Distributed Stochastic Gradient Methods over Networks

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Kun Huang [1] Shi Pu [1]

[1]School of Data Science, the Chinese University of Hong Kong, Shenzhen (CUHK-Shenzhen)

Problem and Motivation

Problem

$$\min_{x\in\mathbb{R}^p} f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

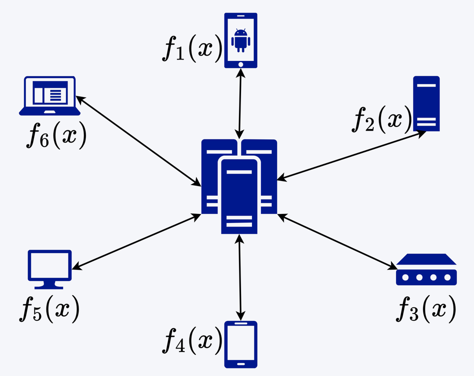


Figure 1. Centralized architecture.

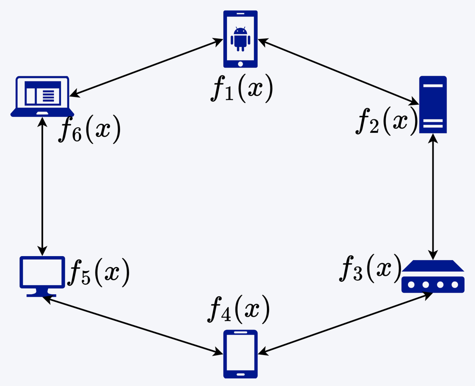


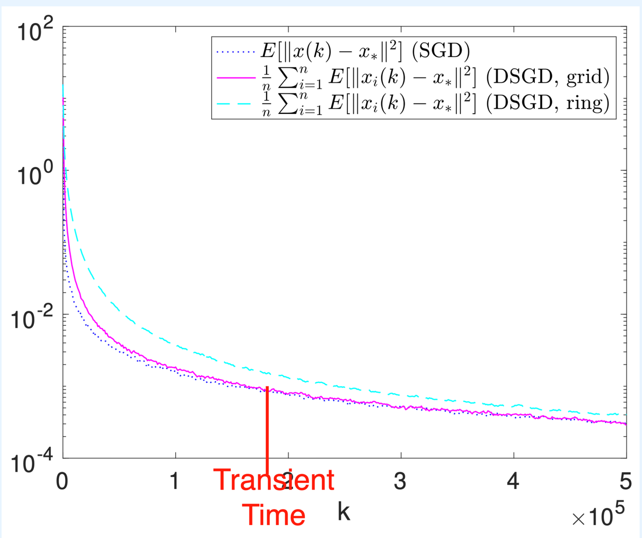
Figure 2. Decentralized architecture.

Motivation

Decentralized architecture reduces the **high latency** and the **robustness bottleneck** caused by the central server. However, decentralization may slow down the optimization process due to the partial communication over sparse networks.

How do the decentralized algorithms compare with the centralized algorithms?

Transient Time



Some decentralized stochastic gradient algorithms achieve **similar performance** compared to centralized stochastic gradient descent (SGD) after a finite transient time (iterations) has passed.

Figure 3. A Illustration of the Transient Time [7].

Starting Point

- A decomposition:

$$\frac{1}{n}\sum_{i=1}^{n}\|x_i - x^*\|^2 = \underbrace{\|\bar{x} - x^*\|^2}_{\text{Optimization error}} + \underbrace{\frac{1}{n}\sum_{i=1}^{n}\|x_i - \bar{x}\|^2}_{\text{Consensus error}},\ \bar{x} := \frac{1}{n}\sum_{i=1}^{n} x_i,\ x^* \in \arg\min f(x).$$

- Most of the distributed algorithms share the same update for $\bar{x}$:

$$\bar{x}_{k+1} = \bar{x}_k - \frac{\alpha_k}{n}\sum_{i=1}^{n} g_i(x_{i,k};\xi_{i,k}),\ \text{where } g_i(x_{i,k};\xi_{i,k}) \text{ is a stochastic gradient of } \nabla f_i(x_{i,k}).$$

- Handling the consensus error depends on the algorithms.
- Assumption: the corresponding mixing matrix W is symmetric and stochastic, i.e., $W^\intercal = W, W\mathbf{1} = \mathbf{1}$.
- Assumption: each f_i is $L-$ smooth and lower bounded.

The Function f_i Has a General Form and $\mathbb{E}[g_i(x;\xi)|x] = \nabla f_i(x)$

Assumption: Bounded Variance

$$\mathbb{E}\left[\|\nabla f_i(x) - g_i(x;\xi)\|^2 \Big| x\right] \le \sigma^2.$$

- EDAS improves the transient time from $\mathcal{O}(n/(1-\lambda)^2)$ to $\mathcal{O}(n/(1-\lambda))$ for minimizing smooth strongly convex objective functions [5].
- EDAS improves the transient time from $\mathcal{O}(n^3/(1-\lambda)^4)$ to $\mathcal{O}(n^3/(1-\lambda)^2)$ for minimizing smooth nonconvex objective functions [4].
- EDAS also improves the transient time when equipping with communication compression [4].

Assumption: ABC Condition

$$\mathbb{E}\left[\|\nabla f_i(x) - g_i(x;\xi)\|^2 \Big| x\right] \le C\left[f_i(x) - f_i^*\right] + \sigma^2.$$

- Federated Averaging (FedAvg) can converge under the ABC condition without any data heterogeneity assumption [3].
- Decentralized algorithms can also converge under the ABC condition. (In progress)
- The ABC condition is satisfied when we calculate the stochastic gradient by sampling the data points with replacement [3].

A Summary of Convergence Results: Smooth Nonconvex

Algorithm	Convergence Rate
Centralized RR	$\mathcal{O}\left(\frac{1}{\boldsymbol{m^{1/3}T^{2/3}}}\right)$
Centralized SGD	$\mathcal{O}\left(\frac{1}{\sqrt{mnT}}\right)$
DSGD	$\mathcal{O}\left(\frac{1}{\sqrt{mnT}} + \frac{n^2}{(1-\lambda)^2 mT}\right)$
D-RR	$\mathcal{O}\left(\frac{1}{(1-\lambda)T^{2/3}}\right)$ [2] (a)
D-RR (**New**)	$\mathcal{O}\left(\frac{1}{\boldsymbol{(1-\lambda)^{2/3}m^{1/3}T^{2/3}}} + \frac{1}{(1-\lambda)T}\right)$
DSGT	$\mathcal{O}\left(\frac{1}{\sqrt{mnT}} + \frac{n}{(1-\lambda)mT} + \frac{n}{(1-\lambda)^4 m^2T^2}\right)$ [1]
GT-RR	$\mathcal{O}\left(\frac{1}{\boldsymbol{(1-\lambda)^{1/3}m^{1/3}T^{2/3}}} + \frac{1}{(1-\lambda)T} + \frac{1}{m^{7/3}(1-\lambda)^{7/3}T^{5/3}}\right)$
ED/ D^2	$\mathcal{O}\left(\frac{1}{\sqrt{mnT}} + \frac{n}{(1-\lambda)mT} + \frac{n}{(1-\lambda)^2 m^2T^2}\right)$ [1]
ED-RR	$\mathcal{O}\left(\frac{1}{\boldsymbol{(1-\lambda)^{1/3}m^{1/3}T^{2/3}}} + \frac{1}{(1-\lambda)T} + \frac{1}{m^{7/3}(1-\lambda)^{4/3}T^{5/3}}\right)$

(a) The result is obtained by minimizing over the arbitrary constant η in the original result $\mathcal{O}(1/(\eta T^{2/3}) + \eta^2/[(1-\lambda)^3T^{2/3}])$ in [2].

Table 1. A summary of related the theoretical results using a constant stepsize.

The Function f_i Has a Finite Sum Form

$$\min_{x\in\mathbb{R}^p} f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x),\ \text{with } f_i(x) = \frac{1}{m}\sum_{\ell=1}^{m} f_{i,\ell}(x).$$

How does Random Reshuffling (RR) Proceed?

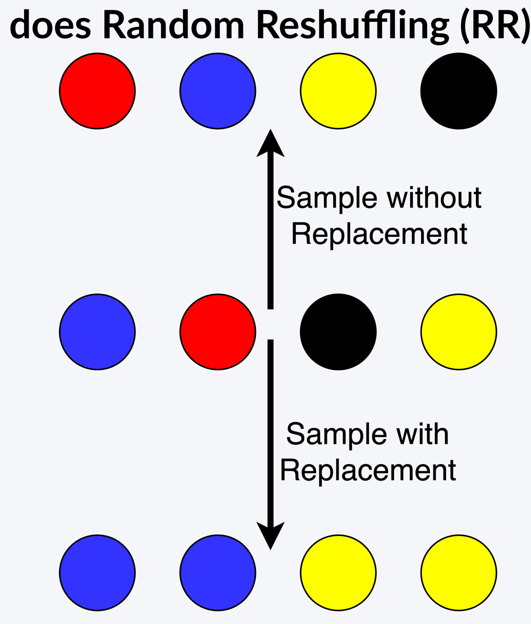


Figure 4. Sample with or without replacement.

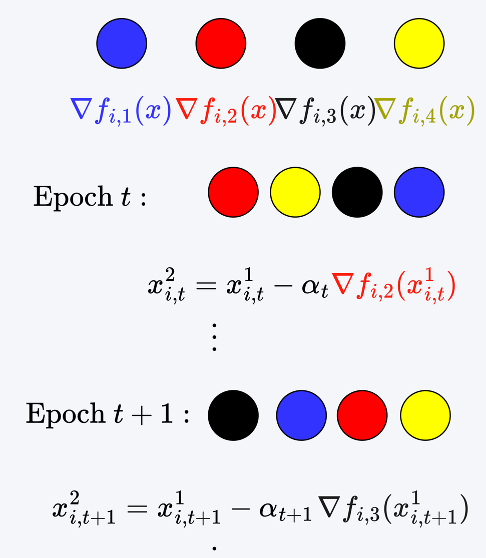


Figure 5. A Illustration of Random Reshuffling.

Why RR?

- **RR improves both numerical performance and theoretical convergence rate.**
- RR is widely used in training machine learning problems.

Distributed RR Methods over Networks

- D-RR achieves similar convergence rate compared to centralized RR [3].
- GT-RR and ED-RR achieves similar convergence rate compared to centralized RR and reduces the impact of decentralization compared to D-RR [6].

References

[1] Sulaiman A Alghunaim and Kun Yuan. A unified and refined convergence analysis for non-convex decentralized learning. *IEEE Transactions on Signal Processing*, 70:3264–3279, 2022.

[2] Kun Huang, Xiao Li, Andre Milzarek, Shi Pu, and Junwen Qiu. Distributed random reshuffling over networks. *IEEE Transactions on Signal Processing*, 71:1143–1158, 2023.

[3] Kun Huang, Xiao Li, and Shi Pu. Distributed stochastic optimization under a general variance condition. *arXiv preprint arXiv:2301.12677*, 2023.

[4] Kun Huang and Shi Pu. Cedas: A compressed decentralized stochastic gradient method with improved convergence, 2023.

[5] Kun Huang and Shi Pu. Improving the transient times for distributed stochastic gradient methods. *IEEE Transactions on Automatic Control*, 68(7):4127–4142, 2023.

[6] Kun Huang, Linli Zhou, and Shi Pu. Distributed random reshuffling methods with improved convergence, 2023.

[7] Shi Pu, Alexander Olshevsky, and Ioannis Ch Paschalidis. A sharp estimate on the transient time of distributed stochastic gradient descent. *IEEE Transactions on Automatic Control*, 2021.

kunhuang@link.cuhk.edu.cn