

B Approachability theory

In this appendix we provide the basic results from approachability theory in the context of compact action sets. We start with the definition of the game. Consider a repeated vector-valued game where player 1 chooses actions in $A\subset\mathbb{R}^a$ and player 2 chooses actions in $B\subset\mathbb{R}^b$. We assume both A and B are compact. We further assume that there exists a continuous vector-valued reward function $m:A\times B\to\mathbb{R}^k$. At every stage t of the game, player 1 selects action $a_t\in A$ and player 2 selects action $b_t\in B$ as a result player 1 obtains a vector-valued reward $m_t=m(a_t,b_t)$. The objective of player 1 is to make sure that the average vector-valued reward

$$\hat{m}_t=\frac{1}{t}\sum_{\tau=1}^{t}m_t$$

converges to a target set $T\subset\mathbb{R}^k$. More precisely, we say that a set T is approachable if there exists a strategy for player 1 such that for every $\epsilon>0$, there exists an integer N such that, for every opponent strategy:

$$\Pr\left(\operatorname{dist}(\hat{m}_t,T)\geq\epsilon \text{ for some } n\geq N\right)<\epsilon,$$

where dist is the point-to-set Euclidean distance.

We further consider mixed actions for both player. A mixed action for player 1 is a probability distribution over

A. We denote such a distribution by $\Delta(A)$. Similarly, we denote by $\Delta(B)$ the set of probability distributions over B. We consider the *lifting* of m from $A \times B$ to $\Delta(A) \times \Delta(B)$ by extending m in the natural way: for $p \in \Delta(A)$ and $q \in \Delta(B)$ we define the expected reward as:

$$m(p,q) = \int_A \int_B p_a q_b m(a,b) \partial a \partial b.$$

The following is the celebrated Blackwell Theorem for convex sets. The proof is essentially identical to Theorem 7.5 of [9] and is therefore omitted.

Theorem 25 *A convex set V is approachable if and only if for every $u \in \mathbb{R}^k$ with $\|u\|_2 = 1$ we have that:*

$$\sup_p \inf_q m(p,q) \cdot u \geq \inf_{x \in V} x \cdot u,$$

where $\cdot$ is the standard inner product in $\mathbb{R}^k$.

The following strategy can be shown to have a convergence rate of $O(\sqrt{k/t})$ as in [9], page 230. At time t calculate $s_t = \arg\min_{s \in V} \|\hat{m}_t - s\|_2$. If $s_t = \hat{m}_t$ act arbitrarily. If $s_t \neq \hat{m}_t$ play $p \in \Delta(A)$ that satisfies:

$$\inf_q m(p,q) \cdot (s_t - \hat{m}_t) \geq s_t \cdot (s_t - \hat{m}_t).$$

(Such a p must exist or else the set V is not approachable according to Theorem 25.)

The following theorem is a variant of Blackwell's original theorem for convex sets (Theorem 3 in [3]).

Theorem 26 *A closed convex set V is approachable if and only if for every $q \in \Delta(B)$ there exists $p \in \Delta(A)$ such that $m(p,q) \in V$.*

Proof: The proof is identical to Blackwell's original proof, noticing that for every q the set $\{m(p,q), p \in \Delta(A)\}$ is a closed convex set. ■

C Proof of Lemma 15

Let $W_1 = W(I)$ and $W_2 = W(J)$. The first inequality follows from the fact that our assumption of the regret of A_2^M implies that $\max\{W_1, W_2\} \leq \frac{1}{1/W_1 + 1/W_2} + \frac{\alpha}{\sqrt{T}}$. This also proves the base of the induction for $r = 1$.

For the proof let $\pi_1 = \prod_{i \in I} L_T(i)$, $\pi_2 = \prod_{j \in J} L_T(j)$ and $\pi = \pi_1 \pi_2$. Also, $s_1 = \sum_{i \in I} \prod_{k \in I \setminus \{i\}} L_T(k)$, $s_2 = \sum_{j \in J} \prod_{k \in J \setminus \{j\}} L_T(k)$, and $s = s_1 \pi_2 + s_2 \pi_1$. One can verify that the optimal cost $opt = (\sum_i 1/L_T(i))^{-1}$ equals to π/s.

Let $\beta = \frac{(r-1)\alpha}{\sqrt{T}}$. From the inductive assumption that we have $W_i \leq \pi_i/s_i + \beta$ for $i \in \{1,2\}$. The proof follows from the following.

$$\begin{aligned}
&\frac{1}{1/W_1 + 1/W_2} + \frac{\alpha}{\sqrt{T}} \\
&\leq \frac{1}{\frac{1}{\pi_1/s_1 + \beta} + \frac{1}{\pi_2/s_2 + \beta}} + \frac{\alpha}{\sqrt{T}} \\
&= \frac{1}{\frac{s_1}{\pi_1 + s_1\beta} + \frac{s_2}{\pi_2 + s_2\beta}} + \frac{\alpha}{\sqrt{T}} \\
&= \frac{(\pi_1 + s_1\beta)(\pi_2 + s_2\beta)}{s_1(\pi_2 + s_2\beta) + s_2(\pi_1 + s_1\beta)} + \frac{\alpha}{\sqrt{T}} \\
&= \frac{\pi_1\pi_2 + \beta(s_1\pi_2 + s_2\pi_1) + \beta^2 s_1 s_2}{s_1\pi_2 + s_2\pi_1 + 2s_2 s_1\beta} + \frac{\alpha}{\sqrt{T}} \\
&= \frac{\pi + \beta s + \beta^2 s_1 s_2}{s + 2s_2 s_1 \beta} + \frac{\alpha}{\sqrt{T}} \\
&\leq \frac{\pi}{s} + \beta\frac{s + \beta s_1 s_2}{s + 2s_2 s_1\beta} + \frac{\alpha}{\sqrt{T}} \\
&\leq opt + \beta + \frac{\alpha}{\sqrt{T}} = opt + r\frac{\alpha}{\sqrt{T}}.
\end{aligned}$$

■

D Lower bound for a concave loss function

Proof:[Theorem 19] Consider the following three sequences of losses: (1) sequence σ_1 has t time steps each with losses (δ, δ), (2) sequence σ_2 starts with sequence σ_1 followed by t time steps each with losses $(1, \delta)$, (3) sequence σ_3 starts with sequence σ_1 followed by t time steps each with losses $(\delta, 1)$ and δ will be determined later

The *opt* for all sequences σ_1, σ_2 and σ_3 has a loss of $f(\delta)$.

Given an online algorithm A, after σ_1 it has an average loss of $\beta\delta$ on alternative 1 and $(1-\beta)\delta$ on alternative 2. Therefore, its loss on σ_1 is $f(\beta\delta) + f((1-\beta)\delta)$. If $\beta \in [1/3, 2/3]$ then the loss of A is at least $f((1/3)\delta) + f((2/3)\delta)$ compared to a loss of $f(\delta)$ of OPT. Hence $C(L_T^A)/C^*(\sigma_1) \geq \gamma_f$.

If $\beta > 2/3$ then consider the performance of A on σ_2. On the first t time steps it behaves the same as for σ_1 and at the remaining t time steps it splits the weight such that its average loss on those steps is λ for alternative 1 and $(1-\lambda)\delta$ for alternative 2. Therefore its cost function is $f((\beta\delta + \lambda)/2) + f((1-\beta+1-\lambda)\delta/2)$. First, we note that for $\lambda > 1/2$, we have that $C(L_T^A) \geq f(1/4)$. Since the cost of OPT is $f(\delta)$ and as δ goes to 0 $C^*(\sigma_2)$ goes to zero, then for sufficiently small δ we have that $C(L_T^A)/C^*(\sigma_2) \geq \gamma_f$. For $\lambda \leq 1/2$ define $h(\lambda) = f((\beta\delta+\lambda)/2) + f((1-\beta+1-\lambda)\delta/2)\delta/2$. First, we would like to show that $h'(\lambda) > 0$ for every $\lambda \in [0, 1/2]$. Consider,

$$h'(\lambda) = f'((\beta\delta + \lambda)/2)/2 - \frac{\delta}{2} f'((1-\beta+1-\lambda)\delta/2) > 0.$$

Since f is concave f' is decreasing and given our assumption that $\delta f'(\delta/2)$ goes to zero as δ goes to zero, then for small enough δ and $\lambda \leq 1/2$ we can bound as follows:

$$\begin{aligned}
f'((\beta\delta+\lambda)/2) &\geq f'(1) \geq 4\delta f'(\delta/4) \\
&\geq \delta f'((1-\beta+1-\lambda)\delta/2).
\end{aligned}$$

Thus $h(\lambda)$ is increasing and its minimum is obtained at $\lambda = 0$. Therefore the cost function of algorithm, A, is at least $f(2\delta/3) + f(\delta/3)$ while the optimum achieves $f(\delta)$. Hence $C(L_T^A)/C^*(\sigma_2) \geq \gamma_f$.

The case of $\beta < 1/3$ is similar to that of $\beta > 2/3$, just using σ_3 rather than σ_2. ■