Optimal Time Complexities of Parallel Stochastic Optimization Methods Under a Fixed Computation Model

Workshop on Nonsmooth Optimization and Applications NOPTA 8-12 April 2024 University of Antwerp, Belgium

Alexander Tyurin, Peter Richtárik

King Abdullah University of Science and Technology, Saudi Arabia

alexandertiurin@gmail.com

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

Abstract

We propose a new protocol, Time Oracle Protocol, that generalizes the classical oracle framework approach. Using this protocol, we establish minimax complexities for parallel optimization methods that have access to an unbiased stochastic gradient oracle with bounded variance. Moreover, we develop optimal algorithms that attain them.

Stochastic Optimization Problem

Consider the nonconvex/convex optimization problem

$$\min_{x\in Q}\left\{f(x) := \mathbb{E}_{\xi\sim\mathcal{D}}\left[f(x;\xi)\right]\right\}, \tag{1}$$

where $f : \mathbb{R}^d \times \mathbb{S}_\xi \to \mathbb{R}$, $Q \subseteq \mathbb{R}^d$, and ξ is a random variable with some distribution $\mathcal{D}$ on $\mathbb{S}_\xi$. Our goal is to find a possibility random point $\bar{x}$, which is called ε–stationary point, such that $\mathbb{E}[\|\nabla f(\bar{x})\|^2] \leq \varepsilon$.

Parallel Computational Setup

We address the following natural setup:

(i) n workers/nodes/CPUs/GPUs/... are available to work in parallel.

(ii) the i^{th} worker requires at most τ_i seconds to calculate a stochastic gradient of f.

Without loss of generality, we assume $0 < \tau_1 \leq \cdots \leq \tau_n$.

Assumptions

Assumption 1 *f is differentiable and L–smooth, i.e.,*

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \quad \forall x, y \in \mathbb{R}^d.$$

Remark 1 *The proposed theory can work with virtually any class of functions, including nonsmooth, nonconvex, convex, and strongly-convex functions.*

Assumption 2 *There exist $f^* \in \mathbb{R}$ such that $f(x) \geq f^*$ for all $x \in \mathbb{R}^d$.*

Assumption 3 *For all $x \in \mathbb{R}^d$, stochastic gradients $\widehat{\nabla} f(x;\xi)$ are unbiased and σ^2-variance-bounded, i.e., $\mathbb{E}[\widehat{\nabla} f(x;\xi)] = \nabla f(x)$ and $\mathbb{E}[\|\widehat{\nabla} f(x;\xi) - \nabla f(x)\|^2] \leq \sigma^2$, where $\sigma^2 \geq 0$.*

Upper and Lower Bounds for # of Stochastic Gradient Calls

Under Assumptions 1, 2, and 3, **with one worker**, stochastic gradient descent (SGD), i.e., the method

$$x^{k+1} = x^k - \gamma\widehat{\nabla} f(x^k;\xi^k),$$

where ξ^k are i.i.d. random samples from $\mathcal{D}$, is known to be **optimal with respect to # of stochastic gradient calls** [3]. SGD guarantees convergence to an ε–stationary point in expectation after

$$\text{Upper and Lower Bound: } \Theta\left(\frac{L\Delta}{\varepsilon} + \frac{\sigma^2 L\Delta}{\varepsilon^2}\right) \tag{2}$$

stochastic gradient evaluations, where $\Delta := f(x^0) - f^*$ and $x^0 \in \mathbb{R}^d$ is a starting point.

How was the Lower Bound Derived?

We need to define a *function class* $\mathcal{F}$, an *oracle class* $\mathcal{O}$, and an *algorithm class* $\mathcal{A}$. We then analyze the complexity of an algorithm $A = \{A^k\}_{k=0}^\infty \in \mathcal{A}$, using the following protocol:

Protocol 1 Classical Oracle Protocol

Input: function $f \in \mathcal{F}$, oracle and distribution $(O, \mathcal{D}) \in \mathcal{O}(f)$, algorithm $A \in \mathcal{A}$
for $k = 0, \ldots, \infty$ **do**
  $x^k = A^k(g^1, \ldots, g^k)$ ▷ $x^0 = A^0$ for $k = 0$.
  $g^{k+1} = O(x^k, \xi^{k+1}), \quad \xi^{k+1} \sim \mathcal{D}$
end for

An algorithm $A = \{A^k\}_{k=0}^\infty \in \mathcal{A}$ is a sequence such that

$$A^k : \underbrace{\mathbb{R}^d \times \cdots \times \mathbb{R}^d}_{k \text{ times}} \to \mathbb{R}^d \ \ \forall k \geq 1, \text{ and } A^0 \in \mathbb{R}^d.$$

Then, in the nonconvex first-order stochastic setting, we analyze the complexity measure

$$\mathfrak{m}_{\text{oracle}}(\mathcal{A}, \mathcal{F}) := \inf_{A\in\mathcal{A}} \sup_{f\in\mathcal{F}} \sup_{(O,\mathcal{D})\in\mathcal{O}(f)} \inf\left\{k \in \mathbb{N} \,\middle|\, \mathbb{E}\left[\left\|\nabla f(x^k)\right\|^2\right] \leq \varepsilon\right\}, \tag{3}$$

where the sequence $\{x^k\}_k$ is generated by Protocol 1. Virtually all previous works are concerned with lower bounds of optimization problems using Protocol 1 and the complexity measure (3) [1, 4]. For the problem (1), [3] show that

$$\mathfrak{m}_{\text{oracle}}(\mathcal{A}, \mathcal{F}) = \Theta\left(\frac{L\Delta}{\varepsilon} + \frac{\sigma^2 L\Delta}{\varepsilon^2}\right).$$

New Framework: Time Oracle Protocol

Protocol 1 seek to quantify the worst-case **number of iterations or oracle calls** that are required to find a solution (see (3)), which is very natural for **sequential** methods. However, and this is a key observation of our work, **Protocol 1 is not convenient if we want to analyze parallel methods**. We now propose an alternative protocol that can be more helpful in this situation:

Protocol 2 Time Oracle Protocol

Input: functions $f \in \mathcal{F}$, oracle and distribution $(O, \mathcal{D}) \in \mathcal{O}(f)$, algorithm $A \in \mathcal{A}$
$s^0 = 0$
for $k = 0, \ldots, \infty$ **do**
  $(t^{k+1}, x^k) = A^k(g^1, \ldots, g^k)$, ▷ $t^{k+1} \geq t^k$
  $(s^{k+1}, g^{k+1}) = O(t^{k+1}, x^k, s^k, \xi^{k+1}), \quad \xi^{k+1} \sim \mathcal{D}$
end for

Let us explain the role of the sequence $\{t^k\}_k$. In Protocol 1, an algorithm outputs a point x^k and then asks the oracle:

Provide me a gradient at the point x^k.

In contrast, in Protocol 2 an algorithm outputs a point x^k and a time t^{k+1}, and asks the oracle:

Start calculating a gradient at x^k at the time t^{k+1}.

We have a constraint that $t^{k+1} \geq t^k$ for all $k \geq 0$, which means that the algorithm is not allowed to travel into the past. Using Protocol 2, we propose to use another complexity measure instead of (3):

$$\mathfrak{m}_{\text{time}}(\mathcal{A}, \mathcal{F}) := \inf_{A\in\mathcal{A}} \sup_{f\in\mathcal{F}} \sup_{(O,\mathcal{D})\in\mathcal{O}(f)} \inf\left\{t \geq 0 \,\middle|\, \mathbb{E}\left[\inf_{k\in S_t}\left\|\nabla f(x^k)\right\|^2\right] \leq \varepsilon\right\}, \tag{4}$$
$$S_t := \left\{k \in \mathbb{N}_0 \,\middle|\, t^k \leq t\right\}.$$

In (3), we seek to find the **worst-case number of iterations** k required to get $\mathbb{E}[\|\nabla f(x^k)\|^2] \leq \varepsilon$ for any $A \in \mathcal{A}$. In (4), we seek to find the **worst-case case time** t required to find an ε-stationary point for any $A \in \mathcal{A}$.

Time Delayed Oracle

Let us define the appropriate oracle for Protocol 2:

$$O_\tau^{\widehat{\nabla} f}(t, x, (s_t, s_x, s_q), \xi) = \begin{cases} ((t, x, 1), & 0), & s_q = 0, \\ ((s_t, s_x, 1), & 0), & s_q = 1 \text{ and } t < s_t + \tau, \\ ((0, 0, 0), & \widehat{\nabla} f(s_x;\xi)), & s_q = 1 \text{ and } t \geq s_t + \tau, \end{cases} \tag{5}$$

and $\widehat{\nabla} f$ is a mapping such that $\widehat{\nabla} f : \mathbb{R}^d \times \mathbb{S}_\xi \to \mathbb{R}^d$.

Note that the oracle $O_\tau^{\widehat{\nabla} f}$ emulates the behavior of a real worker that requires τ seconds for calculate a stochastic gradient.

Lower Bound of Parallel Methods

We now consider a protocol that works with multiple oracles:

Protocol 3 Time Multiple Oracles Protocol

Input: function(s) $f \in \mathcal{F}$, oracles and distributions $((O_1, ..., O_n), (\mathcal{D}_1, ..., \mathcal{D}_n)) \in \mathcal{O}(f)$, algorithm $A \in \mathcal{A}$
$s_i^0 = 0$ for all $i \in [n]$
for $k = 0, \ldots, \infty$ **do**
  $(t^{k+1}, i^{k+1}, x^k) = A^k(g^1, \ldots, g^k)$, ▷ $t^{k+1} \geq t^k$
  $(s_{i^{k+1}}^{k+1}, g^{k+1}) = O_{i^{k+1}}(t^{k+1}, x^k, s_{i^{k+1}}^k, \xi^{k+1})$
end for

Compared to Protocol 2, Protocol 3 works with multiple oracles, and algorithms return the indices i^{k+1} of the oracle they want to call. This minor add-on to the protocol enables the possibility of analyzing parallel optimization methods.

Theorem 1 *Let us take any $\sigma^2 > 0$ and $0 < \tau_1 \leq \ldots \leq \tau_n$. We fix any $L, \Delta > 0$ and $0 < \varepsilon \leq c'L\Delta$. In view Protocol 3, for any algorithm $A \in \mathcal{A}_{\text{zr}}$, there exists a function $f \in \mathcal{F}_{\Delta,L}$ and oracles and distributions $((O_{\tau_1}^{\widehat{\nabla} f}, \ldots, O_{\tau_n}^{\widehat{\nabla} f}), (\mathcal{D}_1, \ldots, \mathcal{D}_n))$ such that $\mathbb{E}\left[\inf_{k\in S_t}\left\|\nabla f(x^k)\right\|^2\right] > \varepsilon$, where $S_t := \left\{k \in \mathbb{N}_0 \,\middle|\, t^k \leq t\right\}$, and $t = c \times \min_{m\in[n]}\left[\left(\frac{1}{m}\sum_{i=1}^m 1/\tau_i\right)^{-1}\left(\frac{L\Delta}{\varepsilon} + \frac{\sigma^2 L\Delta}{m\varepsilon^2}\right)\right]$. The quantities c' and c are universal constants.*

Theorem 1 states that $\mathfrak{m}_{\text{time}}(\mathcal{A}_{\text{zr}}, \mathcal{F}_{\Delta,L}) =$

$$\Omega\left(\min_{m\in[n]}\left[\left(\frac{1}{m}\sum_{i=1}^m \frac{1}{\tau_i}\right)^{-1}\left(\frac{L\Delta}{\varepsilon} + \frac{\sigma^2 L\Delta}{m\varepsilon^2}\right)\right]\right). \tag{6}$$

Example with One Worker

Consider the case when we have one worker, $n = 1$, with the performance τ_1. Then, we get

$$\mathfrak{m}_{\text{time}}(\mathcal{A}_{\text{zr}}, \mathcal{F}_{\Delta,L}) = \Omega\left(\tau_1\left(\frac{L\Delta}{\varepsilon} + \frac{\sigma^2 L\Delta}{m\varepsilon^2}\right)\right).$$

The same result can be naturally derived from (2). Our result is more general and includes the cases when $n > 1$ with arbitrarily heterogeneous time delays τ_i.

Optimal Method: Rennala SGD

We now present a new method that attains the lower bound (6).

Method 4 Rennala SGD

Input: starting point x^0, stepsize γ, batch size S
Run Method 5 in all workers
for $k = 0, 1, \ldots, K-1$ **do**
  Init $g^k = 0$ and $s = 1$
  while $s \leq S$ **do**
    Wait for the next worker
    Receive gradient and iteration index (g, k')
    if $k' = k$ **then**
      $g^k = g^k + \frac{1}{S}g; \quad s = s + 1$
    end if
    Send (x^k, k) to the worker
  end while
  $x^{k+1} = x^k - \gamma g^k$
end for

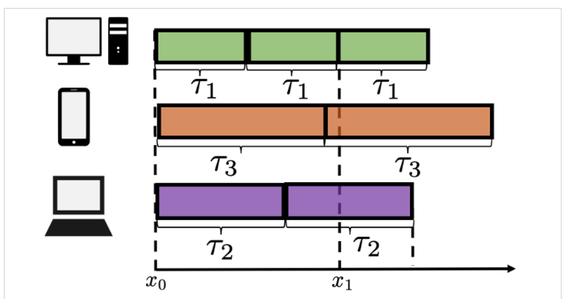
Method 5 Worker's Infinite Loop

Init $g = 0$ and $k' = -1$
while True **do**
  Send (g, k') to the server
  Receive (x^k, k) from the server
  $k' = k$
  $g = \widehat{\nabla} f(x^k;\xi), \quad \xi \sim \mathcal{D}$
end while

Theorem 2 *Assume that Assumptions 1, 2 and 3 hold. Let us take the batch size $S = \max\{\lceil\sigma^2/\varepsilon\rceil, 1\}$, and $\gamma = \min\left\{\frac{1}{L}, \frac{\varepsilon S}{2L\sigma^2}\right\} = \Theta(1/L)$ in Method 4. Then after*

$$96 \times \min_{m\in[n]}\left[\left(\frac{1}{m}\sum_{i=1}^m \frac{1}{\tau_i}\right)^{-1}\left(\frac{L\Delta}{\varepsilon} + \frac{\sigma^2 L\Delta}{m\varepsilon^2}\right)\right]$$

seconds, Method 4 guarantees to find an ε-stationary point.



Rennala SGD collects S stochastic gradients ($S = 4$ in this example). Only then does it calculate the new iterate.

Remark 2 *In [2], we extend our theory to the* heterogeneous *case, in which the workers have access to different distributions, and provide a lower bound and a new method, Malenia SGD, that attains it. We provide the optimal time complexities in the* convex *setting.*

References

[1] Nemirovskij A. and Yudin D. Problem complexity and method efficiency in optimization. 1983.

[2] Tyurin A. and Richtárik P. Optimal time complexities of parallel stochastic optimization methods under a fixed computation model. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[3] Arjevani Y., Carmon Y., Duchi J., Foster D., Srebro N., and Woodworth B. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, pages 1–50, 2022.

[4] Nesterov Y. *Lectures on convex optimization*, volume 137. Springer, 2018.

Download

PAPER

POSTER