Exponential Graphs are Provably Efficient in Decentralized Deep Training

Bicheng Ying[*1,3], Kun Yuan[*2], Yiming Chen[*2], Hanbin Hu[4], Pan Pan[2], and Wotao Yin[2]

1. University of California, Los Angeles 2. DAMO Academy, Alibaba Group, 3. Google Inc. 4. University of California, Santa Barbara
* equal contributions

2021 Conference on Neural Information Processing Systems (NeurIPS)

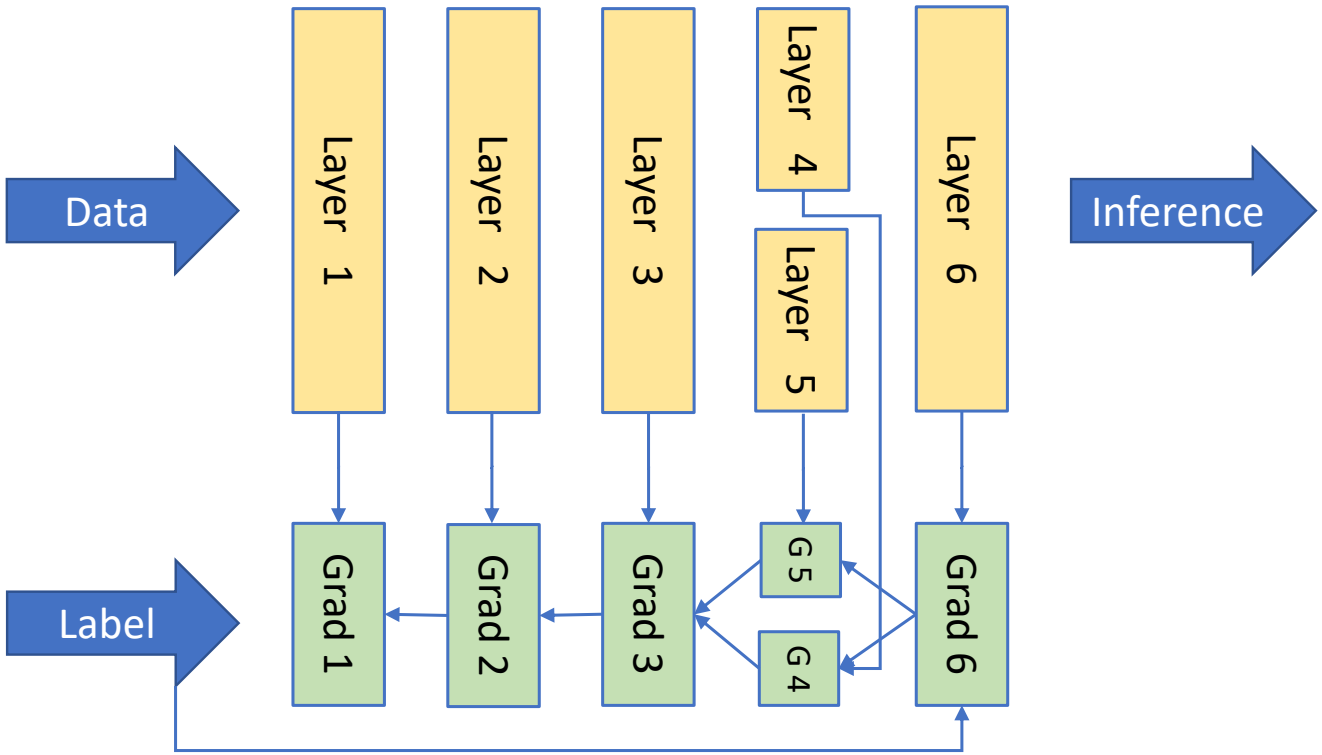
Outline

NEURAL INFORMATION PROCESSING SYSTEMS

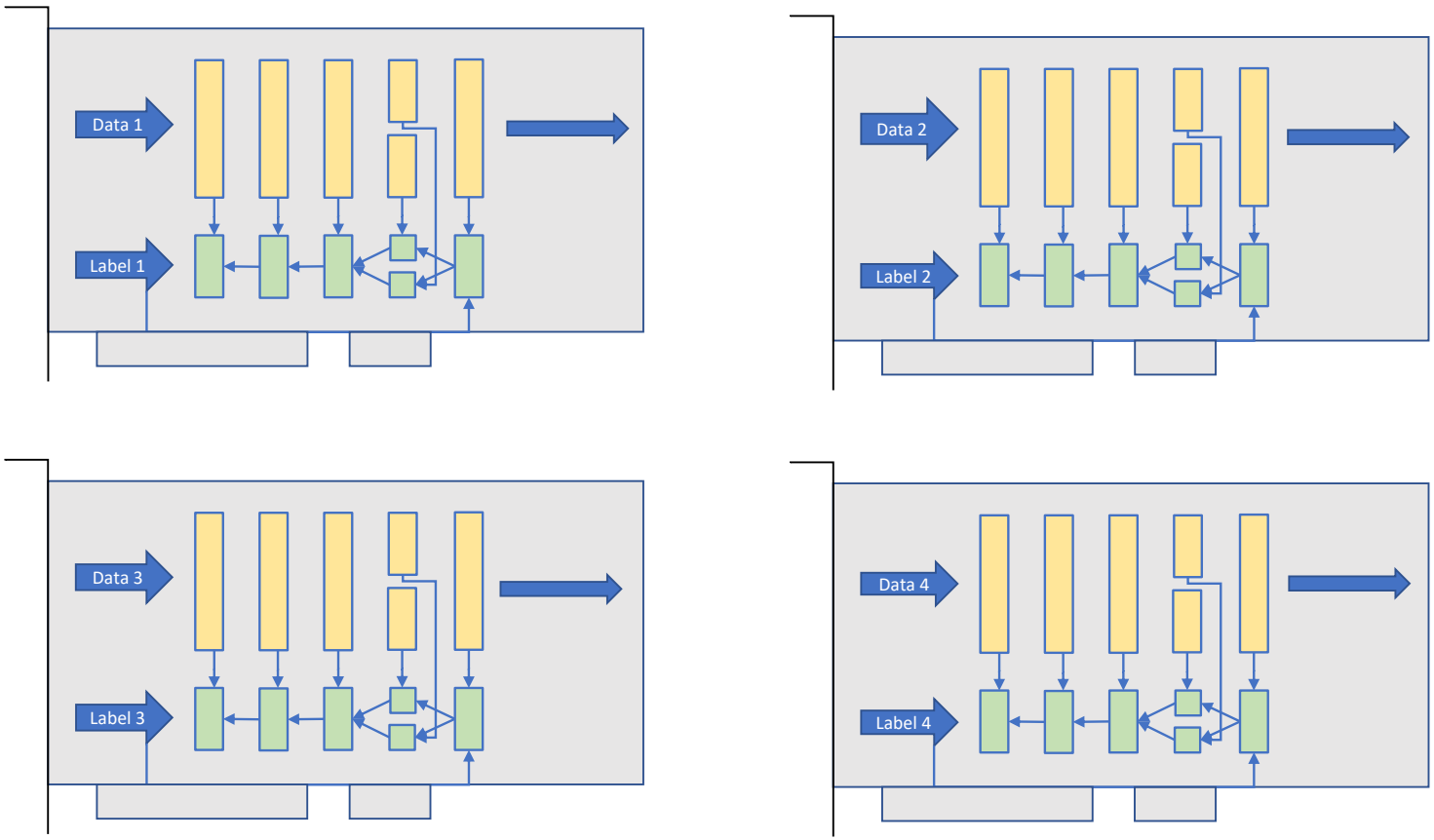
- Quick review of distributed optimization; parallel SGD; communication cost
- Decentralized SGD: efficient communication through partial-averaging
- Exponential topology enables both fast and high-performance training
- Validate our claims over various large-scale deep training tasks [1]

[1] All implementations are based on the open-source library BlueFog, which is available at GitHub: `https://github.com/Bluefog-Lib/bluefog`

DNN training



Data parallel training



Formulate DNN training as distributed optimization

- A network of n nodes (GPUs) collaborate to solve the problem:

$$\min_{x\in\mathbb{R}^d} \quad f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \quad \text{where} \quad f_i(x) = \mathbb{E}_{\xi_i\sim D_i} F(x;\xi_i).$$

- Each component $f_i : \mathbb{R}^d \to \mathbb{R}$ is local and private to node i
- Random variable ξ_i denotes the local data that follows distribution D_i
- Each local distribution D_i may be different; data heterogeneity

Parallel stochastic gradient descent (SGD)

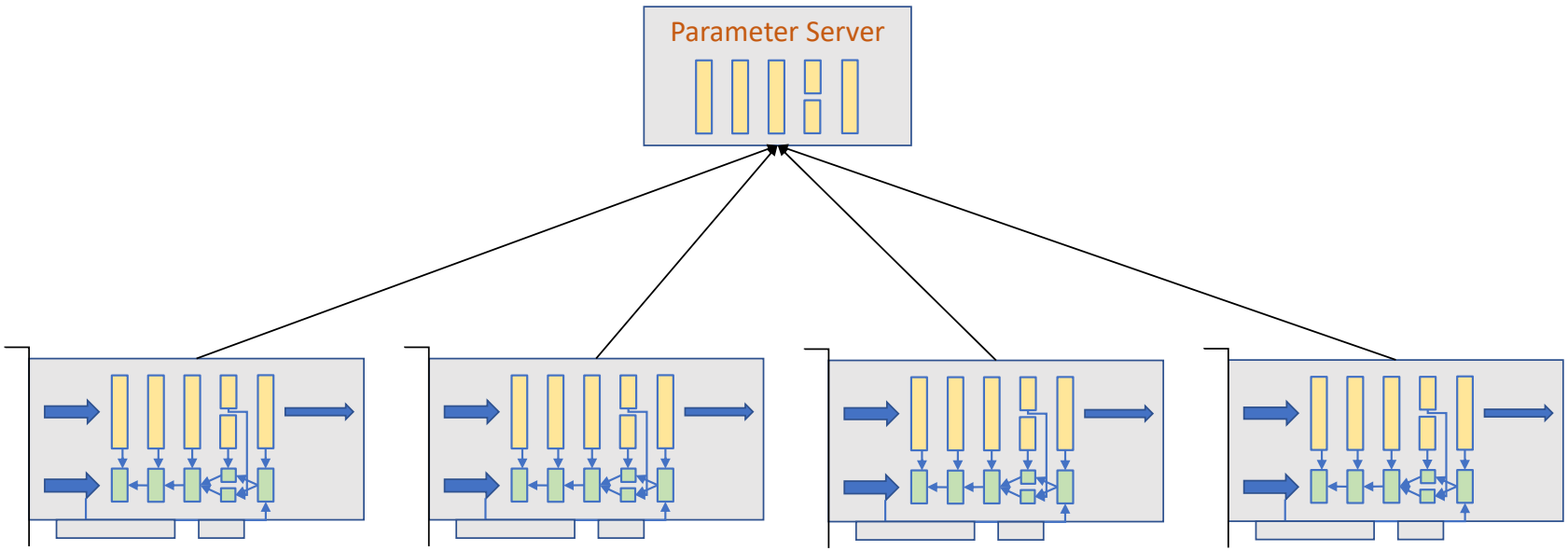
NEURAL INFORMATION PROCESSING SYSTEMS

$$g_i^{(k)} = \nabla F(x^{(k)}; \xi_i^{(k)}) \qquad \text{(Local compt.)}$$

$$x^{(k+1)} = x^{(k)} - \frac{\gamma}{n} \sum_{i=1}^{n} g_i^{(k)} \qquad \text{(Global comm.)}$$

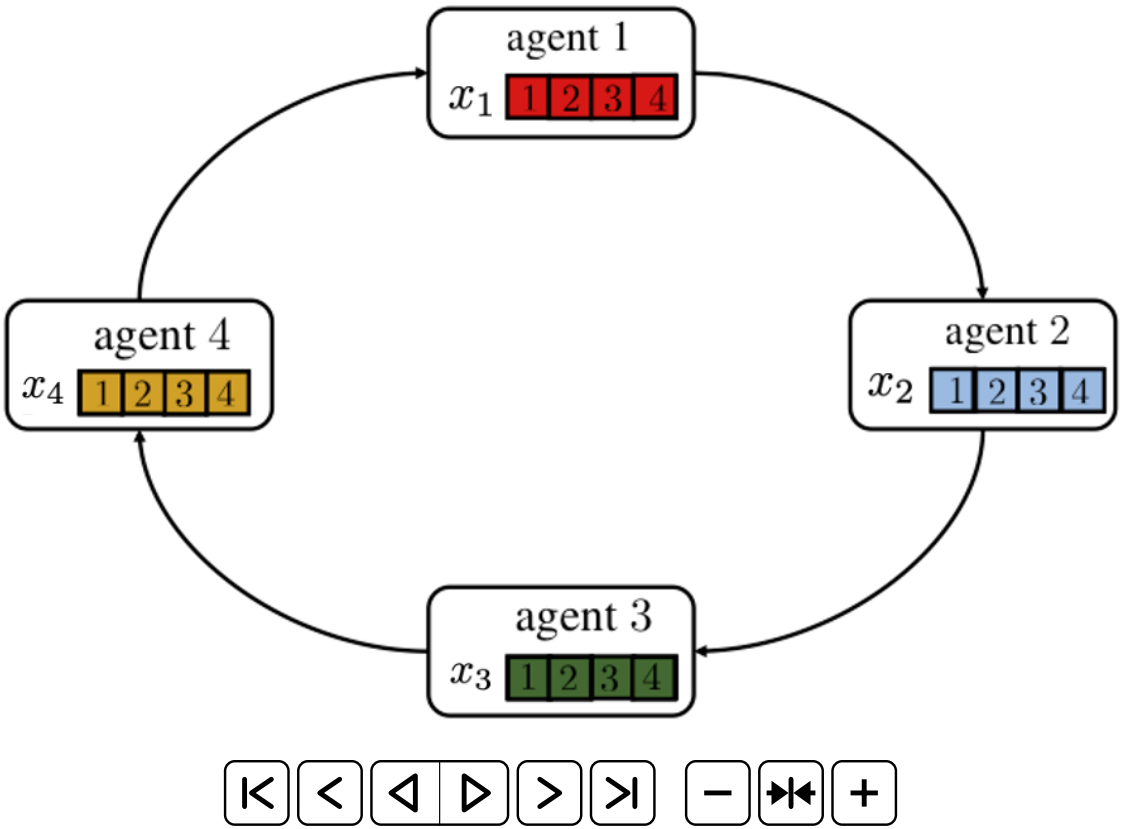
- Each node i samples data $\xi_i^{(k)}$ and computes gradient $\nabla F(x^{(k)}; \xi_i^{(k)})$
- All nodes synchronize (i.e. global averaged) to update model x
- Global average incurs significant comm. cost; hinders training scalability

Global average via Parameter-Server[2]



[2] [Li et.al., 2014]

Global average via Ring-Allreduce[3]

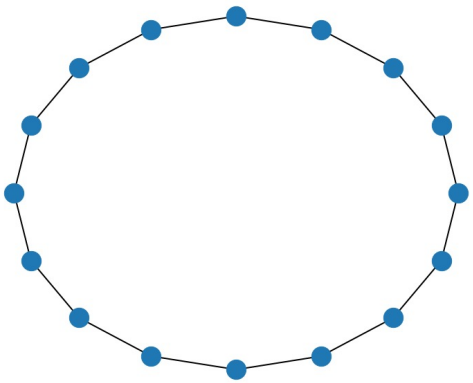


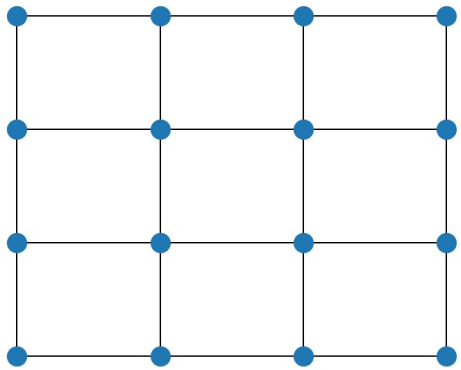
[3] [Patarasuk and Yuan, 2009]

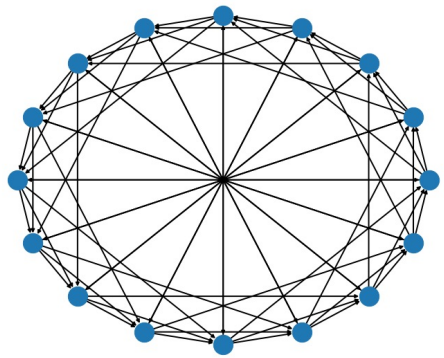
Decentralized SGD: topology

NEURAL INFORMATION PROCESSING SYSTEMS

- Assume we connect all nodes with some topology (n=16)







- Communication is only allowed between neighbors
- No global information is computed.

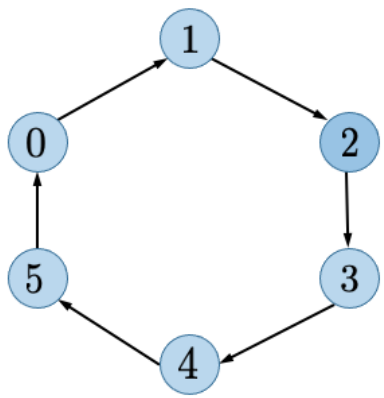
Decentralized SGD: weight matrix

- The weight matrix associated with the topology is defined as

$$w_{ij} \begin{cases} > 0 & \text{if node } j \text{ is connected to } i, \text{ or } i = j; \\ = 0 & \text{otherwise.} \end{cases}$$

- We assume W is doubly stochastic: $W\mathbb{1} = \mathbb{1}$ and $\mathbb{1}^T W = \mathbb{1}^T$.

- An example:



$$W = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

Figure: A directed ring topology and its associated combination matrix W.

Decentralized SGD: recursion[4]

$$x_i^{(k+\frac{1}{2})} = x_i^{(k)} - \gamma \nabla F(x_i^{(k)}; \xi_i^{(k)}) \quad \text{(Local update)}$$

$$x_i^{(k+1)} = \sum_{j \in \mathcal{N}_i} w_{ij} x_j^{(k+\frac{1}{2})} \quad \text{(Partial averaging)}$$

- Decentralized SGD (D-SGD) = local SGD update+ paritial averaging
- Per-iteration communication: $\Omega(d_{\max}) \ll \Omega(n)$ when topology is sparse, where d is the degree of a node.
- Incurs $\Omega(1)$ comm. overhead on sparse topology (ring or grid)

[4] [Lopes and Sayed, 2008; Nedic and Ozdaglar, 2009; Chen and Sayed, 2012]

However, D-SGD has slower convergence

NEURAL INFORMATION PROCESSING SYSTEMS

- The efficient communication comes with a cost: slow convergence
- Partial averaging is less effective to aggregate information
- The average effectiveness can be evaluated by:

$$\rho = \max\{|\lambda_2(W)|, |\lambda_N(W)|\},$$

where λ_i is the i-th largest eigenvalue and $1-\rho$ is also commonly referred as spectral gap.

  - Assume W is doubly-stochastic, it holds that $\rho \in (0, 1)$.
  - Well-connected topology has $\rho \to 0$, e.g. fully-connected topology
  - Sparsely-connected topology has $\rho \to 1$, e.g., ring has $\rho = O(1 - \frac{1}{n^2})$

Convergence rate of D-SGD

NEURAL INFORMATION PROCESSING SYSTEMS

- Convergence comparison (non-convex; i.i.d data distribution)[5]:

$$\text{P-SGD}: \quad \frac{1}{T}\sum_{k=1}^{T}\mathbb{E}\|\nabla f(\bar{x}^{(k)})\|^2 = O\Big(\frac{\sigma^2}{\sqrt{nT}}\Big)$$

$$\text{D-SGD}: \quad \frac{1}{T}\sum_{k=1}^{T}\mathbb{E}\|\nabla f(\bar{x}^{(k)})\|^2 = O\Big(\frac{\sigma^2}{\sqrt{nT}} + \underbrace{\frac{n\sigma^2}{T(1-\rho)}}_{\text{extra overhead}}\Big)$$

  where σ^2 is the gradient noise, and T is the number of iterations.

- D-SGD can asymptotically converge as fast as P-SGD when $T \to \infty$; the first term dominates

- But it requires more iteration (i.e., T has to be large enough) to reach that stage due to the extra overhead in rate caused by partial averaging

[5] [Lian et.al. 2017; Assran, Ballas, Rabbat 2019; Koloskova et.al. 2020]

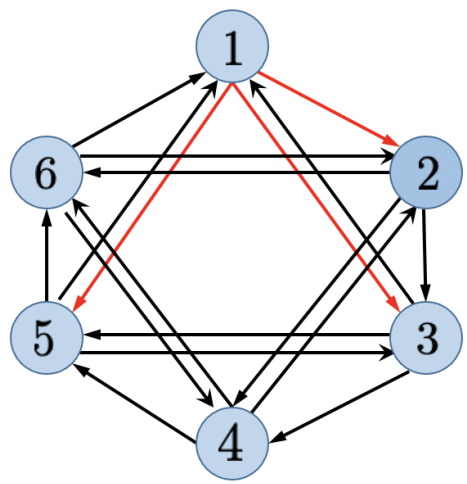
Trade-off between communication and convergence

- Recall per-iter comm. $\Omega(d_{\max})$ and rate's extra overhead $\Omega((1-\rho)^{-1})$
- Dense topology: expensive comm. but faster convergence
- Sparse topology: cheap comm. but slower convergence

Static exponential graph

NEURAL INFORMATION PROCESSING SYSTEMS

- Static exponential graph[6] is widely-used in deep training
- Empirically successful but less theoretically understood
- Each node links to neighbors that are $2^0, 2^1, \cdots, 2^{\lfloor \log_2(n-1) \rfloor}$ hops away
- In the figure, node 1 connects to $2, 3$ and 5.



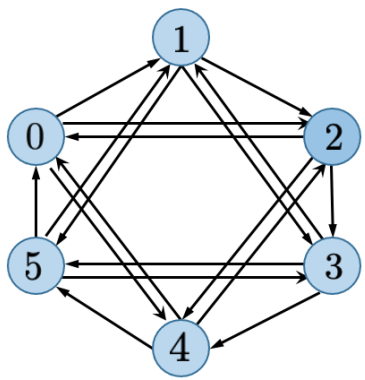
[6] [Lian et.al. 2017; Lian et.al. 2018; Assran, Ballas, and Rabbat 2019]

Static exponential graph

- The weight matrix W associated with static exp. graph is defined as

$$w_{ij}^{\exp} = \begin{cases} \frac{1}{\lceil \log_2(n) \rceil + 1} & \text{if } \log_2(\mathrm{mod}(j-i, n)) \text{ is an integer or } i = j \\ 0 & \text{otherwise.} \end{cases}$$

- An illustrating example



$$W = \begin{bmatrix} \frac{1}{4} & 0 & \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} \\ \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

Figure: A 6-node static exponential graph and its associated weight matrix.

Static exponential graph: spectral gap

- Each node has $\lceil \log_2(n) \rceil$ neighbors; per-iter comm. cost is $\Omega(\log_2(n))$
- The following theorem clarifies that $\rho(W^{\exp}) = O(1 - 1/\log_2(n))$; a non-trivial proofs; requires smart utilization of Discrete Fourier transform.

Theorem

Let $\tau = \lceil \log_2(n) \rceil$, and ρ as the second largest eigenvalue in magnitude of W, ($1 - \rho$ is also known as the spectral gap). It holds that

$$\rho(W^{\exp}) \begin{cases} = 1 - \dfrac{2}{\tau + 1}, & \text{when } n \text{ is even} \\ < 1 - \dfrac{2}{\tau + 1}, & \text{when } n \text{ is odd} \end{cases}$$

Further, it also holds that $\|W - \frac{1}{n}\mathbb{1}\mathbb{1}^T\|_2 = \rho(W^{\exp})$.

Spectral gap: numerical illustration

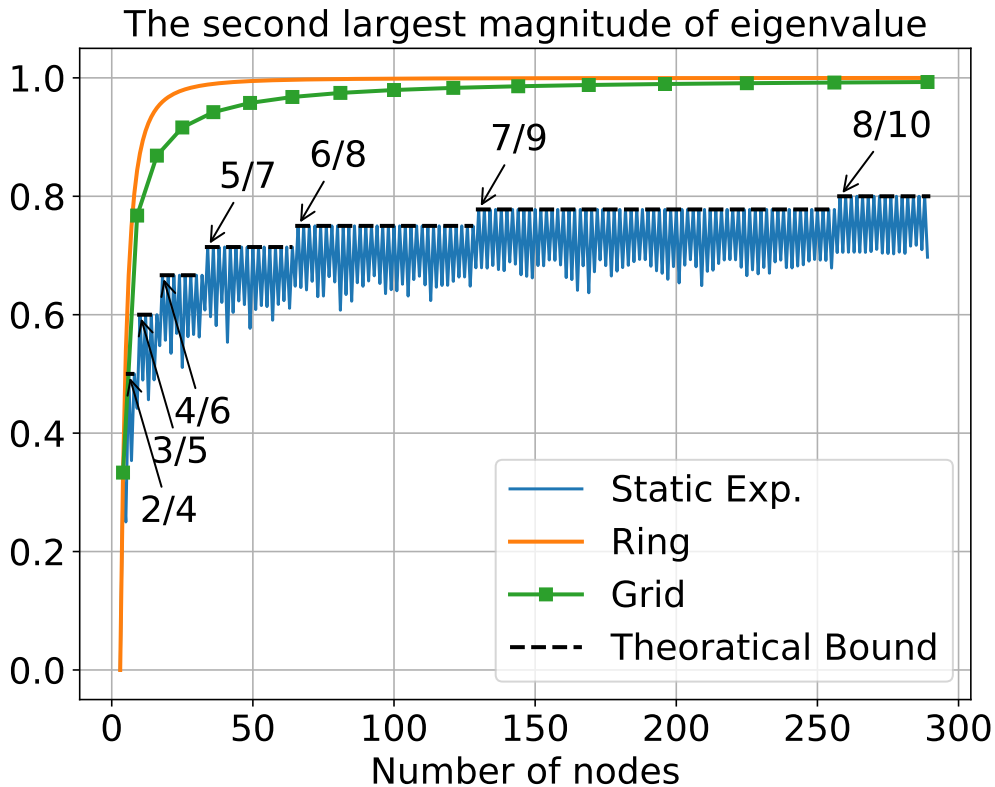
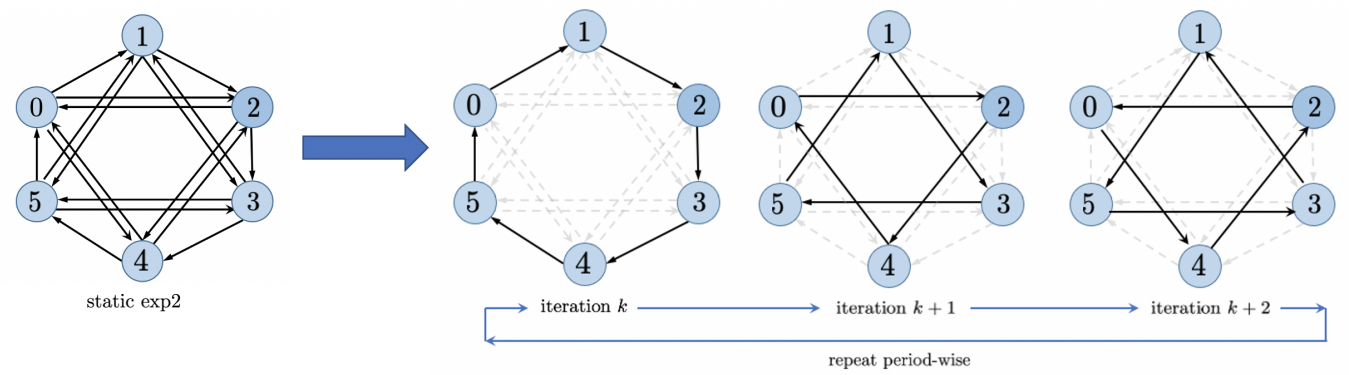


Figure: Illustration of the spectral gaps for ring, grid and static exp. graphs.

One-peer exponential graph

NEURAL INFORMATION PROCESSING SYSTEMS

- Static exponential graph has $\Omega(\log_2(n))$ per-iteration comm.
- Such overhead is still more expensive than ring or grid
- Split exponential graph into a sequence of one-peer realizations[7]



- Each realization has $\Omega(1)$ per-iteration communication

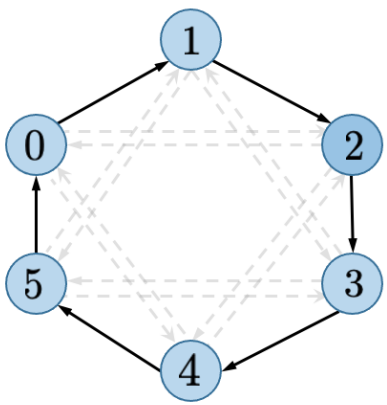
[7] [Assran, Ballas, and Rabbat 2019]

One-peer exponential graph: weight matrix

- We let $\tau = \lceil \log_2(n) \rceil$. The weight matrix $W^{(k)}$ is time-varying

$$w_{ij}^{(k)} = \begin{cases} \frac{1}{2} & \text{if } \log_2(\text{mod}(j-i, n)) = \text{mod}(k, \tau) \\ \frac{1}{2} & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

- An illustrating example



$$W = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

Decentralized SGD over one-peer exponential graph

- The D-SGD recursion over one-peer exponential graph:

Sample $W^{(k)}$ over one-peer exponential graph

$$x_i^{(k+\frac{1}{2})} = x_i^{(k)} - \gamma \nabla F(x_i^{(k)}; \xi_i^{(k)}) \qquad \text{(Local update)}$$

$$x_i^{(k+1)} = \sum_{j \in \mathcal{N}_i} w_{ij}^{(k)} x_j^{(k+\frac{1}{2})} \qquad \text{(Partial averaging)}$$

- One-loop algorithm; each node has one neighbor; per-iter comm. is $\Omega(1)$
- Since each realization is sparser than static exp., will it enable DSGD with larger extra overhead in rate?

One-peer exp. graphs: periodic exact average

NEURAL INFORMATION PROCESSING SYSTEMS

Theorem (Periodic Global-averaging)

Suppose $\tau = \log_2(n)$ *is a positive integer. It holds that*

$$W^{(k+\ell)} W^{(k+\ell-1)} \cdots W^{(k+1)} W^{(k)} = \frac{1}{n} \mathbb{1}\mathbb{1}^T$$

for any integer $k \geq 0$ *and* $\ell \geq \tau - 1$.

While each realization of one-peer graph is sparser, a sequence of one-peer graphs will enable effective global averaging.

Applying one-peer exp. graphs to DSGD

NEURAL INFORMATION PROCESSING SYSTEMS

Assumption

(1) Each $f_i(x)$ is L-smooth; (2) Each gradient noise is unbiased and has bounded variance σ^2; (3) Each local distribution D_i is identical

Theorem (DSGD CONVERGENCE WITH ONE-PEER EXP.)

Under the above assumptions and with $\gamma = O(1/\sqrt{T})$, let $\tau = \log_2(n)$ be an integer, DSGD with one-peer exponential graph will converge at

$$\frac{1}{T}\sum_{k=1}^{T}\mathbb{E}\|\nabla f(\bar{x}^{(k)})\|^2 = O\Big(\frac{\sigma^2}{\sqrt{nT}} + \underbrace{\frac{n\log_2(n)\sigma^2}{T}}_{\text{extra overhead}}\Big)$$

Convergence rate for decentralized momentum SGD (DmSGD) with heterogeneous data distributions is also established in the paper.

Static exp. v.s. one-peer exp.

- Convergence rate for DSGD over static and one-peer exp. graphs

$$\text{Static exp.} \quad O\left(\frac{\sigma^2}{\sqrt{nT}} + \frac{n\sigma^2}{T(1-\rho)}\right) \quad (\text{where } 1-\rho = O(1/\log_2(n)))$$

$$\text{One-peer exp.} \quad O\left(\frac{\sigma^2}{\sqrt{nT}} + \frac{n\log_2(n)\sigma^2}{T}\right)$$

- DSGD with one-peer exp. converges as fast as static exp. in terms of the established bounds; a surprising result.
- DSGD with both graphs are with the same rate's overhead $O(\log_2(n))$
- The same results hold for heterogeneous data scenario, and for DmSGD.

One-peer graph is the state-of-the-art topology

NEURAL INFORMATION PROCESSING SYSTEMS

Topology	Per-iter. Comm.	Extra overhead in rate (iid)
Ring	$\Omega(2)$	$\Omega(n^2)$
Star	$\Omega(n)$	$\Omega(n^2)$
2D-Grid	$\Omega(4)$	$\Omega(n)$
2D-Torus	$\Omega(4)$	$\Omega(n)$
$\frac{1}{2}$-RandGraph	$\Omega(\frac{n}{2})$	$\Omega(1)$
Static Exp.	$\Omega(\log_2(n))$	$\Omega(\log_2(n))$
One-peer Exp.	$\Omega(1)$	$\Omega(\log_2(n))$

- Both static and one-peer exp are nearly best (up to $\log_2(n)$) in terms of Per-iter comm. and extra overhead in rate.
- Since one-peer exp. incurs less per-iter comm., it is recommended for DL.

Experiments

NEURAL INFORMATION PROCESSING SYSTEMS

We focuse on two main metrics:

- Wall-clock time to finish K epochs of training; measures per-iter comm.
- Validation accuracy after K epochs of training; measures convgt. rate

We run the experiment through BlueFog – a library dedicated for running large-scale decentralized algorithms

Available at `https://github.com/Bluefog-Lib/bluefog`

Convergence curves: one-peer exp. v.s. static exp.

Image classification: ResNet-50 for ImageNet; $8 \times 8 = 64$ GPUs.

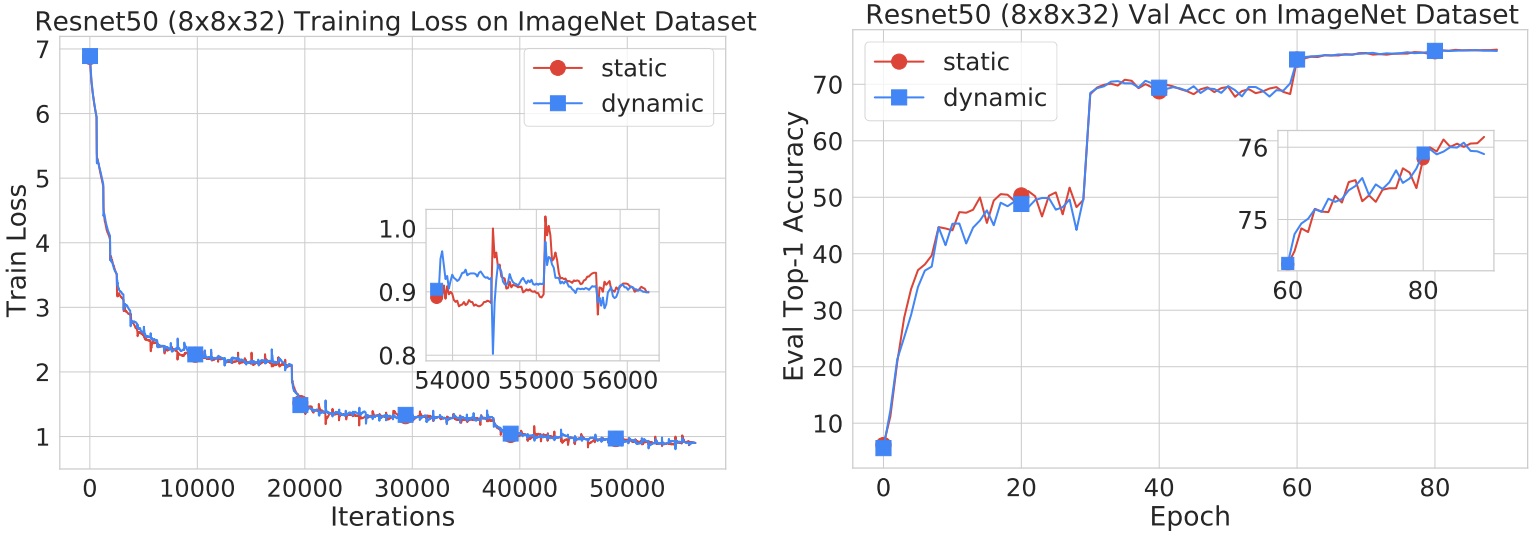


Figure: DmSGD over one-peer exp. converges as fast as over static exp.

Comparing different models/methods

NEURAL INFORMATION PROCESSING SYSTEMS

model	ResNet-50			MobileNet-v2			EfficientNet		
Topology	static	one-peer	diff	static	one-peer	diff	static	one-peer	diff
Parallel SGD	76.21	-	-	70.12	-	-	77.63	-	-
vanilla DmSGD	76.14	76.06	-0.08	69.98	69.81	-0.17	77.62	77.48	-0.14
DmSGD	76.50	76.52	+0.02	69.62	69.98	+0.36	77.44	77.51	+0.07
QG-DmSGD	76.43	76.35	-0.08	69.83	69.81	-0.02	77.60	77.72	+0.12

- setting: ImageNet; $8 \times 8 = 64$ GPUs; diff = o.e - s.e.
- both topo. achieve similar accuracy across different models and algorithms
- accuracy difference is minor (except for MobileNet with DmSGD)
- QG-DmSGD and DmSGD outperform PSGD in ResNet-50

Scaling effects for DmSGD over various topologies

Image classification: ResNet-50 for ImageNet;

Table 1: Top-1 validation accuracy(%) and training time (hours) after 90 epochs.

nodes	4(4x8 GPUs)		8(8x8 GPUs)		16(16x8 GPUs)		32(32x8 GPUs)	
topology	acc.	time	acc.	time	acc.	time	acc.	time
Ring	76.16	11.6	76.14	6.5	76.16	3.3	75.62	1.8
Grid	76.10	11.6	76.39	6.7	75.92	3.4	75.80	2.0
Random Graph	76.03	11.5	76.07	7.1	76.25	6.7	76.32	4.7
static exp.	76.26	11.6	76.50	6.9	76.50	4.1	76.29	2.5
one-peer exp.	76.34	11.1	76.52	5.7	76.47	2.8	76.27	1.5

- training time (32 nodes): OE $<$ Ring $<$ Grid $<$ SE $<$ Random
- accuracy (32 nodes): Random $\approx$ SE $\approx$ OE $>$ Grid $>$ Ring
- one-peer exp. promises fast and high-quality deep training
- one-peer exp. has the best linear speedup among these topologies

Summary

- Both per-iter comm. and convergence overhead of exponential graphs are nearly the best (up to $\log_2(n)$ factors) among known topologies
- While one-peer exp. is sparser, it can converge as fast as static exp.
- One-peer exponential graph is recommended for decentralized DL

References

[Li et.al., 2014] Scaling distributed machine learning with the parameter server

[Patarasuk and Yuan, 2009] Bandwidth optimal all-reduce algorithms for clusters of workstations

[Ben-Nun and Hoefler, 2018] Demystifying Parallel and Distributed Deep Learning: An In-Depth Concurrency Analysis

[Lopes and Sayed, 2008] Diffusion least-mean squares over adaptive networks: Formulation and performance analysis

[Nedic and Ozdaglar, 2009] Distributed subgradient methods for multi-agent optimization

[Chen and Sayed, 2012] Diffusion adaptation strategies for distributed optimization and learning over networks

References

[Lian et.al. 2017] Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent

[Assran, Ballas, Rabbat 2019] Stochastic gradient push for distributed deep learning

[Koloskova et.al. 2020] A unified theory of decentralized sgd with changing topology and local updates

[Pu, Olshevsky, Paschalidis, 2019] A Sharp Estimate on the Transient Time of Distributed Stochastic Gradient Descent

[Nedic, Olshevsky, and Rabbat, 2018] Network Topology and Communication-Computation Tradeoffs in Decentralized Optimization

[Lian et.al. 2018] Asynchronous decentralized parallel stochastic gradient descent

[Wang et.al., 2019] SlowMo: Improving communication-efficient distributed sgd with slow momentum

References

[Yuan and Alghunaim, 2021] Removing Data Heterogeneity Influence Enhances Network Topology Dependence of Decentralized SGD

[Yuan et.al., 2021] DecentLaM: Decentralized Momentum SGD for Large-batch Deep Training

[Shi et.al. 2015] Extra: An exact first-order algorithm for decentralized consensus optimization

[Xu et.al., 2015] Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes

[Lorenzo and Scutari, 2016] Next: In-network nonconvex optimization

[Nedic et.al. 2017] Achieving geometric convergence for distributed optimization over time-varying graphs

[Qu and Li, 2017] Harnessing Smoothness to Accelerate Distributed Optimization

[Shi, et.al., 2016] Finite-time Convergent Gossiping