ROYAL INSTITUTE
OF TECHNOLOGY

EP3260: Machine Learning Over Networks

Lecture 8: Communication Efficiency

Carlo Fischione

Division of Network and Systems Engineering
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology, Stockholm, Sweden

https://sites.google.com/view/mlons2020/home

April 2020

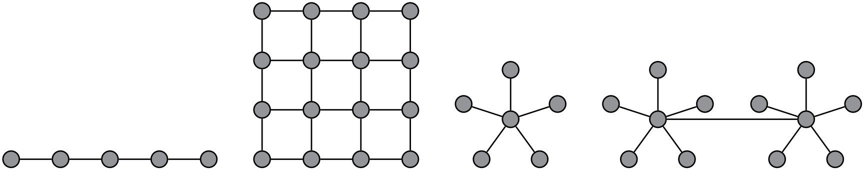
Learning outcomes

- What is the computation-communication tradeoff in a general approach to primal-dual optimizations in ML?
- How quantization affects Gradient Descent Algorithm in ML?
- How quantization affects Stochasitc Gradient Descent Algorithm in ML?

Outline

1. Computation-communication tradeoff in a general approach
2. Quantized Distributed Gradient Descent
3. Parallel Quantized Stochastic Gradient Descent

Recap of previous two lectures



- ML over Master-Workers networks
  - Duality methods (Lec 6)
  - Alternating Direction Methods of Multipliers (ADMM) (Lec 7)
- ML over general networks
  - Duality methods with consensus (Lec 6)
  - ADMM with consensus (Lec 7)

Outline

1. Computation-communication tradeoff in a general approach
2. Quantized Distributed Gradient Descent
3. Parallel Quantized Stochastic Gradient Descent

A general framework for primal-dual methods

- **Definition** (L-Lipschitz Continuity). A function $h : \mathbb{R}^m \to \mathbb{R}$ is L-Lipschitz Continuos if $\forall$ $\boldsymbol{u}$ and $\boldsymbol{v} \in \mathbb{R}^m$, we have $|h(\boldsymbol{u}) - h(\boldsymbol{v})| \leq L\|\boldsymbol{u} - \boldsymbol{v}\|$

- **Definition** (L-Bounded Support). A function $h : \mathbb{R}^m \to \mathbb{R} \cup +\infty$ has L bounded support if its effective domain is bounded by L
$h(\boldsymbol{u}) < +\infty \implies \|\boldsymbol{u}\| \leq L$

- **Definition** ($\frac{1}{\mu}$-Smoothness). A function $h : \mathbb{R}^m \to \mathbb{R}$ is $\frac{1}{\mu}$ smooth if it is differentiable and its derivative is $\frac{1}{\mu}$-Lipschitz continuous

$$h(\boldsymbol{u}) \leq h(\boldsymbol{v}) + \nabla h(\boldsymbol{v})^T(\boldsymbol{u} - \boldsymbol{v}) + \frac{1}{2\mu}\|\boldsymbol{u} - \boldsymbol{v}\|^2 \quad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^m$$

- **Definition** (μ-Strong Convexity). A function $h : \mathbb{R}^m \to \mathbb{R}$ is μ strongly convex for $\mu \geq 0$ if

$$h(\boldsymbol{u}) \geq h(\boldsymbol{v}) + \boldsymbol{s}^T(\boldsymbol{u} - \boldsymbol{v}) + \frac{\mu}{2}\|\boldsymbol{u} - \boldsymbol{v}\|^2 \quad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^m$$

for any $\boldsymbol{s} \in \partial h(\boldsymbol{v})$, where $\partial h(\boldsymbol{v})$ denotes the subgradient of h at $\boldsymbol{v}$

A general framework for primal-dual methods

- We now study a general framework to ML problems having the form

$$\min_{\boldsymbol{u}\in\mathbb{R}^n} \ell(\boldsymbol{u}) + r(\boldsymbol{u}) \qquad (I)$$

for convex functions $\ell(\boldsymbol{u}) = \sum_i \ell_i(\boldsymbol{u})$ (the loss function) and $r(\boldsymbol{u})$ (the regularizer function, e.g. $\lambda\|\boldsymbol{u}\|_p$).

- This formulation includes ML problems such as Support Vector Machines, Linear and Logistic Regression, Lasso or Sparse Logistic Regression

- This general framework maps the ML problem (I) into one of the two following problems

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n} O_A(\boldsymbol{\alpha}) = f(A\boldsymbol{\alpha}) + g(\boldsymbol{\alpha}) \qquad (A)$$

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} O_B(\boldsymbol{w}) = f^*(\boldsymbol{w}) + g^*(-A^T\boldsymbol{w}) \qquad (B)$$

where $\boldsymbol{\alpha} \in \mathbb{R}^n$ and $\boldsymbol{w} \in \mathbb{R}^m$, $A = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{m\times n}$ is a data matrix with column vectors $\boldsymbol{x}_i \in \mathbb{R}^m\ \forall i$, and f^* and g^* are the convex conjugates of f and g respectively. (A) is called primal. (B) dual.

A general framework for primal-dual methods

- Optimization Problem (A) and (B) are equivalent according to the Fenchel-Rockafellar duality
- Given $\boldsymbol{\alpha}$ from (A), we achieve $\boldsymbol{w}$ of (B) as $\boldsymbol{w} = \boldsymbol{w}(\boldsymbol{\alpha}) := \nabla f(A\boldsymbol{\alpha})$
- (A) and (B) give the duality gap $G(\boldsymbol{\alpha}) := O_A(\boldsymbol{\alpha}) - [-O_B(\boldsymbol{w}(\boldsymbol{\alpha}))]$
- Recall that the duality gap is always non negative and is zero if the pair $(\boldsymbol{\alpha}^*, \boldsymbol{w}^*)$ is optimal. It gives an upper bound on the unknown primal or dual optimization error (certificate of the suboptimality) since

$$O_A(\boldsymbol{\alpha}) \geq O_A(\boldsymbol{\alpha}^*) \geq -O_B(\boldsymbol{w}^*) \geq -O_B(\boldsymbol{w}(\boldsymbol{\alpha}))$$

- Assumption: Problem (A) is with f $\frac{1}{\tau}$-smooth and the function g are separable $g(\boldsymbol{\alpha}) = \sum_i g_i(\boldsymbol{\alpha})$, with $g_i(\boldsymbol{\alpha})$ having L-bounded support.
- Given the equivalence between (A) and (B), this gives that in problem (B) f^* is τ-strongly convex and the function $g^*(-A^T\boldsymbol{w}) = \sum_i g_i^*(-\boldsymbol{x}_i^T\boldsymbol{w})$ is separable with each g_i^* being L-Lipschitz

Common Losses and Regularizers

(i) Losses

Loss	Obj	f / g^*
Least Squares	(A)	$f=\frac{1}{2}\|A\boldsymbol{\alpha}-\mathbf{b}\|_2^2$
	(B)	$g^*=\frac{1}{2}\|A^\top\mathbf{w}-\mathbf{b}\|_2^2$
Logistic Reg.	(A)	$f=\frac{1}{m}\sum_j\log(1+\exp(b_j\mathbf{x}_j^\top\boldsymbol{\alpha}))$
	(B)	$g^*=\frac{1}{n}\sum_i\log(1+\exp(b_i\mathbf{x}_i^\top\mathbf{w}))$
SVM	(B)	$g^*=\frac{1}{n}\sum_i\max(0,1-y_i\mathbf{x}_i^\top\mathbf{w})$
Absolute Dev.	(B)	$g^*=\frac{1}{n}\sum_i\lvert\mathbf{x}_i^\top\mathbf{w}-y_i\rvert$

(ii) Regularizers

Regularizer	Obj	g / f^*
Elastic Net	(A)	$g=\lambda(\eta\|\boldsymbol{\alpha}\|_1+\frac{1-\eta}{2}\|\boldsymbol{\alpha}\|_2^2)$
	(B)	$f^*=\lambda(\eta\|\mathbf{w}\|_1+\frac{1-\eta}{2}\|\mathbf{w}\|_2^2)$
L_2	(A)	$g=\frac{\lambda}{2}\|\boldsymbol{\alpha}\|_2^2$
	(B)	$f^*=\frac{\lambda}{2}\|\mathbf{w}\|_2^2$
L_1	(A)	$g=\lambda\|\boldsymbol{\alpha}\|_1$
Group Lasso	(A)	$g=\lambda\sum_p\|\boldsymbol{\alpha}_{\mathcal{I}_p}\|_2,\ \mathcal{I}_p\subseteq[n]$

Assumptions

- Our main interest is now to apply (A) or (B) for deriving a distributed solution to the initial ML problem (I).

- The data set A is distributed over K machines according to a partition $\{\mathcal{P}_k\}_{k=1}^{K}$ of the columns of $A \in \mathbb{R}^{m \times n}$. The size of the partition on the machine k is $n_k = |\mathcal{P}_k|$

- For machine $k \in \{1, \ldots, K\}$ and vector $\boldsymbol{\alpha} \in \mathbb{R}^n$, let $\boldsymbol{\alpha}_{[k]} \in \mathbb{R}^n$ a vector with elements $(\boldsymbol{\alpha}_{[k]})_i := \alpha_i$ if $i \in \mathcal{P}_k$ and $(\boldsymbol{\alpha}_{[k]})_i := 0$ otherwise

- Analogously, let $A_{[k]}$ be a matrix with columns corresponding to those of A according to the partition, and zeros elsewehere

- The function g in (A) can be easily distributed, since $g(\boldsymbol{\alpha}) = \sum_i g_i(\boldsymbol{\alpha})$

- However, the function $f(A\boldsymbol{\alpha})$ is not in general separable

- The main idea of the general framework for primal-dual methods is a separable approximation of the function $O_A(\boldsymbol{\alpha})$. See next

Approximation of $O_A(\boldsymbol{\alpha})$

- Let $\boldsymbol{v} := A\boldsymbol{\alpha} \in \mathbb{R}^m$ and let $\boldsymbol{\alpha}_{[k]}^{(t+1)} := \boldsymbol{\alpha}_{[k]}^{(t)} + \gamma \Delta\boldsymbol{\alpha}_{[k]}$, where $\Delta\boldsymbol{\alpha}_{[k]}$ denotes a certain change of variables $\boldsymbol{\alpha}_i$ for $i \in \mathcal{P}_k$ and $(\Delta\boldsymbol{\alpha}_{[k]})_i := 0 \; \forall i \ni \mathcal{P}_k$
- Then, $O_A(\boldsymbol{\alpha})$ can be exactly decomposed as follows

$$\sum_{i\in[n]} g_i(\alpha_i^{(t)} + \Delta\alpha_i) + f(\boldsymbol{v}^{(t)}) + \nabla f(\boldsymbol{v}^{(t)})^T A\Delta\boldsymbol{\alpha} +$$

$$\frac{\sigma'}{2\tau}\Delta\boldsymbol{\alpha}^T \begin{bmatrix} A_{[1]}^T A_{[1]} & & 0 \\ & \ddots & \\ 0 & & A_{[K]}^T A_{[K]} \end{bmatrix} \Delta\boldsymbol{\alpha} = \sum_{k=1}^{K} G_k^{\sigma'}(\Delta\boldsymbol{\alpha}_k; \boldsymbol{v}^{(t)}, \boldsymbol{\alpha}_{[k]})$$

$$G_k^{\sigma'}(\Delta\boldsymbol{\alpha}_k; \boldsymbol{v}^{(t)}, \boldsymbol{\alpha}_{[k]}) := \frac{1}{K} f(\boldsymbol{v}^{(t)}) + \boldsymbol{w}^T A_{[k]}\Delta\boldsymbol{\alpha}_{[k]} + \frac{\sigma'}{2\tau}\left\|A_{[k]}\Delta\boldsymbol{\alpha}_{[k]}\right\|^2 +$$

$$\sum_{i\in\mathcal{P}_k} g_i(\alpha_i^{(t)} + \Delta\boldsymbol{\alpha}_{[k]_i})$$

Approximation of $O_A(\boldsymbol{\alpha})$

- The function $G_k^{\sigma'}(\Delta\boldsymbol{\alpha}_k; \boldsymbol{v}^{(t)}, \boldsymbol{\alpha}_{[k]}^{(t)})$ is completely local at processor k except the coupling variable $\boldsymbol{v}^{(t)} = A\boldsymbol{\alpha}^{(t)}$ which is global
- The decomposition of $O_A(\boldsymbol{\alpha})$ suggests that we can iteratively solve local problems and exchange $\boldsymbol{\alpha}_k$ to reconstruct $\boldsymbol{v}^{(t)}$

$$\min_{\Delta\boldsymbol{\alpha}_k \in \mathbb{R}^n} G_k^{\sigma'}(\Delta\boldsymbol{\alpha}_k; \boldsymbol{v}^{(t)}, \boldsymbol{\alpha}_{[k]}^{(t)})$$

- Each processor can do the local minimisation and just exchange to the others the variables $\boldsymbol{\alpha}_k$ at each iteration t
- Note that the minimization is done independently from other processors k and thus the resulting $G_k^{\sigma'}(\Delta\boldsymbol{\alpha}_k; \boldsymbol{v}^{(t)}, \boldsymbol{\alpha}_{[k]}^{(t)})$ will not give the exact term to perfectly reconstruct $O_A(\boldsymbol{\alpha})$. However, this is enough to approximately compute the optimal solution with approximation Θ

Algorithm 1: Generalized primal-dual algorithm

Algorithm 1: Generalized primal-dual algorithm

Input Data matrix A distributed column-wise according to the partition $\{\mathcal{P}_k\}_{k=1}^{K}$, aggregation parameter $\gamma \in (0, 1]$, and σ'.

Starting point $\boldsymbol{\alpha}^{(0)} := 0 \in \mathbb{R}^n$, $\boldsymbol{v}^{(0)} := 0 \in \mathbb{R}^m$

for $t = 0, 1, \ldots$ **do**

  for $k = 1, 2, \ldots, K$ in parallel in each processor **do**

    Compute a Θ approximate solution to

$$\min_{\Delta\boldsymbol{\alpha}_k \in \mathbb{R}^n} G_k^{\sigma'}(\Delta\boldsymbol{\alpha}_k; \boldsymbol{v}^{(t)}, \boldsymbol{\alpha}_{[k]}^{(t)})$$

    $\boldsymbol{\alpha}_{[k]}^{(t+1)} := \boldsymbol{\alpha}_{[k]}^{(t)} + \gamma \Delta\boldsymbol{\alpha}_{[k]}$

    $\Delta\boldsymbol{v}_k := A_{[k]} \Delta\boldsymbol{\alpha}_{[k]}$. Transmit to the other processors $\Delta\boldsymbol{v}_k$

  end for

  Compute $\boldsymbol{v}^{(t+1)} = \boldsymbol{v}^{(t)} + \gamma \sum_{k=1}^{K} \Delta\boldsymbol{v}_k$

end for

Application to primal and dual

Algorithm 2: Primal mapping

Map Problem (I) into (A)

Distribute dataset A by columns (here typically features) according to the partition $\{\mathcal{P}_k\}_{k=1}^{K}$

Run Algorothm 1 with appropriate choice of parameter γ and subproblem parameter σ'

Algorithm 3: Dual mapping

Map Problem (I) into (B)

Distribute dataset A by columns (here typically training points) according to the partition $\{\mathcal{P}_k\}_{k=1}^{K}$

Run Algorothm 1 with appropriate choice of parameter γ and subproblem parameter σ'

Algorithm 1 for convex g_i and L-Lipschiz g_i^*

- **Theorem 1**: Consider Algorithm 1 with $\gamma := 1$, and let Θ be the quality of the local solver at processor k. Let g_i have L bounded support, and let f be $\frac{1}{\tau}$-mooth. Let T be such that

$$T \geq T_0 + \max\left(\left\lceil \frac{1}{1-\Theta} \right\rceil, \frac{4L^2}{\tau\varepsilon_G(1-\Theta)}\right)$$

$$T_0 \geq t_0 + \left[\frac{2}{1-\Theta}\left(\frac{8L^2}{\tau\varepsilon_G} - 1\right)\right]$$

$$t_0 \geq \max\left(0, \left\lceil \frac{1}{1-\Theta} \log\left(\frac{\tau n(O_A(\boldsymbol{\alpha}^{(0)})) - O_A(\boldsymbol{\alpha}^*))}{2L^2K}\right)\right\rceil\right)$$

Then

$$\mathbb{E}[O_A(\bar{\boldsymbol{\alpha}}) - (-O_B(\boldsymbol{w}(\bar{\boldsymbol{\alpha}})))] \leq \varepsilon_G \qquad \bar{\boldsymbol{\alpha}} = \frac{1}{T-T_0}\sum_{t=T_0+1}^{T-1} \boldsymbol{\alpha}^{(t)}$$

Algorithm 1 for strong. convex g_i and smooth g_i^*

- **Theorem 2**: Consider Algorithm 1 with $\gamma := 1$, and let Θ be the quality of the local solver. Let g_i be μ strongly convex $\forall i$ and let f be $\frac{1}{\tau}$-mooth. Let T be such that

$$T \geq \frac{1}{1-\Theta} \frac{\mu\tau + 1}{\mu\tau} \log \frac{1}{\varepsilon_{O_A}}$$

Then $\mathbb{E}[O_A(\boldsymbol{\alpha}^{(T)}) - O_A(\boldsymbol{\alpha}^*)] \leq \varepsilon_{O_A}$
Moreover, if

$$T \geq \frac{1}{1-\Theta} \frac{\mu\tau + 1}{\mu\tau} \log \left(\frac{1}{1-\Theta} \frac{\mu\tau + 1}{\mu\tau} \frac{1}{\varepsilon_{O_A}} \right)$$

then the expected duality gap

$$\mathbb{E}[O_A(\boldsymbol{\alpha}^{(T)}) - (-O_B(\boldsymbol{w}(\boldsymbol{\alpha}^T))] \leq \varepsilon_G$$

Criteria for Running Algorithms 2 vs. 3

	Smooth ℓ	Non-smooth and separable ℓ
Strongly convex r	Alg. 2 or 3	Alg. 3
Non-strongly convex and separable r	Alg. 2	-

	Smooth ℓ	Non-smooth and separable ℓ
Strongly convex r	Theorem 3	Theorem 2
Non-strongly convex and separable r	Theorem 2	-

Comparison with ADMM

- We can apply consensus ADMM to (B) (or (A)):

$$\min_{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_k,\boldsymbol{w}} \sum_{k=1}^{K} \sum_{i\in\mathcal{P}_k} g^*(-\boldsymbol{x}_i^T\boldsymbol{w}_k) + f^*(\boldsymbol{w}) \qquad \text{s.t.} \quad \boldsymbol{w}_k = \boldsymbol{w} \quad \forall k$$

- We solve the problem by the augmented Lagrangian

$$\boldsymbol{w}_k^{(t+1)} = \arg\min_{\boldsymbol{w}_k} \sum_{i\in\mathcal{P}_k} g^*(-\boldsymbol{x}_i^T\boldsymbol{w}_k) + \rho {\boldsymbol{u}_k^{(t)}}^T(\boldsymbol{w}_k - \boldsymbol{w}^{(t)}) + \frac{\rho}{2}\|\boldsymbol{w}_k - \boldsymbol{w}^{(t)}\|^2$$

$$\boldsymbol{w}^{(t+1)} = \arg\min_{\boldsymbol{w}} f^*(\boldsymbol{w}) + \frac{\rho K}{2}\|\boldsymbol{w} - (\bar{\boldsymbol{w}}_k^{(t+1)} - \bar{\boldsymbol{u}}_k^{(t)})\|^2$$

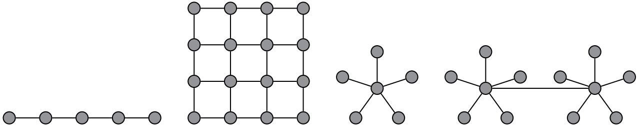
$$\boldsymbol{u}_k^{(t+1)} = u_k^{(t)} + \boldsymbol{w}_k^{(t+1)} - \boldsymbol{w}^{(t+1)}$$

- ADMM has the drawback of the proximal updating

Outline

1. Computation-communication tradeoff in a general approach
2. Quantized Distributed Gradient Descent
3. Parallel Quantized Stochastic Gradient Descent

Problem formulation



- Set of n nodes $\mathcal{V} = (1, \ldots, n)$, a set of edges $\mathcal{E} = \mathcal{V} \times \mathcal{V}$. The nodes communicate over a connected an undirected graph $\mathcal{G} = (\mathcal{G}, \mathcal{E})$
- $\mathcal{N}_i$ is the set of neighbours that node i communicates with
- Each node i has a strongly convex and smooth function $f_i(\boldsymbol{w}) : \mathbb{R}^p \to \mathbb{R}$
- All the nodes wish to solve the ML optimization problem $\underset{\boldsymbol{w} \in \mathbb{R}^p}{\text{minimize}}\ f(\boldsymbol{w}) = \underset{\boldsymbol{w} \in \mathbb{R}^p}{\text{minimize}}\ \frac{1}{|\mathcal{N}_i|} \sum_{i \in \mathcal{N}_i} f_i(\boldsymbol{w})$
- Clearly, $f(\boldsymbol{w})$ is strongly convex and smooth and there is a unique minimizer $\boldsymbol{w}^*$

Problem formulation

- A node has only access to its local function and it can communicate only with the neighbours $\mathcal{N}_i$

- As we have seen in the previous lectures, we can equivalently rewrite the ML optimization problem by the consensus method as

$$\underset{\boldsymbol{w}\in\mathbb{R}^p}{\text{minimize}} \quad \frac{1}{|\mathcal{N}_i|}\sum_{i\in\mathcal{N}_i} f_i(\boldsymbol{w})$$
$$\text{s.t.} \quad \boldsymbol{w}_i = \boldsymbol{w}_j \quad \forall i,j\in\mathcal{N}_i$$

- We could solve the problem by the methods of the previous lectures with local iterates

- However, the nodes cannot exchange the decision variables $\boldsymbol{w}_{i,t}$, but a quantized version $\boldsymbol{z}_{i,t} = Q(\boldsymbol{w}_{i,t})$, where $Q(\cdot)$ is a quantizer function

- The quantization can substantially reduce the amount of information to exchange, which is very important, e.g., in IoT applications

Quantized Distributed Gradient Descent (QDGD)

Algorithm 4: QDGD

Node i requires Weights $\{a_{i,j}\}_{j=1}^{n}$

Set $\boldsymbol{w}_{i,0} = 0$ and compute $\boldsymbol{z}_{i,0} = Q(\boldsymbol{w}_{i,0})$

for $t = 0, 1, \ldots, T-1$ **do**

  Transmit $\boldsymbol{z}_{i,t} = Q(\boldsymbol{w}_{i,t})$ to $\mathcal{N}_i$ and receive $\boldsymbol{z}_{j,t}$

  Compute the local decision variable as

$$\boldsymbol{w}_{i,t+1} = (1 - \varepsilon + \varepsilon a_{i,i})\boldsymbol{w}_{i,t} + \varepsilon \sum_{j \in \mathcal{N}_i} a_{i,j}\boldsymbol{z}_{j,t} - \alpha\varepsilon \nabla f_i(\boldsymbol{w}_{i,t})$$

end for

Return $\boldsymbol{w}_{i,T}$

- ε and α are positive step sizes to be appropriately chosen
- There are no particular restrictions on the type of quantizer (see later)

QDGD Convergence analysis

- **Assumption 1**: $\forall\ \boldsymbol{w} \in \mathbb{R}^p$, $\boldsymbol{y} \in \mathbb{R}^p$, f_i is differentiable and smooth with parameter L
$$\|\nabla f_i(\boldsymbol{w}) - \nabla f_i(\boldsymbol{y})\| \leq L\|\boldsymbol{w} - \boldsymbol{y}\| \qquad \forall i$$
- **Assumption 2**: $\forall\ \boldsymbol{w} \in \mathbb{R}^p$, $\boldsymbol{y} \in \mathbb{R}^p$, f_i is strongly convex with parameter μ
$$(\nabla f_i(\boldsymbol{w}) - \nabla f_i(\boldsymbol{y}))^T(\boldsymbol{w} - \boldsymbol{y}) \geq \mu\|\boldsymbol{w} - \boldsymbol{y}\|^2 \qquad \forall i$$
- **Assumption 3**: The quantizer is unbiased and has a bounded variance:
$$\mathbb{E}[Q(\boldsymbol{w})|\boldsymbol{w}] = \boldsymbol{w} \qquad \mathbb{E}\left[\|Q(\boldsymbol{w}) - \boldsymbol{w}\|^2|\boldsymbol{w}\right] \leq \sigma^2$$
- **Assumption 4**: The matrix $\boldsymbol{A} = [a_{i,j}] \in \mathbb{R}^{n,n}$ is symmetric and doubly stochastic:
$$\boldsymbol{A} = \boldsymbol{A}^T \qquad \boldsymbol{A}\mathbf{1} = \mathbf{1} \qquad \boldsymbol{A}^T\mathbf{1} = \mathbf{1}$$

QDGD Convergence analysis

- **Theorem 4**: Consider the QDGD Algorithm. Suppose Assumptions 1 $\sim$ 4 hold. Let δ be an arbitrary scalar in $(0, 1/2)$ and let $\varepsilon = c_1/T^{3\delta/2}$ and $\alpha = c_2/T^{\delta/2}$, where c_1 and c_2 are arbitrary positive constants independent of T. The, for each node i

$$\mathbb{E}\left[\|\boldsymbol{w}_{i,T} - \boldsymbol{w}^*\|^2\right] \leq \mathcal{O}\left(\left(\frac{4nc_2^2 D^2(3+2L/\mu)^2}{(1-\beta)^2} + \frac{2c_1 n\sigma^2\|\boldsymbol{A} - \boldsymbol{A}_D\|}{\mu c_2}\right)\frac{1}{T^\delta}\right)$$

where

$$D^2 = 2L\sum_{i=1}^{n}(f_i(0) - f_i^*), \qquad f_i^* = \min_{\boldsymbol{w}\in\mathbb{R}^p} f_i(\boldsymbol{w})$$

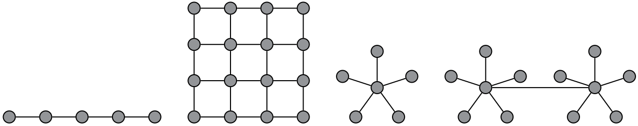
- The theorem shows that QDGD provides an approximation solution with vanishing deviation from the optimal solution, despite the quantization noise that does not vanish with the iterations

- The convergence rate is sublinear

Outline

1. Computation-communication tradeoff in a general approach
2. Quantized Distributed Gradient Descent
3. Parallel Quantized Stochastic Gradient Descent

Stochastic Gradient Descent (SGD)



- Set of n nodes $\mathcal{V} = (1, \ldots, n)$, a set of edges $\mathcal{E} = \mathcal{V} \times \mathcal{V}$. The nodes communicate over a connected an undirected graph $\mathcal{G} = (\mathcal{G}, \mathcal{E})$

- Let $\mathcal{W}$ be a known convex set. There is a global function $f(\boldsymbol{w}) : \mathcal{W} \to \mathbb{R}$ which is unknown to the nodes

- Each node i has access to its measurement of the stochastic gradient of $f(\boldsymbol{w})$

- All the nodes wish to solve the ML optimization problem $\underset{\boldsymbol{w} \in \mathbb{R}^p}{\text{minimize}} \ f(\boldsymbol{w})$

SGD

- **Definition 1**: Given the function $f(\boldsymbol{w}) : \mathcal{W} \to \mathbb{R}$, a stochastic gradient of f is a random function $\tilde{g}(\boldsymbol{w})$ so that $\mathbb{E}\left[\tilde{g}(\boldsymbol{w})\right] = \nabla f(\boldsymbol{w})$

- **Definition 2**: The stochastic gradient has second oder moment at most B if $\mathbb{E}\left[\|\tilde{g}(\boldsymbol{w})\|^2\right] \leq B$ for $\boldsymbol{w} \in \mathcal{W}$

- **Definition 3**: The stochastic gradient has variance at most σ^2 if $\mathbb{E}\left[\|\tilde{g}(\boldsymbol{w}) - \nabla f(\boldsymbol{w})\|^2\right] \leq \sigma^2$ for $\boldsymbol{w} \in \mathcal{W}$.

SGD

- A standard instance of the Stochastic Gradient Descent (SGD) is

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \tilde{g}(\boldsymbol{w}_t)$$

  where η_t is variable step size

- **Theorem 5**: Let $\mathcal{W} \subseteq \mathbb{R}^n$ be convex and let the function $f(\boldsymbol{w}) : \mathcal{W} \to \mathbb{R}$ be unknown, convex, and L-smooth. Let $\boldsymbol{w}_0 \in \mathcal{W}$ be given and let $R^2 = \sup_{\boldsymbol{w} \in \mathcal{W}} \|\boldsymbol{w} - \boldsymbol{w}_0\|^2$. Let $T \geq 0$ be fixed. Given repeated and independent access to stocastic gradients with variance bound σ^2, the SGD with constant step size $\eta_t = 1/(L + 1/\gamma)$ where $\gamma = R/\sigma\sqrt{2/T}$ achieves

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=0}^{T} \boldsymbol{w}_t\right)\right] - \min_{\boldsymbol{w} \in \mathcal{W}} f(\boldsymbol{w}) \leq R\sqrt{\frac{2\sigma^2}{T}} + \frac{LR^2}{T}$$

Parallel SGD

- If we have K processors each making an independent measurement of the stochastic gradient $\tilde{g}^i(\boldsymbol{w})$, and each processor i communicates to each other such measurement at every time step t, a parallel SGD is

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\eta_t}{K} \sum_{i=1}^{K} \tilde{g}^i(\boldsymbol{w}_t)$$

- **Corolary 1**: Let $\mathcal{W}$, $f(\boldsymbol{w})$, $\boldsymbol{w}_0$ and R as in the previous theorem. Fix $\varepsilon \geq 0$. Suppose to run parallel SGD on K processors each with access to independent stochastic gradients with second moment bound B, with step size $\eta_t = 1/(L + \sqrt{K}/\gamma)$ with γ as in the previous theorem. If $T = \mathcal{O}\left(R^2 \max\left(\frac{2B}{K\varepsilon^2}, \frac{L}{\varepsilon}\right)\right)$ then

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=0}^{T} \boldsymbol{w}_t\right)\right] - \min_{\boldsymbol{w}\in\mathcal{W}} f(\boldsymbol{w}) \leq \varepsilon$$

Parallel Quantized SGD

Algorithm 5: PQSGD

for $t = 0, 1, \ldots, T-1$ **do**

Let $\tilde{g}^i(\boldsymbol{w}_t)$ be an independent stochastic gradient

Broadcast $\boldsymbol{z}_{i,t} = Q(\tilde{g}^i(\boldsymbol{w}_t))$ to all nodes and receive $\boldsymbol{z}_{j,t}$

Compute the local estimate of the global decision variable as

$$\boldsymbol{w}_{i,t+1} = \boldsymbol{w}_{i,t} - \frac{\eta_t}{K} \sum_{i=1}^{K} \boldsymbol{z}_{i,t}$$

end for

Return $\boldsymbol{w}_{i,T}$

- Where $Q(\cdot)$ is a quantizer (see below)
- Does the algorithm converge? Not in general...

Quantization

- Let $\boldsymbol{v} \in \mathbb{R}^n$ with $\boldsymbol{v} \neq 0$, and let $s \geq 1$. The "low precision quantizer" is

$$Q_s(\boldsymbol{v}) = [Q_s(v_i) = \|\boldsymbol{v}\|_2 \operatorname{sgn}(v_i) \xi_i(\boldsymbol{v}, s)]$$

where $\xi_i(\boldsymbol{v}, s)$ are independent random variables with outcome

$$\xi_i(\boldsymbol{v}, s) = \begin{cases} \ell/s & \text{with probability } 1 - p\left(\frac{|v_i|}{\|\boldsymbol{v}\|_2}, s\right) \\ (\ell+1)/s & \text{otherwise} \end{cases}$$

with $p(a, s) = as - \ell$ for any $a \in [0, 1]$, and the integer $0 \leq \ell < s$ to be chosen such that $|w_i|/\|\boldsymbol{w}\| \in [\ell/s, (l+1)/s]$

- ℓ is the quantization index, and s is the upper bound of the quantization levels

- Example: if $s = 1$, the quantization levels are $0, 1, -1$

Quantization

- Motivation: $\xi_i(\boldsymbol{v}, s)$ has minimal variance over distributions with support $\{0, 1/s, \ldots, 1\}$

- **Lemma**: For any vector $\boldsymbol{v} \in \mathbb{R}^n$, 1) $\mathbb{E}[Q_s(\boldsymbol{v})] = \boldsymbol{v}$ (unbiasedness) 2) $\mathbb{E}[\|Q_s(\boldsymbol{v}) - \boldsymbol{v}\|_2^2 \leq \min(n/s^2, \sqrt{n}/s)\|\boldsymbol{v}\|_2^2$ (variance bund), and 3) $\mathbb{E}[\|Q_s(\boldsymbol{v})\|_0] \leq s(s + \sqrt{n})$

- **Theorem**: Let $f : \mathbb{R}^n \to \mathbb{R}$ be fixed, and let $\boldsymbol{w} \in \mathbb{R}^n$ be arbitrary. Fix $s \geq 2$ quantization levels. If $\tilde{g}(\boldsymbol{w})$ is a stochastic gradient for f at $\boldsymbol{w}$ with second order moment B, then $Q_s(\tilde{g}(\boldsymbol{w}))$ is a stochastic gradient for f at $\boldsymbol{w}$ with variance bound $\min(n/s^2, \sqrt{n}/s)B$. There is an encoding scheme so that in expectation, the number of bits to communicate $Q_s(\tilde{g}(\boldsymbol{w}))$ is upper bounded by

$$\left(3 + \left(\frac{3}{2} + o(1)\right) \log\left(\frac{2(s^2 + n)}{s(s + \sqrt{n})}\right)\right) s(s + \sqrt{n}) + 32$$

Convergence of Parallel QSGD

- **Theorem 6** (Smooth Convex Parallel QSGD). Let $\mathcal{W}$, $f(\boldsymbol{w})$, $\boldsymbol{w}_0$, R and γ as in the main SGD convergence theorem. Let $\varepsilon > 0$. Suppose to run the Parallel QSGD algorithm on K processors accessing independent stochastic gradients with second moment bound B, with step size $\eta_t = 1/(L + \sqrt{K}/\gamma)$ with $\sigma = B'$ with $B' = \min(\frac{n}{s^2}, \frac{\sqrt{n}}{s})B$. If $T = \mathcal{O}\left(R^2 \max\left(\frac{2B'}{K\varepsilon^2}, \frac{L}{\varepsilon}\right)\right)$ then

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=0}^{T}\boldsymbol{w}_t\right)\right] - \min_{\boldsymbol{w}\in\mathcal{W}} f(\boldsymbol{w}) \leq \varepsilon$$

  Moreover, the Parallel QSGD requires a number of bits given by the previous theorem per communication round. If $s = \sqrt{n}$, the number of bits is reduced to $2.8n + 32$.

Convergence of Parallel QSGD

- **Theorem** (Smooth non Convex Parallel QSGD). Let $\mathcal{W}$, $\boldsymbol{w}_0$, R and γ as in the main SGD convergence theorem. Let $f(\boldsymbol{w})$ be an L-smooth possibly non-convex function, and let $\boldsymbol{w}_1$ be an arbitrary initial point. Let $T > 0$ be fixed, and $s > 0$.

  Then there is a random stopping time R supported on $\{1, \ldots, N\}$ so that the Parallel QSGD with quantization level s constant stepsizes $\eta = \mathcal{O}(1/L)$ and access to stochastic gradients of f with second moment bound B satisfies

$$\frac{1}{L}\mathbb{E}\left[\|\nabla f(\boldsymbol{w})\|_2^2\right] \leq \mathcal{O}\left(\frac{\sqrt{L(f(\boldsymbol{w}_1) - f^*}}{N} + \frac{B\min(n/s^2, \sqrt{n}/s)}{L}\right)$$

  Moreover, the number of bits to communicate for each gradient transmission is the same as in the previous theorem

CA6: Communication efficiency

Split the "MNIST" dataset to 10 random disjoint subsets, each for one worker, and consider SVM classifier in the form of $\min_{\boldsymbol{w}} \frac{1}{N} \sum_{i \in [N]} f_i(\boldsymbol{w})$ with $N = 10$. An alternative approach to improve communication-efficiency is to compress the information message to be exchanged (usually gradients – either in primal or dual forms). Consider two compression/quantization methods for a vector: (Q1) keep only K values of a vector and set the rest to zero and (Q2) represent every element with fewer bits (e.g., 4 bits instead of 32 bits).

a) Repeat parts a-b from CA5 using Q1 and Q2. Can you integrate Q1/Q2 to your solution in part c from CA5? Discuss.

b) How do you make SVRG and SAG communication efficient for large-scale ML?

Some references

- V. Smith, S. Forte, C. M. M. Takáč, M. I. Jordan, M. Jaggi, "CoCoA: A general Framework for Communication-Efficient Distributed Optimization", JMLR, 2018.

- A. Reisizadeh, A. Mokhtari, H- Hassani, R. Pedarsani, "An Exact Quantized Decentralized Gradient Descent Algorithm," arXiv, 2018.

- D. Alistarh, D. Grubic, J. Li, R. Tomioka, M. Vojnovic, "QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding",' NIPS, 2017.