On the Computation and Communication Complexity of Parallel SGD with Dynamic Batch Sizes for Stochastic Non-Convex Optimization

{ Hao Yu, Rong Jin}
Machine Intelligence Technology Lab, Alibaba Group (US) Inc, Bellevue, WA

1. Stochastic Non-Convex Opt

- **Non-convex** stochastic optimization

$$\min_{\mathbf{x}\in\mathbb{R}^m} \quad f(\mathbf{x}) \triangleq \mathbb{E}_{\zeta\sim D}[F(\mathbf{x};\zeta)]$$

- Typical applications: training deep neural networks
- **Mini-batch** SGD used in practice

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma \frac{1}{B}\sum_{i=1}^{B} \nabla F(\mathbf{x}_t;\zeta_i)$$

- Effects of batch size B in SGD
  - Single node case: Larger B improves the utilization of computing hardware.
  - Data parallel training: Larger B decreases # of aggregation/communication rounds when Stochastic First-order Oracle (**SFO**) budget is given.
- Should we always choose B as large as possible?
  - As B increases, mini-batch SGD is more similar to GD.
  - GD has exponential convergence for strongly convex opt. Does this suggest GD is preferred?
  - **No!** when SFO budget is given.
- SGD with $B = 1$ has better SFO convergence than GD [Bottou&Bousquet'08] [Bottou et. al.'18].

2. Parallel SGD with dynamic BS

- Complexity of N node parallel SGD with fixed small BS
  - Strongly convex case: $O(1/(NT))$ SFO convergence with $O(T)$ comm rounds
  - Non-convex case: $O(1/\sqrt{NT})$ SFO convergence with $O(T)$ comm rounds
- This paper explores using dynamic batch sizes in parallel SGD to achieve same SFO convergence with less comm.

3. Non-Convex under PL

Polyak-Lojasiewicz (P-L) condition

$$\frac{1}{2}\|\nabla f(\mathbf{x})\|^2 \geq \mu(f(\mathbf{x}) - f^*), \forall \mathbf{x}$$

- Strongly convex functions satisfy P-L condition.
- CR-PSGD: parallel SGD with exponentially increasing BS

Alg1: CR-PSGD $(f, N, T, \mathbf{x}_1, B_1, \rho, \gamma)$

1: **Input:** $N, T, \mathbf{x}_1 \in \mathbb{R}^m, \gamma, B_1$ and $\rho > 1$.
2: Initialize $t = 1$
3: **while** $\sum_{\tau=1}^{t} B_\tau \leq T$ **do**
4: Each worker obtains individual batch stochastic gradient average $\bar{\mathbf{g}}_{t,i} = \frac{1}{B_t}\sum_{j=1}^{B_t} F(\mathbf{x}_t;\zeta_{i,j})$.
5: Each worker aggregates all $\bar{\mathbf{g}}_{t,i}$ to compute average $\bar{\mathbf{g}}_t = \frac{1}{N}\sum_{i=1}^{N}\bar{\mathbf{g}}_{t,i}$.
6: Each worker updates in parallel via: $\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma\bar{\mathbf{g}}_t$.
7: Set batch size $B_{t+1} = \lfloor \rho^t B_1 \rfloor$.
8: Update $t \leftarrow t + 1$.
9: **end while**
10: **Return:** $\mathbf{x}_t$

4. General Non-Convex

- For general non-convex without PL, we have a new catalyst-like algorithm:

Alg2: CR-PSGD-Catalyst $(f, N, T, \mathbf{y}_0, B_1, \rho, \gamma)$

1: **Input:** $N, T, \theta, \mathbf{y}_0 \in \mathbb{R}^m, \gamma, B_1$ and $\rho > 1$.
2: Initialize $\mathbf{y}^{(0)} = \mathbf{y}_0$ and $k = 1$.
3: **while** $k \leq \lfloor\sqrt{NT}\rfloor$ **do**
4: Define $h_\theta(\mathbf{x};\mathbf{y}^{(k-1)}) \triangleq f(\mathbf{x}) + \frac{\theta}{2}\|\mathbf{x} - \mathbf{y}^{(k-1)}\|^2$.
5: Update $\mathbf{y}^{(k)} =$ CR-PSGD$(h_\theta(\cdot;\mathbf{y}^{(k-1)}), N, \lfloor\sqrt{T/N}\rfloor, \mathbf{y}^{(k-1)}, B_1, \rho, \gamma)$
6: Update $k \leftarrow k + 1$.
7: **end while**

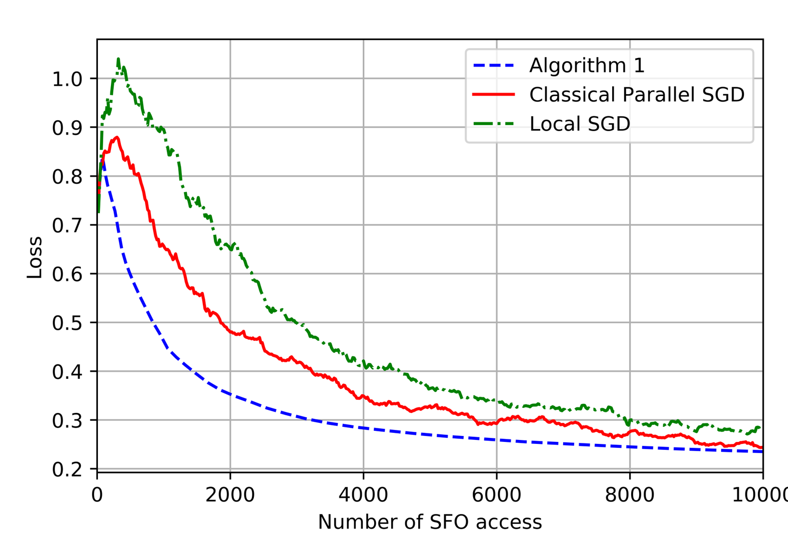
- Like "catalyst acceleration" proposed in [Lin et al.'15] [Paquette et al.'18], our CR-PSGD-Catalyst uses a proximal point outer-loop inside which CR-PSGD is called.

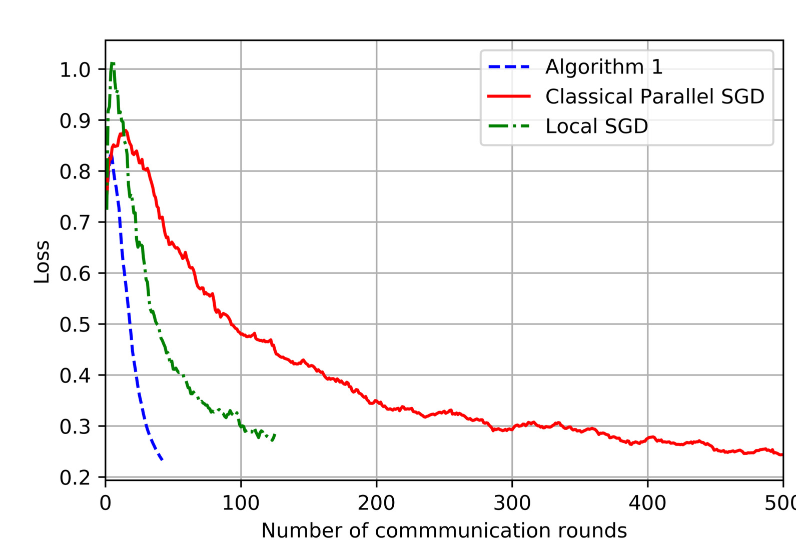
5. Performance Analysis

- **Non-Convex under PL**:
  - **CR-PSGD** has $O(1/(NT))$ SFO convergence with $O(\log T)$ comm rounds
  - Compared with parallel SGD, same SFO convergence but less comm (v.s. $O(T)$)
  - Strongly convex special case: tie with best known $O(1/(NT))$ SFO with $O(\log T)$ comm attained by local SGD [Stich'18]
- **General Non-Convex**:
  - **CR-PSGD-Catalyst** has $O(1/\sqrt{NT})$ SFO convergence with $O(\sqrt{NT}\log(T/N))$ comm rounds
  - Better than parallel SGD with $O(1/\sqrt{NT})$ SFO convergence and $O(T)$ comm; or parallel restarted SGD (local SGD for non-convex) with $O(1/\sqrt{NT})$ SFO convergence and $O(N^{3/4}T^{3/4})$ comm [Yu et al.'18].

6. Experiments

- Distributed Logistic Regression ($N = 10$)





- Train ResNet20 over CIFAR10 ($N = 8$)