Learning with Global Cost in Stochastic Environments

Eyal Even-dar
Google Research
evendar@google.com

Shie Mannor*
Technion
shie@ee.technion.ac.il

Yishay Mansour†
Tel Aviv Univ.
mansour@cs.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, where the losses are assumed to come from a joint distribution. 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. Specifically, we consider the global cost functions: (1) the makespan (the maximum over the alternatives) and (2) the L_d norm (over the alternatives) for $d > 1$. We design algorithms that guarantee logarithmic regret for this setting, where the regret is measured with respect to the best static decision (one selects the same distribution over alternatives at every time step). We also show that the least loaded machine, a natural algorithm for minimizing the makespan, has a regret of the order of $\sqrt{T}$. We complement our theoretical findings with supporting experimental results.

1 Introduction

Consider a decision maker that has to repeatedly select between multiple actions, while having uncertainty regarding the results of its action. This basic setting motivated a large body of research in machine learning, as well as theoretical computer science, operation research, game theory, control theory and elsewhere. Online learning in general, and regret minimization in particular, focus on this case. In regret minimization, in each time step the decision maker has to select between N actions, and only then observes the loss of each action. The cumulative loss of the decision maker is the sum of its losses at the various time steps. The main goal of regret minimization is to compare the decision maker's cumulative loss to the best strategy in a benchmark class, which many times is simply the set of all actions (i.e., each strategy will play the same action in every time step). The regret is the difference between the performance of the decision maker and the best strategy in the benchmark class. The main result is that if we allow the decision maker to play a mixture of the actions, the decision maker can guarantee to almost match the best single action even in the case that the losses are selected by an adversary: the regret would be of the order of $O(\sqrt{T \log N})$. (See [3] for an excellent exposition of the topic.)

In this work we are interested in extending the regret minimization framework to handle the case where the global cost is not simply additive across the time steps as was initiated in [4] for adverserial environments. The best motivating examples are load balancing and job scheduling. Assume that each action is a machine, and at each time step we need to map an incoming task to one of the machines, without knowing the cost that it will introduce on each machine (where the cost can be a function of the available resources on the machine and the resources the task requires). In such a setting we are interested in the load of each machine, which is the sum of the costs of the tasks map to it. A natural measure of the imbalance between the machines (actions) is either the makespan (the maximum load) or the L_d norm of the loads, both widely studied in

*This research was partially supported by the Israel Science Foundation under contract 890015 and by a Horev Fellowship and the EU under a Reintegration Grant.

†This work was supported in part by a grant from the Ministry of Science grant No. 3-6797, by a grant from the Israel Science Foundation (grant No. 709/09) and grant No. 2008-321 from the United States-Israel Binational Science Foundation (BSF), and by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886. This publication reflects the authors' views only.

job scheduling.[1] This setup, in the adversarial model where losses are assumed to be a deterministic (but unknown) sequence that is possibly even generated by an adversary, was introduced and studied in [4]. It was proved that a regret of the order of $O(\sqrt{TN})$ can be guaranteed where T is the time horizon and N is the number of actions or machines. For the specific case of makespan an improved regret of $O(\log N\sqrt{T})$ can be achieved.

In our model losses are generated from a joint distribution D. After every stage the decision maker observes the sampled loss vector and we are interested in both cases of full and partial information (that is: both cases where D is known and D is not known are of interest). The main result of our work is to show a logarithmic regret bounds for general cost functions (makespan and L_d norm). Contrary to most bandits setups, the distribution D can have arbitrary correlations between the losses of various actions, capturing the idea that some instances are inherently harder or easier. Note that the decision maker observes the entire loss vector, and thus our work is in the perfect observation model. We emphasize that the regret is never negative, and in the case that the decision maker global cost is less than the static optimum global cost then the regret is zero. Namely, the runs in which the decision maker outperforms the best static strategy are regraded as having zero regret.

To better understand our setting and results, it would be helpful to consider the following example where there are two actions and the global cost function is the makespan. The distribution D with probability half returns the loss vector $(0,1)$, where action 1 has zero loss and action 2 has a loss of one, and with probability half D returns the loss vector $(1,0)$. A realization of D of size T, will have $T/2+\Delta$ losses of type $(0,1)$ and $T/2-\Delta$ losses of type $(1,0)$. The most natural strategy the decision maker can use is to fix the best static strategy for D, and use it in all time steps. In this case the best strategy would be $(1/2,1/2)$, and if there are $T/2+\Delta$ losses of type $(0,1)$ and $T/2-\Delta$ losses of type $(1,0)$ then the load on action 1 would be $T/4+\Delta/2$, the load on action 2 would be $T/4-\Delta/2$ and the makespan would be $T/4+|\Delta|/2$. One can show that the best static strategy in hindsight would have both actions with the same load and both loads would be at most $T/4$. Since we expect that $|\Delta|$ be of the order of $\sqrt{T}$, this would give a regret bound of $\Theta(\sqrt{T})$. At the other extreme we can use a dynamic strategy that greedily selects at each time step the action with the lower load. This is the well-known *Least Loaded Machine* (LLM) strategy. For the analysis of the LLM we can consider the sum of the loads, since in the LLM strategy the max is just half of the sum for two machines. Since at each time step the LLM strategy selects deterministically an action, the sum of loads would be the sum of T Bernoulli random variables, which would be $T/2+\Delta$. Since with constant probability we have $\Delta=\Theta(\sqrt{T})$, the regret would be $\Omega(\sqrt{T})$. The starting point of this research is whether the decision maker can do better than $\Theta(\sqrt{T})$ regret? The main result of this work is an affirmative answer to this question, showing logarithmic regret bounds.

In this work we consider two stochastic models, in the *known distribution model* the distribution D is known to the decision maker, while in the *unknown distribution model* the decision maker only knows that there is some distribution D that generates the examples. We consider two global cost function, the makespan, where the global cost is the maximum load on any action, and L_d norm for $d>1$, where the global cost is an L_d norm of the loads. For both the makespan and the L_d norm we devise algorithms with a regret bound of $O(\log T\log\log T)$. In the unknown distribution model we show a regret bound of $O(\log^2 T\log\log T)$. The above regret bounds depend on knowing the exact number of time steps T in advance, and hold only at the last time step. We define *anytime regret* to be the case where we are given a bound T on the number of time steps and the regret bound has to hold at any time $t<T$. We present an algorithm with an anytime regret bound of $O(T^{1/3})$. We analyze the LLM strategy, showing that it has an anytime regret upper of $O(\sqrt{T}\log T)$ and lower bound of $\Omega(\sqrt{T})$. We also perform experiments that support our theoretical finding, and show the benefits of the algorithms that we developed.

It would be instructive to compare our stochastic model and results to other regret minimization models in stochastic environments under additive loss. First, note that in the known distribution model, the best action is known, and therefore the optimal algorithm for the additive loss would simply select the best action in every time step. For the makespan or L_d norm global cost functions, the online algorithm needs to compensate for the stochastic variations in the loss sequence, even in the known distribution model. Second, in the unknown distribution model, when the decision maker observes all the losses, i.e., the perfect observation model, the simple greedy algorithm is optimal. The greedy algorithm for makespan is the LLM, and we show that its regret is at least $\Omega(\sqrt{T})$ and at most $O(\sqrt{T}\log T)$. Finally, most of the work regarding stochastic environments was devoted to the multi-armed bandit problem, where the key issue is partial observation (which induces the *exploration vs exploitation tradeoff*): the decision maker observes *only* the loss of the action (arm) it chooses. The multi-armed bandit has been studied since [6] with the main result being logarithmic regret ([5] and [1]),

[1] We remark that our information model differs from the classical job scheduling model in that we observe the costs only after we select an action, while in job scheduling you first observe the costs and only then select the action (machine).

with a constant that is a function of the distribution.

Regret for stochastic environments with memory has been studied in [2, 7] in the context of Markov decision problems (MDPs). The algorithms developed there obtain logarithmic regret but requires a finite state space and require that the MDP is either unichain or irreducible. Our problem can also be modelled as an MDP where the states are the load vector (or their differences) and the costs are additive. The state space in such a model is, however, continuous[2] and is neither unichain nor irreducible. Moreover, efficient exploration of the state space which is the hallmark of these works is not really relevant to online learning with global cost that is more focused on taking advantage of the local stochastic deviations from expected behavior.

2 Model

We consider an online learning setup where a scheduler has a finite set $N = \{1, \ldots, n\}$ of n actions (machines) to choose from. At each time step $t \in [1, T]$, the scheduler A selects a distribution $\alpha_t^A \in \Delta(N)$ over the set of actions (machines) N, where $\Delta(N)$ is the set of distributions over N. Following that a vector of losses (loads) $\ell_t \in [0,1]^n$ is drawn from a fixed distribution D, such that $p(i) = \mathbb{E}[\ell_t(i)]$ and $p_{\min} = \min_{i \in N} p(i)$. We consider both cases where D is known and unknown. We stress that D is an arbitrary distribution over $[0,1]^n$, and can have arbitrary correlations between the losses of different actions. We do assume that loss vectors of different time steps are drawn independently from D.

Our goal is to minimize a given global cost function C which is defined over the average loss (or load) of each action. In order to define this more formally we will need to introduce a few notations. Denote the average loss (or load) of the online scheduler A on action (machine) i by $L_T^A(i) = \frac{1}{T}\sum_{t=1}^T \alpha_t^A(i)\ell_t(i)$ and its average loss (or load) vector is $L_T^A = (L_T^A(1), \ldots, L_T^A(n))$.

We now introduce global cost functions. In this work we consider two global cost functions, the makespan, i.e., $C_\infty(x) = \max_{i \in N} x(i)$ or the L_d norm, i.e., $C_d(x) = (\sum_{i \in N} x(i)^d)^{1/d}$ for $d > 1$. This implies that the objective of the online scheduler A is to minimize either the *makespan*, i.e., $C_\infty(L_T^A) = \max_{i \in N} L_T^A(i)$ or the L_d norm, i.e., $C_d(L_T^A) = (\sum_{i \in N}(L_T^A(i)^d)^{1/d}$. Note that both the makespan and the L_d norm introduce a very different optimization problem in contrast to traditional online learning setup (adversarial or stochastic) where the cost is an additive function, i.e., $\sum_{i=1}^n L_T^A(i)$.

In order to define a regret we need to introduce a comparison class. Our comparison class is the class of static allocations for $\alpha \in \Delta(N)$. Again, we need to first introduce a few notations. Denote the average loss (or load) of machine i by $L_T(i) = \frac{1}{T}\sum_{t=1}^T \ell_t(i)$. The loss vector of a static allocation $\alpha \in \Delta(N)$ is $L_T^\alpha = \alpha \odot L_T$ where $x \odot y = (x(1)y(1), \ldots, x(n)y(n))$. We define the *optimal cost function* $C^*(L_T)$ as the minimum over $\alpha \in \Delta(N)$ of $C(L_T^\alpha)$ and denote by $\alpha_C^*(L_T)$ a minimizing $\alpha \in \Delta(N)$, called the *optimal static allocation*, i.e,

$$C^*(L_T) \quad = \quad \min_{\alpha \in \Delta(N)} C(L_T^\alpha) = \min_{\alpha \in \Delta(N)} C(\alpha \odot L_T).$$

For the makespan we denote the optimal cost by C_∞^* and by α_∞^* the optimal static allocation, i.e., we have

$$C_\infty^*(L_T) = \min_{\alpha \in \Delta(N)} \max_{i \in N} \alpha(i) L_T(i) = \max_{i \in N} \alpha_\infty^*(i) L_T(i).$$

Similarly for the L_d-norm we denote the optimal cost by C_d^* and by α_d^*.

As we mentioned before, We distinguish between two cases. In the *known distribution* model the scheduler A has as an input $p = (p(1), \ldots, p(n))$, the expected loss of each action, while in the *unknown distribution* model A has no information (and has to estimate these quantities from data). We stress that in the known distribution model the online algorithm does not know the realization of the losses but has access only to the expectation under the distribution D.

The regret of scheduler A at time T after the loss sequence L_T is defined as,

$$R_T(L_T, A) = \max\{C(L_T^A) - C^*(L_T), 0\}.$$

The following claim proved in [4], prescribes the static optimum of makespan and L_d norm explicitly.

Claim 1 ([4]) *For the makespan global cost function, we have* $\alpha_\infty^*(L_T) = (\frac{1/L_T(i)}{\sum_{i=1}^k 1/L_T(j)})_{i \in K}$ *and* $C_\infty^*(L_T) = (\frac{1}{\sum_{j=1}^k 1/L_T(j)})$. *For the* L_d *norm we have* $C_d^*(L_T) = (\frac{1}{\sum_{j=1}^k 1/L_j^{\frac{d}{d-1}}})^{\frac{d-1}{d}}$ *and* $\alpha_d^*(L_T) = (\frac{1/L_T(i)^{\frac{d}{d-1}}}{\sum_{j=1}^k 1/L_T(j)^{\frac{d}{d-1}}})_{i \in K}$. *The functions* C_∞ *and* C_d *are convex and the functions* C_∞^* *and* C_d^* *are concave.*

[2] Even if one discretizes the MDP, the size will grow with time.

The following lemma is a standard concentration bound.

Lemma 2 (concentration bound - additive) *Let $Z_1, \ldots, Z_m$ be i.i.d. random variable in $[0,1]$ with mean μ. Then, for any $\gamma > 0$, $\Pr\left[|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu| > \gamma\right] \leq 2e^{-2\gamma^2 m}$*

3 A Generic Load Balancing Algorithm

In this section we describe a generic load balancing algorithm G with low regret. The algorithm is described in a parametrized way. Later on we will provide concrete instances of G with regret rates for both of known and unknown distribution models and for the makespan and the L_d-norm global cost functions.

3.1 Overview

The basic idea of the generic algorithm is to partition the time in to m phases, where the length of phase k is T^k time steps.

In the known distribution model it is tempting to use always the optimal weights w^* derived from the expectations, essentially assuming that the realization will exactly match the expectation. In this case the regret would depend on the difference between the realization and the expectation. Such a strategy will yield a regret of the order of $O(\sqrt{T})$, which is the order of the deviations we are likely to observe between the realization and the expectation of the losses. Our main goal is to have a much lower regret, namely a logarithmic regret.

The main idea is the following. We start with the base weights derived from w^*. In phase k we perturb the base weights w^* depending on the deviation between w^* and opt^{k-1}, the optimal weights given the realization in phase $k-1$. At first sight it could appear counterintuitive that we can use the optimal allocation opt^{k-1} of phase $k-1$ to perturb the weights in phase k. Essentially, the optimal allocations in different phases are independent, given the known loss distribution D. However, consider the makespan for illustration, then any suboptimal allocation will have the load of some actions strictly larger than the loads of other actions. This suggests that for the actions with observed lower loads we can increase the weight, since we are concerned only with the action whose load is maximal. Essentially, our perturbations attempt to take advantage of those imbalances in order to improve the performance of the online algorithm and make it match better the optimal static allocation.

3.2 Generic Load Balancing Algorithm

The generic algorithm G depends on the following parameters: (1) C and C^*, the global cost and the optimal cost functions, respectively; (2) the number of phases m; (3) the phases' lengths, $(T^1, \ldots, T^m)$ where T^k is the length of phase k; and (4) $w^* \in \Delta(N)$ which is a distribution over the actions N.

The generic algorithm G runs in m phases where the length of the k-th phase is T^k. Let $q^k(i)$ be the observed average loss of action i at phase k, i.e., $q^k(i) = \sum_{t \in T^k} \ell_t(i)/T^k$. Let $opt^k(i)$ be the weight of action i in the optimal allocation for phase k and let $OPT^k(i) = opt^k(i)q^k(i)T^k$ be the load on action i in phase k using the optimal allocation for phase k.

The generic algorithm G has a parameter $w^* \in \Delta(N)$ that is the base weight vector (different applications of G will use different base weights w^*). During phase k the weight of action i the algorithm does not change, and it equals

$$w^k(i) = w^*(i) + \frac{1}{q^{k-1}(i)} \frac{OPT^{k-1}(i) - X^{k-1}(i)}{T^k} = w^*(i) + \frac{T^{k-1}}{T^k}(opt^{k-1}(i) - w^*(i)), \qquad (1)$$

where $X^{k-1}(i) = w^*(i)q^{k-1}(i)T^{k-1}$ if all $w^k(i)$'s are positive. Otherwise we set $w^k(i) = w^*$. First note that G has all the information to compute the weights. Second, note that $X^{k-1}(i)$ depends on $w^*(i)$ and not on $w^{k-1}(i)$. Third, at first sight it might look that $w^k(i)$ and $w^*(i)$ can be very far apart, however, in the analysis we will require that $opt^{k-1}(i)$ and $w^*(i)$ are close, and therefore $w^k(i)$ and $w^*(i)$ will be close.

3.3 Analysis of the Generic Algorithm

We now turn to deriving the properties of the generic algorithm G that will be used later for specific setups. Before we start the analysis of the generic algorithm G we would like to state a few properties that will be essential to our analysis. Some of the properties depend on the realized losses while other depend on the behaviour of the algorithm G and its parameters.

Definition 3 *Phase k of an algorithm is (α, β)-***opt-stable** *if it has the following properties:*

P1 For any action i we have that $|q^k(i) - p(i)| \leq \beta/\sqrt{T^k}$ and that $q^k(i) > 0$;

P2 For any action i we have that $|w^*(i) - opt^k(i)| = \epsilon^k(i)$, where $\epsilon^k(i) < \alpha/\sqrt{T^k}$; and

P3 The phase lengths satisfies that $T^{k-1} \leq 4T^k$.

Let us explain and motivate the above properties. Property P1 is a property of the realization of the losses in a phase, and it states that the empirical frequency of the losses are close their true expectations. Property P3 depends solely on the parameters of G and relates the length of adjacent phases, requiring that the length does not shrink too fast. Property P2 requires the base weights to be close to the optimal weight for all actions. In this section we assume that all properties hold, while for each instance of G we will need to show that they indeed hold for most of the phases, with high probability, for the specified parameters.

The first step in the analysis is to show that the weights assigned by the generic algorithm G are indeed valid. First note that if we had a negative value in Eq. (1) (i.e., $w^*(i) + \frac{T^{k-1}}{T^k}(opt^{k-1}(i) - w^*(i)) < 0$), then we set $w^k(i) = w^*$ and the weights are valid by definition. Claim 4 shows that the weights always sum to 1. Claim 5 shows that Eq. (1) is non-negative if the phase is (α, β)-**opt-stable**.

Claim 4 *For any phase k we have $\sum_{i \in N} w^k(i) = 1$.*

Proof: First note that if we set $w^k = w^*$ then the the claim holds. The proof follows from the following identities.

$$\begin{aligned}\sum_{i \in N} w^k(i) &= \sum_{i \in N}\left[w^*(i) + \frac{T^{k-1}}{T^k}(w^*(i) + \sum_{i \in N} opt^{k-1}(i) - w^*(i))\right] \\ &= 1 + \frac{T^{k-1}}{T^k}\sum_{i \in N} opt^{k-1}(i) - \frac{T^{k-1}}{T^k}\sum_{i \in N} w^*(i) = 1.\end{aligned}$$

■

The following claim shows that $w^k(i)$ in Eq. (1) are non-negative and close to the base weights $w^*(i)$.

Claim 5 *If phase k is (α, β)-**opt-stable** and $T_k > (\frac{4\alpha}{w^*(i)})^2$ then for every action i, we have*

$$|w^k(i) - w^*(i)| \leq \frac{T^{k-1}}{T^k}\frac{\alpha}{\sqrt{T^k}} \leq \frac{4\alpha}{\sqrt{T^k}}.$$

Proof: First note that if we set $w^k(i) = w^*(i)$ then the the claim holds. Otherwise

$$w^k(i) \geq w^*(i) - \frac{T^{k-1}}{T^k}|opt_i^{k-1} - w^*(i)| \geq w^*(i) - \frac{T^{k-1}}{T^k}\frac{\alpha}{\sqrt{T^k}} \geq 0,$$

where the first inequality uses $P2$ and the last uses property $P3$ that $T^{k-1}/T^k \leq 4$ and the fact that $T_k > (4\alpha/w^*(i))^2$. ■

The most important observation is made in the next lemma that considers the increase in the load due to the generic algorithm G. It shows that the increase in the load of action i in phase k can be decomposed to three parts. The first is the optimal cost of action i in the previous phase, phase $k-1$. The second is the difference between the load of the base weight in phase k and $k-1$. The third can be viewed as a constant, and will contribute at the end to the regret.

Lemma 6 *Suppose that phase k is (α, β)-**opt-stable**. Then the increase for action i in the phase is bounded by*

$$OPT^{k-1}(i) + X^k(i) - X^{k-1}(i) + \alpha\beta\left(1 + \sqrt{\frac{T^{k-1}}{T^k}}\right).$$

Proof: The increase of the load of the online algorithm for action i during phase k is $w^k(i)q^k(i)T^k$. Now,

$$\begin{aligned}w^k(i)q^k(i)T^k &= w^*(i)q^k(i)T^k + \frac{q^k(i)}{q^{k-1}(i)}(OPT^{k-1}(i) - X^{k-1}(i)) \\ &= X^k(i) + (1 + \frac{q^k(i) - q^{k-1}(i)}{q^{k-1}(i)})(OPT^{k-1}(i) - X^{k-1}(i)) \\ &= X^k(i) + OPT^{k-1}(i) - X^{k-1}(i) + \frac{q^k(i) - q^{k-1}(i)}{q^{k-1}(i)}(OPT^{k-1}(i) - X^{k-1}(i)) \\ &= X^k(i) + OPT^{k-1}(i) - X^{k-1}(i) + (q^k(i) - q^{k-1}(i))(opt_i^{k-1} - w^*(i))T^{k-1} \\ &= X^k(i) + OPT^{k-1}(i) - X^{k-1}(i) + (q^k(i) - q^{k-1}(i))\epsilon^{k-1}(i)T^{k-1} \\ &\leq X^k(i) + OPT^{k-1}(i) - X^{k-1}(i) + (\beta/\sqrt{T^k} + \beta/\sqrt{T^{k-1}})(\alpha/\sqrt{T^{k-1}})T^{k-1} \\ &= X^k(i) + OPT^{k-1}(i) - X^{k-1}(i) + \alpha\beta + \alpha\beta\sqrt{\frac{T^{k-1}}{T^k}},\end{aligned}$$

where the inequality is due to properties $P1$ and $P2$. ■

The following theorem summarizes the performance of the generic algorithm G, showing that its regret depends on two parts, the regret of the base weights in the last phase and a term which in linear in the number of phases.

Theorem 7 *Assume that C^* is concave, C is convex, and $C(a,\ldots,a) = a$ for $a > 0$. Suppose that phases $1 \ldots m'$ of the generic algorithm G with its parameters are (α,β)-**opt-stable** and that $T^{m'} \geq \max_i (4\alpha/w^*(i))^2$. Then its cost in these phases is bounded by*

$$OPT + R^{m'} + 3m'\alpha\beta,$$

where OPT is the optimal cost, $R^{m'} = \max_i R^{m'}(i)$ and $R^{m'}(i) = \max\{X^{m'}(i) - OPT^{m'}(i), 0\}$. Furthermore, if $m' < m$ then its cost is bounded by

$$OPT + 3m'\alpha\beta + \sum_{k=m'}^{m} R^k.$$

Proof: Since C is convex, for any $Z > 0$

$$C\left(\sum_{k=1}^{m} OPT^k(1) + Z, \ldots, \sum_{k=1}^{m} OPT^k(N) + Z\right) \leq \sum_{k=1}^{m} C(OPT^k(1), \ldots, OPT^k(N)) + Z\,.$$

Let L^k be the losses in phase k. By definition, $C(OPT^k(1),\ldots,OPT^k(N)) = C^*(L^k)$. Since C^* is concave

$$\sum_{k=1}^{m} C^*(L^k) \leq C^*(\sum_{k=1}^{m} L^k) = OPT.$$

We first deal with regret in the first m' rounds. By Lemma 6, the increase of load at the kth phase is bounded by

$$X^k(i) + OPT^{k-1}(i) - X^{k-1}(i) + \alpha\beta(1 + \sqrt{\frac{T^{k-1}}{T^k}}) \leq X^k(i) + OPT^{k-1}(i) - X^{k-1}(i) + 3\alpha\beta.$$

Summing over the different phases we obtain that the loss of G on action i is at most

$$[\sum_{k=1}^{m'} OPT^k(i) + 3\alpha\beta] + X^{m'}(i) - OPT^{m'}(i).$$

Applying the cost C to this we bound the cost on the G online algorithm by

$$C(L_T^G) \leq OPT + R^{m'} + 3m'\alpha\beta.$$

For the last $m - m'$ phases since C is convex and C^* is concave the total regret is bounded by the sum of the regrets over the phases completing the proof of the second part of the theorem. ■

We can summarize the theorem in the following corollary, for the cases that are of interest to us, the makespan and the L_d norm.

Corollary 8 *Assuming that the first m' phases of the generic algorithm G with its parameters are (α,β)-**opt-stable** with probability at least $1-\delta$ and $T^{m'} \geq \max_i (4\alpha/w^*(i))^2$ for the makespan and L_d norm, then its regret is bounded by*

$$3m\alpha\beta + \sum_{k=m'}^{m} T^k,$$

with probability at least $1-\delta$.

4 Makespan

In this section we consider the makespan as the global cost function. We first analyze the case of where the distribution is known and then turn to the case of where the distribution is unknown.

4.1 Known distribution

To apply the generic algorithm G we need to specify its parameters: (1) the number of phases and their duration, and (2) the base weights.

- Set $w^*(i) = \frac{1/p(i)}{P}$, where $P = \sum_{i=1}^{n} 1/p(i)$, i.e., the optimal allocation for p.
- Set the number of phases $m = \log(T)$.
- Set the length of phase k to be $T^k = T/2^k$ for $k \in [1, m]$.

Let G_∞^{kn} be the generic algorithm G with the above parameters and the cost functions makespan.

The analysis divides the phases into two sets: we show that the first $\log T/A$ phases, where A will be defined shortly, are (α, β)-**opt-stable** with high probability; for the remaining phases we show that $\sum_{k=\log T/A}^{m} T^k$ is small; the result would follow from Corollary 8.

Let $A = \lceil \max\{4\beta^2/p_{\min}^2, \max_i 4\alpha^2/w^*(i)^2\} \rceil$, $\beta = \sqrt{(1/2)\ln(1/\eta)}$, and $\alpha = 6\beta/(Np_{min}^4)$, where η is a parameter that controls the success probability. Since for the first $\log T/A$ phases we have $T^k > 4\alpha^2/w^*(i)^2$, once we show that with high probability all the first $\log T/A$ phases are (α, β)-**opt-stable**, we establish the following theorem.

Theorem 9 *Suppose we run Algorithm G_∞^{kn} with parameters $\alpha = 6\beta/(Np_{\min}^4)$, $\beta = \sqrt{(1/2)\ln(1/\eta)}$, and $\eta = \delta/Nm$. Then with probability at least $1-\delta$ the regret is bounded by*

$$O\left(\log(T)\log\left(\frac{N\log T}{\delta}\right)\frac{1}{Np_{\min}^4} + \frac{1}{p_{\min}^{10}}\log\left(\frac{N\log T}{\delta}\right)\right) = O(\log T \log\log T).$$

The next sequence of lemmas show that the first phases of G_∞^{kn} is (α, β)-**opt-stable**, where the last lemma bounds the contribution fro the last phases.

The simplest claim is showing that P3 holds, which is immediate from the definition of T^k.

Claim 10 *G_∞^{kn} satisfies property P3.*

The additive concentration bound (Lemma 2) together with the fact that $T^k \geq \max\{4\beta^2/p_{\min}^2, 4\alpha^2/w^*(i)^2\}$ for the first $\log T/A$ phases imply that for these phases property $P1$ is satisfied with high probability.

Claim 11 *With probability $1 - mN\eta$, for any phase $k < \log T/A$ and action i we have, $|q^k(i) - p(i)| < \beta/\sqrt{T^k}$ and $q^k(i) > 0$ where $\beta = \sqrt{(1/2)\ln(1/\eta)}$.*

The following lemma proves that property $P2$ is satisfied.

Lemma 12 *With probability $1 - Nm\eta$, for any phase $k < \log T/A$ and action i,*

$$|w^*(i) - opt^k(i)| \leq \frac{\alpha}{\sqrt{T^k}},$$

where $\beta = \sqrt{\frac{1}{2}\ln(1/\eta)}$ and $\alpha = 6\beta/(Np_{\min}^4)$.

Proof: First,

$$\left|\frac{1}{p(i)} - \frac{1}{q^k(i)}\right| = \frac{|p(i) - q^k(i)|}{p(i)q^k(i)} \leq \frac{\beta}{\sqrt{T^k}}\frac{2}{(p(i))^2},$$

where we used the bound on $|p(i) - q^k(i)|$ from Claim 11 and the fact that for the first $\log T/A$ phases we have $T^k \geq \max\{4\beta^2/p_{min}^2, 4\alpha^2/w^*(i)^2\}$ implies that $q^k(i) \geq p(i) - \beta/\sqrt{T^k} \geq p(i)/2$ for these phases.

Second, we show that,

$$\left|P - Q^k\right| = \left|\sum_{i=1}^{n}\frac{1}{p(i)} - \frac{1}{q^k(i)}\right| \leq \sum_{i=1}^{n}\left|\frac{1}{p(i)} - \frac{1}{q^k(i)}\right| \leq N\frac{\beta}{\sqrt{T^k}}\frac{2}{p_{\min}^2}.$$

Since $1 \leq 1/p(i) \leq 1/p_{\min}$, we have that $N \leq P \leq N/p_{\min}$. By Claim 11 we have $|p(i) - q^k(i)| \leq \beta/\sqrt{T^k}$, and this implies that,

$$\left|p(i)P - q^k(i)Q^k\right| \leq P\left|p(i) - q^k(i)\right| + q^k(i)\left|P - Q^k\right| \leq \frac{N}{p_{\min}}\frac{\beta}{\sqrt{T^k}} + N\frac{\beta}{\sqrt{T^k}}\frac{2}{p_{\min}^2} \leq \frac{3N\beta}{p_{\min}^2\sqrt{T^k}}.$$

Since $P \geq N$ and similarly $Q^k \geq N$, this implies that,

$$\left|w_i^* - opt^k(i)\right| = \left|\frac{1/p(i)}{P} - \frac{1/q^k(i)}{Q^k}\right| = \frac{|p(i)P - q^k(i)Q^k|}{p(i)Pq^k(i)Q^k} \leq \frac{2(|p(i)P - q^k(i)Q^k|)}{p_{\min}^2 N^2} = \frac{6\beta}{Np_{\min}^4\sqrt{T^k}},$$

which establishes the lemma. ■

To establish the number of time steps in phases which are shorter than A, we need to lower bound $w^*(i)$.

Lemma 13 *For any action i we have $w^*(i) \geq p_{\min}/N$, which implies that the total time at the last $\log A$ phases is bounded by $O(A) = O(\log(1/\eta)/p_{\min}^{10})$.*

Proof: Since the length of the phases decays by factor of two, it is enough to bound $A = \max\{4\beta^2/p_{\min}^2, 4\alpha^2/w^*(i)^2\}$. Consider action i. We have that $P = \sum_{j=1}^n 1/p(j) \geq 1/p(i) + (1/p_{\min})(N-1)$. Thus,

$$w^*(i) = \frac{1/p(i)}{P} \geq \frac{1/p(i)}{1/p(i) + (N-1)/p_{\min}} = \frac{p_{\min}}{p(i)p_{\min} + p(i)(N-1)} \geq \frac{p_{\min}}{N}.$$

Substituting the value of α we obtain the lemma. ■

Proof of Theorem 9 : By Lemma 12 and Claims 11, 10 show the first $\log T/A$ phases are (α, β)-**opt-stable**. Lemma 13 bounds the length of the other phases. Applying Corollary 8 proves the theorem. ■

4.2 Unknown Distribution

The algorithm for an unknown distribution relies heavily on the application developed in the previous subsection for the known distribution model. We partition the time T to $\log(T/2)$ blocks, where the r-th block, B^r, has 2^r time steps. In block r we run G_∞^r, the generic algorithm G with the following parameters:

- Set $w^{r,*}(i)$ using the observed probabilities in block B^{r-1} as follows. Let $opt^{r-1}(i)$ be the optimal weight for action i in block B^{r-1}, then $w^{r,*}(i) = opt^{r-1}(i)$. (For $r = 1$ set $w^{1,*}$ arbitrarily, note that there are only two time steps in B^1.)
- In block B^r we have $m = r$ phases, where the duration of phase k is $T^{r,k} = |B^r|/2^k = 2^{r-k}$.

We now need to show that the algorithm G_∞^r is (α, β)-**opt-stable**. We start by showing that property $P3$ is satisfied.

Claim 14 *With probability $1 - mN\eta$ we have that $|w^{r,*}(i) - opt^{r,k}(i)| \leq 2\alpha/\sqrt{T^{r,k}}$, where $\beta = \sqrt{\frac{1}{2}\ln(1/\eta)}$ and $\alpha = 6\beta/(Np_{\min}^4)$.*

Proof: Let $w^*(i) = (1/p(i))/P$. Then,

$$\begin{aligned} |w^{r,*}(i) - opt^{r,k}(i)| = |opt^{r-1}(i) - opt^{r,k}(i)| &\leq |opt^{r-1}(i) - w^*(i)| + |w^*(i) - opt^{r,k}(i)| \\ &\leq \frac{\alpha}{2^{(r-1)/2}} + \frac{\alpha}{2^{(r-k)/2}} \leq \frac{2\alpha}{\sqrt{T^{r,k}}}, \end{aligned}$$

where we used Lemma 12 twice for the second inequality. ■

Remember that $A = \max\{4\beta^2/p_{min}^2, 4\alpha^2/w^*(i)^2\}$. Thus for all blocks with $B^r < A$, our regret bounds will be trivial. A more subtle point is that A is actually A_r as $w^*(i)$ changes between the different blocks. So we still need to compute A_r for any block to be able to compute the regret bounds. Note that by definition $P1$ and $P3$ are satisfied by the same reasoning as in the previous section,

Lemma 15 *If $B^{r-1} > 4\beta^2/p_{\min}^2$, then $w^{r,*} \geq p_{\min}/(2N)$ and $A_r = O(\log 1/\eta/p_{\min}^{10})$ with probability at least $1 - \eta$.*

Proof: Let $\hat{q}$ be the average loss at the $r-1$ block. Since $B^{r-1} > 4\beta^2/p_{\min}^2$, then $\hat{q}_r(i) \geq p(i) - \beta/\sqrt{B^{r-1}} \geq p(i)/2$. Thus $1/\hat{q}_r(i) \leq p_{\min}/2$. Thus

$$\frac{1/\hat{q}_r(i)}{\sum_{j\in N} 1/\hat{q}_r(j)} \geq \frac{1}{\sum_{j\in N} 1/\hat{q}_r(j)} \geq \frac{p_{\min}}{2N}.$$

The second of the part of the proof is identical to the proof in Lemma 13 ■

In each block B^r, Theorem 9 bounds the regret of G_∞^r, and we derive the following,

Lemma 16 *With probability $1 - rN\eta$, the regret during B^r is at most*

$$O\left(\log(|B^r|)\frac{\log(1/\eta)}{Np_{\min}^4} + \frac{1}{p_{\min}^{10}}\log(1/\eta)\right).$$

Summing over the m blocks we obtain the following theorem.

Theorem 17 *With probability $1 - \delta$, the regret is at most*

$$O\left(\log^2 T \log\left(\frac{N\log T}{\delta}\right)\frac{1}{p_{\min}^4} + \frac{\log T}{p_{\min}^{10}}\log\left(\frac{N\log T}{\delta}\right)\right) = O(\log^2 T \log\log T).$$

5 L_d norm

As in the makespan case (in the previous section) we apply the generic algorithm to the L_d norm global cost function. Note that the generic algorithm behaves differently for different global cost functions, even if the same parameters are used since OPT is different. From the implementation perspective, the only difference is that we set the base distribution w^* to the optimal one with respect to the L_d norm rather than the makespan. From the proof perspective, the key difference between the makespan and the L_d norm is the proof of property P2. Also, since the optimal allocation is different, we will have a different value of α. (The proofs of this section are omitted.)

5.1 Known distribution

In this section we define G_d^{kn}, which will have a low regret for the L_d norm in the known distribution model. Let G_d^{kn} be the generic algorithm G the following parameters.

- Set $w^*(i) = (p(i))^{-\lambda}/P_\lambda$, where $\lambda = d/(d-1)$ and $P_\lambda = \sum_{i=1}^n (p(i))^{-\lambda}$, i.e., the optimal allocation for p under the L_d norm.
- Set the number of phases $m = \log(T)$ and the length of phase k to be $T^k = T/2^k$ for $k \in [1, m]$.

The following lemma will be used to show property P2, and is similar to Lemma 12 for the makespan.

Lemma 18 *With probability $1 - Nm\eta$, for any action i and phase $k < \log(T/A)$ and $A = (4\beta^2/p_{\min}^2)$, we have*

$$\left|w^*(i) - opt^k(i)\right| \le \frac{\alpha}{\sqrt{T^k}},$$

where $\beta = \sqrt{0.5\ln(1/\eta)}$, and $\alpha = O\left(d\sqrt{\ln(1/\eta)}/\left((d-1)Np_{\min}^{4d/(d-1)}\right)\right)$.

To apply Corollary 8 we need to bound the sum of the lengths of phases of size less than A, which is done in the following lemma.

Lemma 19 *For any action i we have $w^*(i) \ge p_{\min}/N$, which implies that the total time in the last $\log A$ phases is bounded by $O(A) = O\left(d^2\log(1/\eta)/\left((d-1)^2 p_{\min}^{10d/(d-1)}\right)\right)$.*

Similarly to the previous section we obtain the following.

Theorem 20 *Algorithm G_d^{kn}, with probability at least $1 - \delta$, has regret at most*

$$O\left(\log(T)\log\left(\frac{N\log T}{\delta}\right)\frac{d}{(d-1)Np_{\min}^{4d/(d-1)}} + \frac{d^2}{(d-1)^2 p_{\min}^{10d/(d-1)}}\log\left(\frac{N\log T}{\delta}\right)\right) = O(\log T\log\log T).$$

5.2 Unknown distribution

Similar to the makespan case, algorithm G_d^{un}, for L_d norm in the unknown distribution model, runs in blocks of increasing size, where in each block uses as the base distribution the optimal allocation for the previous block. The proofs are similar to the makespan case, where the differences are due to the different global cost function.

For algorithm G_d^{un}, we partition the time first to $\log(T/2)$ blocks, where the r-th block, B^r, has 2^r time steps. In block r we run G_d^r, the generic algorithm G with the following parameters:

- Set $w^{r,*}(i)$ using the observed probabilities in block B^{r-1} as follows. Let $opt^{r-1}(i)$ be the optimal weight for action i in block B^{r-1}, then $w^{r,*}(i) = opt^{r-1}(i)$. (For $r = 1$ set $w^{1,*}$ arbitrarily, note that there are only two time steps in B^1.)

- In block B^r we have $m = r$ phases, where the duration of phase k is $T^{r,k} = |B^r|/2^k = 2^{r-k}$.

Similarly to the previous section we obtain the following.

Theorem 21 *Algorithm G_d^{un}, with probability $1-\delta$, has regret at most*

$$O\left(\log^2 T \log\left(\frac{N\log T}{\delta}\right)\frac{d}{(d-1)p_{\min}^{4d/(d-1)}} + \frac{d^2\log T}{(d-1)^2 p_{\min}^{10d/(d-1)}}\log\left(\frac{N\log T}{\delta}\right)\right) = O(\log^2 T \log\log T).$$

6 Any Time Regret

The regret algorithms presented so are optimizing the regret at the termination time T. It is not hard to see that the previous algorithms have regret of $O(\sqrt{T})$ at the initial part of the sequence. In this section we develop algorithms that have low regret at any time $t \in [1,T]$, and develop a different application of the generic algorithm which guarantees at an anytime a regret of $O(T^{1/3})$. We start from the case of known distribution and move to the case of an unknown one. (The proofs are omitted.)

6.1 Known Distribution

For the known distribution model we will describe two algorithms: G_∞^{any} for the makespan and G_d^{any} for the L_d norm. We set the parameters of the algorithms as follows:

- For G_∞^{any} set $w^*(i) = (p(i))^{-1}/P$, where $P = \sum_{i=1}^n (p(i))^{-1}$. For G_d^{any} set $w^*(i) = (p(i))^{-\lambda}/P_\lambda$, where $P_\lambda = \sum_{i=1}^n (p(i))^{-\lambda}$ and $\lambda = d/(d-1)$.
- For both G_∞^{any} and G_d^{any} set $m = T^{1/3}$, the number of phases, and let the phase length be constant: $T^k = T^{2/3}$ for $k \in [1,m]$.

The main result of this section is the following theorem.

Theorem 22 *Given the expected loss $p(i)$ of each action i, then with probability at least $1 - Nm\eta$, the regret of G_∞^{any} and G_d^{any} at any $t \in [1,T]$ is $O(\alpha_\infty T^{1/3})$ and $O(\alpha_d T^{1/3})$, respectively, where $\alpha_\infty = O\left(\sqrt{\log(1/\eta)}/(Np_{\min}^4)\right)$, $\alpha_d = O\left(d\sqrt{\ln(1/\eta)}/\left((d-1)Np_{\min}^{4d/(d-1)}\right)\right)$, and $\beta = O(\sqrt{\log(1/\eta)})$.*

When all the phases have identical length, using Eq. (1), we observe that $w^k(i)$ is simply $opt^{k-1}(i)$.

Observation 23 *If $T^k = T^{k-1}$ then $w^k(i) = opt^{k-1}(i)$.*

One advantage of the above observation is that we are guarantee that w^k is a distribution. Therefore, we can drop the requirement that the phase length is at least $4\alpha^2/w^*(i)^2$, which comes from Claim 5.

Claim 24 *Suppose that all phases have identical length $T^k > 4\beta^2/p_{min}^2$. Then for $\beta = O(\sqrt{\log(1/\eta)})$, for the makespan $\alpha_\infty = O(\sqrt{\log(1/\eta)}/(Np_{\min}^4))$, for the L_d norm $\alpha_d = O\left(d\sqrt{\ln(1/\eta)}/\left((d-1)Np_{\min}^{4d/(d-1)}\right)\right)$, then with probability at least $1 - \eta Nm$ all phases are (α,β)-**opt-stable**.*

The next lemma bounds the regret from the start of phase k until some time t in phase k, compared to the optimal allocation for this time period.

Lemma 25 *Suppose that the global cost C is convex and $C((a,\ldots,a)) = a$ and that an algorithm is (α,β)-**opt-stable** algorithm at stages $1,2,\ldots,k$. Then, at any time t during phase k, the regret is at most $O(\alpha T^{1/3})$.*

Proof: By Observation 23 during phase k we use weight $w^k(i) = opt^{k-1}(i)$ for action i. Since all phases are (α,β)-**opt-stable** then this weight is close to the base weight, namely, we have that $|w^k(i) - w^*(i)| = |opt^{k-1}(i) - w^*(i)| \le \alpha/\sqrt{T^k}$. Let $\tau = 1 + \sum_{j=1}^{k-1} T^j$ be the start of phase k. Let $opt^{k,t}(i)$ be the weight of the optimal allocation for the period starting at phase k and until time t, i.e., during time steps $[\tau,t]$. Similarly to the proofs of Claims 12 and 18, just for $t-\tau$ time steps (rather than T_k) we get that $|w^*(i) - opt^{k,t}(i)| \le \alpha/\sqrt{t-\tau}$. Combining the two and using the fact that $T^k = T^{k-1}$ we obtain that$|w^k(i) - opt^{k,t}(i)| \le 2\alpha/\sqrt{t-\tau}$. The difference between weight of the algorithm and the optimal allocation during $[\tau,t]$ in phase k, for any action i is at most $2\alpha(t-\tau)/\sqrt{t-\tau} \le 2\alpha\sqrt{T^{2/3}}$. Since the global cost C is convex and $C((a,\ldots,a)) = a$ the regret is at most $O(\alpha T^{1/3})$. ■

We are now ready to bound the anytime regret.

Proof of Theorem 22 : Consider a time t in phase k. By Claim 24, the algorithm is (α, β)-**opt-stable**. Applying Lemma 25 to phase $k-1$ we have that we have that $R^{k-1} = O(\alpha_\infty T^{1/3})$ for makespan and $R^{k-1} = O(\alpha_d T^{1/3})$ for L_d norm. By Lemma 25 the regret in the period $[\tau, t]$ is at most $O(\alpha_\infty T^{1/3})$ for makespan and $O(\alpha_d t^{1/3})$ for L_d norm. We apply Corollary 8 to derive the theorem.[3] ■

6.2 Unknown distribution

Similar to the case before, of the makespan and L_d norm, we use blocks of increasing size to (implicitly) estimate the expectation of the losses. Formally, we have $m = \log(T/2)$ blocks, where the r-th block, B^r, has 2^r time steps. In each block we run our G_∞^{any} and G_d^{any} algorithms. The parameters are:

- Set $w^{r,*}(i)$ to be the optimal allocation in the previous block, i.e., $opt^{k-1}(i)$, where in G_∞^{any} we use the makespan and in G_d^{any} we use the L_d norm.
- In block B^r we have $m = |B^r|^{1/3}$ phases, where the duration of phase k is $T^{r,k} = |B^r|^{2/3}$.

Let $A = 4\beta^2/p_{\min}^2$. For all blocks which are smaller than $A^{3/2}$ we simply bound their regret by their time. For the longer blocks, we show that in *each* block all phases are (α, β)-**opt-stable** (with high probability). Therefore, we obtain the following theorem.

Theorem 26 *With probability $1 - N\eta T^{1/3}$, at anytime the regret is at most $O(T^{1/3}\alpha + \log^{3/2} 1/\eta/p_{\min}^3) = O(T^{1/3})$, where $\alpha = O((\sqrt{\log(1/\eta)})/(Np_{\min}^4))$ for makespan and $\alpha = O(\frac{d\sqrt{\ln \frac{1}{\eta}}}{(d-1)Np_{\min}^{4d/(d-1)}})$ for L_d norm.*

7 Least Loaded Machine Scheduler

The Least load machine (LLM) scheduler is an intuitive and frequently used algorithm to minimize the makespan. The LLM scheduler puts all the weight of the next job on the least loaded machine machine, or, in our terminology, LLM selects the action with the least observed losses. The LLM scheduler is geared towards minimizing the makespan, and we will show that the regret of the LLM scheduler is at most $O(\sqrt{T}\log T)$ in the anytime model. On the other hand, the LLM scheduler suffers a regret which is $\Omega(\sqrt{T})$, a property that is common to all deterministic schedulers that allocate all the weight to a single machine.

In our setting all losses are bounded by 1 so at any time t the LLM scheduler will satisfy that the difference between the load of different actions is bounded by 1.

Lemma 27 *At time $t \in [t, T]$ the difference between the load of any two actions is bounded by 1, i.e., $|L_t^{LLM}(i) - L_t^{LLM}(j)| \leq 1$ for any $t \in [1, T]$ and $i \in N$.*

We prove in the following sequence of lemmas that the frequency of LLM using machine i is proportional to $1/p(i)$. The first step is to define when the realized loses are "representative" of the expectations. A realization of the losses is a matrix M of size $n \times T$, where the entry (i, k) is distributed according to $\ell(i)$ (using D). We can view the generation process of the losses as follows. The k-th time LLM uses action i we return the loss in entry $M[i, k]$. One can see that this gives an identical distribution to D.

Definition 28 *A realization matrix M is* representative *if for any i and k we have $\left|\sum_{j=1}^{k} M[i, j]/k - p(i)\right| \leq \sqrt{(\log 1/\eta)/(2k)}$.*

Using a simple concentration bound (Lemma 2) we have the following lemma.

Lemma 29 *With probability $1 - 2NT\eta$ the realization matrix M is representative.*

We are interested in cases where the loads on the actions is almost balanced. This definition will formally specify when an action selection vector results in an almost balanced loads.

Definition 30 *An integer vector $(k(1), \ldots k(N))$ is ℓ balanced for a matrix M if $\sum_{i=1}^{N} k(i) = T$ and for every i we have $\sum_{j=1}^{k(i)} M[i, j] \in [\ell, \ell + 1]$.*

The following lemma shows that if the loads are almost balanced then the makespan is close to T/P.

[3] Technically, the bound of the minimum phase required in Corollary 8 does not hold, but using Observation 23 we can derive an identical statement with an almost identical proof.

Lemma 31 *Given a representative matrix M and an ℓ balanced vector $(k(1), \ldots, k(N))$, we have that $|\ell - T/P| = O(\sqrt{T \log(1/\eta)})$.*

Proof: Since M is representative, for each i we have that $|\sum_{j=1}^{k(i)} M[i,j] - p(i)k(i)| \leq \sqrt{0.5k(i)\log(1/\eta)}$. Since $(k(1), \ldots, k(N))$ is ℓ balanced we have that $|\ell - p(i)k(i)| \leq \sqrt{0.5k(i)\log 1/\eta)} + 1$. Let $k(i) = \ell/p(i) + \lambda(i)/p(i)$, where we have shown that $|\lambda(i)| \leq \sqrt{0.5k(i)\log(1/\eta)} + 1$. Summing over all actions, we obtain that $T = \sum_{i=1}^{N} k(i) = \ell P + \Lambda$, where $\Lambda = \sum_{i=1}^{N} \lambda(i)/p(i)$. Therefore $T/P - \ell = \Lambda/P = \sum_{i=1}^{N} \lambda(i)(1/p(i))/P$. The lemma follows since $\Lambda/P < \max_i \lambda(i) \leq \sqrt{0.5T\log(1/\eta)} + 1$. ■

The next claim states that the LLM scheduler produces balanced vectors, given its selection of actions.

Claim 32 *Let $k(i)$ be the number of times LLM used action i. Then $(k(1), \ldots, k(N))$ is ℓ balanced, where the makespan of the LLM in in the range $[\ell, \ell + 1]$.*

In order to bound the regret, we need to lower bound the performance of the optimal allocation.

Lemma 33 *Let $\hat{p}(i)$ be the empirical loss of action i, and $\hat{p}(i) = p(i)(1 + \delta(i))$. Then the optimal makespan is at least $(1 - \delta)T/P$, where $\delta = \max_i |\delta(i)|$. In addition, with probability $1 - N\eta$, we have that $\delta < \sqrt{(\log(1/\eta))/T}$.*

Proof: The optimal makespan has value $T/\hat{P}$ where $\hat{P} = \sum_{i=1}^{N} 1/\hat{p}(i)$. By our assumption $\hat{p}(i) > p(i)(1 - \delta)$, and the lemma follows. ■

We bounded, with high probability, the deviation of the LLM scheduler above the T/P bound (Lemma 31), and the deviation of the optimal below the T/P bound (Lemma 33). Therefore, we derived the following theorem.

Theorem 34 *With probability $1 - 3NT\eta$ we have that the regret is at most $O(\sqrt{(\log(1/\eta))/T})$.*

The LLM scheduler, as any deterministic scheduler that always assign at each time step all the weight to a single action, is bound to have a regret of the order of at least $\sqrt{T}$ with some constant probability. As stated in the the introduction, for any such scheduler, the sum of the loads is a sum of T IID Bernoulli random variables, and with constant probability some load would be of the order of $\sqrt{T}$ above the expectation. We summarize the result in the following theorem.

Theorem 35 *The expected regret of any deterministic scheduler (including LLM) is $\Omega(\sqrt{T})$, for $N = 2$.*

The influence of the number of actions N is interesting. Somewhat counterintuitively, the regret may *drop* with the increase in N. To slightly de-mystify this, note that when $T = N$ the makespan of LLM and also the uniform weights is bounded by 1, and therefore the regret is at most 1.

8 Simulations

We demonstrate the algorithms introduced in this paper by running several toy simulations with makespan cost. Of particular interest in the simulations below is the behaviour of the algorithms in different phases the way their variance changes.

We compare the algorithms we developed to several standard algorithms and show considerable gain for all variants of the generic algorithm. As in the theoretical part of the paper we consider two settings, the case where the distribution D is known and where the distribution D is unknown. In the known distribution model we compare the optimal allocation given the distribution to several applications of our algorithm, namely both anytime and the logarithmic regret. In the unknown distribution we compare our algorithms to the least loaded machine algorithm, and the algorithm presented by [4] that is suited to an adversarial setting. We use our algorithm for unknown distribution, with a small modification where in the first phase instead of using random weights we run least loaded machine algorithm (to avoid random guess in the beginning).

Table 1 summarizes the behavior of the different algorithms that were run 200 times where each run consisted of 10^7 time steps. The table demonstrates a few interesting points. First, all of variants of our algorithm enjoy lower regret, and even more striking they enjoy a very low standard deviation compared to the more intuitive algorithms (LLM and the a-priori optimal). This suggests that one can regard our method as a method for a variance reduction. The first two rows of Table 1 present the case where the distribution D is a product distribution, where each mean value is chosen uniform at random at $[0, 1]$. The last two rows present a case where the distribution D is correlated and in particular is the multinomial distribution.

	G_∞^{kn}	G_∞^{un}	G_∞^{kn}(anytime)	LLM	A-priori Optimal	Adverserial
Uniform D						
Average regret	1.6380	6.502	10.3375	20.3173	75.3860	169.3265
STD regret	1.2586	6.0256	5.5891	31.9719	39.8705	104.1208
Multinomial D						
Average regret	1.8890	5.4982	8.1424	12.0866	47.0461	160.6345
Std regret	1.3544	4.5942	4.0043	20.5329	24.9777	98.2591

Table 1: Regret of different algorithms for uniform and correlated D.

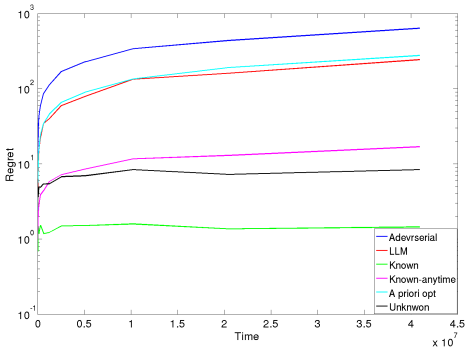

Figure 1: A demonstration of the regret growth of the algorithm when the termination time is known. The number of machines is 8 and each point is an average of 100 runs. In each time D is a product distribution where the expectation of each machine is chosen at uniform at $[0, 1]$.

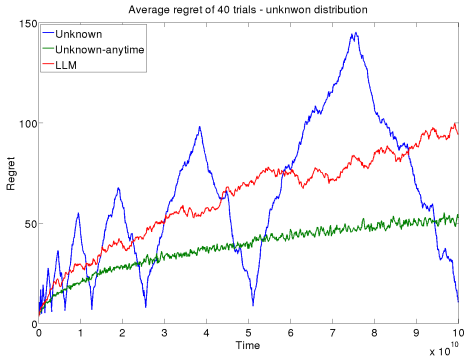

Figure 2: A demonstration of the regret the different algorithms as the time grows. All algorithms know the termination time which is 10^8 and each point is an average of 20 runs.

As can be observed, the results of for correlated and product D are similar demonstrating the ability of our algorithm to work even when there are correlations. Figure 1 depicts the regret of each algorithm as a function of the termination time T. [4] As expected, the algorithm which balances at the end outperforms the other algorithms significantly. Another point of interest is that although the adversarial and the LLM algorithm have both $O(\sqrt{T})$ regret, the LLM algorithm outperform the adversarial one.

Figure 2 provides some intuition on how the logarithmic regret algorithm G_∞^{un} behaves during each phase. Namely, the regret grows at the beginning of a phase and then starts to decrease until it is very small. In addition, it demonstrates the advantage of the anytime algorithm over other algorithms.

References

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

[2] P. Auer and R. Ortner. Logarithmic online regret bounds for undiscounted reinforcement learning. In *Advances in Neural Information Processing Systems 19*, pages 49–56. MIT Press, 2007.

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

[4] E. Even-Dar, R. Kleinberg, S. Mannor, and Y. Mansour. Online learning for global cost functions. In *The 22nd Annual Conference on Learning Theory (COLT) 2009*, pages 41–52, 2009.

[5] T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6:4–22, 1985.

[6] H. Robbins. Some aspects of sequential design of experiments. *Bull. Amer. Math. Soc.*, 55:527–535, 1952.

[7] A. Tewari and P. L. Bartlett. Optimistic linear programming gives logarithmic regret for irreducible MDPs. In *Advances in Neural Information Processing Systems Conference (NIPS)*, 2007.

[4] Note that algorithm at time 10000 and the one at time 20000 will be different since they both know the end time.