Online Resource Allocation with Machine Variability: A Bandit Perspective

Huanle Xu*, *Member, IEEE,* Yang Liu*, *Student Member, IEEE,* Wing Cheong Lau, *Senior Member, IEEE,* Tiantong Zeng, Jun Guo, *Member, IEEE,* Alex Liu, *Fellow, IEEE,*

***Abstract*—Approximation jobs that allow partial execution of their many tasks to achieve valuable results have played an important role in today's large-scale data analytics [2], [3]. This fact can be utilized to maximize the system utility of a big data computing cluster by choosing proper tasks in scheduling for each approximation job. A fundamental challenge herein, however, is that the machine service capacity may fluctuate substantially during a job's lifetime, which makes it difficult to assign valuable tasks to well-performing machines. In addition, the cluster scheduler needs to make online scheduling decisions without knowing future job arrivals according to machine availabilities. In this paper, we tackle this online resource allocation problem for approximation jobs in parallel computing clusters. In particular, we model a cluster with heterogeneous machines as a multi-armed bandit where each machine is treated as an arm. By making estimations on machine service rates while balancing the exploration-exploitation trade-off, we design an efficient online resource allocation algorithm from a bandit perspective. The proposed algorithm extends existing online convex optimization techniques and yields a sublinear regret bound. Moreover, we also examine the performance of the proposed algorithm via extensive trace-driven simulations and demonstrate that it outperforms the baselines substantially.**

***Index Terms*—Multi-armed bandit, online optimization, approximation jobs, regret bound.**

I. Introduction

Parallel and distributed computing has become an important paradigm for supporting large-scale data analytics in today's computing clusters. In such applications, approximation jobs are starting to see considerable interests since the wide-adoption of machine learning [2]. An approximation job consists of multiple small tasks and aims at performing partial execution with acceptable accuracy rather than obtaining an exact result that takes days to months to complete [4], [5]. Taking Distributed Stochastic Gradient Descent as an example, in each round, all the tasks located in different worker nodes compute the gradient for a separate data partition in parallel, and then send the results to several parameter severs, which will aggregate the results to update the parameter for the next round processing [6], [7]. With the number of rounds increasing, the training accuracy will also improve. Usually, the training accuracy is a concave function with respect to the total number of completed iterations [8]. As such, users need to make a trade-off between accuracy and computation cost when running approximation jobs under cluster environments.

With the growing number of applications to be supported, a modern computing cluster can easily consist of tens of thousands of machines. The machines within a cluster are usually heterogeneous with highly diverse costs [9]. A more powerful instance may incur a much higher price, e.g., in Amazon EC2 service, the xlarge-instance costs $3.2 per hour while the nano-instance only needs $0.0058 for an hour. Another problem arising in today's clusters is that intermittent/partial component failures, resource contention and congestion across networks have become a common phenomenon [10], [11]. As such, the actual service capacity of a machine may vary significantly when a task is being processed, which can lead to a large variation in the task progresses within the same job [12]. These issues make efficient resource allocation in distributed computing clusters extremely difficult.

Based on the above observation, in this paper, we investigate the resource allocation problem for approximation jobs with concave objectives in computing clusters under machine service variability and budget constraints [13]. Our goal is to design online resource allocation algorithms that can dynamically assign unscheduled tasks to different machines with certain performance guarantees. A fundamental challenge herein comes from two aspects. On the one hand, due to the heterogeneity and machine service variability, the system needs to estimate the service rates for all machines, so as to yield a good assignment from tasks to machines. However, to the best of our knowledge, no existing work in the literature has explicitly characterize and analyze the effect of machine variability on the task service process. On the other hand, the resource allocation decisions need to be made in an online manner, i.e., without knowing future job arrivals. The situation is further complicated when each job has a deadline constraint [14], which makes it difficult to choose between jobs that are more valuable though less urgent and less valuable but have small deadlines. Worse still, different approximation jobs may have different utility functions (e.g., training accuracy) with respect to the total amount of work that has been finished. As such, the system needs to evaluate the utility gain in a task level over time, so as to schedule as many valuable tasks as

*Co-first authors. The research was supported in part by the CUHK Direct Grant #4055108, the CUHK MobiTeC R&D Fund and in part by the National Natural Science Foundation of China under Grant Numbers 61802060, 61872082, 61472184 and in part by Dongguan University of Technology under Grant Numbers KCYKYQD2017008, KCYKYQD2017007.

Huanle Xu, Jun Guo and Alex Liu are with the School of CyberSpcae Security, Dongguan University of Technology, Dongguan, Guangdong. E-mail: {xhlcuhk@gmail.com, j.guo@ieee.org, alexliu@cse.msu.edu}.

Yang Liu, Wing Cheong Lau and Tiantong Zeng are with the Department of Information Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong. E-mail: {ly016,wclau}@ie.cuhk.edu.hk, Tracezeng72@link.cuhk.edu.hk.

Part of this work has been presented in IEEE Infocom 2019 [1].

possible to yield an optimized performance.

To tackle the aforementioned challenges, in this work, we build a stochastic framework to model a computing cluster consisting of multiple heterogeneous machines, with the aim of maximizing the total utility gain from all jobs. Under this framework, we consider that time is slotted and each approximation job has a unique utility function with a deadline constraint. To handle the issue of machine service variability, we model the whole cluster as a multi-armed bandit (henceforth, MAB) where each machine is treated as an arm. As such, our problem of selecting different machines becomes a combinatorial MAB problem [15]. To solve this problem, we estimate the service rate for each arm independently based on the amount of work completed by all tasks in the previous time slots. By carefully embodying the tension between exploration and exploitation [16], we construct an upper bound for the mean of the service rate for each arm, which will then be used to choose a proper set of arms for each individual job.

In our model, each job has a cost budget that limits the total amount of resource it can make use of in the cluster. Whenever a job executes its work on a particular node in a time slot, it needs to pay a price to the system which is specified by that node. Due to this budget constraint, the resultant optimization problem includes both the long-term constraint, i.e., the long-term computation cost for each job, as well as the short-term constraint characterized by the amount of resource that can be used in each time slot. However, these two types of constraints are further coupled with the combinatorial MAB problem as mentioned above. As such, the traditional MAB methods do not work in this case since they cannot handle long-term constraints [17]. In this paper, we solve this optimization problem by adopting the online convex optimization (OCO) techniques [18], [19]. Specifically, we first study a problem wherein there is only one job in the cluster. By transforming the long-term constraint to yield a series of short ones and implementing new convex updates with random sampling, we present a simple online primal-dual algorithm to decide which machines to assign in the single-job setting. Finally, we extend the single-job scenario to the multi-job case and design efficient scheduling algorithms.

We also extend the definitions of dynamic regret and fit in recent OCO literature [20] to quantify the performance of our proposed online scheduling algorithms. Here, dynamic regret measures the difference of the achieved utility between our scheme and that of the optimal offline solution. And the fit characterizes the total amount of resources that have been used beyond the budget. Based on these two metrics, we show that, our scheme can achieve a dynamic regret of $O(\sqrt{T\log\frac{T}{\delta}})$ with probability $(1-\delta)$ while guaranteeing a small fit for both the single-job and multi-job cases over a duration of T time slots. To summarize, we have made the following technical contributions in this paper:

- After reviewing the related work in Section II, we present a new combinatorial MAB framework to address the resource allocation problem for multiple approximation jobs in a heterogeneous cluster with machine service variability in Section III. Moreover, our framework generalizes the recent online learning problems in the sense that it includes both the long-term and short-term constraints.
- When there is only one job in the cluster, we design an efficient online algorithm of low complexity by applying a simple primal-dual update with random sampling in Section IV. This algorithm applies a new method to update the dual variables, which can yield a small constraint violation. Comparing to the previous combinatorial MAB algorithms, our designed algorithm is more scalable to implement. In addition, we also design an online algorithm in Section V to handle the multi-job scenario and guarantee sublinear dynamic regret in some special cases. Note that, in addition to the resource allocation problems, one can also leverage our techniques to study other general OCO problems with uncertainty.
- Before concluding our work in Section VII, we conduct trace-driven simulations in Section VI and demonstrate that our proposed algorithm can increase the overall job utilities by nearly 50% while incurring less computation costs comparing to existing scheduling schemes.

II. Related Work

Partial execution with concave utility functions for approximation jobs has been studied extensively in the research literature, e.g., [3], [14], [21]. In particular, [14] does not consider the parallelism and each job can only be processed on one single machine. By contrast, [3], [21] assume that each job can be processed in parallel. However, one fundamental limitation of [3], [21] is that they do not take the machine service variability into account and model the whole cluster resources as one single pool. Deadline-sensitive job scheduling is also an active research area currently [22], [23]. Most of the existing work focused on designing algorithms with constant competitive ratios. Resource allocation with budget constraint is another hot research topic [13], [24], [25]. In particular, [13] presents an approach to allocate resource in terms of virtual machines with the objective to ensure all jobs are finished within their deadlines at minimum financial cost. As a comparison, [25] considers the problem where there is a fixed time-limit as well as a resource budget for execution of adaptive applications in a cloud environment.

Recently, researchers began to analyze the effectiveness of cloning via modeling the variability of machine service rates [12], [26], [27]. However, all of the work do not explore the exact dynamics of machines since they either adopt the mean of machine service rates to explore the job service process, or simply assume the machine service rate follows a exponential distribution. By contrast, in this paper, we explicitly estimate the machine service rates from a bandit perspective.

The MAB problem has been the predominant theoretical model for sequential decision problems that balance the trade-off between exploration and exploitation. To resolve the conflict between taking actions which yield immediate reward and taking actions whose benefit will come only later, [28] presents the upper confidence bound (UCB) algorithm to choose one arm per time slot. Under UCB, all the rewards generated by one arm follow the same distribution and the overall objective

is to maximize the total rewards pulled from M arms within T rounds. Based on the design principle of UCB, Chen *et al.* consider a more general case wherein multiple arms can be chosen in each round [15], [17]. Recently, researchers have started to investigate bandit problems with knapsacks, e.g., [16], [29]–[31]. In these problems, a cost is associated once an arm is pulled and the goal is to maximize the total rewards with limited budget. One drawback of the work is that the proposed schemes require one to solve a complicated optimization problem in each round, which is quite inefficient. As comparison, in this paper, we only perform a simple projected gradient descent operation with random sampling when making scheduling decisions.

OCO has emerged as a widely-adopted framework for designing online algorithms, especially when the sequence of costs varies in an unknown and adversary manner and the decision needs to be made before the future cost functions are revealed [18], [19]. While early OCO frameworks can only deal with unconstrained convex problems, recent advances in the field have enabled one to analyze and solve OCO problems with long-term constraints, e.g., [20], [32]–[35]. In [32], authors present an online convex-concave approach to achieve an $O(\sqrt{T})$ bound on the static regret and an $O(T^{3/4})$ bound on the violation of constraints. By contrast, the authors in [34] develop an adaptive algorithm to choose better step sizes, which lead to cumulative bounds of $O(T^{\max\{\beta,1-\beta\}})$ and $O(T^{1-\beta/2})$ for the static regret and constraint violations respectively. In [33], [35], authors adopt the online saddle-point method to deal with the long-term constraints and show that a static regret bound of $O(\sqrt{T})$ and finite constraint violations are achievable. Lately, [20] presents a modified saddle-point method to achieve a bound of $O(T^{2/3})$ on the constraint violations and a sublinear dynamic regret, under the assumption that the variations in both the optimal solutions and constraint functions over time are small.

III. System Model and Problem Formulation

Consider a computing cluster which consists of M servers (machines/instances), where the servers are indexed from 1 to M. Time is slotted and job j arrives at the cluster in time slot a_j and the job arrival process, $(a_1, a_2, \ldots, a_N)$, is an arbitrary deterministic time sequence. In addition, job j has a deadline d_j which specifies the last time slot after which the job needs to leave the cluster. In this model, we consider all jobs are approximation jobs which allow partial execution and each job can be run in parallel on multiple servers. Job j has a utility of $f_j(\cdot)$, which is a concave function with respect to the amount of work that has been done by its deadline d_j. For a machine learning application, f_j can be viewed as the training/ learning accuracy with respect to the number of completed iterations.

We model the multi-machine cluster as a MAB where each machine is treated as one arm. Specifically, letting θ_i^t be the amount of service delivered by machine i in time slot t. $\mathbb{E}[\theta_i^t] = \theta_i$ where $\theta_i \leq 1$ for all i. In addition, each machine is associated with a price p_i, which is the cost that a job needs to pay when it runs a task on machine i in a time slot. Without loss of generality, we assume $p_i \leq 1$ for all i. Moreover, each job j has a fixed budget of B_j and is reported to the cluster when it arrives. On the one hand, the budget gives an upper bound for the amount of work to be processed and limits the number of tasks that can be executed in parallel for a job. On the other hand, the budget also limits the amount of resource a job can make use of in the cluster by the deadline d_j.

Let $x_j^i(t)$ be a binary variable that indicates whether a task of job j is executed on machine i in time slot t. Then $r_j^i(t) = \theta_i^t \cdot x_j^i(t)$ is the amount of work that has been executed during time slot t on machine i for job j. In this model, $r_j^i(t)$ is known to the system only at the end of each time slot t.

Denote by $X_j(t)$ the total amount of work completed in time slot t for job j, then $X_j(t)$ can be represented as:

$$X_j(t) = \sum_{i=1}^{M} r_j^i(t) = \sum_{i=1}^{M} \theta_i^t \cdot x_j^i(t), \tag{1}$$

As such, the total amount of work completed by job j is:

$$X_j = \sum_{t=a_j+1}^{d_j} X_j(t). \tag{2}$$

Since all the tasks within a job share the same setting when performing distributed training (parallel processing), the number of finished iterations (processed data samples) is in proportion to the total amount of work that has been completed. As such, we model the job utility as a function of X_j only. In fact, one can introduce another variable w_j to quantity the amount of work for each iteration performed by a task from job j, then the number of iterations completed for job j can be characterized by X_j/w_j and therefore the utility is $f_j(X_j/w_j)$. Under this setting, the following algorithm design and analysis are still applicable. For ease of presentation, we shall only focus on the setting of $f_j(X_j)$ in the sequel.

In this paper, we also make the following assumption on θ_i^t:

Assumption 1. *For each machine i, $\{\theta_i^t | t \in \mathbb{N}^+\}$ are i.i.d. random variables.*

Remark 1. *Assumption 1 basically models a stochastic MAB. With this assumption, one can adopt conventional UCB methods and their extensions to estimate the machine service rate in each time slot and further design efficient online scheduling algorithms. Extension of this assumption to more practical settings, i.e., the adversary MAB will be the next step of this work. The subsequent evaluation results also show that, even though the machine service rates within different time slots are highly correlated instead of i.i.d. distributed, our designed UCB-based estimation method can still yield a good performance.*

A. Problem formulation

In this model, our objective is to maximize the total utility gain from all jobs in the cluster subject to both the resource

and budget constraints. Letting $\boldsymbol{x}(t) \triangleq \{x_j^i(t)\}$, formally, we study the following optimization problem P1:

$$\max_{\{\boldsymbol{x}(t)\}} \sum_{j=1}^{N} f_j(X_j) \tag{P1}$$

$$\text{s.t.} \sum_{j\in J_t} x_j^i(t) \le 1, \quad \forall i,t, \tag{3}$$

$$\sum_{i=1}^{M}\sum_{t=a_j+1}^{d_j} p_i \cdot x_j^i(t) \le B_j, \quad \forall j, \tag{4}$$

$$x_j^i(t) \in \{0,1\}, \quad \forall\ i,j,t, \tag{5}$$

where $J_t = \{j : t \ge a_j + 1, t \le d_j\}$ denotes the set of jobs that are still in the cluster at time slot t. Constraint (3) ensures that each machine can only hold one task at any time and Constraint (4) is due to that the computation cost a job j incurs cannot exceed its budget B_j.

P1 turns out to be a combinatorial MAB problem with the aim to find in each time slot, a subset of arms for all jobs that are still in the cluster, so as to maximize the total utility gain. However, P1 is quite different from the traditional combinatorial MAB problem, since we include in P1 both long-term constraints (i.e., Constraint (4)) and short-term constraints (i.e., Constraint (3) and (5)).

In this paper, we consider the communications/ coordinations between tasks within a same job do not lead to bottlenecks and therefore only model the behaviors of workers. By applying the recent advances in distributed machine learning, i.e., adopting the techniques of round-robin synchronization and batch size tuning [36], one can significantly reduce the communication overhead in a heterogeneous cluster and in the meanwhile, do not hurt the convergence performance. We will leave the modeling of the exact communication overhead between workers and parameter servers as future works.

B. Objective and Performance Metrics

Due to the uncertainties associated with future job arrivals and job cost functions, in general, it is not possible to find the optimal solution for P1 in an online manner. However, OCO literatures have formalized ways to guide the design of good online solutions by comparing their cumulative utility with that of some reference benchmark schemes. One such approach is to measure the difference between the cumulative utility of a proposed online solution and that of the best, static or dynamic solution in the hindsight by defining the regret metric [28]. For static regret, the optimal solution in the benchmark scheme does not change over time. Definitely, a static optimal solution does not provide a meaningful benchmark. This is because, in such cases, even if an online solution can minimize or achieve small static regret, its performance can still be very poor in practice. By contrast, dynamic regret can allow the benchmark solution to adapt over time when the environment changes or new job comes. However, dynamic regret is generally difficult to analyze as it may require additional assumptions on the objective function [20].

Regarding the behavior of online decisions $\{\boldsymbol{x}(t)\}_{t=1}^{T}$ produced by our solution where $T = \max_{j\in\{1,2,\ldots,N\}} d_j$, we adopt the dynamic regret as an evaluation metric. However, it is nontrivial to find a meaningful definition of regret. The reason behind is that, the budget in Constraint (4) limits the total resource that a job can make use of in the cluster within its lifetime. When $p_i = 1$ for all i, X_j is upper bounded by B_j and therefore $f_j(X_j) \le f_j(B_j)$. As such, the difference between the optimal value and the objective achieved by any algorithm is bounded by $\sum_{j=1}^{N} f_j(B_j)$. If $f_j(X_j)$ is adopted in the regret formulation, achieving a sublinear regret bound with respect to T does not make sense, especially when $f_j(B_j)$ is much smaller than T. To tackle this issue, we adopt $(d_j - a_j) f_j\big(X_j/(d_j - a_j)\big)$ instead of $f_j(X_j)$ to characterize the utility achieved under our solution (the benchmark). With this newly defined job utility, we formulate the dynamic regret as follows:

$$\text{Reg}_T := \sum_{j=1}^{N}(d_j - a_j) f_j\big(X_j^*/(d_j - a_j)\big) - \sum_{j=1}^{N}(d_j - a_j) f_j\big(X_j/(d_j - a_j)\big), \tag{6}$$

where X_j^* is given by:

$$X_j^* = \sum_{t=a_j+1}^{d_j}\sum_{i=1}^{M} \theta_i \cdot x_j^{i,*}(t), \tag{7}$$

and $\{x_j^{i,*}(t)\}$ is an optimal solution to P1 with θ_i^t replaced by θ_i. As such, our regret definition is consistent with that in the traditional bandit setting, e.g., [16], [29]. In these works, the optimal offline benchmark is formulated using the expected value of the corresponding random variables. We will show in the sequel that, even we change θ_i to θ_i^t in this benchmark, we can still achieve the same order of regret as that under the original definition (i.e., Eq. (7)).

Furthermore, we follow the same regret definition for constraints as that in [16], [30] to capture the constraint violations. To be specific, we force our online solutions, $\{\boldsymbol{x}(t)\}_{t=1}^{T}$ to strictly satisfy the short-term constraints, while allowing them to violate the long-term constraints. Towards this end, we introduce the following metric to characterize the accumulative violation of constraints for all jobs:

$$\text{Fit}_T = \sum_{j=1}^{N}\Big(\sum_{i=1}^{M}\sum_{t=a_j+1}^{d_j} p_i \cdot x_j^i(t) - B_j\Big), \tag{8}$$

where $\Big(\sum_{i=1}^{M}\sum_{t=a_j+1}^{d_j} p_i \cdot x_j^i(t) - B_j\Big)$ is treated as the constraint violation for each job j.

C. Approximation of job utilities

It is worth noting that, in P1, the utility of a job reveals only when a job completes and leaves the cluster. However, the online decisions for task assignments should be made in each time slot before a job leaves. As such, the online version of P1 is difficult to handle. To tackle this issue, we solve another approximated optimization problem using a time-varying objective instead. In particular, we adopt the following function as an approximation of the original objective:

$$\sum_{j\in J_t}(t - a_j) f_j\Big(\sum_{\tau=a_j+1}^{t} X_j(\tau)/(t - a_j)\Big), \tag{9}$$

where $(t-a_j)$ is the number of time slots that job j has stayed in the cluster since its arrival and $f_j\big(\sum_{\tau=a_j+1}^{t} X_j(\tau)/(t-a_j)\big)$ is the average utility achieved by job j in a time slot. With (9), we only need to maximize the accumulative utility till time slot t. We shall show in the sequel that such an approximation yields a good regret performance.

IV. Online Optimization for a Single Job Case

Before we present the algorithm for a general multi-job scenario, in this section, we study a special case where there is only one job in the cluster. For this case, we omit the subscript for the variable $x_j^i(t)$ ($r_j^i(t)$) in P1 and use $x^i(t)$ ($r^i(t)$) instead. In addition, we also omit the subscript for f_j and B_j. Without loss of generality, we also consider that $a_j = 0$ and $d_j = T$.

We adopt the OCO approach to tackle this single-job case. To begin with, we first separate the long-term constraint (4) into each time slot as follows:

$$g(\boldsymbol{x}(t)) = \sum_{i=1}^{M} p_i \cdot x^i(t) - \frac{(1-\epsilon)B}{T} \leq 0, \tag{10}$$

where ϵ is a parameter to be defined in the subsequent sections. Here, the contraint (10) can be violated temporarily during some time slots, as long as the accumulated constraint violations are under control.

With the use of (9) and (10), P1 becomes:

$$\max_{\boldsymbol{x}(t)\in\{0,1\}^M} \quad \sum_{t=1}^{T} t \cdot f\Big(\sum_{\tau=1}^{t} X(\tau)/t\Big) \tag{P2}$$

$$\text{s.t.} \quad \sum_{t=1}^{T} g(x(t)) \leq 0. \tag{11}$$

A. Estimation on the service rate

In this simplified optimization problem P2, the parameter θ_i^t is still unknown at the beginning of each time slot t. To address this issue, we need to make an estimate of θ_i^t before making online decisions. In particular, we present algorithms derived from the UCB family of algorithms [28]. The basic idea behind is to use the observations from the past plays of each arm (machine) i at time slot t to construct estimations for the mean service rate θ_i. More importantly, we also balance the trade-off between exploration and exploitation by adding a confidence radius to the averaged value [29]. As such, our estimation of θ_i in time t, $\widehat{\theta}_i^t$ is given by:

$$\widehat{\theta}_i^t = \min\left\{1, \overline{\theta}_i^t + 2\text{rad}(\overline{\theta}_i^t, \sum_{\tau=1}^{t-1} x^i(\tau) + 1)\right\}, \tag{12}$$

where

$$\overline{\theta}_i^t = \frac{\sum_{\tau=1}^{t-1} r^i(\tau)}{\sum_{\tau=1}^{t-1} x^i(\tau) + 1}, \tag{13}$$

is the empirical average of the service rate from arm i by time t and $\text{rad}(\nu, P)$ is the confidence radius characterized by the following formula:

$$\text{rad}(\nu, P) = \sqrt{\frac{\gamma\nu}{P}} + \frac{\gamma}{P}. \tag{14}$$

Here, γ is an important parameter to address the trade-off between the exploration and exploitation process. The meaning of Eq. (14) with γ is characterized by the following concentration inequality.

Theorem 1. *Consider some distribution with values in* $[0,1]$ *and expectation* ν. *Let* $\overline{\nu}$ *be the average of* P *independent samples from this distribution [29], [37], then:*

$$\Pr[|\nu - \overline{\nu}| \leq rad(\overline{\nu}, P) \leq 3rad(\nu, P)] \leq 1 - e^{-\Omega(\gamma)}. \tag{15}$$

More generally, (15) also holds if $Z_1, \ldots, Z_P \in [0,1]$ *are random variables,* $\overline{\nu} = \frac{\sum_{i=1}^{P} Z_i}{P}$ *is the empirical average and* $\nu = \frac{\sum_{i=1}^{P} \mathbb{E}[Z_i|Z_1,\ldots,Z_{i-1}]}{P}$.

B. Algorithm Design Using Online Primal-Dual Approach

With the estimated service rates and transformed constraint characterized by Eq. (10), P2 turns out to be an OCO problem with long-term constraints [32]. However, the solution approach proposed in [32] can only deal with a convex set and requires one to introduce a regularizer. In this paper, we adopt a primal-dual approach [38] that carefully update the primal-dual variables followed by random sampling.

Let $\lambda(t)$ be the multiplier in time-slot t. We then incorporate the short-term constraint into the objective at time slot t by multiplying it by $\lambda(t+1)$, which yields the following Lagrangian function:

$$L_t(\boldsymbol{x}, \boldsymbol{\lambda}) = tf\Big(\big(\sum_{\tau=1}^{t-1}\sum_{i=1}^{M} \widehat{\theta}_i^{\tau} x^i(\tau) + \sum_{i=1}^{M} \widehat{\theta}_i^t x^i\big)/t\Big) - \lambda(t+1)g(\boldsymbol{x}). \tag{16}$$

The online primal-dual approach works as follows. Given the dual variable $\lambda(t+1)$, $\boldsymbol{x}(t+1)$ is determined via maximizing the combination of the first order approximation of the Lagrangian function at $\boldsymbol{x}(t)$ and a regularization term, i.e.,

$$\boldsymbol{x}(t+1) = \arg\max_{\boldsymbol{x}\in\Omega} \boldsymbol{\sigma}_t^T(\boldsymbol{x} - \boldsymbol{x}(t)) - \lambda(t+1)g(\boldsymbol{x}) - \frac{||\boldsymbol{x} - \boldsymbol{x}(t)||_2^2}{2\alpha}, \tag{17}$$

where $\boldsymbol{\sigma}_t = t \cdot \nabla_{\boldsymbol{x}(t)} f\Big(\big(\sum_{\tau=1}^{t}\sum_{i=1}^{M} \widehat{\theta}_i^{\tau} x^i(\tau)\big)/t\Big)$ and α is a regularization parameter. The regularization term controls the distance between $\boldsymbol{x}(t)$ and $\boldsymbol{x}(t-1)$. We shall show in the experimental results that α has a heavy impact on the migration costs. One can also choose other regularizers. However, it can be readily shown that, with this particular regularization term, the update of $\boldsymbol{x}(t+1)$ can be easily done via implementing the following gradient ascent step:

$$\boldsymbol{x}(t+1) = \Pi_{\Omega}\Big(\boldsymbol{x}(t) + \alpha \cdot \nabla_{\boldsymbol{x}} L_t(\boldsymbol{x}(t), \boldsymbol{\lambda})\Big), \tag{18}$$

where $\Pi_{\Omega}(\boldsymbol{c})$ is the projection of $\boldsymbol{c}$ onto the compact set $\Omega = \{\boldsymbol{x} : \boldsymbol{0} \leq \boldsymbol{x} \leq \boldsymbol{1}\}$. As such, α can also be interpreted as the learning rate of the primal update.

Since Π_{Ω} only contains vectors of elements between 0 and 1, $\Pi_{\Omega}(\boldsymbol{c})$ can be easily computed as follows:

$$\big(\Pi_{\Omega}(\boldsymbol{c})\big)^i = \begin{cases} c^i, & 0 \leq c^i \leq 1, \\ 1, & c^i > 1, \\ 0, & \text{otherwise}, \end{cases} \tag{19}$$

where $(\cdot)^i$ is the ith element of a vector. Furthermore, we also perform gradient descent on the Lagrangian function to update the dual variable, i.e.,

$$\lambda(t+1) = \max\Big\{0\ ,\ \lambda(t) + 2\mu\cdot g(\boldsymbol{x}(t)) - \mu\cdot g(\boldsymbol{x}(t-1))\Big\}, \quad (20)$$

where μ is a step-size to be designed later. In (20), we borrow the idea from Nesterov's Accelerated Gradient Descent to update the dual variable [39], which can lead to a tighter regret bound than normal gradient descent methods.

After computing $\boldsymbol{x}(t)$ in (18), we then round $x^i(t)$ to an integer solution $\widetilde{x^i(t)}$ by applying a simple random sampling method as follows:

$$\widetilde{x^i(t)} = \begin{cases} 1, & \text{with prob. } x^i(t), \\ 0, & \text{with prob. } (1 - x^i(t)). \end{cases} \quad (21)$$

As such, we have:

$$\mathbb{E}[\widetilde{x^i(t)}] = x^i(t). \quad (22)$$

That is, $\widetilde{x^i(t)}$ is an unbiased estimator of $x^i(t)$. We call this online algorithm *OPS* (Online Primal-Dual Algorithm with Sampling for A Single Job) and its corresponding pseudo-code is shown in Algorithm 1.

Algorithm 1: Online Primal-Dual Algorithm with Sampling for A Single Job

```
1  Initialize λ(0) = 0 and choose any x(0) ∈ [0,1]^M ;
2  for t = 1, 2, ..., do
3      Estimate the service rates for all machines
         following (12);
4      Update the dual variable λ(t) via (20);
5      Update the primal variable x(t) via (18);
6      Generate integer solutions x̃(t) via (21);
7      for k = 1, 2, ..., M do
8          if Σ_{τ=1}^{t-1} Σ_{i=1}^{M} p_i x̃^i(τ) + Σ_{i=1}^{k} p_i x̃^k(t) < B then
9              if x̃^k(t) == 1 then
10                 Execute the job on machine k in time
                     slot t;
11         else
12             exit;
```

It is worthing noting that, in Line 7 of Algorithm 1, we need to check whether the budget has run out before making a scheduling decision. We shall show in the sequel that, by carefully choosing ϵ, the budget will run out a small constant before the deadline with a high probability.

C. Main Results

We characterize the long-term performance of OPS via the following theorem.

Theorem 2. *When f is a π-Lipschitz function and the algorithm has run T time slots, by choosing $\gamma = \ln\frac{MT}{\delta}$, $\mu = \frac{\sqrt{T}}{M}$, $\alpha = \frac{1}{2\sqrt{T}}$ and $\epsilon = \frac{8}{\rho^2 T}$ where $\rho = \frac{B}{MT}$, with prob. $(1-\delta)$, the constraint violation in (8) under OPS is bounded by:*

$$Fit_T = \sum_{i=1}^{M}\sum_{t=1}^{T} p_i\cdot\widetilde{x^i(t)} - B \le \ln\frac{1}{\delta}\sqrt{B} + O(1), \quad (23)$$

and the regret defined in (6) is upper bounded by:

$$Reg_T \le O\Big(M\pi\sqrt{T\ln\frac{MT}{\delta}}\Big) + 2\pi\ln\frac{MT}{\delta} + \epsilon OPT, \quad (24)$$

where $OPT = Tf(X^*/T)$.

To prove Theorem 2, we first apply the Lyapunov optimization based approach to analyze the constraint violation and the regret separately given the machine estimations in each time slot. In particular, when analyzing the constraint violation, we introduce a special feasible solution which has a zero objective and substitute it into Eq. (17) to derive an upper bound for the dual variable. For analyzing the regret performance, we adopt a convex analysis to transform the approximated objective (Per Eq. (9)) to the true objective (Per Eq. (6)). After that, we apply the concentration inequalities to bound the difference between the estimated values and the corresponding true values, which is quite different from the bandit analysis in [32] since the latter only analyzes the expected performance via making approximations. The major novelty of this analysis is that we build a systematic way to combine stochastic bandit methods with OCO techniques.

1) The Lyapunov-drift Approach: For ease of presentation, we let $\omega(t) = \lambda(t) - \mu g\big(\boldsymbol{x}(t-1)\big)$. We shall first characterize several important properties for $\omega(t)$, which will be used to analyze the fit and dynamic regret.

Lemma 1. *$\omega(t) \ge 0$ and $\omega(t+1) \le \omega(t) + \mu g\big(\boldsymbol{x}(t)\big)$ for all t. Moreover, $\omega(t+1)^2 \le 2\mu^2 g^2(\boldsymbol{x}(t)) + \omega^2(t) + 2\mu\omega(t)\cdot g(\boldsymbol{x}(t))$.*

Proof. With $\omega(t)$, Eq. (20) can be reformulated as:

$$\omega(t+1) = \max\Big\{-\mu g\big(\boldsymbol{x}(t)\big)\ ,\ \omega(t) + \mu\cdot g\big(\boldsymbol{x}(t)\big)\Big\}. \quad (25)$$

Therefore, $\omega(t+1) \le \omega(t) + \mu g\big(\boldsymbol{x}(t)\big)$. In addition, since $\omega(1) \ge 0$, we have $\omega(t+1) \ge 0$ if $g(\boldsymbol{x}(t)) \le 0$ and $\omega(t+1) \ge \omega(t)$ if $g(\boldsymbol{x}(t)) \ge 0$. Using the method of mathematical induction, we conclude that $\omega(t) \ge 0$. Finally, squaring both sides of (25), we have:

$$\begin{aligned}\omega(t+1)^2 =& \Big(\max\Big\{-\mu g(\boldsymbol{x}(t))\ ,\ \omega(t) + \mu g(\boldsymbol{x}(t))\Big\}\Big)^2 \\ \le& \mu^2 g^2(\boldsymbol{x}(t)) + \Big(\omega(t) + \mu g(\boldsymbol{x}(t))\Big)^2 \\ =& 2\mu^2 g^2(\boldsymbol{x}(t)) + \omega^2(t) + 2\mu\omega(t)\cdot g(\boldsymbol{x}(t)).\end{aligned} \quad (26)$$

This completes the proof of Lemma 1. □

Lemma 2. *The $\omega(t)$ is upper bounded by $(2\mu M + \frac{M\pi+\mu M^2}{\beta} + \frac{M}{2\alpha\beta} + \mu\beta)$ where $\beta = \frac{(1-\epsilon)B}{T}$.*

Proof. Since $\boldsymbol{x}(t)$ is an optimal solution to the per-slot optimization problem in Eq. (17), for all $\boldsymbol{x} \in \mathcal{X}$, it follows that:

$$\begin{aligned} &-\boldsymbol{\sigma}_{t-1}^T(\boldsymbol{x}(t)-\boldsymbol{x}(t-1))+\lambda(t)g(\boldsymbol{x}(t))+\frac{||\boldsymbol{x}(t)-\boldsymbol{x}(t-1)||_2^2}{2\alpha} \\ &\leq -\boldsymbol{\sigma}_{t-1}^T(\boldsymbol{x}-\boldsymbol{x}(t-1))+\lambda(t)g(\boldsymbol{x})+\frac{||\boldsymbol{x}-\boldsymbol{x}(t-1)||_2^2}{2\alpha}. \end{aligned} \tag{27}$$

In particular, when $\boldsymbol{x}=\mathbf{0}$, we have:

$$\begin{aligned} &-\boldsymbol{\sigma}_{t-1}^T(\boldsymbol{x}(t)-\boldsymbol{x}(t-1))+\lambda(t)g(\boldsymbol{x}(t)) \\ &\leq \boldsymbol{\sigma}_{t-1}^T\boldsymbol{x}(t-1)+\lambda(t)g(\mathbf{0})+\frac{||\boldsymbol{x}(t-1)||_2^2}{2\alpha}. \end{aligned} \tag{28}$$

In addition, following Eq. (10), $g(\mathbf{0})$ is equal to $\frac{-(1-\epsilon)B}{T}$, thus, by rearranging terms in (28) and ignoring negative terms, we further obtain:

$$\lambda(t)g(\boldsymbol{x}(t)) \leq \boldsymbol{\sigma}_{t-1}^T\boldsymbol{x}(t)-\beta\lambda(t)+\frac{||\boldsymbol{x}(t)||_2^2}{2\alpha}. \tag{29}$$

Moreover, it can be readily shown that $||\boldsymbol{x}|| \leq \sqrt{M}$, applying the result $||\boldsymbol{\sigma}_{t-1}|| \leq \pi\sqrt{M}$, it follows that:

$$\lambda(t)g(\boldsymbol{x}(t)) \leq M\pi-\beta\lambda(t)+\frac{M}{2\alpha}, \tag{30}$$

Since $\lambda(t)=\omega(t)+\mu g(\boldsymbol{x}(t-1))$ and $g(\boldsymbol{x}(t))g(\boldsymbol{x}(t-1)) \geq -\beta M$, Eq. (30) can be rewritten as:

$$\omega(t)g(\boldsymbol{x}(t)) \leq M\pi-\beta\omega(t)+\frac{M}{2\alpha}+\mu\beta M+\mu\beta^2, \tag{31}$$

Combine Eq. (31) with the result from Lemma 1 yields:

$$\begin{aligned} &\frac{||\omega(t+1)||^2-||\omega(t)||^2}{2\mu} \\ &\leq M\pi-\beta\omega(t)+\frac{M}{2\alpha}+\mu\beta M+\mu M^2+\mu\beta^2. \end{aligned} \tag{32}$$

As such, $\omega(t)$ is upper bounded by:

$$\omega(t) \leq 2\mu M+\frac{M\pi+\mu M^2}{\beta}+\frac{M}{2\alpha\beta}+\mu\beta, \tag{33}$$

as otherwise, if (t_1+1) is the first time that $\omega(t_1+1)$ violates Equation (33), i.e.,

$$\omega(t_1+1) \geq 2\mu M+\frac{M\pi+\mu M^2}{\beta}+\frac{M}{2\alpha\beta}+\mu\beta. \tag{34}$$

Applying the result from Lemma 1 again with Eq.(34) yields:

$$\begin{aligned} \omega(t_1) &\geq \omega(t_1+1)-\mu M \\ &\geq \mu M+\frac{M\pi+\mu M^2}{\beta}+\frac{M}{2\alpha\beta}+\mu\beta. \end{aligned} \tag{35}$$

Eqs.(35) and (32) together imply that $\omega(t_1+1) \leq \omega(t_1)$, which leads to a contradiction. As such, we conclude that Equation (33) holds, this completes the proof of Lemma 2. □

Lemma 3. *By choosing* $\mu=\frac{\sqrt{T}}{M}$, $\alpha=\frac{1}{2\sqrt{T}}$ *and* $\epsilon=\frac{8}{\rho^2 T}$, $\widehat{Fit_T} \leq 0$ *where* $\widehat{Fit_T}=\sum_{t=1}^T\sum_{i=1}^M p_i\cdot x^i(t)-B$ *and* $\rho=\frac{B}{MT}$.

Proof. First, $\widehat{\text{Fit}_T}$ can be reformulated as:

$$\begin{aligned} \widehat{\text{Fit}_T} &= \sum_{t=1}^T\Big(\sum_{i=1}^M p_i\cdot x^i(t)-\frac{(1-\epsilon)B}{T}\Big)-\epsilon B \\ &= \sum_{t=1}^T g(\boldsymbol{x}(t))-\epsilon B. \end{aligned} \tag{36}$$

Following Lemma 1, we obtain:

$$g(\boldsymbol{x}(t)) \leq \frac{\omega(t+1)-\omega(t)}{\mu}. \tag{37}$$

Substitute Eq.(37) into Eq.(36) and use the result from Lemma 2, it follows that:

$$\begin{aligned} \widehat{\text{Fit}_T} \leq& \frac{\omega(T+1)}{\mu}-\epsilon B \\ \leq& 2M+\frac{M\pi}{\mu\beta}+\frac{M^2}{\beta}+\frac{M}{2\alpha\mu\beta}+\beta-\epsilon B. \end{aligned} \tag{38}$$

Since $B=\rho MT$, by choosing $\mu=\frac{\sqrt{T}}{M}$, $\alpha=\frac{1}{2\sqrt{T}}$ and $\epsilon=\frac{8}{\rho^2 T}$, we have that:

$$\begin{aligned} \widehat{\text{Fit}_T} &\leq 2M+\frac{M\pi}{(1-\epsilon)\rho\sqrt{T}}+\frac{2M}{(1-\epsilon)\rho}+(1-\epsilon)\rho M-\epsilon\rho MT \\ &\leq \frac{5M}{(1-\epsilon)\rho}-\epsilon\rho MT \\ &\leq \frac{6M}{\rho}-\epsilon\rho MT \leq 0. \end{aligned} \tag{39}$$

This completes the proof of Lemma 3. □

Next, we analyze the regret performance of OPS under the estimated values of machine service rates. The following lemma states that the optimal scheduling decisions are the same across different time slots.

Lemma 4. *Let* $\{\hat{\boldsymbol{x}}^*(t)\}_t$ *be the optimal solution to P1 with* B_j *replaced by* $(1-\epsilon)B_j$ *and* $\hat{\boldsymbol{x}}^*(t) \in [0,1]^M$. *Then,* $\{\hat{\boldsymbol{x}}^*(t)\}_t$ *satisfies:*

$$\hat{x}^{i,*}(1)=\hat{x}^{i,*}(2)=\cdots=\hat{x}^{i,*}(T), \forall i \in \{1,2,\ldots,M\}.$$

Proof. Since f is a concave function, using the same argument as that in Lemma 3.1 of [31], the result immediately follows. This completes the proof. □

We will make use of Lemma 4 and apply convex analysis to bound the regret performance under the estimated values.

Lemma 5. *By choosing* $\mu=\frac{\sqrt{T}}{M}$ *and* $\alpha=\frac{1}{2\sqrt{T}}$, *we have:*

$$\begin{aligned} &\sum_{t=1}^T f\Big(\sum_{i=1}^M \widehat{\theta}_i^t\hat{x}^{i,*}\Big)-Tf\Big(\sum_{t=1}^T\sum_{i=1}^M \widehat{\theta}_i^t x^i(t)/T\Big) \\ &\leq \frac{\sqrt{T}M\pi^2}{2}+\frac{3M\sqrt{T}}{2}. \end{aligned} \tag{40}$$

Proof. Paralleling Eq.(27) and applying the argument that the R.H.S. of (17) is a strongly-convex function with module $\frac{1}{\alpha}$, we have that:

$$\begin{aligned}
&-\boldsymbol{\sigma}_{t-1}^T(\boldsymbol{x}(t)-\boldsymbol{x}(t-1))+\lambda(t)g(\boldsymbol{x}(t))+\frac{||\boldsymbol{x}(t)-\boldsymbol{x}(t-1)||_2^2}{2\alpha}\\
&\leq -\boldsymbol{\sigma}_{t-1}^T(\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t-1))+\lambda(t)g(\hat{\boldsymbol{x}}^*)+\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t-1)||_2^2}{2\alpha}\\
&\quad-\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t)||_2^2}{2\alpha}.
\end{aligned}\tag{41}$$

Adding $\Phi_{t-1}=(t-1)f(\sum_{\tau=1}^{t-1}\sum_{i=1}^{M}\widehat{\theta_i^\tau}x^i(\tau))/(t-1))$ on both sides of Eq.(41) yields,

$$\begin{aligned}
&-\Phi_{t-1}-\boldsymbol{\sigma}_{t-1}^T(\boldsymbol{x}(t)-\boldsymbol{x}(t-1))+\lambda(t)g(\boldsymbol{x}(t))+\frac{||\boldsymbol{x}(t)-\boldsymbol{x}(t-1)||_2^2}{2\alpha}\\
&\leq \Phi_{t-1}-\boldsymbol{\sigma}_{t-1}^T(\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t-1))+\lambda(t)g(\hat{\boldsymbol{x}}^*)+\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t-1)||_2^2}{2\alpha}\\
&\quad-\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t)||_2^2}{2\alpha}\\
&\overset{\text{(i)}}{\leq} -(t-1)f((\sum_{\tau=1}^{t-2}\sum_{i=1}^{M}\widehat{\theta_i^\tau}x_i(\tau)+\sum_{i=1}^{M}\widehat{\theta_i^{t-1}}\hat{x}^{i,*})/(t-1))\\
&\quad+\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t-1)||_2^2}{2\alpha}-\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t)||_2^2}{2\alpha},
\end{aligned}\tag{42}$$

where (i) is due to the convexity of f and $g(\hat{\boldsymbol{x}}^*)\leq 0$. For ease of presentatin, let $\Psi_{t-1}=(t-1)f((\sum_{\tau=1}^{t-2}\sum_{i=1}^{M}\widehat{\theta_i^\tau}x^i(\tau)+\sum_{i=1}^{M}\widehat{\theta_i^{t-1}}\hat{x}^{i,*})/(t-1))$, rearranging terms in Eq.(42), we obtain:

$$\begin{aligned}
&\Psi_{t-1}-\Phi_{t-1}+\frac{||\boldsymbol{x}(t)-\boldsymbol{x}(t-1)||_2^2}{2\alpha}\\
&\leq \boldsymbol{\sigma}_{t-1}^T(\boldsymbol{x}(t)-\boldsymbol{x}(t-1))-\lambda(t)g(\boldsymbol{x}(t))+\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t-1)||_2^2}{2\alpha}\\
&\quad-\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t)||_2^2}{2\alpha}\\
&\leq \frac{\eta||\boldsymbol{\sigma_{t-1}}||_2^2}{2}+\frac{||\boldsymbol{x}(t)-\boldsymbol{x}(t-1)||_2^2}{2\eta}-\omega(t)g(\boldsymbol{x}(t))\\
&\quad+\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t-1)||_2^2-||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t)||_2^2}{2\alpha}-\mu g(\boldsymbol{x}(t-1))g(\boldsymbol{x}(t)).
\end{aligned}\tag{43}$$

On the one hand, Lemma 1 implies that:

$$\omega(t)\cdot g(\boldsymbol{x}(t))\geq\frac{||\omega(t+1)||^2-||\omega(t)||^2}{2\mu}-\mu g^2(\boldsymbol{x}(t)).\tag{44}$$

On the other hand, we have:

$$\begin{aligned}
&g(\boldsymbol{x}(t-1))g(\boldsymbol{x}(t))\\
&=\frac{1}{2}(g^2(\boldsymbol{x}(t))+g^2(\boldsymbol{x}(t-1))-(g(\boldsymbol{x}(t))-g(\boldsymbol{x}(t-1)))^2)\\
&\overset{\text{(ii)}}{\geq}\frac{1}{2}(g^2(\boldsymbol{x}(t))+g^2(\boldsymbol{x}(t-1)))-M||\boldsymbol{x}(t)-\boldsymbol{x}(t-1)||_2^2,
\end{aligned}\tag{45}$$

where (ii) is due to that g is a $\sqrt{M}$-Lipschitz function. Combine Eqs.(43), (44) and (45), $(\Psi_{t-1}-\Phi_{t-1})$ is upper bounded by:

$$\begin{aligned}
&\Psi_{t-1}-\Phi_{t-1}\\
&\leq(\frac{1}{2\eta}-\frac{1}{2\alpha}+\frac{\mu M}{2})||\boldsymbol{x}(t)-\boldsymbol{x}(t-1)||_2^2+\frac{\eta M\pi^2}{2}\\
&\quad+\frac{||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t-1)||_2^2-||\hat{\boldsymbol{x}}^*-\boldsymbol{x}(t)||_2^2}{2\alpha}\\
&\quad+\frac{\mu}{2}(g^2(\boldsymbol{x}(t))-g^2(\boldsymbol{x}(t-1)))+\frac{||\omega(t)||^2-||\omega(t+1)||^2}{2\mu}.
\end{aligned}\tag{46}$$

Summing up Eq.(46) over all time t, we have:

$$\begin{aligned}
&\sum_{t=1}^{T}(\Psi_t-\Phi_t)\\
&\leq(\frac{1}{2\eta}-\frac{1}{2\alpha}+\frac{\mu M}{2})\sum_{t=1}^{T}||\boldsymbol{x}(t+1)-\boldsymbol{x}(t)||_2^2\\
&\quad+\frac{\eta TM\pi^2}{2}+\frac{||\hat{\boldsymbol{x}}^*||_2^2}{2\alpha}+\frac{\mu}{2}g^2(\boldsymbol{x}(T+1))\\
&\leq(\frac{1}{2\eta}-\frac{1}{2\alpha}+\frac{\mu M}{2})\sum_{t=1}^{T}||\boldsymbol{x}(t+1)-\boldsymbol{x}(t)||_2^2\\
&\quad+\frac{\eta TM\pi^2}{2}+\frac{M}{2\alpha}+\frac{\mu M^2}{2}.
\end{aligned}\tag{47}$$

By choosing $\eta=\frac{1}{\sqrt{T}}$, $\mu=\frac{\sqrt{T}}{M}$ and $\alpha=\frac{1}{2\sqrt{T}}$, one upper bound of $\sum_{t=1}^{T}(\Psi_t-\Phi_t)$ is given by:

$$\sum_{t=1}^{T}(\Psi_t-\Phi_t)\leq\frac{\sqrt{T}M\pi^2}{2}+\frac{3M\sqrt{T}}{2}.\tag{48}$$

Moreover, since f is a concave function, the relationship between Ψ_{t-1} and Φ_{t-1} is characterized by:

$$\begin{aligned}
\Psi_t&=t\cdot f\left(\left(\sum_{\tau=1}^{t-1}\sum_{i=1}^{M}\widehat{\theta_i^\tau}x_i(\tau)+\sum_{i=1}^{M}\widehat{\theta_i^t}\hat{x}^{i,*}\right)/t\right)\\
&\geq(t-1)\cdot f\left(\left(\sum_{\tau=1}^{t-1}\sum_{i=1}^{M}\widehat{\theta_i^\tau}x^i(\tau)\right)/(t-1)\right)+f\left(\sum_{i=1}^{M}\widehat{\theta_i^t}\hat{x}^{i,*}\right)\\
&=\Phi_{t-1}+f\left(\sum_{i=1}^{M}\widehat{\theta_i^t}\hat{x}^{i,*}\right).
\end{aligned}\tag{49}$$

Combining Eq.(48) and Eq.(49) yields the result. □

2) Bandit Analysis: In the sequel, we shall first apply the concentration inequality to characterize the optimal objective in the following lemma.

Lemma 6. *With prob.* $(1-MTe^{-\Omega(\gamma)})$*, we have*

$$(1-\epsilon)Tf(X^*/T)\leq\sum_{t=1}^{T}f\left(\sum_{i=1}^{M}\widehat{\theta_i^t}\hat{x}^{i,*}\right).\tag{50}$$

Proof. The result in Theorem 1 indicates that, $\theta_i\leq\widehat{\theta_i^t}$ holds with prob. $(1-e^{-\Omega(\gamma)})$. Since f is an increasing and concave function, applying union bounds, the result immediately follows and this completes the proof of Lemma 6. □

We proceed to give an upper bound for Fit_T which is defined in Eq. (10). Based on Eq. (22), for all $t \in \{1, 2, \ldots, T\}$, we have:

$$\mathbb{E}[\widetilde{x^i(t)}] = x^i(t). \tag{51}$$

For ease of presentation, let $C = \sum_{t=1}^{T}\sum_{i=1}^{M} p_i \cdot x^i(t)$ and $\widetilde{C} = \sum_{t=1}^{T}\sum_{i=1}^{M} p_i \cdot \widetilde{x^i(t)}$. Then, Theorem 1 implies that, with probability at least $(1 - e^{-\Omega(\gamma)})$, we have:

$$\left|C/T - \widetilde{C}/T\right| \leq \text{rad}(\widetilde{C}/T, T) = \sqrt{\frac{\gamma\widetilde{C}}{T^2} + \frac{\gamma}{T}}, \tag{52}$$

which leads to

$$\widetilde{C} - C \leq \sqrt{\gamma\widetilde{C}} + \gamma. \tag{53}$$

(53) implies that $\widetilde{C} \leq C + \gamma\sqrt{C} + \gamma^2 + \gamma$, this together with Lemma 3, we conclude that

$$\text{Fit}_T \leq \gamma\sqrt{B} + \gamma^2 + \gamma, \tag{54}$$

holds with prob. $(1 - e^{-\Omega(\gamma)})$. This completes the proof of the first part in Theorem 2.

Now, it remains to characterize the relationship between $f\big(\sum_{\tau=1}^{T}\sum_{i=1}^{M}\widehat{\theta}_i^{\tau} x^i(\tau)/T\big)$ and $f\big(\sum_{\tau=1}^{T}\sum_{i=1}^{M}\theta_i^{\tau}\widetilde{x^i(\tau)}/T\big)$. Since f is a π-Lipschitz function, it follows:

$$\begin{aligned}
&\left|T \cdot f\Big(\sum_{\tau=1}^{T}\sum_{i=1}^{M}\widehat{\theta}_i^{\tau} x^i(\tau)/T\Big) - T \cdot f\Big(\sum_{\tau=1}^{T}\sum_{i=1}^{M}\theta_i^{\tau}\widetilde{x^i(\tau)}/T\Big)\right| \\
\leq &\pi \cdot \left|\sum_{\tau=1}^{T}\sum_{i=1}^{M}\widehat{\theta}_i^{\tau} x^i(\tau) - \sum_{\tau=1}^{T}\sum_{i=1}^{M}\theta_i^{\tau}\widetilde{x^i(\tau)}\right| \\
\leq &\pi \cdot \Bigg|\underbrace{\sum_{\tau=1}^{T}\sum_{i=1}^{M}\widehat{\theta}_i^{\tau} x^i(\tau)}_{R_1} - \underbrace{\sum_{\tau=1}^{T}\sum_{i=1}^{M}\widehat{\theta}_i^{\tau}\widetilde{x^i(\tau)}}_{R_2}\Bigg| \\
&+ \pi \cdot \Bigg|\sum_{\tau=1}^{T}\sum_{i=1}^{M}\widehat{\theta}_i^{\tau}\widetilde{x^i(\tau)} - \underbrace{\sum_{\tau=1}^{T}\sum_{i=1}^{M}\theta_i^{\tau}\widetilde{x^i(\tau)}}_{R_3}\Bigg|.
\end{aligned} \tag{55}$$

On the one hand, paralleling (52), it follows that, with prob. $(1 - e^{-\Omega(\gamma)})$,

$$|R_1 - R_2| \leq T\text{rad}(R_2/T, T) = \sqrt{\gamma R_2} + \gamma \leq \sqrt{\gamma MT} + \gamma, \tag{56}$$

On the other hand, the second term on the RHS of the last inequality of (55) can be further decoupled as

$$|R_2 - R_3| \leq |R_2 - R_4| + |R_4 - R_3|, \tag{57}$$

where $R_4 = \sum_{\tau=1}^{T}\sum_{i=1}^{M}\theta_i\widetilde{x^i(\tau)}$.

Moreover, $|R_2 - R_4|$ can be reformulated as

$$|R_2 - R_4| = \left|\sum_{i=1}^{M}\sum_{\tau=1}^{T}(\widehat{\theta}_i^{\tau} - \theta_i)\widetilde{x^i(\tau)}\right|. \tag{58}$$

Since $\widehat{\theta}_i^{\tau} = \overline{\theta_i^{\tau}} + 2\text{rad}(\overline{\theta_i^{\tau}}, \sum_{t=1}^{\tau}\widetilde{x^i(t)} + 1)$, $|R_2 - R_4|$ is upper bounded by

$$\begin{aligned}
|R_2 - R_4| \leq &\left|\sum_{i=1}^{M}\sum_{\tau=1}^{T}(\overline{\theta_i^{\tau}} - \theta_i)\widetilde{x^i(\tau)}\right| \\
&+ 2\sum_{i=1}^{M}\sum_{\tau=1}^{T}\text{rad}(\overline{\theta_i^{\tau}}, \sum_{t=1}^{\tau}\widetilde{x^i(t)} + 1) \cdot \widetilde{x^i(\tau)}.
\end{aligned} \tag{59}$$

Based on Lemma B.3 of [30], it follows that $\overline{\theta_i^{\tau}} - \theta_i \leq 2\text{rad}(\overline{\theta_i^{\tau}}, \sum_{t=1}^{\tau}\widetilde{x^i(t)} + 1)$ holds with prob. $(1 - e^{-\Omega(\gamma)})$. As such, by applying union bound, it can be readily shown that, with prob. $(1 - MTe^{-\Omega(\gamma)})$, we have:

$$\begin{aligned}
|R_2 - R_4| &\leq 4\sum_{i=1}^{M}\sum_{\tau=1}^{T}\text{rad}(\overline{\theta_i^{\tau}}, \sum_{t=1}^{\tau}\widetilde{x^i(t)} + 1) \cdot \widetilde{x^i(\tau)} \\
&\leq 12\sum_{i=1}^{M}\sum_{\tau=1}^{T}\text{rad}(\theta_i, \sum_{t=1}^{\tau}\widetilde{x^i(t)} + 1) \cdot \widetilde{x^i(\tau)}.
\end{aligned} \tag{60}$$

Letting $\rho_i(\tau) = \sum_{t=1}^{\tau}\widetilde{x^i(t)} + 1$, $|R_2 - R_4|$ is further bounded by:

$$\begin{aligned}
|R_2 - R_4| &\leq 12\sum_{i=1}^{M}\sum_{K=1}^{\rho_i(T)+1}\text{rad}(\theta_i, K) \\
&= 12\sum_{i=1}^{M}\sum_{K=1}^{\rho_i(T)+1}\Big(\sqrt{\frac{\gamma\theta_i}{K}} + \frac{\gamma}{K}\Big) \\
&\leq 24\sum_{i=1}^{M}\sqrt{\gamma\theta_i \cdot T} + 12M\ln(T+1) + 12M.
\end{aligned} \tag{61}$$

Since $\mathbb{E}[\theta_i^{\tau}] = \theta_i$, using the same argument as that in (52), with prob. $(1 - e^{-\Omega(\gamma)})$, we have:

$$|R_3 - R_4| \leq T \cdot \text{rad}(R_3/T, T) = \sqrt{\gamma R_3} + \gamma \leq \sqrt{\gamma MT} + \gamma. \tag{62}$$

Combining (55), (56), (57), (61) and (62) yields, with prob. $(1 - MTe^{-\Omega(\gamma)})$, we have

$$\begin{aligned}
&\left|T \cdot f\Big(\sum_{\tau=1}^{T}\sum_{i=1}^{M}\widehat{\theta}_i^{\tau} x^i(\tau)/T\Big) - T \cdot f\Big(\sum_{\tau=1}^{T}\sum_{i=1}^{M}\theta_i^{\tau}\widetilde{x^i(\tau)}/T\Big)\right| \\
\leq &O(M\pi\sqrt{\gamma T}) + 2\pi\gamma.
\end{aligned} \tag{63}$$

Together Eq. (63) with results from lemmas 4, 5 and 6, Theorem 2 immediately follows. □

Using the same argument as that in Eq. (57), we can also show that, $\left|\sum_{t=1}^{T} f(\sum_{i=1}^{M}\widehat{\theta}_i^t\hat{x}^{i,*}) - \sum_{t=1}^{T} f(\sum_{i=1}^{M}\theta_i^t\hat{x}^{i,*})\right| \leq O(M\pi\sqrt{\gamma T})$ holds with high probability. As such, even we adopt θ_i^t to evaluate the offline benchmark, the resulting regret has the same order as that using θ_i.

Discussion: The conventional OCO framework with long-term constraints e.g., [32] adopts a primal-dual method to bound the regret and fit, and can only achieve a sublinear fit of $O(T^{3/4})$. Another body of OCO works with long-term constraints also adopt the Lyapunov framework to analyze the online learning algorithm, e.g., [33], [35]. However, the analytical framework requires the constraint to satisfy the slater condition. As a comparison, our developed approach automatically transfers the long-term constraint to short ones, and more importantly, manages to achieve a sublinear fit with respect to B only, which is more meaningful than existing works. Moreover, we also adopt a quite different update of dual variables so as to provide a more convenient way for bounding the overall violation of constraints. In addition, the traditional bandit methods solve a convex optimization problem in each time slot to deal with the short-term constraints, and therefore is not computationally efficient, e.g., [16], [29]. By contrast, our framework leverages a simple gradient descent method to solve the learning problem and is more scalable.

V. Resource Allocation for Multiple Jobs

In this section, we study a general setting where multiple jobs arrive at the cluster over time. In this case, we need to make scheduling decisions for multiple jobs simultaneously at the beginning of each time slot.

Paralleling (10), we first separate the budget constraint of each job into multiple time slots as follows:

$$g_j(\boldsymbol{x_j}(t)) = \sum_{i=1}^{M} p_i \cdot x_j^i(t) - (1-\epsilon)\beta_j \leq 0. \tag{64}$$

where $\beta_j = \frac{B_j}{d_j - a_j}$. In this multi-job setting, the online optimization problem becomes:

$$\max_{\boldsymbol{x}(t)\in\mathcal{X}_t} \sum_{t=1}^{T} \sum_{j\in J_t} (t-a_j) f_j\left(\frac{\sum_{\tau=a_j+1}^{t} X_j(\tau)}{t-a_j}\right) \tag{P3}$$

$$\text{s.t.} \sum_{t=a_j+1}^{d_j} g_j(\boldsymbol{x}_j(t)) \leq 0, \quad \forall j, \tag{65}$$

where $\mathcal{X}_t = \left\{\boldsymbol{x}(t) : \boldsymbol{x}(t) \in \{0,1\}^M, \sum_{j\in J_t} x_j^i(t) \leq 1, \forall i.\right\}$.

A. Algorithm Design

To solve the optimization problem P3, we adopt the same approach as that in Section IV. Specifically, we first estimate the service rate for each arm i at the beginning of time slot t as follows:

$$\widehat{\theta}_i^t = \min\left\{1, \overline{\theta_i^t} + 2\text{rad}(\overline{\theta_i^t}, \sum_{\tau=1}^{t-1}\sum_{j\in J_\tau} x_j^i(\tau) + 1)\right\}, \tag{66}$$

where $\overline{\theta_i^t}$ is given by:

$$\overline{\theta_i^t} = \frac{\sum_{\tau=1}^{t-1}\sum_{j\in J_\tau} r_j^i(\tau)}{\sum_{\tau=1}^{t-1}\sum_{j\in J_\tau} x_j^i(\tau) + 1}. \tag{67}$$

We then introduce a Lagrangian multiplier $\lambda_j(t)$ for each job j at time slot t. Let $\boldsymbol{\lambda}(t) = \{\lambda_1(t), \lambda_2(t), \ldots, \lambda_N(t)\}$. Thus, the per-slot Lagrangian function is given by

$$\begin{aligned} &L_t(\boldsymbol{x}, \boldsymbol{\lambda}(t+1)) \\ &= \sum_{j\in J_t} (t-a_j) f_j\left(\frac{\sum_{i=1}^{M}\sum_{\tau=1}^{t-1} \widehat{\theta}_i^\tau x_j^i(\tau) + \sum_{i=1}^{M} \widehat{\theta}_i^t x_j^i}{t-a_j}\right) \\ &\quad - \sum_{j\in J_t} \lambda_j(t+1) g_j(\boldsymbol{x}_j). \end{aligned} \tag{68}$$

We proceed to derive solutions for $\boldsymbol{x}(t)$ via applying the same update as that in (18). In this case, we need to be more careful in dealing with the projection operation since multiple jobs may compete for the resource on the same machine. Let

$$\boldsymbol{y}_j(t) = \boldsymbol{x}_j(t-1) + \alpha \cdot \nabla_{\boldsymbol{x}_j} L_{t-1}(\boldsymbol{x}(t-1), \boldsymbol{\lambda}(t)), \tag{69}$$

be the primal update without projection in time slot t. Then, the projection operation requires one to solve the following optimization problem on each machine i:

$$\min_{\boldsymbol{z}} \sum_{j\in J_t} \left(z_j^i - y_j^i(t)\right)^2 \tag{P4}$$

$$\text{s.t.} \sum_{j\in J_t} z_j^i \leq 1, \ \forall i. \tag{70}$$

$$z_j^i \geq 0, \ \forall j \in J_t. \tag{71}$$

The optimal solution of P4 can be derived efficiently with a complexity of at most $O(N)$ by solving the corresponding KKT equations. Thus, the update of $x_j^i(t)$ can be simply attained by taking $x_j^i(t) = z_j^{i,*}$ where $\{z_j^{i,*}\}_{i,j}$ is the optimal solution to P4.

We then apply a similar random sampling approach as that in (21) to round each $x_j^i(t)$ to a random variable $\widetilde{x_j^i(t)} \in \{0,1\}$. Moreover, paralleling (22), the random sampling procedure also guarantees:

$$\mathbb{E}[\widetilde{x_j^i(t)}] = x_j^i(t). \tag{72}$$

Finally, each dual variable $\lambda_j(t+1)$ is updated as

$$\lambda_j(t+1) = \max\left\{0, \ \lambda_j(t) + 2\mu \cdot g_j(\boldsymbol{x}_j(t)) - \mu \cdot g_j(\boldsymbol{x}_j(t-1))\right\}. \tag{73}$$

Towards this end, we design an online scheduling algorithm in this multi-job setting and call it OPM (Online Primal- Dual Algorithm with Sampling for Multiple Jobs).

B. Performance of OPM

Theorem 3. *When f is a π-Lipschitz function, by choosing $\gamma = \ln\frac{MT}{\delta}$, $\mu = \frac{\sqrt{T}}{M}$, $\alpha = \frac{1}{2\sqrt{T}}$ and $\epsilon = 0$, with prob. $(1-\delta)$, the constraint violation in (8) under OPM is bounded by:*

$$Fit_T \leq \ln\frac{1}{\delta}\sqrt{\sum_{j=1}^{N} B_j} + \frac{\sqrt{N}M^2 + \beta^{max}\sqrt{N}M}{\beta^{min}}, \tag{74}$$

where $\beta^{max} = \max_{j\in\{1,2,\ldots,N\}}\beta_j$ and $\beta^{min} = \min_{j\in\{1,2,\ldots,N\}}\beta_j$, and the regret is upper bounded by:

$$Reg_T \leq O\left(M\pi\sqrt{T\ln\frac{MT}{\delta}}\right) + \frac{M\pi\sqrt{T}}{\beta^{min}} \cdot \sum_{j=1}^{N} B_j. \tag{75}$$

where $Reg_T := \sum_{j=1}^{N}\sum_{t\geq a_j+1}^{d_j} f(X_j^*(t)) - \sum_{j=1}^{N}(d_j-a_j) f_j(\frac{X_j}{d_j-a_j})$.

Proof. In the first part of the proof, we bound the constraint violation. To begin with, let

$$\omega_j(t+1) = \max\left\{-\mu g_j(\boldsymbol{x}_j(t)), \ \omega_j(t) + \mu \cdot g_j(\boldsymbol{x}_j(t))\right\}. \tag{76}$$

Paralleling Eq. (26), we have:

$$\omega_j(t+1)^2 \leq 2\mu^2 g_j^2(\boldsymbol{x}_j(t)) + \omega_j^2(t) + 2\mu\omega_j(t) \cdot g_j(\boldsymbol{x}_j(t)). \tag{77}$$

Since $\boldsymbol{x}(t)$ is an optimal solution to the per-slot optimization problem determined by Eq. (69) and the program P4, for all $\boldsymbol{x} \in \mathcal{X}_t$, it follows that:

$$\begin{aligned}&\sum_{j\in J_t} -\boldsymbol{\sigma}_{j,t-1}^T(\boldsymbol{x}_j(t)-\boldsymbol{x}_j(t-1))\\&+\lambda_j(t)g_j(\boldsymbol{x}_j(t))+\frac{||\boldsymbol{x}_j(t)-\boldsymbol{x}_j(t-1)||_2^2}{2\alpha}\\\le&\sum_{j\in J_t} -\boldsymbol{\sigma}_{j,t-1}^T(\boldsymbol{x}_j-\boldsymbol{x}_j(t-1))+\lambda(t)g_j(\boldsymbol{x}_j)+\frac{||\boldsymbol{x}_j-\boldsymbol{x}_j(t-1)||_2^2}{2\alpha}.\end{aligned} \tag{78}$$

where $\boldsymbol{\sigma}_{j,t} = t\cdot\nabla_{\boldsymbol{x}_j(t)}f_j\Big((\sum_{\tau=1}^t\sum_{i=1}^M\widehat{\theta}_i^\tau x_j^i(\tau))/(t-a_j)\Big)$. In particular, when $\boldsymbol{x}_j=\mathbf{0}$ for all $j\in J_t$, we have:

$$\begin{aligned}&\sum_{j\in J_t} -\boldsymbol{\sigma}_{j,t-1}^T(\boldsymbol{x}_j(t)-\boldsymbol{x}_j(t-1))+\lambda_j(t)g_j(\boldsymbol{x}_j(t))\\&\le\boldsymbol{\sigma}_{j,t-1}^T\boldsymbol{x}_j(t-1)+\lambda_j(t)g_j(\mathbf{0})+\frac{||\boldsymbol{x}_j(t-1)||_2^2}{4\alpha}.\end{aligned} \tag{79}$$

In addition, following Eq. (64), $g_j(\mathbf{0})$ is equal to $\frac{-B_j}{d_j-a_j}$, thus, by rearranging terms in (79) and ignoring negative terms, we further get:

$$\sum_{j\in J_t}\lambda_j(t)g_j(\boldsymbol{x}_j(t))\le\sum_{j\in J_t}\boldsymbol{\sigma}_{j,t-1}^T\boldsymbol{x}_j(t)-\beta_j\lambda_j(t)+\frac{||\boldsymbol{x}_j(t)||_2^2}{4\alpha}. \tag{80}$$

Moreover, it can be readily shown that,

$$\sum_{j\in J_t}||\boldsymbol{x}_j(t)||_2^2\le M, \tag{81}$$

and

$$\sum_{j\in J_t}\boldsymbol{\sigma}_{j,t-1}^T\boldsymbol{x}_j(t)\le MD. \tag{82}$$

Substitute Eq.(81),(82) into Eq. (80), we have:

$$\sum_{j\in J_t}\lambda_j(t)g_j(\boldsymbol{x}_j(t))\le MD-\sum_{j\in J_t}\beta_j\lambda_j(t)+\frac{M}{4\alpha}. \tag{83}$$

Since $\lambda_j(t)=\omega_j(t)+\mu g_j(\boldsymbol{x}_j(t-1))$, Eq. (83) can be rewritten as:

$$\sum_{j\in J_t}\omega_j(t)g_j(\boldsymbol{x}_j(t))\le MD-\sum_{j\in J_t}\beta_j\omega_j(t)+\frac{M}{4\alpha}+\mu\beta^{\max}M, \tag{84}$$

Let $\beta^{\min}=\min_{j\in\{1,2,\ldots,N\}}\beta_j$, paralleling Eq. (32), we have:

$$\begin{aligned}&\frac{||\boldsymbol{\omega}(t+1)||^2-||\boldsymbol{\omega}(t)||^2}{2\mu}\\\le&MD-\beta^{\min}||\boldsymbol{\omega}(t)||+\frac{M}{4\alpha}+\mu\beta^{\max}M+\mu M^2.\end{aligned} \tag{85}$$

As such, $||\boldsymbol{\omega}(t)||$ is upper bounded by:

$$||\boldsymbol{\omega}(t)||\le\sqrt{2}\mu M+\frac{MD+\mu M^2}{\beta^{\min}}+\frac{M}{4\alpha\beta^{\min}}+\mu\frac{\beta^{\max}}{\beta^{\min}}M, \tag{86}$$

as otherwise, if (t_1+1) is the first time that $||\boldsymbol{\omega}(t_1+1)||$ violates Equation (86), i.e.,

$$||\boldsymbol{\omega}(t_1+1)||\ge\sqrt{2}\mu M+\frac{MD+\mu M^2}{\beta^{\min}}+\frac{M}{4\alpha\beta^{\min}}+\mu\frac{\beta^{\max}}{\beta^{\min}}M. \tag{87}$$

Per Eq.(87) and Eq.(76), we have:

$$\begin{aligned}||\boldsymbol{\omega}(t_1)||&\ge||\boldsymbol{\omega}(t_1+1)||-\sqrt{2}\mu M\\&\ge\frac{MD+\mu M^2}{\beta^{\min}}+\frac{M}{4\alpha\beta^{\min}}+\mu\frac{\beta^{\max}}{\beta^{\min}}M.\end{aligned} \tag{88}$$

Eqs.(88) and (85) together imply that $||\boldsymbol{\omega}(t_1+1)||\le||\boldsymbol{\omega}(t_1)||$, which leads to a contradiction. As such, we conclude Equation (86) holds and

$$\sum_{j=1}^N g_j(\boldsymbol{x}_j(t))\le\sum_{j=1}^N\frac{\omega_j(t+1)-\omega_j(t)}{\mu}, \tag{89}$$

which implies:

$$\begin{aligned}\sum_{t=1}^T\sum_{j=1}^N g_j(\boldsymbol{x}_j(t))&\le\sum_{j=1}^N\frac{\omega_j(T+1)}{\mu}\\&\le\sqrt{N}\big\|\boldsymbol{\omega}(T+1)\big\|/\mu\\&\le\frac{\sqrt{N}M^2+\beta^{\max}\sqrt{N}M}{\beta^{\min}}+O(1).\end{aligned} \tag{90}$$

Let

$$C^m=\sum_{t=1}^T\sum_{j\in J_t}\sum_{i=1}^M p_i x_j^i(t),$$

and

$$\widetilde{C^m}=\sum_{t=1}^T\sum_{j\in J_t}\sum_{i=1}^M p_i\widetilde{x_j^i(t)}.$$

With the same argument as that in the analysis of the single-job case, we have, with probability at least $(1-e^{-\Omega(\gamma)})$,

$$\begin{aligned}\widetilde{C^m}&\le C^m+\gamma\sqrt{C^m}+\gamma^2+\gamma\\&\le\sum_{j=1}^N B_j+\gamma\sqrt{\sum_{j=1}^N B_j}+\frac{\sqrt{N}M^2+\beta^{\max}\sqrt{N}M}{\beta^{\min}}+O(1).\end{aligned} \tag{91}$$

This completes the first part of the proof. We proceed to prove the second part of this theorem. Let

$$\Phi_{j,t}=(t-a_j)f\Big(\sum_{\tau=1}^t\sum_{i=1}^M\widehat{\theta}_i^\tau x_i(\tau))/(t-a_j)\Big),$$

and

$$\Psi_{j,t}=(t-a_j)f\Big((\sum_{\tau=1}^{t-1}\sum_{i=1}^M\widehat{\theta}_i^\tau x_j^i(\tau)+\sum_{i=1}^M\widehat{\theta}_i^t x_j^{i,*})/(t-a_j)\Big),$$

paralleling Eq. (43), we have:

$$\begin{aligned}&\sum_{j\in J_t}\Psi_{j,t}-\Phi_{j,t}+\frac{||\boldsymbol{x}_j(t+1)-\boldsymbol{x}_j(t)||_2^2}{2\alpha}+\lambda_j(t+1)g_j(\boldsymbol{x}_j(t+1))\\\le&\sum_{j\in J_t}\boldsymbol{\sigma}_{j,t}^T(\boldsymbol{x}_j(t+1)-\boldsymbol{x}_j(t))+\lambda_j(t+1)g_j(\boldsymbol{x}_j^*(t+1))\\&+\frac{||\boldsymbol{x}_j^*(t+1)-\boldsymbol{x}_j(t)||_2^2}{2\alpha}-\frac{||\boldsymbol{x}_j^*(t+1)-\boldsymbol{x}_j(t+1)||_2^2}{2\alpha}.\end{aligned} \tag{92}$$

In addition, it can be readily shown that:

$$\boldsymbol{\sigma}_{j,t}^T(\boldsymbol{x}_j(t+1)-\boldsymbol{x}_j(t))\le\frac{\alpha}{2}||\boldsymbol{\sigma}_{j,t}^T||^2+\frac{||\boldsymbol{x}_j(t+1)-\boldsymbol{x}_j(t)||_2^2}{2\alpha}. \tag{93}$$

Observe that:

$$g_j(\boldsymbol{x}_j^*(t)) - g_j(\boldsymbol{x}_j(t)) = \boldsymbol{p} \cdot (\boldsymbol{x}_j^*(t) - \boldsymbol{x}_j(t)), \tag{94}$$

where $\boldsymbol{p} = \{p_1, p_2, \ldots, p_M\}$. Together this result with Eq. (92) and Eq. (93) yields:

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{j\in J_t}(\Psi_{j,t} - \Phi_{j,t}) \\
\leq &\sum_{t=1}^{T}\sum_{j\in J_t}\frac{\alpha}{2}||\boldsymbol{\sigma}_{j,t}||_2^2 + \lambda_j(t+1)\boldsymbol{p}\cdot\left(\boldsymbol{x}_j^*(t+1) - \boldsymbol{x}_j(t+1)\right) + \\
&\sum_{j=1}^{N}\sum_{t=a_j+1}^{d_j}\frac{\left\|\boldsymbol{x}_j^*(t+1) - \boldsymbol{x}_j(t)\right\|_2^2}{2\alpha} - \frac{\left\|\boldsymbol{x}_j^*(t+1) - \boldsymbol{x}_j(t+1)\right\|_2^2}{2\alpha}.
\end{aligned} \tag{95}$$

Since there are at most N job arrivals during a job's lifetime, $\boldsymbol{x}_j^*(t+1)$ can thus change at most N times during the time interval $[a_j, d_j]$. As such, the second term in the R.H.S. of Eq. (95) can be upper bounded by:

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{t=a_j+1}^{d_j}\frac{\left\|\boldsymbol{x}_j^*(t+1) - \boldsymbol{x}_j(t)\right\|_2^2}{2\alpha} - \frac{\left\|\boldsymbol{x}_j^*(t+1) - \boldsymbol{x}_j(t+1)\right\|_2^2}{2\alpha} \\
\leq &\sum_{t:t\in\{a_j\}}\sum_{j=1}^{N}\frac{\left\|\boldsymbol{x}_j^*(t+1)\right\|_2^2}{2\alpha} + \frac{\left\|\boldsymbol{x}_j(t)\right\|_2^2}{2\alpha} \\
\leq &\frac{MN}{\alpha}.
\end{aligned} \tag{96}$$

Let $\lambda^{\max} = \max_{j\in\{1,2,\ldots,N\}}\max_{t\in\{1,2,\ldots,T\}}\lambda_j(t)$, the first term in the R.H.S. of Eq. (95) can be upper bounded by:

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{j\in J_t}\frac{\alpha}{2}||\boldsymbol{\sigma}_{j,t}||_2^2 + \lambda_j(t+1)\boldsymbol{p}\cdot\left(\boldsymbol{x}_j^*(t) - \boldsymbol{x}_j(t)\right) \\
\leq &\frac{\alpha MNTD^2}{2} + \lambda^{\max}\sum_{j=1}^{N}\sum_{t=a_j+1}^{d_j}\boldsymbol{p}\cdot\boldsymbol{x}_j^*(t) \\
\overset{(a)}{\leq} &\frac{\alpha MNTD^2}{2} + \frac{\mu M^2 + \mu\beta^{\max}M}{\beta^{\min}}\cdot\sum_{j=1}^{N}B_j,
\end{aligned} \tag{97}$$

where (a) is due to that $||\boldsymbol{\sigma}_{j,t}|| \leq \sqrt{M}D$ with $\lambda^{\max} \leq \max_{t\in 1,2,\ldots,T}||\boldsymbol{\lambda}(t)||$ and $\sum_{t=a_j+1}^{d_j}\boldsymbol{p}\cdot\boldsymbol{x}_j^*(t) \leq B_j$.

Substitute Eq. (96) and Eq. (97) into Eq. (95) and parallel Eq. (49) yields:

$$\begin{aligned}
&\sum_{j=1}^{N}\sum_{t\geq a_j+1}^{d_j} f\Big(\sum_{i=1}^{M}\widehat{\theta}_i^t x_j^{i,*}(t)\Big) - (d_j - a_j)f_j\Big(\frac{\sum_{t=a_j+1}^{d_j}\sum_{i=1}^{M}\widehat{\theta}_i^t x_j^i(t)}{d_j - a_j}\Big) \\
\leq &\frac{\alpha MNTD^2}{2} + \frac{\mu M^2 + \mu\beta^{\max}M}{\beta^{\min}}\cdot\sum_{j=1}^{N}B_j + \frac{MN}{\alpha}.
\end{aligned} \tag{98}$$

Applying the same result as that in Eq. (99), we have, with prob. $(1 - MTe^{-\Omega(\gamma)})$,

$$\begin{aligned}
&\Bigg|\sum_{j=1}^{N}(d_j - a_j)f_j\Big(\frac{\sum_{t=a_j+1}^{d_j}\sum_{i=1}^{M}\widehat{\theta}_i^t x_j^i(t)}{d_j - a_j}\Big) \\
&- (d_j - a_j)f_j\Bigg(\frac{\sum_{t=a_j+1}^{d_j}\sum_{i=1}^{M}\theta_i^t\widetilde{x_j^i(t)}}{d_j - a_j}\Bigg)\Bigg| \\
\leq &O(M\pi\sqrt{\gamma T}) + 2\pi\gamma.
\end{aligned} \tag{99}$$

This completes the proof of the second part of the theorem. □

In the multi-job case, the optimal solution can change across time slots and thus, it is difficult to bound the term $\sum_{t=1}^{T}\sum_{j\in J_t}\lambda_j(t+1)g_j(\boldsymbol{x}_j^*(t+1))$ since $g_j(\boldsymbol{x}_j^*(t+1))$ may exceed zero. In the proof, we bound this term by utilizing the budget constraint for each job, i.e.,

$$\sum_{i=1}^{M}\sum_{t=a_j+1}^{d_j}p_i\cdot x_j^{i,*}(t) \leq B_j, \quad \forall j. \tag{100}$$

The regret bound may not be sublinear in T if $(d_j - a_j)$ is the order of time T. However, when $a_j = 0$ and $d_j = T$ for all j, we can adopt the same analysis as that in the single-job case to yield a much tighter regret bound.

Corollary 1. *When all the jobs have the same arrival time and the same deadline, i.e., $a_j = 0$ and $d_j = T$ for all j. The regret defined in (6) is upper bounded by:*

$$Reg_T \leq O\left(M\pi\sqrt{T\ln\frac{MT}{\delta}}\right) + \frac{MN\pi\sqrt{T}}{2}. \tag{101}$$

C. Extension to more general settings

The above study considers each server can only hold one task at a time. In this part, we discuss how to extend the original setting to a more general case, i.e., each server can hold multiple tasks and jobs have different resource demands. Specifically, we let r_j denote the resource demand of tasks from job j and the capacity of sever i is denoted by C_i. Under this setting, Constraint (3) is changed to:

$$\sum_{j\in J_t}r_j x_j^i(t) \leq C_i, \quad \forall i, t. \tag{102}$$

To be more general, we also limit the maximum number of tasks from job j that can be executed in parallel to be u_j, i.e.,

$$\sum_{i=1}^{M}x_j^i(t) \leq u_j, \quad \forall j \in J_t, t. \tag{103}$$

In this setting, we still adopt the gradient descent methods to design scheduling algorithms. After running the primal and

dual updates given by Eqs. (69) and (73), we then perform the following projection operation via solving a convex program:

$$\min_{\boldsymbol{z}} \sum_{j\in J_t} \left(z_j^i - y_j^i(t)\right)^2 \tag{P5}$$

$$\text{s.t.} \sum_{j\in J_t} r_j z_j^i \le C_i, \ \forall i. \tag{104}$$

$$\sum_{i=1}^{M} z_j^i \le \mu_j, \quad 0 \le z_j^i \le 1, \ \forall j \in J_t. \tag{105}$$

Similarly, the update of $x_j^i(t)$ can be attained by taking $x_j^i(t) = z_j^{i,*}$ where $\{z_j^{i,*}\}_{i,j}$ is the optimal solution to P5. At this moment, we should be more careful conducting the sampling procedure to round each $x_j^i(t)$ to a random variable $\widetilde{x_j^i(t)}$ such that, the rounded values satisfy Constraint (102) and (103). To achieve this, we incorporate prior work on randomized rounding schemes (RRS) for linear programs, e.g., [40].

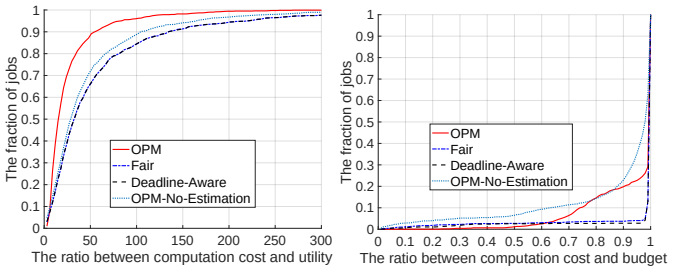
The RRS scheme works as follows. It takes a feasible fractional solution $\boldsymbol{x} = \left\{x_j^i(t) \middle| j \in J_t, i \in \{1,2,\ldots,M\}\right\}$ and the linear equations describing $\boldsymbol{x}$ as inputs (i.e., Eq.(102) and (103)), and produces a random vector $\widetilde{\boldsymbol{x}}$, which also satisfies the linear equations. The RRS scheme returns an unbiased result, i.e., $\mathbb{E}[\widetilde{\boldsymbol{x}}] = \boldsymbol{x}$. More importantly, $\widetilde{\boldsymbol{x}}$ also satisfies the corresponding concentration inequality. As such, the analysis in Section V-B is still applicable to bound the regret performance of the designed online algorithm under this new setting.

VI. Performance Evaluation

In this section, we run extensive simulations driven by Google job trace during a 10-hour period to evaluate the performance of OPM in a cluster that consists of 1000 heterogeneous machines.

Simulation Setup: From the Google traces, we obtain the arrival time a_j and deadline d_j for each job j. The budget of job j is set to $B_j = q_j \cdot (d_j - a_j)$ where q_j is uniformly distributed in [2,100]. Similar to [3], the utility of job j is modeled as a concave function of the form $f_j(X_j) = v_j X_j^{\beta}$. Here, we consider $\beta \in [0.5, 1)$ and the coefficient v_j is generated from a uniform distribution ranging from 1 to 5.

As pointed out by the work of Kondo [10], the Gamma distribution is a good fit for the failure model of most parallel and distributed computing clusters. As such, we apply the Gamma distribution to generate machine service rates in a cluster consisting of 1000 machines over a period which lasts 50,000 units of time. To be more specific, we categorize the processing periods of each machine into two types, namely, the available period (AP) and the unavailable period (UP). During an available period, the service rate of machine i is uniformly distributed in $[0.7, 1]$. By contrast, when the machine is processing jobs in an unavailable period, its rate is uniformly distributed in $[0, 0.1]$. In addition, we adopt the statistics of the trace data collected from a computational grid platform in France [10] to generate a series of available and unavailable periods for each machine independently. The length of an AP is Gamma distributed with $k = 0.34$ and $\theta = 94.35$ where k and θ are the shape parameter and scale parameter respectively. By contrast, the length of an UP is Gamma distributed with $k = 0.19$ and $\theta = 39.92$. In addition, the price for each machine is set to two times of the mean service rate.



(a) The CDF of jobs with respect to computation cost to utility ratios when $\beta = 0.5$.

(b) The CDF of jobs with respect to computation cost to budget ratios when $\beta = 0.5$.

TABLE I: Overall Utility achieved under Different schemes when $\beta = 0.5, 0.6, 0.7$.

	$\beta = 0.5$	$\beta = 0.6$	$\beta = 0.7$
OPM	2.23×10^5	6.18×10^5	1.71×10^6
OPM without estimation	2.06×10^5	5.67×10^5	1.54×10^6
Fair Algorithm	2.02×10^5	5.09×10^5	1.29×10^5
Deadline-aware Algorithm	2.00×10^5	5.08×10^5	1.28×10^6

Baseline Algorithms: We use the following three algorithms as the baselines for comparison with our proposed OPM Algorithm:

- OPM Without Machine Estimations: This approach adopts the same online optimization method as that in OMP. However, it does not estimate the machine service variability and treats all machine service rates as the same constant.
- Fair Scheduler: We implement the fair scheduler under which the cluster resource is shared equally among all ongoing jobs whenever the budget has not run out.
- Deadline-Aware Scheduler: Jobs with tighter deadlines are given scheduling priorities, and we assign as many machines to such jobs as possible unless the corresponding budget has run out.

To make a fair comparison across all the schemes, we do not violate the budget constraint when implementing the OPM algorithm. Once the amount of consumed resources of a job reaches the budget limit, the algorithm shall remove it from the scheduling list even its deadline is not due yet.

Evaluation Metrics: We evaluate the overall utility achieved by all jobs contained in the traces across different algorithms. In addition, we study the ratio between the amount of computation cost and the achieved utility for all jobs. Moreover, we also investigate a study on the migration costs.

We first characterize the utility achieved for all jobs under different algorithms to evaluate the scheduling efficiency when $\beta = 0.5$. More specifically, we study the ratio between the job computation cost and the achieved utility in terms of the cumulative density function (CDF) in Fig. 1a. It shows that, more than 90% of jobs achieve a ratio of less than 50

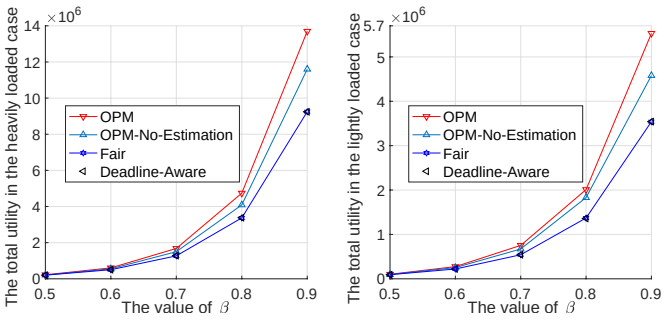
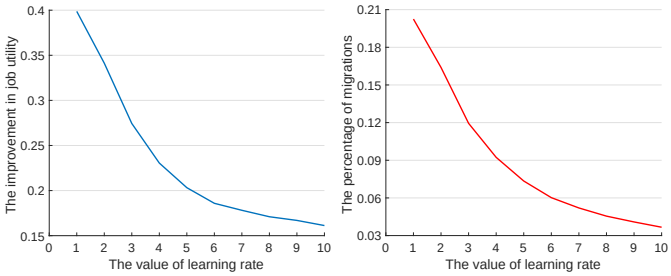


Fig. 2: The overall job utility achieved under all schemes with different utility functions.



(a) Utility improvement w.r.t. the Fair Scheduler

(b) Migration cost

Fig. 3: The impact on of learning rate on the job performance.

under OPM whereas only 65% of jobs achieve this ratio under Fair Algorithm and Deadline-Aware Algorithm. From Fig. 1a, we also observe that, the mechanism of machine service estimation can improve the performance of OPM substantially, since only 72% of jobs can achieve the computation cost to utility ratio of 50 when where is no estimation for machine service rates in OPM.

In Fig. 1b, we depict the CDF for all jobs with respect to the percentage of consumed budget, i.e., the ratio between the computation cost and the job budget. This metric characterizes the cost effectiveness under different algorithms since a higher percentage means a larger computation cost incurred and thus a lower effectiveness. One can note in Fig. 1b that the OPM Algorithm is the most effective one when compared to the other three baselines. In particular, more then 20% of jobs consume a computation cost of less than 90% of their budget. By contrast, only 3% of jobs consume a computation cost of this level under Fair Algorithm and Deadline-aware Algorithm.

We proceed to evaluate the key optimization metric in this paper, i.e., the overall job utilities achieved under different algorithms when the utility function changes. Table. I depicts the comparison results when β changes from 0.5 to 0.7. It shows that, when $\beta = 0.5$, OPM manages to increase the overall utility by 10% and 11% comparing to Fair and Deadline-Aware Scheduler respectively. In addition, OPM can dramatically increase the achieved overall utility by 30% and 34% comparing to the same baselines when β becomes 0.7. It is also noteworthy that, with the use of our designed method for estimating the machine service rates, OPM can increase the performance by nearly 11% according to this metric under all settings of β. In the job traces, the machine service rates are highly correlated and each machine stays in a normal/abnormal state for a long time before it goes to another state. Nevertheless, the UCB-based estimation can still help to increase the overall job utility substantially.

We also evaluate the impact of the loading conditions on the efficiency of OPM and the results are illustrated in Fig. 2. The left panel corresponds to the heavily-loaded case and the right panel corresponds to the lightly-loaded case where we only keep half of the jobs from the original trace. We tune β from 0.5 to 0.9 in the job utility functions and evaluate the overall utility achieved under different β. As shown in Fig. 2, in the heavily-loaded case, OPM outperforms the Fair Scheduler by 47% in terms of this metric when $\beta = 0.9$. By contrast, OPM increases the overall utility by 56% comparing to the Fair Scheduler, when the cluster is lightly loaded and $\beta = 0.9$. These results indicate that, the improvement of OPM over baseline schemes is more significant when the cluster is lightly-loaded. The reason behind is that, OPM can find better job-to-machine matchings when the cluster resource is enough.

Our designed online resource allocation algorithm can potentially migrate tasks between machines. As such, we also study the impact of the learning rate α on the migration cost as well as the improvement of job utility when the cluster is heavily loaded and $\beta = 0.8$. We depict the evaluation results in Fig. 3. As shown in Fig. 3a, the improvement of the overall job utilities under OPM with respect to the Fair Scheduler decreases with α. In particular, when we increase the value of α by ten times, the utility improvement of OPM drops from 40% to 17%. Fig. 3b illustrates the migration costs of OPM under different settings of α. When α is small, migration happens on nearly 20% of servers. By contrast, when the value of α increases by ten times, only 3% of servers have migrations. As such, tuning α up shall significantly reduce the number of migrations and $\alpha = 6$ will lead to a good trade-off between the overall job utility and the migration costs.

VII. Conclusion and Future Works

This paper makes the first attempt to address the impact of machine service variability in parallel/distributed computing clusters from a bandit perspective. Our primary contribution was to provide the fundamental understanding on designing efficient algorithms via combining bandit methods with OCO techniques. Extensions of this work to other bandit problems with more general rewards [41], e.g, the reward is treated as the training loss for approximation jobs rather than the amount of work completed, are likely next steps towards developing the fundamental theory and associated algorithms. Possible extension of this work also includes dynamic resource scaling for long-running applications such as training a deep-learning job. Moreover, applying the Thompson Sampling Method [42] to estimate the machine service rate may also be an interesting future research direction.

References

[1] Huanle Xu, Yang Liu, Wing Cheong Lau, Jun Guo, and Alex Liu. Efficient online resource allocation in heterogeneous clusters with machine variability. In *Proceedings of IEEE Infocom*, 2019.

[2] Ganesh Ananthanarayanan, Michael Chien-Chun Hung, Xiaoqi Ren, and Ion Stoica. Grass: Trimming stragglers in approximation analytics. In *NSDI*, April 2014.

[3] Zizhan Zheng and Ness B. Shroff. Online multi-resource allocation for deadline sensitive jobs with partial values in the cloud. In *Proceedings of IEEE Infocom*, 2016.

[4] Ruiting Zhou, Zongpeng Li, and Chuan Wu. Scheduling frameworks for cloud container services. *IEEE/ACM Transactions on Networking*, 2018.

[5] J. Brutlag. Speed matters for google web search. http://services.google.com/fh/files/blogs/googledelayexp.pdf, 2009.

[6] Mu Li, David G. Andersen, Jun Woo Park, Alexander J. Smola, and Amr Ahmed. Scaling distributed machine learning with the parameter server. In *Proceedings of OSDI*, pages 583–598, 2014.

[7] Mu Li, David G. Andersen, Alexander Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. In *Proceedings of NIPS*, 2014.

[8] Chen Chen, Wei Wang, and Bo Li. Performance-aware fair scheduling: Exploiting demand elasticity of data analytics jobs. In *Proceedings of IEEE Infocom*, 2018.

[9] Gunho Lee, Byung-Gon Chun, and Randy H. Katz. Heterogeneity-aware resource allocation and scheduling in the cloud. In *HotCloud*, 2011.

[10] Derrick Kondo, Bahman Javadi, Alexandru Iosup, and Dick Epema. The failure trace archive: Enabling comparative analysis of failures in diverse distributed systems. In *CCGrid*, pages 398–407, 2010.

[11] Eric Heien, Derrick Kondo, Ana Gainaru, Dan LaPine, Bill Kramer, and Franck Cappello. Modeling and tolerating heterogeneous failures in large parallel system. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, 2011.

[12] Huanle Xu, Gustavo de Veciana, and Wing Cheong Lau. Addressing job processing variability through redundant execution and opportunistic checkpointing: A competitive analysis. In *Proceeding of Infocom*, 2017.

[13] Ming Mao and Marty Humphrey. Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, 2011.

[14] Yuxiong He, Sameh Elnikety, James Larus, and Chenyu Yan. Zeta: Scheduling interactive services with partial execution. In *Proceedings of Socc*, 2012.

[15] Wei Chen, Yajun Wang, and Yang Yuanu. Combinatorial multi-armed bandit: General framework, results and applications. In *Proceeding of ICML*, 2013.

[16] Shipra Agrawal and Nikhil R. Devanur. Linear contextual bandits with knapsacks. In *Proceedings of NIPS*, 2016.

[17] Wei Chen, Wei Hu, Fu Li, Jian Li, Yu Liu, and Pinyan Lu. Combinatorial multi-armed bandit with general reward functions. In *Proceeding of NIPS*, 2016.

[18] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of ICML*, 2003.

[19] Elad Hazan, Adam Kalai, and Satyen. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, pages 2503–2528, 2007.

[20] T. Chen, Q. Ling, and G. B. Giannakis. An online convex optimization approach to dynamic network resource allocation. arXiv preprint:1701.03974, 2017.

[21] Yixin Bao, , Yanghua Peng, Chuan Wu, and Zongpeng Li. Online job scheduling in distributed machine learning clusters. In *Proceedings of IEEE Infocom*, 2018.

[22] Navendu Jain, Ishai Menache, Jeseph Naor, and Jonathan Yaniv. Near-optimal scheduling mechanisms for deadline-sensitive jobs in large computing clusters. In *ACM Transactions on Parallel Computing*, 2015.

[23] Brendan Lucier, Ishai Menache, Joseph (Seffi) Naor, and Jonathan Yaniv. Efficient online scheduling for deadline-sensitive jobs. In *Proceedings of SPAA*, 2013.

[24] Robabeh Ghafouri, Ali Movaghar, and Mehran Mohsenzadeh. A budget constrained scheduling algorithm for executing workflow application in infrastructure as a service clouds. In *Peer-to-Peer Networking and Applications*, 2018.

[25] Qian Zhu and Gagan Agrawal. Resource provisioning with budget constraints for adaptive applications in cloud environments. In *Proceedings of HPDC*, 2010.

[26] Da Wang, Gauri Joshi, and Gregory Wornell. Efficient straggler replication in large-scale parallel computing. In *ACM Trans. on Modeling and Perf. Eval. of Comp. Systems*, 2019.

[27] Kristen Gardner, Samuel Zbarsky, Sherwin Doroudi, Mor Harchol-Balter, and Esa Hyytiä and Alan Scheller-Wolf. Reducing latency via redundant requests: Exact analysis. In *Proceedings of ACM Sigmetrics*, 2015.

[28] Peter Auer. Using confidence bounds for exploitation-exploration tradeoffs. In *Journal of Machine Learning Research*, 2002.

[29] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *Journal of the ACM*, 2018.

[30] S Agrawal and NR Devanur. Bandits with concave rewards and convex knapsacks. In *ACM Conference on Economics & Computation*, 2014.

[31] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2013.

[32] M. Mahdavi, R. Jin, and T. Yang. Trading regret for efficiency: Online convex optimization with long term constraints. *Journal of Machine Learning Research*, 13:2503–2528, 2012.

[33] H. Yu and M. J. Neely. A low complexity algorithm with $O(\sqrt{T})$ regret and constraint violations for online convex optimization with long term constraints. arXiv preprint:1604.02218, 2016.

[34] Rodolphe Jenatton, Jim C. Huang, and Cedric Archambeau. Adaptive algorithms for online convex optimization with long-term constraints. In *ICML*, 2016.

[35] M. J. Neely and H. Yu. Online convex optimization with time-varying constraints. arXiv preprint:1702.04783, 2017.

[36] Chen Chen, Wei Wang, and Bo Li. Round-robin synchronization: Mitigating communication bottlenecks in parameter servers. In *Proceedings of IEEE Infocom*, 2019.

[37] Moshe Babaioff, Shaddin Dughmi, Robert D. Kleinberg, and Aleksandrs Slivkins. Dynamic pricing with limited supply. In *ACM Transactions on Economics and Computation*, 2015.

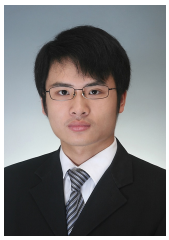
[38] Huanle Xu, Pili Hu, Wing Cheong Lau, Qiming Zhang, and Yang Wu. DPCP: A protocol for optimal pull coordination in decentralized social networks. In *Proceedings of IEEE Infocom*, 2015.

[39] Chi Jin, Praneeth Netrapalli, and Michael I. Jordan. Accelerated gradient descent escapes saddle points faster than gradient descent. In *Proceedings of Machine Learning Research*, 2018.

[40] Karthik Abinav Sankararaman and Aleksandrs Slivkins. Combinatorial semi-bandits with knapsacks. In *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[41] Lihong Li, Yu Lu, and Dengyong Zhou. Provably optimal algorithms for generalized linear contextual bandits. In *Proceedings of ICML*, 2017.

[42] Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *25th Annual Conference on Learning Theory*, 2012.

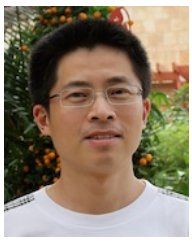


Huanle Xu received his BSc(Eng) degree from the Department of Information Engineering, Shanghai Jiao Tong University (SJTU) in 2012 and his Ph.D. degree from the Department of Information Engineering, The Chinese University of Hong Kong (CUHK) in 2016, and is currently a Postdoc Fellow at CUHK. His primary research interests focus on Job Scheduling and Resource Allocation in Cloud Computing, Decentralized social networks, Parallel Graph Algorithms and Machine Learning. Huanle is particularly interested in designing and implementing wonderful algorithms for large-scale systems and their applications using optimization tools and theories.

Yang Liu Yang Liu received his BSc(Eng) degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University (SJTU) in 2016 and is currently a PhD student supervised by Prof. Wing Cheong Lau in the Department of Information Engineering, The Chinese University of Hong Kong (CUHK). His research interests includes: Job Scheduling and Resource Management in the Cloud and Computing Cluster, Online Learning and Multi-armed Bandit Algorithms. He is particularly interested in designing and implementing resource management algorithms for distributed systems.

Wing Cheong Lau is currently an Associate Professor in the Department of Information Engineering and the Director of the Mobile Technologies Center at the Chinese University of Hong Kong (CUHK). Before joining CUHK, he spent 10 years in the US with Bell Labs, Holmdel and Qualcomm, San Diego. Wing received his BSEE degree from the University of Hong Kong and MS and PhD degrees in Electrical and Computer Engineering from the University of Texas at Austin. His research interests include Networking Protocol Design and Performance Analysis, Network/ Systems Security, Mobile Computing and System Modeling. His recent research projects include: Resource Allocation and Management for Data-center-scale Computing, Online Social Network Privacy and Vulnerabilities, Authenticated 2D barcodes, Decentralized Social Networking protocols/systems. He is/has been on the Technical Program Committee for various international conferences including ACM Sigmetrics Mobihoc, IEEE Infocom, SECON, ICC, Globecom, WCNC, VTC, and ITC. He also served as the Guest Editor for the Special Issue on High-Speed Network Security of the IEEE Journal of Selected Areas in Communications (JSAC). Wing holds 19 US patents. Related research findings have culminated in more than 100 scientific papers in major international journals and conferences. Wing is a Senior Member of IEEE and a member of ACM and Tau Beta Pi.

Tiantong Zeng received his BSc(Eng) degree from the Department of Information Engineering, The Chinese University of Hong Kong (CUHK) in 2020, and is currently will join Tencent Technology as an Engineer. His primary research interests focus on deep learning and big data analytics.



Jun Guo received the Ph.D. degree in electrical and electronic engineering from The University of Melbourne, Australia, in 2006. He was with the School of Computer Science and Engineering, The University of New South Wales, Australia, as a Senior Research Associate, from 2006 to 2008. He was supported by the Australian Research Council, through an Australian Postdoctoral Fellowship, from 2009 to 2011. From 2012 to 2016, he was with the Department of Electronic Engineering, City University of Hong Kong. In 2017, he joined the School of Cyberspace Security, Dongguan University of Technology, China. His current research interests include networking and security.

Alex X. Liu received his Ph.D. degree in Computer Science from The University of Texas at Austin in 2006, and is currently an adjunct professor of Dongguan University of Technology. Before that, he was a Professor of the Department of Computer Science and Engineering at Michigan State University. He received the IEEE & IFIP William C. Carter Award in 2004, a National Science Foundation CAREER award in 2009, the Michigan State University Withrow Distinguished Scholar (Junior) Award in 2011, and the Michigan State University Withrow Distinguished Scholar (Senior) Award in 2019. He has served as an Editor for IEEE/ACM Transactions on Networking, and he is currently an Associate Editor for IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Mobile Computing, and an Area Editor for Computer Communications. He has served as the TPC Co-Chair for ICNP 2014 and IFIP Networking 2019. He received Best Paper Awards from SECON-2018, ICNP-2012, SRDS-2012, and LISA-2010. His research interests focus on networking, security, and privacy. He is an IEEE Fellow and an ACM Distinguished Scientist.