Efficient Distributed Machine Learning via Combinatorial Multi-Armed Bandits

Maximilian Egger*, **Rawad Bitar***, **Antonia Wachter-Zeh***, and **Deniz Gündüz**†
*Technical University of Munich, Germany, {maximilian.egger, antonia.wachter-zeh, rawad.bitar}@tum.de
†Imperial College London, United Kingdom, {d.gunduz}@imperial.ac.uk

***Abstract*—We consider the distributed stochastic gradient descent problem, where a main node distributes gradient calculations among n workers from which at most $b \leq n$ can be utilized in parallel. By assigning tasks to all the workers and waiting only for the k fastest ones, the main node can trade-off the error of the algorithm with its runtime by gradually increasing k as the algorithm evolves. However, this strategy, referred to as *adaptive k-sync*, can incur additional costs since it ignores the computational efforts of slow workers. We propose a cost-efficient scheme that assigns tasks only to k workers and gradually increases k. As the response times of the available workers are unknown to the main node a priori, we utilize a combinatorial multi-armed bandit model to learn which workers are the fastest while assigning gradient calculations, and to minimize the effect of slow workers. Assuming that the mean response times of the workers are independent and exponentially distributed with different means, we give empirical and theoretical guarantees on the regret of our strategy, i.e., the extra time spent to learn the mean response times of the workers. Compared to adaptive k-sync, our scheme achieves significantly lower errors with the same computational efforts while being inferior in terms of speed.**

I. Introduction

We consider a distributed machine learning setting, in which a central entity, referred to as the *main node*, possesses a large amount of data on which it wants to run a machine learning algorithm. To speed up the computations, the main node distributes the computational tasks to several *worker* machines. The workers compute smaller tasks in parallel and send back their results to the main node, which then aggregates the results to obtain the desired large computation. A naive distribution of the tasks to the workers suffers from the presence of stragglers, i.e., slow or even unresponsive workers [1], [2].

The negative effect of the stragglers can be mitigated by assigning redundant computations to the workers and ignoring the responses of the slowest ones, e.g., [3], [4]. However, in gradient descent algorithms, assigning redundant tasks to the workers can be avoided when a (good) estimate of the gradient is sufficient. On a high level, gradient descent is an iterative algorithm requiring the main node to compute the gradient of a loss function at every iteration. Simply ignoring the stragglers is equivalent to stochastic gradient descent (SGD) [5], [6], which advocates computing an estimate of the gradient of the loss function at every iteration [2], [7]. As a result, SGD trades-off the time spent per iteration with the total number of iterations for convergence, or until a desired result is reached. The authors of [8] show that for distributed SGD algorithms, it is faster for the main node to assign tasks to all the workers but wait for only a small subset of the workers to return their results. As the algorithm evolves in iterations, in order to improve the convergence speed, the main node increases the number of workers it waits for. Despite reducing the runtime of the algorithm, i.e., the total time needed to reach the desired result, this strategy requires the main node to occupy (and pay for) all the available workers while only using the computations of the fastest ones.

In this work, we take into account the cost of employing the workers for computations, independently of whether the results of their computations are used or not. In contrast to [8], we propose a cost-efficient scheme that distributes tasks only to the fastest workers and waits for completion of all their computations. The downside of our scheme is that, in practice, the main node does not know which workers are the fastest. To this end, we introduce the use of a stochastic multi-armed bandit (MAB) framework to learn the speeds of the workers while efficiently assigning them computational tasks. Stochastic MABs, introduced in [9], are iterative algorithms initially designed to maximize the gain of a user gambling with multiple slot machines, termed "armed bandits". At each iteration, the user is allowed to pull one arm from the available armed bandits. Each arm pull yields a random reward following a known distribution with unknown mean. The user wants to design a strategy to learn the expected reward of the arms while maximizing the accumulated rewards.

Following the literature on distributed computing [3], [10], we model the response times of the workers by independent and exponentially distributed random variables. We additionally assume that the workers are heterogeneous, i.e., have different mean response times. To apply MABs to distributed computing, we model the rewards by the response times and aim to minimize the rewards. Under this model, we show that compared to adaptive k-sync, using a MAB to learn the mean response times of the workers on the fly cuts the average cost (reflected by the total number of worker employments) but comes at the expense of significantly increasing the total runtime of the algorithm.

A. Related Work

1) Distributed Gradient Descent: Assigning redundant tasks to the workers and running distributed gradient descent is

This project has received funding from a TUM-ICL JADS project (COALESCENCE), the German Research Foundation (DFG) under Grant no. WA3907/7-1 and was partly supported by the TUM - Institute for Advanced Studies, funded by the German Excellence Initiative and European Union Seventh Framework Programme under Grant Agreement No. 291763. D. Gündüz received funding from the UK EPSRC through CHIST-ERA project CONNECT (CHISTERA-18-SDCDN-001, EPSRC-EP/T023600/1).

known as gradient coding [4], [11]–[15]. Approximate gradient coding is introduced to reduce the required redundancy and run SGD in the presence of stragglers [16]–[23]. The schemes in [14], [15] use redundancy but no coding to avoid encoding/decoding overheads. In contrast, [2], [7], [8] advocate running distributed SGD without redundant task assignment to the workers. Assigning redundant computations to the workers increases the computation time spent per worker and may slow down the overall computation process. In [7], the convergence speed of the algorithm is analyzed in terms of the wall-clock time rather than the number of iterations. It is assumed that the main node waits for k out of n workers and ignores the rest. The authors of [8] show that gradually increasing k, i.e., gradually decreasing the number of tolerated stragglers as the algorithm evolves, increases the convergence speed of the algorithm. In this work, we consider a similar analysis to the one in [8]; however, instead of assigning tasks to all the workers and ignoring the stragglers, we require the main node to only employ (assign tasks to) the required amount of workers. To learn the speeds of the workers and choose the fastest ones, we use ideas from the literature on MABs.

2) MABs: Since the problem introduction in [9], MABs have been extensively studied for decision-making under uncertainty. A MAB strategy is evaluated by its *regret* defined as the difference between the actual cumulative reward and the one that could be achieved should the user know the expected reward of the arms a priori. The works of [24], [25] introduced the use of upper confidence bounds (UCBs) based on previous rewards to decide which arm to pull at each iteration. Those schemes are said to be asymptotically optimal since their regret becomes negligible as the number of iterations goes to infinity. In [26], the regret of a UCB algorithm is bounded for a finite number of iterations. While most of the works assume a finite support for the reward, MABs with unbounded rewards were studied in [27], [28]. In the class of combinatorial MABs (CMABs), the user is allowed to pull multiple arms at each iteration. The authors of [29] extended the asymptotically efficient allocation rules of [24] to a CMAB scenario. General frameworks for the CMAB with bounded reward functions are investigated in [30]–[32]. The analysis in [33], [34] for linear reward functions with finite support is an extension of the classical UCB strategy, and comes closest to our work.

B. Contributions

We employ a MAB model to reduce the cost of distributed gradient descent. Our cost-efficient policy increases the number of employed workers as the algorithm evolves. We show theoretically and through numerical simulations that the introduced scheme reduces the cost of distributing the computations at the expense of increasing the overall run-time of the algorithm.

II. System Model and Preliminaries

Notations. Vectors and matrices are denoted in bold lower and upper case letters, e.g., $\mathbf{z}$ and $\mathbf{Z}$, respectively. For integers κ, τ with $\kappa < \tau$, the set $\{\kappa, \kappa+1, \dots, \tau\}$ is denoted by $[\kappa, \tau]$. Sub-gamma distributions are expressed by shape α and rate β, i.e., SubΓ (α, β), and sub-Gaussian distributions by variance σ^2, i.e., SubG $\left(\sigma^2\right)$. The identity function $\mathbb{1}\{z\}$ is 1 if z is true, and 0 otherwise. Throughout the paper, we use the terms arm and worker interchangeably.

We denote by $\mathbf{X} \in \mathbb{R}^{m \times d}$ a data matrix with m samples, where each sample $\mathbf{x}_\ell \in \mathbb{R}^d$, $\ell \in [1, m]$, is the ℓ-th row of $\mathbf{X}$ and by $\mathbf{y} \in \mathbb{R}^m$ the vector containing the labels y_ℓ for every sample $\mathbf{x}_\ell$. The goal is to find a model $\mathbf{w} \in \mathbb{R}^d$ that minimizes an additively separable loss function $F(\mathbf{X}, \mathbf{y}, \mathbf{w}) \coloneqq \sum_{\ell=1}^{m} F(\mathbf{x}_\ell, y_\ell, \mathbf{w})$, i.e., to find $\mathbf{w}_\star = \arg\min_{\mathbf{w} \in \mathbb{R}^d} F(\mathbf{X}, \mathbf{y}, \mathbf{w})$.

To enable flexible distributed computing schemes that use at most b workers[1] out of n available in parallel, we employ mini-batch gradient descent. At each iteration j, the main node employs a set of workers, indexed by $\mathcal{A}(j)$, $|\mathcal{A}(j)| \leq b$. Every worker $i \in \mathcal{A}(j)$ computes a partial gradient estimate $\nabla F(\mathbf{X}_{i,j}, \mathbf{y}_{i,j}, \mathbf{w}_j)$ using a random subset $\mathbf{X}_{i,j}$ (batch) of $\mathbf{X}$ consisting of $s = \frac{m}{b}$ samples, i.e., $\mathbf{X}_{i,j} \in \mathbb{R}^{s \times d}$. The data $\mathbf{X}$ and $\mathbf{y}$ is stored on a shared memory, and can be accessed by all n workers. The main node waits for $\mathcal{R}(j) \subseteq \mathcal{A}(j)$ responsive workers and updates the model $\mathbf{w}$ as

$$\mathbf{w}_{j+1} = \mathbf{w}_j - \frac{\eta}{|\mathcal{R}(j)| \cdot s} \sum_{i \in \mathcal{R}(j)} \nabla F(\mathbf{X}_{i,j}, \mathbf{y}_{i,j}, \mathbf{w}_j),$$

where η denotes the learning rate. According to [7], [35], fixing the value of $|\mathcal{R}(j)| = k$ and running j iterations of gradient descent with a mini-batch size of $s \cdot k$ results in an expected deviation from the optimal loss $F^\star$ bounded as[2]

$$\begin{aligned} E(k, j) &= \mathbb{E}[F(k, \mathbf{w}_j) - F^\star] \\ &\leq \frac{\eta L \sigma^2}{2cks} + (1 - \eta c)^j \left(F(\mathbf{w}_0) - F^\star - \frac{\eta L \sigma^2}{2cks} \right). \quad (1) \end{aligned}$$

III. Policy and Main Results

We group the iterations into rounds $r \in [1, b]$ such that the main node employs $|\mathcal{A}(j)| = r$ workers and waits for all of them to respond, i.e., $\mathcal{A}(j) = \mathcal{R}(j)$. As in [8], we let each round r run for a predetermined number of iterations. That is, at a switching point $j = T_r$, the algorithm advances to round $r+1$. We define $T_0 \coloneqq 0$, i.e., the algorithm starts in round one, and T_b as the last iteration, i.e., the algorithm ends in round b. The total budget B is defined as $B \coloneqq \sum_{r=1}^{b} r \cdot (T_r - T_{r-1})$, which gives the total number of worker employments.

We assume exponentially distributed response times of the workers $Z_i^j \sim \exp(\lambda_i)$ with rate λ_i and mean $\mu_i = \frac{1}{\lambda_i}$ resulting from the sum of communication and computation delays. The minimum rate of all workers is $\lambda_{\min} \coloneqq \min_{i \in [1,n]} \lambda_i$. The goal is to assign tasks only to the r fastest workers. We denote by policy π_{cmab} a decision process that chooses the r expected fastest workers. The optimal policy $\pi^\star$ assumes knowledge of the μ_i's and chooses r workers with the smallest μ_i's. However, in practice the μ_i's are typically unknown. Our objective is two-fold. First, we want to find confident estimates

[1] For ease of analysis, we assume that b divides m. This can be satisfied by adding all-zero rows to $\mathbf{X}$ and corresponding zero labels to $\mathbf{y}$.

[2] This holds under the assumptions detailed in [7], [35], i.e., a Lipschitz-continuous gradient with bounds on the first and second moments of the objective function characterized by L and σ^2, respectively, strong convexity with parameter c, the stochastic gradient being an unbiased estimate, and a sufficiently small learning rate η.

$\hat{\mu}_i$ of the mean response times μ_i to correctly identify (explore) the fastest workers, and second, we want to leverage (exploit) this knowledge to employ the fastest workers. To trade off this exploration-exploitation dilemma, we utilize the MAB framework where each arm $i \in [1,n]$ corresponds to a different worker and r arms are pulled at each iteration. A superarm $\mathcal{A}^r(j) \subseteq [1,n]$ with $|\mathcal{A}^r(j)| = r$ is the set of indices of the arms pulled at iteration j and $\mathcal{A}^{r,\star}$ is the optimal choice containing the indices of the r workers with the smallest means. Let $\mathcal{W}^r$ denote the set of all superarms with cardinality r. For every worker, we maintain a counter $T_i(j)$ for the number of times this worker has been employed until iteration j, and a counter $M_i(j)$ for the sum of its response times, i.e., $M_i(j) = \sum_{y=1}^{j} \mathbb{1}\{i \in \mathcal{A}^r(y)\} Z_i^y$. Motivated by [26], we describe a policy π_{cmab} such that in iteration $j \in [T_{r-1}+1, T_r]$ we choose the superarm $\mathcal{A}^r(j)$ as the r arms with the lowest lower confidence bounds (LCBs) calculated as

$$\text{LCB}_i(j-1) \coloneqq \begin{cases} -\infty & \text{if } T_i(j-1) = 0 \\ \hat{\mu}_i(j-1) - \theta_i(j-1) & \text{otherwise,} \end{cases}$$

where the confidence radius $\theta_i(j)$ is a parameter of the policy π_{cmab} and is a function of the iteration j and the counter $T_i(j)$, and $\hat{\mu}_i(j) \coloneqq \frac{M_i(j)}{T_i(j)}$. Our choice of $\theta_i(j)$, clarified in the sequel, affects the performance of the policy. The estimates $\hat{\mu}_i$ and the confidence radii are updated after every iteration according to the responses of the chosen workers. For an algorithm summarizing the update procedure, we refer the interested reader to [36, Algorithm 1].

In contrast to most works on MABs, we minimize an unbounded objective, i.e., the overall response time $Z^j_{\mathcal{A}^r(j)} \coloneqq \max_{i \in \mathcal{A}^r(j)} Z_i^j$ in iteration j. This corresponds to waiting for the slowest worker. The expected response time of a superarm $\mathcal{A}^r(j)$ is then defined as $\mu_{\mathcal{A}^r(j)} \coloneqq \mathbb{E}[Z^j_{\mathcal{A}^r(j)}]$ and can be calculated according to Proposition 1.

Proposition 1. *The mean of the maximum of r independently distributed exponential random variables with different means, indexed by a set $\mathcal{I}$, i.e., $Z_p \sim \exp(\lambda_p)$, $p \in \mathcal{I}$, is given as*

$$\mathbb{E}\left[\max_{p \in \mathcal{I}} Z_y\right] = \sum_{S \in \mathcal{P}(\mathcal{I}) \setminus \emptyset} (-1)^{|S|-1} \frac{1}{\sum_{\xi \in S} \lambda_\xi}, \tag{2}$$

with $\mathcal{P}(\mathcal{I})$ denoting the power set of $\mathcal{I}$.

The suboptimality gap of a chosen (super-)arm describes the expected difference in time compared to the optimal choice.

Definition 1. For a superarm $\mathcal{A}^r(j)$ and for $\mathcal{A}^r_{\text{worst}}$ defined as the set of indices of the r slowest workers, we define the following superarm suboptimality gaps

$$\begin{aligned} \Delta_{\mathcal{A}^r(j)} &\coloneqq \mu_{\mathcal{A}^r(j)} - \mu_{\mathcal{A}^{r,\star}}, \\ \Delta_{\mathcal{A}^r,\max} &\coloneqq \mu_{\mathcal{A}^r_{\text{worst}}} - \mu_{\mathcal{A}^{r,\star}}. \end{aligned} \tag{3}$$

For $\nu \leq r$, $\mathcal{A}^r_\nu(j)$ and $\mathcal{A}^{r,\star}_\nu$ denote the indices of the ν^{th} fastest worker in $\mathcal{A}^r(j)$ and $\mathcal{A}^{r,\star}$, respectively. Then, we define the suboptimality gap for the *employed* arms as

$$\delta_{\mathcal{A}^r_\nu(j)} \coloneqq \mu_{\mathcal{A}^r_\nu(j)} - \mu_{\mathcal{A}^{r,\star}_\nu}.$$

We define the minimum suboptimality gap for *all* the arms as

$$\delta_{\min} \coloneqq \min_{r \in [1,b], \mathcal{A}^r \in \mathcal{W}^r} \; \min_{\nu \in [1,r]: \mu_{\mathcal{A}^r_\nu} > \mu_{\mathcal{A}^{r,\star}_\nu}} \delta_{\mathcal{A}^r_\nu}. \tag{4}$$

Definition 2. We define the regret R^π_j of a policy π run until iteration j as the expected difference in run-time of the policy π compared to the optimal policy $\pi^\star$, i.e.,

$$R^\pi_j = \mathbb{E}\left[\sum_{y=1}^{j} Z^y_{\mathcal{A}^r(y)}\right] - \sum_{r \in [1,b]: j > T_{r-1}} (\min\{j, T_r\} - T_{r-1}) \min_{\mathcal{A}^r \in \mathcal{W}^r} \mu_{\mathcal{A}^r}.$$

Our main results are summarized in Theorem 1 and Theorem 2. The proof of Theorem 1 is given in the sequel. For the proof of Theorem 2, the reader is referred to [36].

Theorem 1. *The regret of the CMAB policy π_{cmab} with gradually increasing superarm size and arms chosen based on LCBs with radius $\theta_i(j) \coloneqq \sqrt{\frac{4f(j)}{T_i(j)}} + \frac{2f(j)}{T_i(j)}$ where $f(j) = 2\log(j)$, and assuming[3] $\lambda_{\min} \geq 1$, is bounded from above as*

$$R^{\pi_{cmab}}_j \leq \max_{r \in [1,b]: j > T_{r-1}} \Delta_{\mathcal{A}^r,\max} \cdot n \cdot \left(\frac{48\log(j)}{\min\{\delta^2_{\min}, \delta_{\min}\}} + 1 + u \cdot \frac{\pi^2}{3}\right), \tag{5}$$

where $u \coloneqq \max_{r \in [1,b]: j > T_{r-1}} r$.

Theorem 1 quantifies the overhead in time spent by π_{cmab} to learn the average speeds of the workers. To provide a guarantee on the run-time of the algorithm using π_{cmab}, we combine in Theorem 2 the regret of π_{cmab} with the run-time guarantee for the optimal policy, with respect to the number of iterations.

Theorem 2. *Given a desired confidence $\epsilon > 0$, the time until policy π_{cmab} reaches iteration j is bounded from above as*

$$t_j \leq R^{\pi_{cmab}}_j + \sum_{r=1}^{b} \mathbb{1}\{j > T_{r-1}\} \mu_{\mathcal{A}^{r,\star}} (\min\{j, T_r\} - T_{r-1})(1+\epsilon)$$

with probability

$$\Pr(j) \geq \prod_{r \in [1,b]: j > T_{r-1}} \left(1 - \frac{\sigma^2_{\mathcal{A}^{r,\star}}}{\mu^2_{\mathcal{A}^{r,\star}} (\min\{j, T_r\} - T_{r-1}) \epsilon^2}\right).$$

IV. Proof of Theorem 1

While we will benefit from the proof strategies in [26] and [34], our analysis differs in that we consider an unbounded distribution of the rewards. Also, compared to [26], we deal with LCBs instead of UCBs as we want to minimize the response time of a superarm, i.e., the time spent per iteration. This problem setting was briefly discussed in [34]. While the authors bound the probability of overestimating an entire suboptimal superarm in [34], we bound the probability of individual suboptimal arm choices. This is justified by independent outcomes across arms and by the combined outcome of a superarm being a monotonically non-decreasing function of the individual arms' rewards, that is, the workers' mean response times. A superarm $\mathcal{A}^r(j)$ is considered suboptimal if $\mu_{\mathcal{A}^r(j)} > \mu_{\mathcal{A}^{r,\star}}$ and a single arm $\mathcal{A}^r_\nu(j)$ is suboptimal if $\mu_{\mathcal{A}^r_\nu(j)} > \mu_{\mathcal{A}^{r,\star}_\nu}$. In addition to the counter $T_i(j)$, we introduce for every arm $i \in [1,n]$ the counter $C_{i,e}(j) \leq T_i(j)$. The integer e refers to the maximum cardinality of all possible superarm choices, i.e., $C_{i,e}(j)$ is valid for all rounds $r \leq e$. If a suboptimal

[3]The assumption $\lambda_{\min} \geq 1$ is needed for our proof to hold. In practice, this assumption amounts to choosing the time unit of our theoretical model such that the average response time of each worker is less than one time unit.

superarm $\mathcal{A}^r(j)$ is chosen[4], $C_{i,e}(j)$ is incremented only for the arm $i \in \mathcal{A}^r(j)$ that has been pulled the least until this point in time, i.e., $i = \arg\min_{\nu \in \mathcal{A}^r(j)} T_\nu(j)$. Hence, $\sum_{i=1}^n C_{i,e}(j)$ equals the number of suboptimal superarm pulls. Let $\mathcal{W}^{\leq e}$ be the set of all superarms with a maximum cardinality of e, i.e., $\mathcal{W}^{\leq e} \coloneqq \bigcup_{r \leq e} \mathcal{W}^r$, and $T_{\mathcal{A}^r(j)}(j)$ the number of times superarm $\mathcal{A}^r(j)$ has been pulled until iteration j. We have

$$\sum_{\mathcal{U} \in \mathcal{W}^{\leq e}: \mu_{\mathcal{U}} > \mu_{\mathcal{A}^{|\mathcal{U}|},\star}} \mathbb{E}\left[T_{\mathcal{U}}(j)\right] = \sum_{i=1}^{n} \mathbb{E}\left[C_{i,e}(j)\right]. \tag{6}$$

Applying [37, Lemma 4.5] to express the regret in terms of the suboptimality gaps in iteration j of round r, i.e., $T_{r-1} < j \leq T_r$ and $e = r$, we obtain

$$\begin{aligned} R_j^{\pi_{\text{cmab}}} &= \sum_{\mathcal{U} \in \mathcal{W}^{\leq e}: \mu_{\mathcal{U}} > \mu_{\mathcal{A}^{|\mathcal{U}|},\star}} \Delta_{\mathcal{U}} \, \mathbb{E}\left[T_{\mathcal{U}}(j)\right] \\ &\leq \max_{r \in [1,b]: j > T_{r-1}} \Delta_{\mathcal{A}^r,\max} \cdot \sum_{i=1}^{n} \mathbb{E}\left[C_{i,e}(j)\right]. \end{aligned} \tag{7}$$

To conclude the proof of Theorem 1, we need the following intermediate results.

Claim 3. *Let $|\mathcal{A}^r(y)| \leq e$ hold $\forall y \in [1,j]$. For any $h \geq 0$, we can bound the expectation of $C_{i,e}(j)$, $i \in [1,n]$, as*

$$\begin{aligned} \mathbb{E}\left[C_{i,e}(j)\right] \leq\ & h + \sum_{y=1}^{j} j^2 \cdot \sum_{\nu=1}^{|\mathcal{A}^r(y)|} \Pr\big(\mu_{\mathcal{A}^r_\nu(y)} > \mu_{\mathcal{A}^{r,\star}_\nu}, \\ & \mathit{LCB}_{\mathcal{A}^r_\nu(y)}(y) \leq \mathit{LCB}_{\mathcal{A}^{r,\star}_\nu}(y)\big). \end{aligned}$$

Lemma 4. *The probability of the ν-th fastest arm of $\mathcal{A}^r(j)$ being suboptimal given that $T_{\mathcal{A}^r_\nu(j)}(j) \geq \left\lceil \frac{48\log(j)}{\min\{\delta^2_{\min}, \delta_{\min}\}} \right\rceil$, and $\lambda_{\min} \geq 1$, is bounded from above as*

$$\Pr\left(\mu_{\mathcal{A}^r_\nu(j)} > \mu_{\mathcal{A}^{r,\star}_\nu}, \mathit{LCB}_{\mathcal{A}^r_\nu(j)}(j) \leq \mathit{LCB}_{\mathcal{A}^{r,\star}_\nu}(j)\right) \leq 2j^{-4\lambda_{\min}}. \tag{8}$$

By applying Claim 3 with $h \geq 0$, we have by construction for a superarm $\mathcal{A}^r(j)$ with $|\mathcal{A}^r(j)| \leq e$, $\nu \in [1, |\mathcal{A}^r(j)|]$ and $i \in \mathcal{A}^r(j)$ that $T_{\mathcal{A}^r_\nu(j)}(j) \geq C_{i,e}(j) \geq h$.

Thus, using the result of Claim 3 with $h = \left\lceil \frac{48\log(j)}{\min\{\delta^2_{\min}, \delta_{\min}\}} \right\rceil$ and Lemma 4, for $\lambda_{\min} \geq 1$ we have

$$\begin{aligned} \mathbb{E}\left[C_{i,e}(j)\right] &\leq h + \sum_{y=1}^{j} j^2 \cdot \sum_{\nu=1}^{|\mathcal{A}^r(y)|} \Pr\big(\mu_{\mathcal{A}^r_\nu(y)} > \mu_{\mathcal{A}^{r,\star}_\nu}, \\ &\qquad \mathrm{LCB}_{\mathcal{A}^r_\nu(y)}(y) \leq \mathrm{LCB}_{\mathcal{A}^{r,\star}_\nu}(y)\big) \\ &\leq \left\lceil \frac{48\log(j)}{\min\{\delta^2_{\min}, \delta_{\min}\}} \right\rceil + \sum_{y=1}^{j} j^2 \cdot \sum_{\nu=1}^{e} 2j^{-4\lambda_{\min}} \\ &\leq \frac{48\log(\min\{j, T_r\})}{\min\{\delta^2_{\min}, \delta_{\min}\}} + 1 + e \cdot \frac{\pi}{3}, \end{aligned} \tag{9}$$

where the last step relates to the Basel problem of a p-series[5]. Plugging the bound in (9) into (7) concludes the proof.

The proof of Claim 3 uses standard techniques from the literature on MABs and is omitted for brevity. The proof of Lemma 4 is given next.

[4]Although there exists an optimal superarm, it is not necessarily unique, i.e., there might exist several superarms $\mathcal{A}^r(j)$ with $\mu_{\mathcal{A}^r(j)} = \mu_{\mathcal{A}^{r,\star}}$.

[5]We need $\lambda_{\min} \geq 1$ so that the p-series converges to a small value.

Proof of Lemma 4. As given in [26], to overestimate the ν-th fastest arm of $\mathcal{A}^r(j)$, i.e., for $\mathrm{LCB}_{\mathcal{A}^r_\nu(j)}(j) \leq \mathrm{LCB}_{\mathcal{A}^{r,\star}_\nu}(j)$ to hold, at least one of the following events must be satisfied:

$$\mu_{\mathcal{A}^{r,\star}_\nu} > \mu_{\mathcal{A}^r_\nu(j)} - 2\theta_{\mathcal{A}^r_\nu(j)}(j), \tag{10}$$

$$\hat{\mu}_{\mathcal{A}^{r,\star}_\nu}(j) \geq \mu_{\mathcal{A}^{r,\star}_\nu} + \theta_{\mathcal{A}^{r,\star}_\nu}(j), \tag{11}$$

$$\hat{\mu}_{\mathcal{A}^r_\nu(j)}(j) \leq \mu_{\mathcal{A}^r_\nu(j)} - \theta_{\mathcal{A}^r_\nu(j)}(j). \tag{12}$$

Let in the following $f(j) \coloneqq 2\log(j)$. We first show that the requirement[6] $T_{\mathcal{A}^r_\nu(j)}(j) \geq \lceil \frac{24f(j)}{\min\{\delta^2_{\min}, \delta_{\min}\}} \rceil = h$ guarantees that $\mu_{\mathcal{A}^{r,\star}_\nu} - \mu_{\mathcal{A}^r_\nu(j)} + 2\theta_{\mathcal{A}^r_\nu(j)}(j) \leq 0$, for all $\mathcal{A}^r(j) \in \mathcal{W}^r$ with $r \in [1,b]$, $\nu \in [1,r]$ and $\mu_{\mathcal{A}^r_\nu(j)} > \mu_{\mathcal{A}^{r,\star}_\nu}$, thus making the event (10) a zero-probability event. We have

$$\begin{aligned} &\mu_{\mathcal{A}^{r,\star}_\nu} - \mu_{\mathcal{A}^r_\nu(j)} + 2\left(\sqrt{\frac{4f(j)}{T_{\mathcal{A}^r_\nu(j)}(j)}} + \frac{2f(j)}{T_{\mathcal{A}^r_\nu(j)}(j)}\right) \\ &\quad \leq \mu_{\mathcal{A}^{r,\star}_\nu} - \mu_{\mathcal{A}^r_\nu(j)} + 2\left(\sqrt{\frac{4f(j)}{h}} + \frac{2f(j)}{h}\right) \\ &\quad \leq -\delta_{\mathcal{A}^r_\nu(j)} + \delta_{\min} \leq 0. \end{aligned}$$

Claim 5. *Given i.i.d. random variables $Z_i^j \sim \exp(\lambda_i)$, $j = 1, \ldots, T$, the deviation of the empirical mean from the true mean $\hat{\mu}_i - \mu_i \coloneqq \frac{1}{T}\sum_{j=1}^{T}\left(Z_i^j - \mathbb{E}[Z_i^j]\right)$ follows a sub-gamma distribution $\mathit{Sub}\Gamma(T, T\lambda_i)$ on the right tail and a sub-Gaussian distribution $\mathit{SubG}\left(\frac{1}{T\lambda_i^2}\right)$ on the left tail.*

Applying Claim 5 (which can be proven by moment generating functions and results from [38]) and taking $\lambda_{\min} \geq 1$, we bound the probability of the event (11) as

$$\begin{aligned} &\Pr\left(\hat{\mu}_{\mathcal{A}^{r,\star}_\nu}(j) \geq \mu_{\mathcal{A}^{r,\star}_\nu} + \theta_{\mathcal{A}^{r,\star}_\nu}(j)\right) \\ &= \Pr\left(\hat{\mu}_{\mathcal{A}^{r,\star}_\nu}(j) \geq \mu_{\mathcal{A}^{r,\star}_\nu} + \sqrt{\frac{4f(j)}{T_{\mathcal{A}^{r,\star}_\nu}(j)}} + \frac{2f(j)}{T_{\mathcal{A}^{r,\star}_\nu}(j)}\right) \\ &\leq \Pr\left(\hat{\mu}_{\mathcal{A}^{r,\star}_\nu}(j) \geq \mu_{\mathcal{A}^{r,\star}_\nu} + \sqrt{\frac{4f(j)}{T_{\mathcal{A}^{r,\star}_\nu}(j)\,\lambda_{\min}}} + \frac{2f(j)}{T_{\mathcal{A}^{r,\star}_\nu}(j)}\right) \\ &\leq \Pr\left(\hat{\mu}_{\mathcal{A}^{r,\star}_\nu}(j) \geq \mu_{\mathcal{A}^{r,\star}_\nu} + \sqrt{\frac{4f(j)\lambda_{\min}}{T_{\mathcal{A}^{r,\star}_\nu}(j)\lambda^2_{\min}}} + \frac{2f(j)\lambda_{\min}}{T_{\mathcal{A}^{r,\star}_\nu}(j)\lambda_{\min}}\right) \\ &\leq \exp -4\log(j)\lambda_{\min} \leq j^{-4\lambda_{\min}}, \end{aligned}$$

where in the penultimate step we used the sub-gamma tail bound in [38, p. 29] with $\varepsilon = 2\log(j)\lambda_{\min}$.

For (12), we have

$$\begin{aligned} &\Pr\left(\hat{\mu}_{\mathcal{A}^r_\nu(j)}(j) \leq \mu_{\mathcal{A}^r_\nu(j)} - \theta_{\mathcal{A}^r_\nu(j)}(j)\right) \\ &= \Pr\left(\hat{\mu}_{\mathcal{A}^r_\nu(j)}(j) \leq \mu_{\mathcal{A}^r_\nu(j)} - \sqrt{\frac{4f(j)}{T_{\mathcal{A}^r_\nu(j)}(j)}} - \frac{2f(j)}{T_{\mathcal{A}^r_\nu(j)}(j)}\right) \\ &\leq \Pr\left(\hat{\mu}_{\mathcal{A}^r_\nu(j)}(j) \leq \mu_{\mathcal{A}^r_\nu(j)} - \sqrt{\frac{4f(j)}{T_{\mathcal{A}^r_\nu(j)}(j)}}\right) \\ &\leq \exp -4\lambda^2_{\mathcal{A}^r_\nu(j)}\log(j) \\ &\leq j^{-4\lambda^2_{\mathcal{A}^r_\nu(j)}} \leq j^{-4\lambda^2_{\min}} \leq j^{-4\lambda_{\min}}, \end{aligned}$$

where we used the sub-Gaussian tail bound in [37, p. 77]. □

[6]The scaling factor $\gamma = 24$ is chosen as an approximation of the exact solution of $2\left(\sqrt{\frac{4}{\gamma}} + \frac{2}{\gamma}\right) = 1$, which is $4\left(3 + 2\sqrt{2}\right) \approx 23.31 \leq 24$.

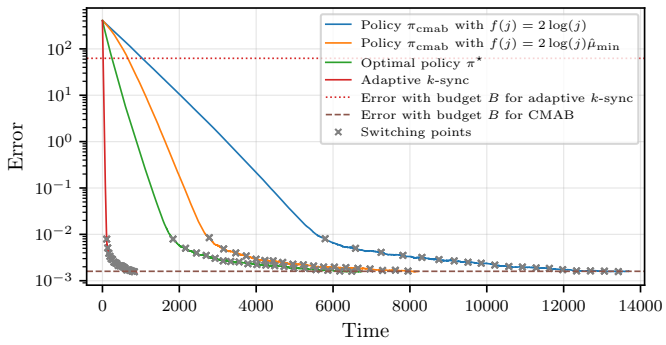

Fig. 1: Comparison to adaptive k-sync [8] with limited budget B.

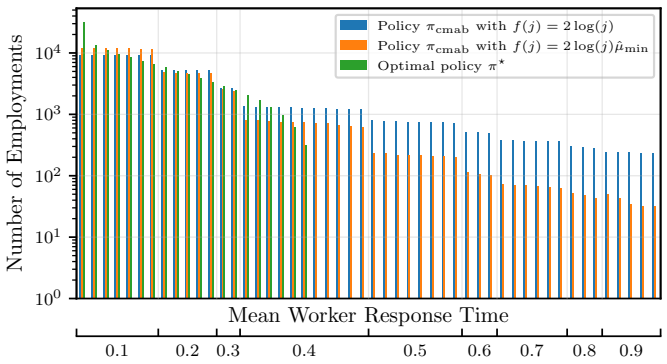

Fig. 2: Number of worker employments using CMAB strategies.

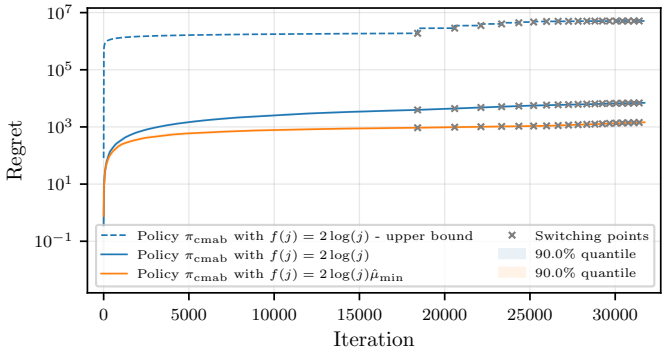

Fig. 3: Comparison of the theoretical and simulated regret.

V. Numerical Simulations

A. Setting

Similarly to [8], we consider $n = 50$ workers with exponentially distributed response times whose means are chosen uniformly at random from $\{0.1, 0.2, \cdots, 0.9\}$ such that $\lambda_{\min} \geq 1$. We limit the budget to $b = 20$ parallel computations. We create $m = 2000$ samples $\mathbf{x}_\ell$ with $d = 100$ entries, each drawn uniformly at random from $[1, 10]$ with labels $y_\ell \sim \mathcal{N}\left(\mathbf{x}_\ell^T \mathbf{w}', 1\right)$, for some $\mathbf{w}'$ drawn uniformly at random from $[1, 100]^d$. The model $\mathbf{w}$ is initialized uniformly at random as $\mathbf{w}_0 \in [1, 100]^d$ and optimized subject to the least squares loss function $F(\mathbf{X}, \mathbf{y}, \mathbf{w}) \coloneqq \frac{1}{2}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2$ with learning rate $\eta = 1 \times 10^{-4}$. We assess the performance of the model by the error function $\|\mathbf{X}^+\mathbf{y} - \mathbf{w}\|_2$ that quantifies the gap with the analytical solution, so that the analysis is largely data and problem independent. For all the simulations, we present the results averaged over at least ten rounds.

B. Switching Points

The switching points T_r, $r \in [1, b]$, are the iterations in which we advance from round r to $r+1$. In [8], Pflug's method [39] is used to determine the T_r's on the fly. However, this method is very sensitive to the learning rate [40], [41], and may result in different T_r's across different runs. While implicit model updates [40] or alternative criteria [41] can avoid this effect, we fix the switching points to ensure comparability across simulation runs. We empirically determine T_1 and necessary statistics to calculate T_r for $r \in [2, b]$ using (1).

C. Simulation Results

In Fig. 1, we analyze the convergence of the CMAB-based algorithm with two different LCB choices, and compare with the optimal policy $\pi^\star$ and the scheme in [8].

As waiting for the r fastest out of $n > r$ workers is on average faster than waiting for all out of r workers, the adaptive k-sync strategy in [8] is faster than the one introduced here. However, with a total budget of $B < 1.3 \times 10^5$, the CMAB-based strategy reaches an error of $\approx 2 \times 10^{-3}$ (cf. dotted red line) while adaptive k-sync achieves an error of only $\approx 6 \times 10^1$ (cf. dashed brown line). Comparing $\pi^\star$ to the performance of π_{cmab} (cf. blue line), learning the mean worker speeds slows down the convergence by a factor of almost three. This is because the chosen confidence radius mostly dominates the mean response time estimates of the workers, which leads to an emphasis on exploration, i.e., more confident estimates at the expense of sampling slow workers more often. This is reflected in Fig. 2, which plots the number of times each worker is employed, with the workers sorted from fastest to slowest. Scaling $f(j)$ with $\hat{\mu}_{\min} \coloneqq \min_{i \in [1,n]} \hat{\mu}_i$ significantly improves the convergence speed, but also delays the determination of the fastest workers. While with $f(j) = 2\log(j)$ the policy correctly determined all b fastest workers in ten simulation runs, with $f(j) = 2\log(j)\hat{\mu}_{\min}$ in one out of ten simulations the algorithm commits to a worker with a suboptimality gap of 0.1. This reflects the trade-off between the competing objectives of best arm identification and regret minimization discussed in [42]. However, since the fastest workers have been determined eventually with an accuracy of 99.5%, the proposed adapted confidence bound seems reasonable to improve the convergence rate. Fig. 3 compares the theoretical regret guarantee to practical results for both choices of $f(j)$. As the theoretical guarantee is a worst case analysis, the true performance is underestimated.

VI. Conclusion

We have introduced a cost-efficient distributed machine learning scheme that assigns random tasks to workers and leverages all computations. To speed up the convergence, we utilized a CMAB model, for which we provided theoretical regret guarantees and simulation results. While our scheme is inferior to the adaptive k-sync strategy in [8] in terms of speed, it achieves significantly lower errors with the same computational efforts. Deriving tighter regret bounds and improving the choice of the confidence bound is left for future work.

REFERENCES

[1] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, pp. 74–80, 2013.

[2] J. Chen, X. Pan, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous SGD," 2017, arXiv preprint arXiv:1604.00981.

[3] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," in *IEEE International Symposium on Information Theory (ISIT)*, 2016, pp. 1143–1147.

[4] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *Proc. International Conference on Machine Learning*, vol. 70, Aug 2017, pp. 3368–3376.

[5] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[6] A. Cotter, O. Shamir, N. Srebro, and K. Sridharan, "Better mini-batch algorithms via accelerated gradient methods," in *Advances in Neural Information Processing Systems*, vol. 24, 2011.

[7] S. Dutta, G. Joshi, S. Ghosh, P. Dube, and P. Nagpurkar, "Slow and stale gradients can win the race: Error-runtime trade-offs in distributed SGD," in *Proc. International Conference on Artificial Intelligence and Statistics*, vol. 84, Apr 2018, pp. 803–812.

[8] S. K. Hanna, R. Bitar, P. Parag, V. Dasari, and S. El Rouayheb, "Adaptive distributed stochastic gradient descent for minimizing delay in the presence of stragglers," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 4262–4266.

[9] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[10] G. Liang and U. C. Kozat, "TOFEC: Achieving optimal throughput-delay trade-off of cloud storage using erasure codes," in *IEEE Conference on Computer Communications*, 2014, pp. 826–834.

[11] M. Ye and E. Abbe, "Communication-computation efficient gradient coding," in *International Conference on Machine Learning*, 2018, pp. 5610–5619.

[12] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient coding from cyclic MDS codes and expander graphs," *IEEE Transactions on Information Theory*, vol. 66, no. 12, pp. 7475–7489, 2020.

[13] E. Ozfatura, S. Ulukus, and D. Gündüz, "Straggler-aware distributed learning: Communication–computation latency trade-off," *Entropy*, vol. 22, no. 5, 2020.

[14] M. M. Amiri and D. Gündüz, "Computation scheduling for distributed machine learning with straggling workers," *IEEE Transactions on Signal Processing*, vol. 67, no. 24, p. 6270–6284, Dec 2019.

[15] S. Li, S. M. Mousavi Kalan, A. S. Avestimehr, and M. Soltanolkotabi, "Near-optimal straggler mitigation for distributed gradient methods," in *IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2018, pp. 857–866.

[16] R. Bitar, M. Wootters, and S. el Rouayheb, "Stochastic gradient coding for straggler mitigation in distributed learning," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, pp. 277–291, 2020.

[17] R. K. Maity, A. Singh Rawat, and A. Mazumdar, "Robust gradient descent via moment encoding and LDPC codes," in *IEEE International Symposium on Information Theory (ISIT)*, 2019, pp. 2734–2738.

[18] Z. Charles, D. Papailiopoulos, and J. Ellenberg, "Approximate gradient coding via sparse random graphs," 2017, arXiv preprint arXiv:1711.06771.

[19] S. Wang, J. Liu, and N. Shroff, "Fundamental limits of approximate gradient coding," *Proc. ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 3, Dec 2019.

[20] H. Wang, Z. Charles, and D. Papailiopoulos, "ErasureHead: Distributed gradient descent without delays using approximate gradient coding," 2019, arXiv preprint arXiv:1901.09671.

[25] R. Agrawal, "Sample mean based index policies with o(log n) regret for the multi-armed bandit problem," *Advances in Applied Probability*, vol. 27, no. 4, pp. 1054–1078, 1995.

[21] S. Horii, T. Yoshida, M. Kobayashi, and T. Matsushima, "Distributed stochastic gradient descent using LDGM codes," in *IEEE International Symposium on Information Theory (ISIT)*, 2019, pp. 1417–1421.

[22] E. Ozfatura, S. Ulukus, and D. Gündüz, "Distributed gradient descent with coded partial gradient computations," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3492–3496.

[23] ——, "Coded distributed computing with partial recovery," *IEEE Transactions on Information Theory*, vol. 68, no. 3, pp. 1945–1959, 2022.

[24] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[26] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, pp. 235–256, May 2002.

[27] W. Jouini and C. Moy, "UCB algorithm for exponential distributions," 2012, arXiv preprint arXiv:1204.1624.

[28] S. Bubeck, N. Cesa-Bianchi, and G. Lugosi, "Bandits with heavy tail," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7711–7717, 2013.

[29] V. Anantharam, P. Varaiya, and J. Walrand, "Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays-part i: I.i.d. rewards," *IEEE Transactions on Automatic Control*, vol. 32, no. 11, pp. 968–976, 1987.

[30] W. Chen, Y. Wang, Y. Yuan, and Q. Wang, "Combinatorial multi-armed bandit and its extension to probabilistically triggered arms," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1746–1778, 2016.

[31] B. Kveton, Z. Wen, A. Ashkan, and C. Szepesvari, "Tight Regret Bounds for Stochastic Combinatorial Semi-Bandits," in *Proc. International Conference on Artificial Intelligence and Statistics*, vol. 38, May 2015, pp. 535–543.

[32] W. Chen, W. Hu, F. Li, J. Li, Y. Liu, and P. Lu, "Combinatorial multi-armed bandit with general reward functions," in *Advances in Neural Information Processing Systems*, vol. 29, 2016, pp. 1651–1659.

[33] Y. Gai, B. Krishnamachari, and R. Jain, "Learning multiuser channel allocations in cognitive radio networks: A combinatorial multi-armed bandit formulation," in *IEEE Symposium on New Frontiers in Dynamic Spectrum (DySPAN)*, 2010, pp. 1–9.

[34] ——, "Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1466–1478, 2012.

[35] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Review*, vol. 60, no. 2, pp. 223–311, 2018.

[36] M. Egger, R. Bitar, A. Wachter-Zeh, and D. Gündüz, "Cost-efficient distributed learning via combinatorial multi-armed bandits," 2022, arXiv preprint arXiv:2202.08302.

[37] T. Lattimore and C. Szepesvári, *Bandit Algorithms*. Cambridge University Press, 2020.

[38] S. Boucheron, G. Lugosi, and P. Massart, *Concentration Inequalities: A Nonasymptotic Theory of Independence*. OUP Oxford, 2013.

[39] G. C. Pflug, "Non-asymptotic confidence bounds for stochastic approximation algorithms with constant step size," *Monatshefte für Mathematik*, vol. 110, no. 3, pp. 297–314, 1990.

[40] J. Chee and P. Toulis, "Convergence diagnostics for stochastic gradient descent with constant learning rate," in *Proc. International Conference on Artificial Intelligence and Statistics*, vol. 84, Apr 2018, pp. 1476–1485.

[41] S. Pesme, A. Dieuleveut, and N. Flammarion, "On convergence-diagnostic based step sizes for stochastic gradient descent," in *Proc. International Conference on Machine Learning*, vol. 119, Jul 2020, pp. 7641–7651.

[42] Z. Zhong, W. C. Cheung, and V. Y. F. Tan, "On the pareto frontier of regret minimization and best arm identification in stochastic bandits," 2021, arXiv preprint arXiv:2110.08627.