南京大學
NANJING UNIVERSITY

Ordered Momentum for Asynchronous SGD

Chang-Wei Shi, Yi-Rui Yang, Wu-Jun Li

{shicw, yangyr}@smail.nju.edu.cn, liwujun@nju.edu.cn

National Key Laboratory for Novel Software Technology,
School of Computer Science, Nanjing University, Nanjing, China

Introduction

南京大學 NANJING UNIVERSITY

- Distributed learning has become a hot research topic in recent years because of its necessity for training large-scale machine learning models.
  - Synchronous distributed learning (SDL) methods: Synchronous SGD (SSGD), SSGD with momentum (SSGDm)…
  - Asynchronous distributed learning (ADL) methods: Asynchronous SGD (ASGD)…
- Momentum has been acknowledged for its benefits in both optimization and generalization in deep model training.
  - In SDL methods, momentum is extensively utilized across various domains.
  - In ADL methods, existing works have found that naively incorporating momentum into ASGD may decrease the convergence rate or even result in divergence.
- In this paper, we propose a novel method called ordered momentum (OrMo) for ASGD.

Preliminary

南京大學
NANJING UNIVERSITY

SSGD & ASGD

- Distributed SGD (DSGD) unifies SSGD and ASGD within a single framework.
- The waiting set is a collection of workers (indexes) that are awaiting the server to send the latest parameter.
- The only difference between SSGD and ASGD is the communication scheduler associated with the waiting set.

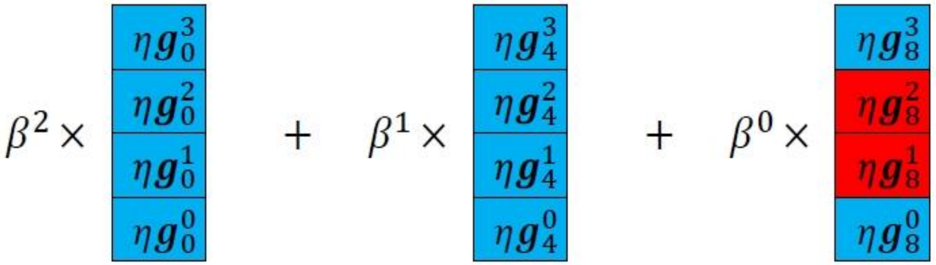
Algorithm 1 Distributed SGD

Server:
Input: number of workers K, number of iterations T, learning rate η, waiting set $\mathcal{C} = \emptyset$;
Initialization: initial parameter $\mathbf{w}_0$;
Send the initial parameter $\mathbf{w}_0$ to all workers;
for iteration $t \in [T]$ **do**
    Receive a stochastic gradient $\mathbf{g}^{k_t}_{ite(k_t,t)}$ from some worker k_t;
    Update the parameter $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{g}^{k_t}_{ite(k_t,t)}$;
    Add the worker k_t to the waiting set $\mathcal{C} = \mathcal{C} \cup \{k_t\}$;
    Execute the communication scheduler:
    Option I: (Synchronous) only when all the workers are in the waiting set, i.e., $\mathcal{C} = [K]$, send the parameter $\mathbf{w}_{t+1}$ to the workers in $\mathcal{C}$ and set $\mathcal{C}$ to $\emptyset$;
    Option II: (Asynchronous) once the waiting set is not empty, i.e., $\mathcal{C} \neq \emptyset$, immediately send the parameter $\mathbf{w}_{t+1}$ to the worker in $\mathcal{C}$ and set $\mathcal{C}$ to $\emptyset$;
end for
Notify all workers to stop;
Worker k **:** $(k \in [K])$
repeat
    Wait until receiving the parameter $\mathbf{w}$ from the server;
    Randomly sample ξ^k and compute the stochastic gradient $\mathbf{g}^k = \nabla f(\mathbf{w}; \xi^k)$;
    Send the stochastic gradient $\mathbf{g}^k$ to the server;
until receive server's notification to stop

Ordered Momentum

☐ Reformulation of SSGD with momentum (SSGDm)

- The momentum in SSGDm can be formulated as $\boldsymbol{u}_{t+1} = \sum_{i=0}^{\left\lfloor \frac{t}{K} \right\rfloor - 1} \left(\beta^{\left\lfloor \frac{t}{K} \right\rfloor - i} \times \sum_{k \in [K]} \eta \boldsymbol{g}_{iK}^{k} \right) + \beta^{0} \times \sum_{j=\left\lfloor \frac{t}{K} \right\rfloor K}^{t} \eta \boldsymbol{g}_{\left\lfloor \frac{t}{K} \right\rfloor K}^{k_j}$.
  - We define $\{\eta \boldsymbol{g}_{iK}^{0}, \eta \boldsymbol{g}_{iK}^{1}, \cdots, \eta \boldsymbol{g}_{iK}^{K-1}\}$ as the i-th *(scaled) gradient group*.
  - The order of the gradient group is based on the iteration indexes of its corresponding gradients.
  - The momentum in SSGDm is a weighted sum of the gradients from the first several gradient groups.
- An example of the momentum $\boldsymbol{u}_{10}$ in SSGDm

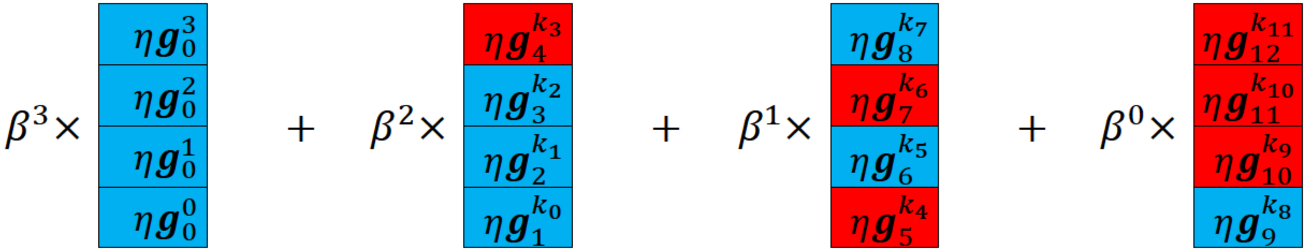


Ordered Momentum

OrMo for ASGD

- Definition of the gradient groups in OrMo for ASGD
  - The sequence of gradients computed in ASGD is given by $\boldsymbol{g}_0^0, \boldsymbol{g}_0^1, \cdots, \boldsymbol{g}_0^{K-1}, \boldsymbol{g}_1^{k_0}, \boldsymbol{g}_2^{k_1}, \cdots, \boldsymbol{g}_K^{k_{K-1}}, \boldsymbol{g}_{K+1}^{k_K}, \boldsymbol{g}_{K+2}^{k_{K+1}}, \cdots, \boldsymbol{g}_{2K}^{k_{2K-1}}, \cdots$.
  - The i-th (scaled) gradient group in OrMo for ASGD is defined as: $\left\{\eta\boldsymbol{g}_{(i-1)K+1}^{k_{(i-1)K}}, \eta\boldsymbol{g}_{(i-1)K+2}^{k_{(i-1)K+1}}, \cdots, \eta\boldsymbol{g}_{iK}^{k_{iK-1}}\right\}$,

    where $i \geq 1$. And the 0-th gradient group in OrMo is $\{\eta\boldsymbol{g}_0^0, \eta\boldsymbol{g}_0^1, \cdots, \eta\boldsymbol{g}_0^{K-1}\}$.
- In OrMo, momentum is incorporated into ASGD by organizing the gradients in order based on their iteration indexes.
  - The momentum is a weighted sum of the gradients from the first several gradient groups.
  - We refer to the gradient group whose gradients are weighted by β^0 as the *latest gradient group*, which contains the latest gradients.
  - An example of the momentum $\boldsymbol{u}_{10}$ in OrMo for ASGD



Ordered Momentum

- OrMo for ASGD
  - Algorithm details

Algorithm 2 OrMo

<u>Server:</u>
Input: number of workers K, number of iterations T, learning rate η, momentum coefficient $\beta \in [0, 1)$, waiting set $\mathcal{C} = \emptyset$;
Initialization: initial parameter $\mathbf{w}_0$, momentum $\mathbf{u}_0 = \mathbf{0}$, index of the latest gradient group $I_0 = 0$;
Send the initial parameter $\mathbf{w}_0$ and its iteration index 0 to all workers;
for iteration $t \in [T]$ **do**
  if the waiting set $\mathcal{C}$ is empty and $\lceil \frac{t}{K} \rceil > I_t$ **then**
    $\mathbf{w}_{t+\frac{1}{2}} = \mathbf{w}_t - \beta \mathbf{u}_t$, $\mathbf{u}_{t+\frac{1}{2}} = \beta \mathbf{u}_t$, $I_{t+1} = I_t + 1$;
  else
    $\mathbf{w}_{t+\frac{1}{2}} = \mathbf{w}_t$, $\mathbf{u}_{t+\frac{1}{2}} = \mathbf{u}_t$, $I_{t+1} = I_t$;
  end if
  Receive a stochastic gradient $\mathbf{g}^{k_t}_{ite(k_t,t)}$ and its iteration index $ite(k_t, t)$ from some worker k_t and then calculate $\lceil \frac{ite(k_t,t)}{K} \rceil$ (i.e., the index of the gradient group that $\mathbf{g}^{k_t}_{ite(k_t,t)}$ belongs to);
  Update the momentum $\mathbf{u}_{t+1} = \mathbf{u}_{t+\frac{1}{2}} + \beta^{I_{t+1} - \lceil \frac{ite(k_t,t)}{K} \rceil} \times \left(\eta \mathbf{g}^{k_t}_{ite(k_t,t)} \right)$;
  Update the parameter $\mathbf{w}_{t+1} = \mathbf{w}_{t+\frac{1}{2}} - \frac{1 - \beta^{I_{t+1} - \lceil \frac{ite(k_t,t)}{K} \rceil + 1}}{1 - \beta} \times \left(\eta \mathbf{g}^{k_t}_{ite(k_t,t)} \right)$;
  Add the worker k_t to the waiting set $\mathcal{C} = \mathcal{C} \cup \{k_t\}$;
  Execute the asynchronous communication scheduler: once the waiting set is not empty, i.e., $\mathcal{C} \neq \emptyset$, immediately send the parameter $\mathbf{w}_{t+1}$ and its iteration index $t + 1$ to the worker in $\mathcal{C}$ and set $\mathcal{C}$ to $\emptyset$;
end for
Notify all workers to stop;
<u>Worker k :</u> $(k \in [K])$
repeat
  Wait until receiving the parameter $\mathbf{w}_{t'}$ and its iteration index t' from the server;
  Randomly sample ξ^k and calculate the stochastic gradient $\mathbf{g}^k_{t'} = \nabla f(\mathbf{w}_{t'}; \xi^k)$;
  Send the stochastic gradient $\mathbf{g}^k_{t'}$ and its iteration index t' to the server;
until receive server's notification to stop

NANJING UNIVERSITY

Convergence Analysis

Assumption 1. *For any stochastic gradient $\nabla f(\mathbf{w};\xi^k)$, we assume that it satisfies:*

$$\mathbb{E}_{\xi^k}[\nabla f(\mathbf{w};\xi^k)] = \nabla F(\mathbf{w}), \quad \mathbb{E}_{\xi^k}\|\nabla f(\mathbf{w};\xi^k) - \nabla F(\mathbf{w})\|^2 \le \sigma^2, \quad \forall \mathbf{w} \in \mathbb{R}^d, \forall k \in [K].$$

Assumption 2. *For any stochastic gradient $\nabla f(\mathbf{w};\xi^k)$, we assume that it satisfies:*

$$\mathbb{E}_{\xi^k}\|\nabla f(\mathbf{w};\xi^k)\|^2 \le G^2, \forall \mathbf{w} \in \mathbb{R}^d, \forall k \in [K].$$

Assumption 3. *$F(\mathbf{w})$ is L-smooth ($L > 0$):*

$$F(\mathbf{w}) \le F(\mathbf{w}') + \nabla F(\mathbf{w}')^T(\mathbf{w} - \mathbf{w}') + \frac{L}{2}\|\mathbf{w} - \mathbf{w}'\|^2, \forall \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d.$$

Assumption 4. *The objective function $F(\mathbf{w})$ is lower bounded by F^*: $F(\mathbf{w}) \ge F^*, \forall \mathbf{w} \in \mathbb{R}^d$.*

Ordered Momentum

南京大學 NANJING UNIVERSITY

Convergence Analysis

- Constant learning rate

Theorem 1. *With Assumptions 1, 2, 3 and 4, letting* $\eta = \min\{\frac{1-\beta}{2KL}, \frac{(1-\beta)\Delta^{\frac{1}{2}}}{(LT)^{\frac{1}{2}}\sigma}, \frac{(1-\beta)^{\frac{5}{3}}\Delta^{\frac{1}{3}}}{(LKG)^{\frac{2}{3}}T^{\frac{1}{3}}}\}$, *Algorithm 2 has the following convergence rate:*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2 \leq \mathcal{O}\left(\sqrt{\frac{L\sigma^2}{T}} + \left(\frac{KLG}{T}\right)^{\frac{2}{3}} + \frac{KL}{T}\right)$$

- Delay-adaptive learning rate

The convergence of OrMo with the above delay-adaptive learning rate (called OrMo-DA) is guaranteed by Theorem 2.

Theorem 2. *With Assumptions 1, 3 and 4, letting* $\eta = \min\{\frac{(1-\beta)^2}{8KL}, \sqrt{\frac{(1-\beta)^3\Delta}{TL\sigma^2}}\}$, *OrMo-DA has the following convergence rate:*

$$\mathbb{E}\|\nabla F(\tilde{\mathbf{w}}_T)\|^2 \leq \mathcal{O}\left(\sqrt{\frac{L\sigma^2}{T}} + \frac{KL}{T}\right)$$

Experiment

Experimental details

- All the experiments are implemented based on the Parameter Server framework. Our distributed platform is conducted with Docker.
  - Each Docker container corresponds to either a server or a worker.
- The experiments are conducted under two settings:
  - [homogeneous]: each worker has similar computing capabilities.
  - [heterogeneous]: some workers ($\frac{1}{16}$ of all) are designated as slow workers.
- We evaluate these methods by training ResNet20 model on CIFAR10 and CIFAR100 datasets.

Empirical results of different methods

- Empirical results on CIFAR10 dataset

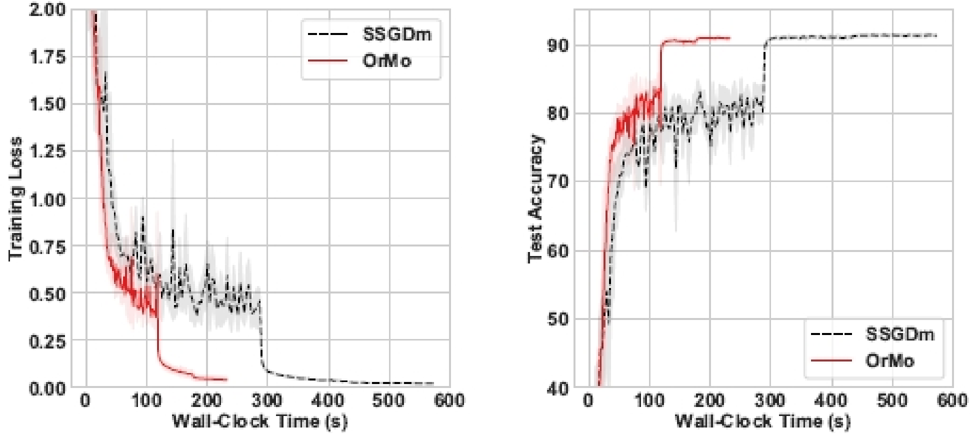
Number of Workers	16 (hom.)		64 (hom.)		16 (het.)		64 (het.)	
Methods	Training Loss	Test Accuracy	Training Loss	Test Accuracy	Training Loss	Test Accuracy	Training Loss	Test Accuracy
ASGD	0.06 ± 0.00	89.77 ± 0.11	0.40 ± 0.02	83.14 ± 0.55	0.06 ± 0.00	89.73 ± 0.19	0.38 ± 0.01	83.94 ± 0.21
naive ASGDm	0.20 ± 0.07	88.15 ± 1.70	0.44 ± 0.06	82.39 ± 1.79	0.58 ± 0.86	73.23 ± 31.61	0.78 ± 0.77	68.75 ± 29.51
shifted momentum	0.08 ± 0.01	90.23 ± 0.27	0.38 ± 0.00	83.72 ± 0.29	0.10 ± 0.02	89.95 ± 0.32	0.37 ± 0.01	83.99 ± 0.23
SMEGA^2	0.05 ± 0.01	90.60 ± 0.42	0.23 ± 0.04	86.82 ± 0.69	0.04 ± 0.01	90.88 ± 0.25	0.22 ± 0.07	86.89 ± 1.42
OrMo	0.04 ± 0.01	90.95 ± 0.27	$\mathbf{0.15 \pm 0.02}$	$\mathbf{88.03 \pm 0.28}$	0.04 ± 0.00	91.01 ± 0.10	0.16 ± 0.03	87.76 ± 0.57
OrMo-DA	$\mathbf{0.03 \pm 0.01}$	$\mathbf{91.17 \pm 0.18}$	0.16 ± 0.02	88.03 ± 0.33	$\mathbf{0.03 \pm 0.01}$	$\mathbf{91.28 \pm 0.37}$	$\mathbf{0.15 \pm 0.02}$	$\mathbf{88.08 \pm 0.38}$

- Empirical results on CIFAR100 dataset

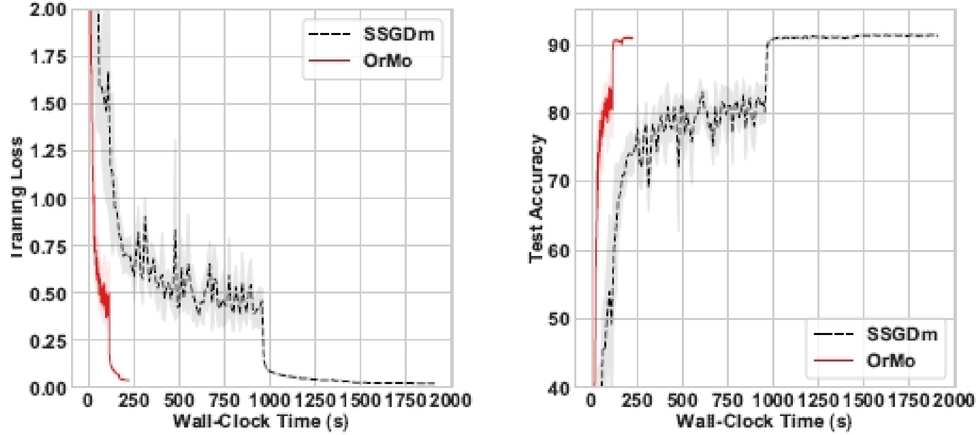
Number of Workers	16 (hom.)		64 (hom.)		16 (het.)		64 (het.)	
Methods	Training Loss	Test Accuracy	Training Loss	Test Accuracy	Training Loss	Test Accuracy	Training Loss	Test Accuracy
ASGD	0.51 ± 0.01	66.16 ± 0.36	0.96 ± 0.03	61.61 ± 0.59	0.51 ± 0.01	65.94 ± 0.39	0.95 ± 0.03	61.74 ± 0.30
naive ASGDm	0.54 ± 0.01	65.46 ± 0.20	1.03 ± 0.05	59.96 ± 0.90	0.53 ± 0.00	65.69 ± 0.42	0.97 ± 0.06	61.13 ± 1.02
shifted momentum	0.47 ± 0.01	66.37 ± 0.14	0.82 ± 0.01	63.55 ± 0.32	0.47 ± 0.00	66.28 ± 0.14	0.82 ± 0.04	63.28 ± 0.66
SMEGA^2	0.41 ± 0.00	67.32 ± 0.22	0.69 ± 0.00	64.16 ± 0.12	0.40 ± 0.01	67.29 ± 0.16	0.68 ± 0.02	64.12 ± 0.53
OrMo	0.41 ± 0.01	67.56 ± 0.34	0.56 ± 0.00	65.48 ± 0.17	0.40 ± 0.01	67.71 ± 0.33	0.58 ± 0.02	65.43 ± 0.35
OrMo-DA	$\mathbf{0.40 \pm 0.00}$	$\mathbf{67.72 \pm 0.21}$	$\mathbf{0.56 \pm 0.01}$	$\mathbf{65.79 \pm 0.12}$	$\mathbf{0.04 \pm 0.00}$	$\mathbf{67.82 \pm 0.20}$	$\mathbf{0.57 \pm 0.01}$	$\mathbf{65.82 \pm 0.30}$

Experiment

- Training curves with respect to wall-clock time on CIFAR10 dataset



(a) homogeneous



(b) heterogeneous

THANKS

南京大學
NANJING UNIVERSITY