ATA: Adaptive Task Allocation for Efficient Resource Management in Distributed Machine Learning

Artavazd Maranjyan, El Mehdi Saad,
Peter Richtárik, Francesco Orabona

Motivation

Imagine you are running Minibatch SGD on a heterogeneous cluster of 1010 workers, using a batch size of 10. The fastest way to collect the batch is to request gradients asynchronously from all workers and accept the first 10 responses. However, this approach wastes the computations of at least 1000 workers whose results arrive too late and are discarded.

Problem setup

We consider n workers where each worker $i \in [n]$ is iid drawn from X_i with mean μ_i.

Each iteration consists of B tasks.

The action set is all possible allocations:

$$\mathcal{A} := \{\boldsymbol{a} \in \mathbb{N}^n : \|\boldsymbol{a}\|_1 = B\}$$

Objective

$$\mathcal{C}_K := \sum_{k=1}^{K} \mathbb{E}[C(\boldsymbol{a}^k)]$$

$$C(\boldsymbol{a}^k) := \max_{i \in \mathrm{supp}(\boldsymbol{a}^k)} \sum_{u=1}^{a_i^k} X_i^{k,u}, \quad \boldsymbol{a}^k \in \mathcal{A}$$

Sub-exponential random variables

$$\|X_i - \mu_i\|_{\psi_1} \le \alpha, \quad \text{for all} \quad i \in [n]$$

$$\|X\|_{\psi_1} := \inf\{C > 0 : \mathbb{E}[\exp(|X|/C)] \le 2\}$$

Confidence Interval

$$s_i^k = \max\left\{\hat{\mu}_i^k - \mathrm{conf}(i,k), 0\right\}$$

$$\mathrm{conf}(i,k) = \begin{cases} 2\alpha\left(\sqrt{\frac{\ln(2k^2)}{K_i^k}} + \frac{\ln(2k^2)}{K_i^k}\right), & K_i^k \ge 1, \\ +\infty, & K_i^k = 0 \end{cases}$$

Theoretical Results

Proxy loss

$$\ell(\boldsymbol{a}, \boldsymbol{\mu}) := \max_{i \in [n]} a_i \mu_i$$

$$\ell(\boldsymbol{a}, \boldsymbol{\mu}) \le \mathbb{E}\left[C(\boldsymbol{a})\right] \le (1 + 4\eta \ln(B))\ell(\boldsymbol{a}, \boldsymbol{\mu})$$

$$\eta := \max_{i \in [n]} \frac{\alpha_i}{\mu_i}$$

Guarantees

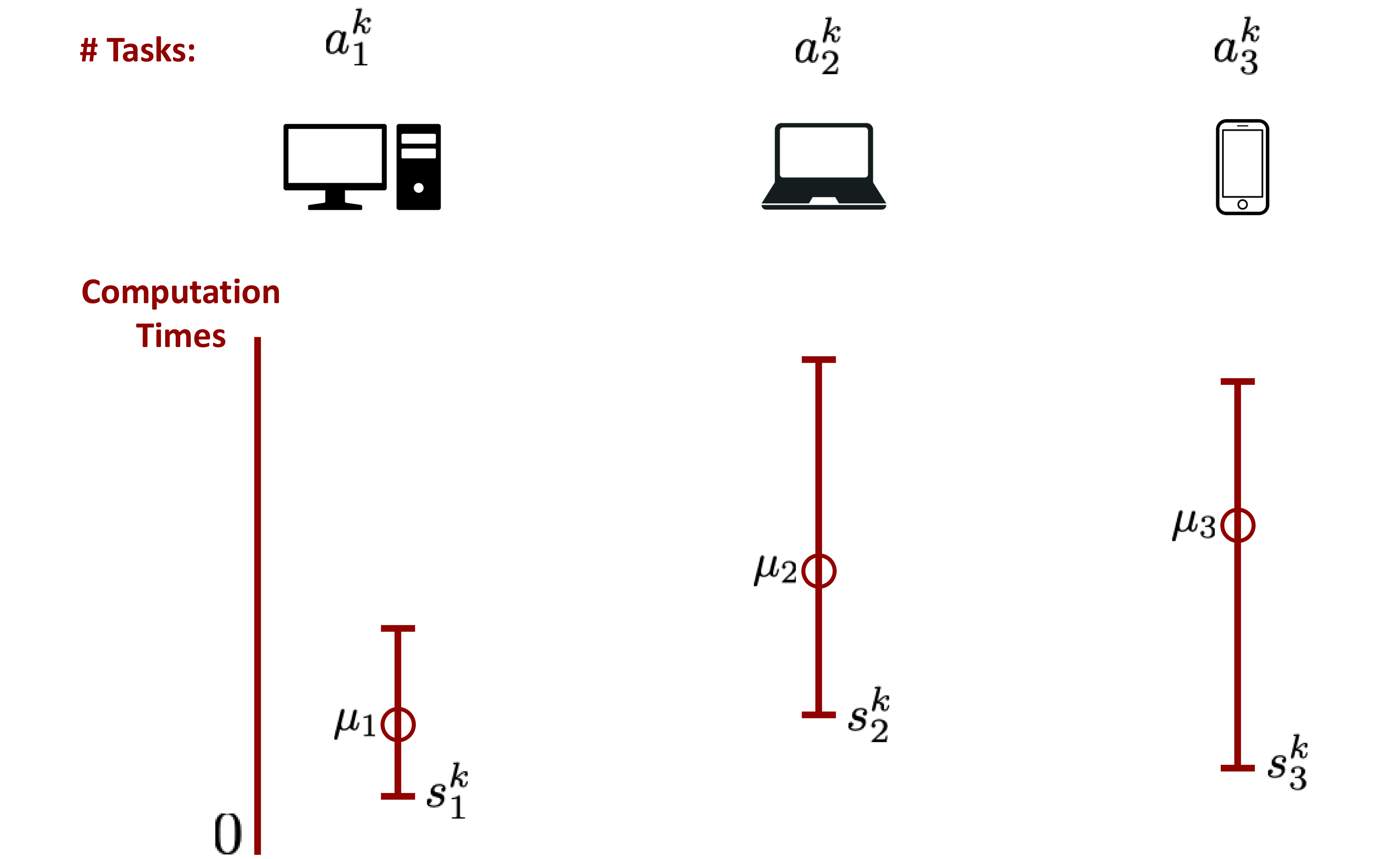
$$\mathcal{C}_K \le (1 + 4\eta\ \ln(B))\, \mathcal{C}_K^* + \mathcal{O}(\ln K)$$

$$\mathcal{C}_K^* := K\mathbb{E}\left[C(\boldsymbol{a}^*)\right], \quad \boldsymbol{a}^* \in \operatorname*{argmin}_{\boldsymbol{a} \in \mathcal{A}} \mathbb{E}\left[C(\boldsymbol{a})\right]$$

Multiple devices, unknown speeds?

No problem.

ATA learns and adapts to minimize computation while maintaining fast distributed ML training.



Next Allocation: $$(a_1^{k+1}, a_2^{k+1}, a_3^{k+1}) = \operatorname*{argmin}_{\sum a_i = B} \left\{\max_j a_j s_j^k\right\}$$

Experiments

Dataset: CIFAR-100
Network: CNN with 3 convolutional layers and 2 fully connected layers
Optimizer: Adam with constant learning rate of 8×10^{-5}

$$B = 23, \quad n = 51$$

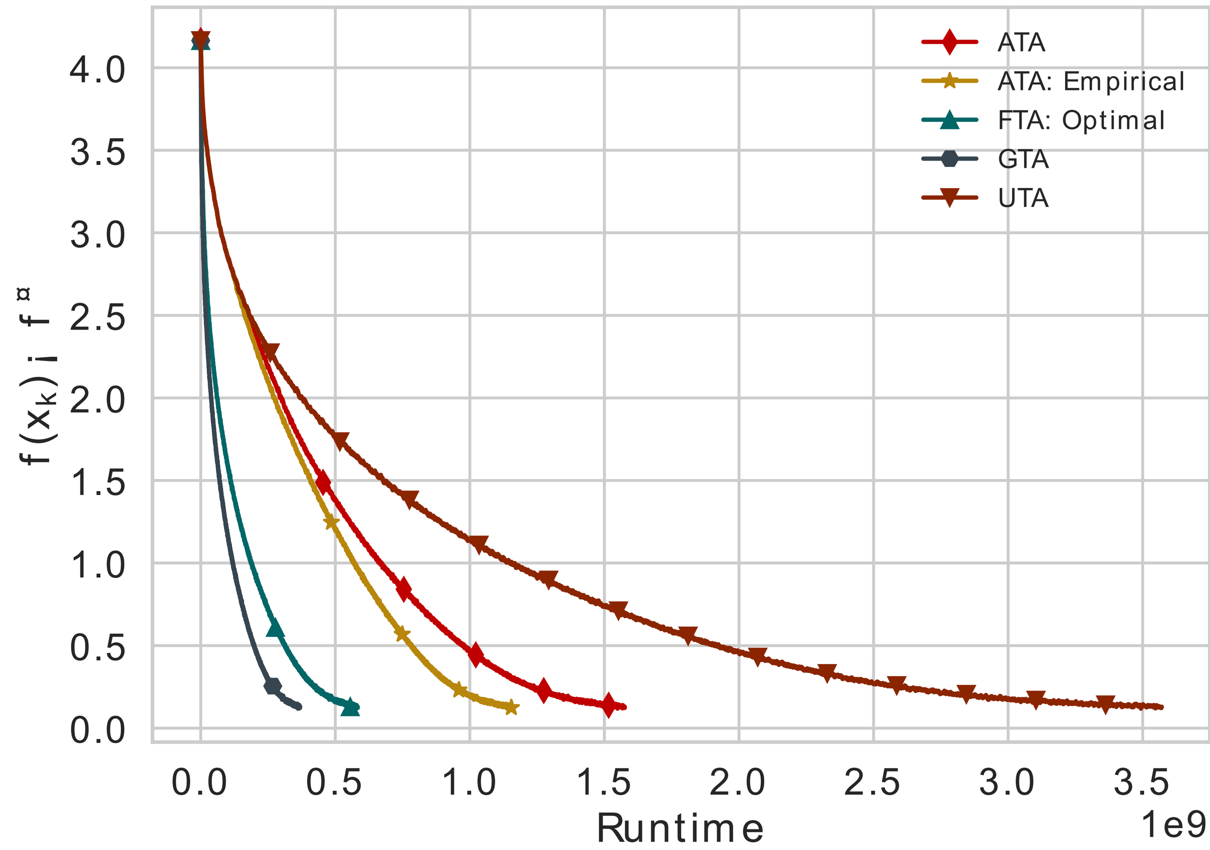
$$X_i \sim 29i + \mathrm{Exp}(29i), \quad \text{for all} \quad i \in [n]$$

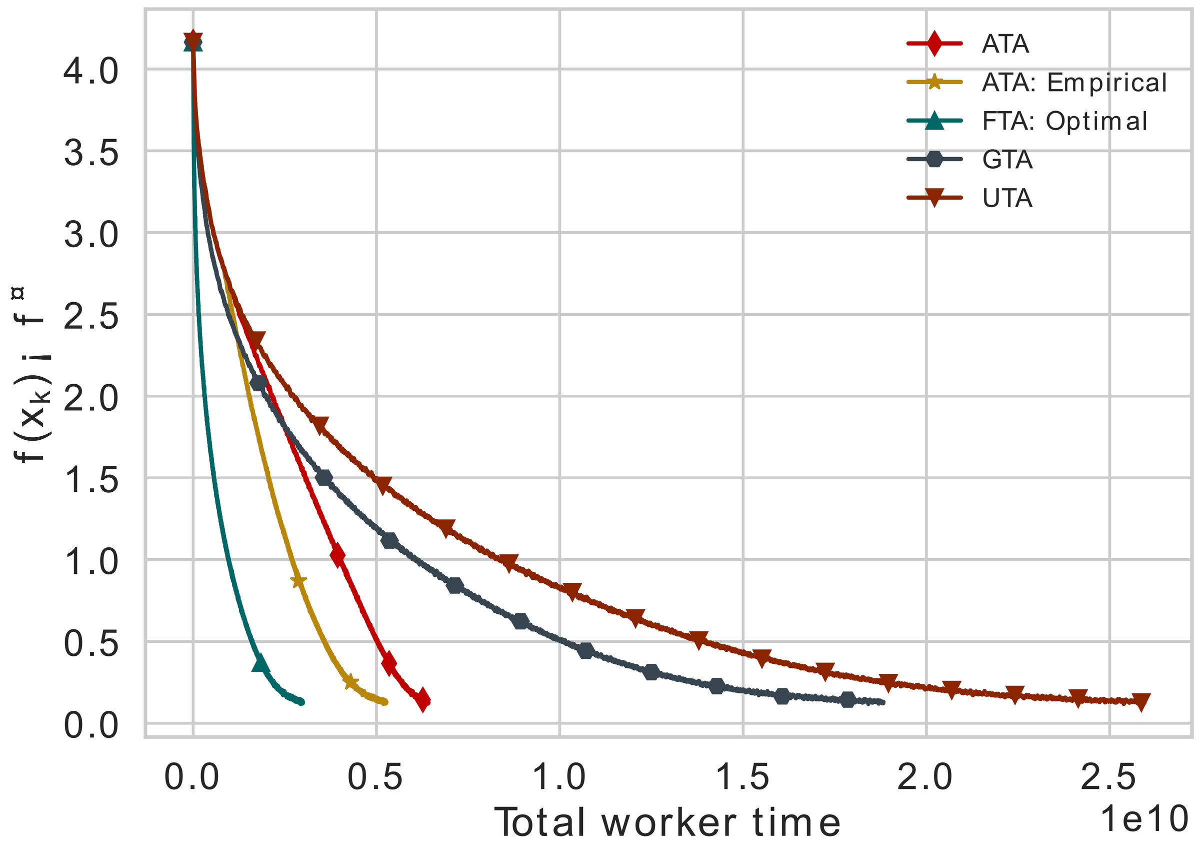
ATA: Empirical

$$\hat{s}_i^k = \hat{\mu}_i^k \max\left\{1 - 2\eta\left(\sqrt{\frac{\ln(2k^2)}{K_i^k}} + \frac{\ln(2k^2)}{K_i^k}\right), 0\right\}$$

Baselines

FTA: Fixed Task Allocation
GTA: Greedy Task Allocation (asynchronous batch collection)
UTA: Uniform Task Allocation





ICML International Conference On Machine Learning

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology