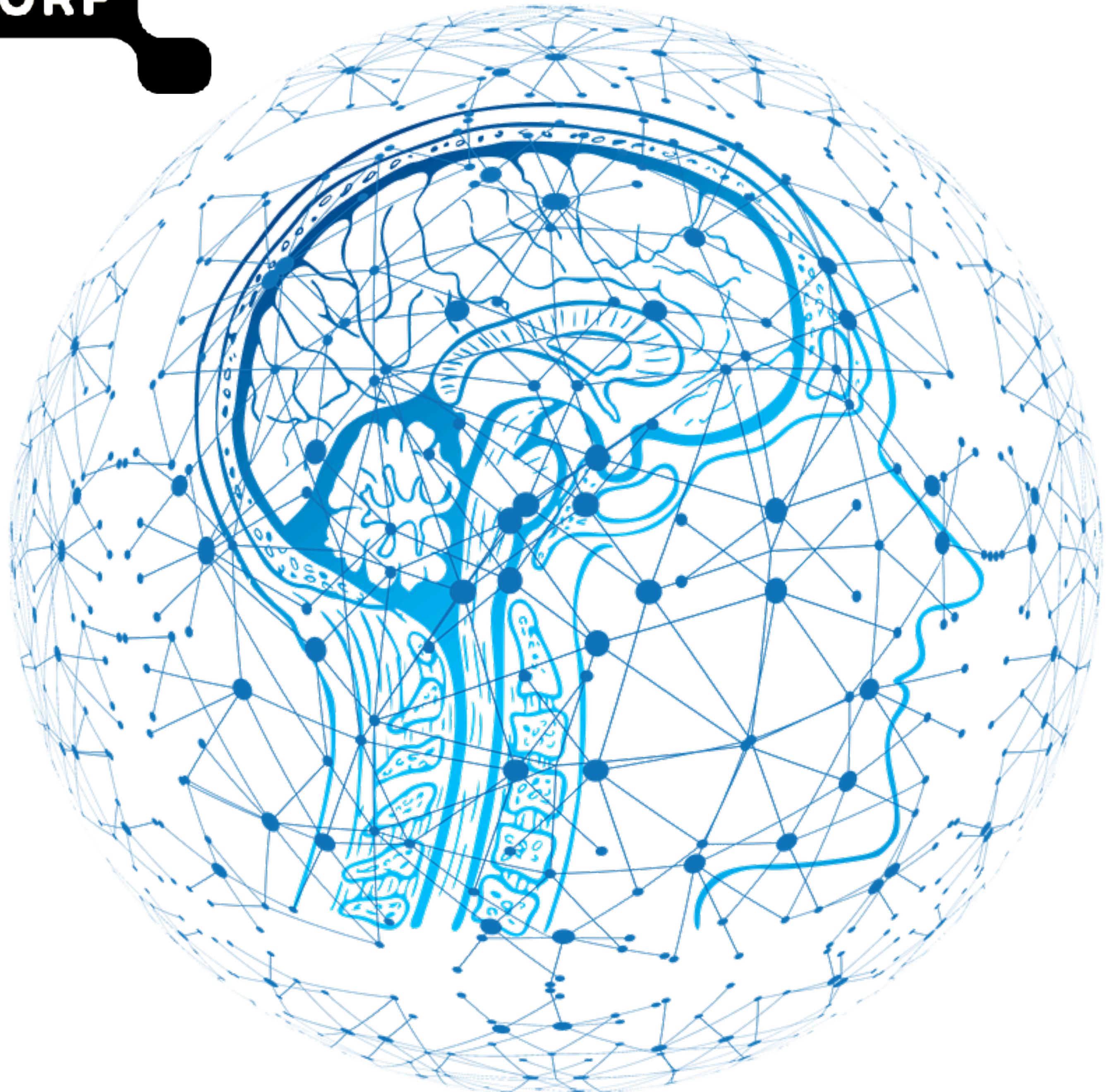


SURF



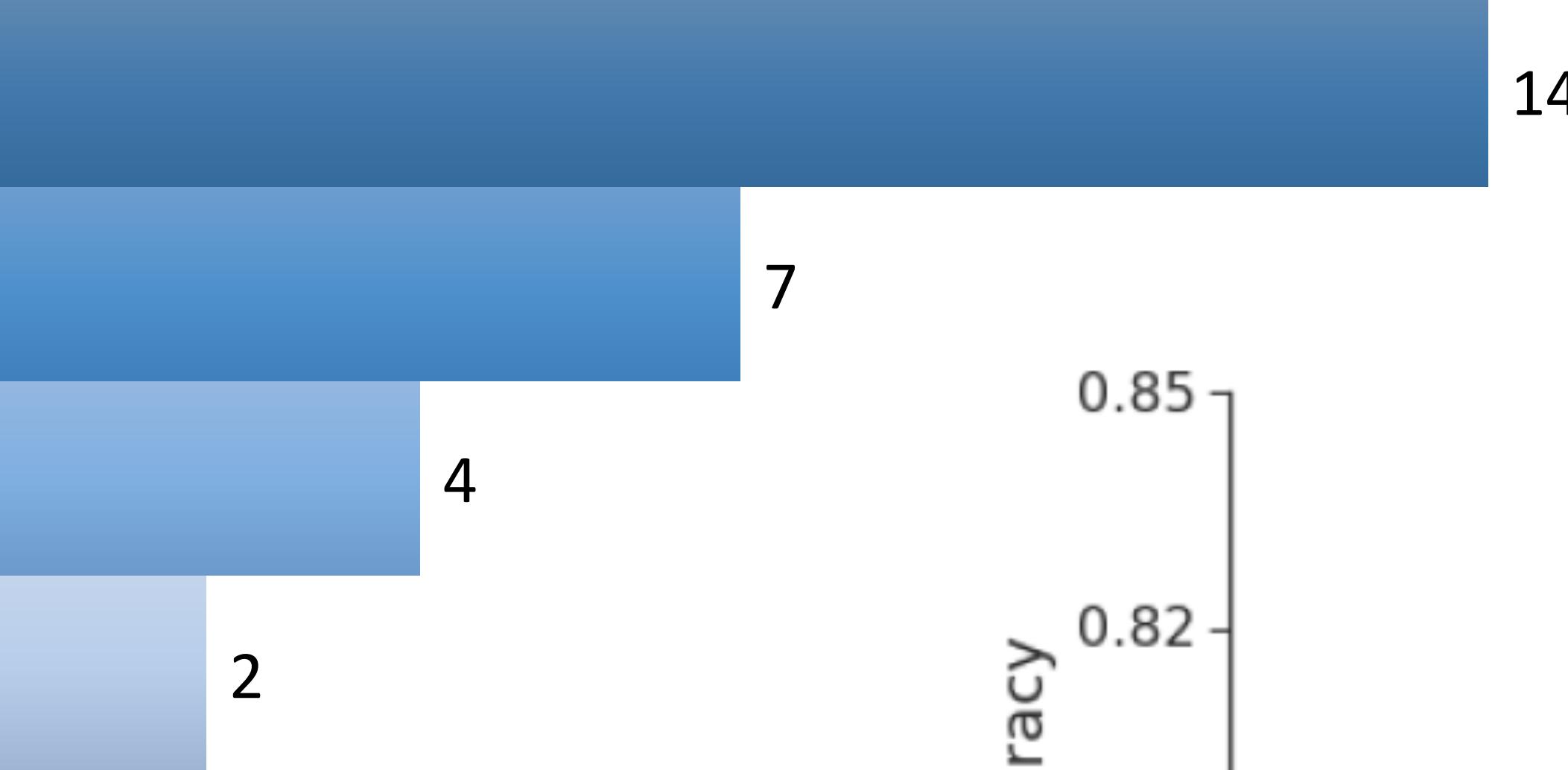
Introduction to High Performance Deep Learning

Maxwell Cai, Caspar van Leeuwen
surf.nl

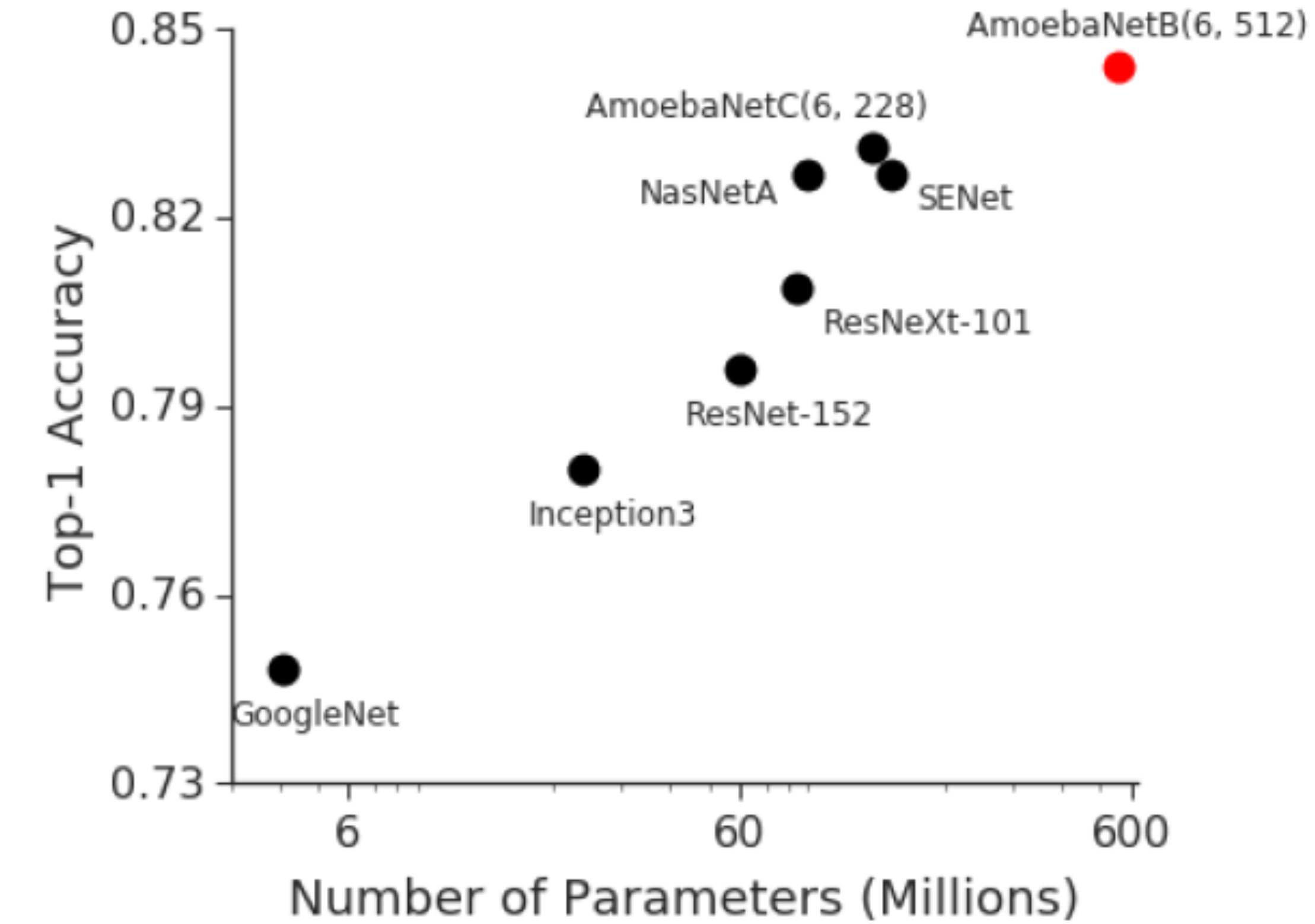
Training time (days)*

- IN5k-ResNeXt-101-64x4d
- ResNeXt-101-32x4d
- ResNet-101
- ResNet-50

Internet-scale data / Videos

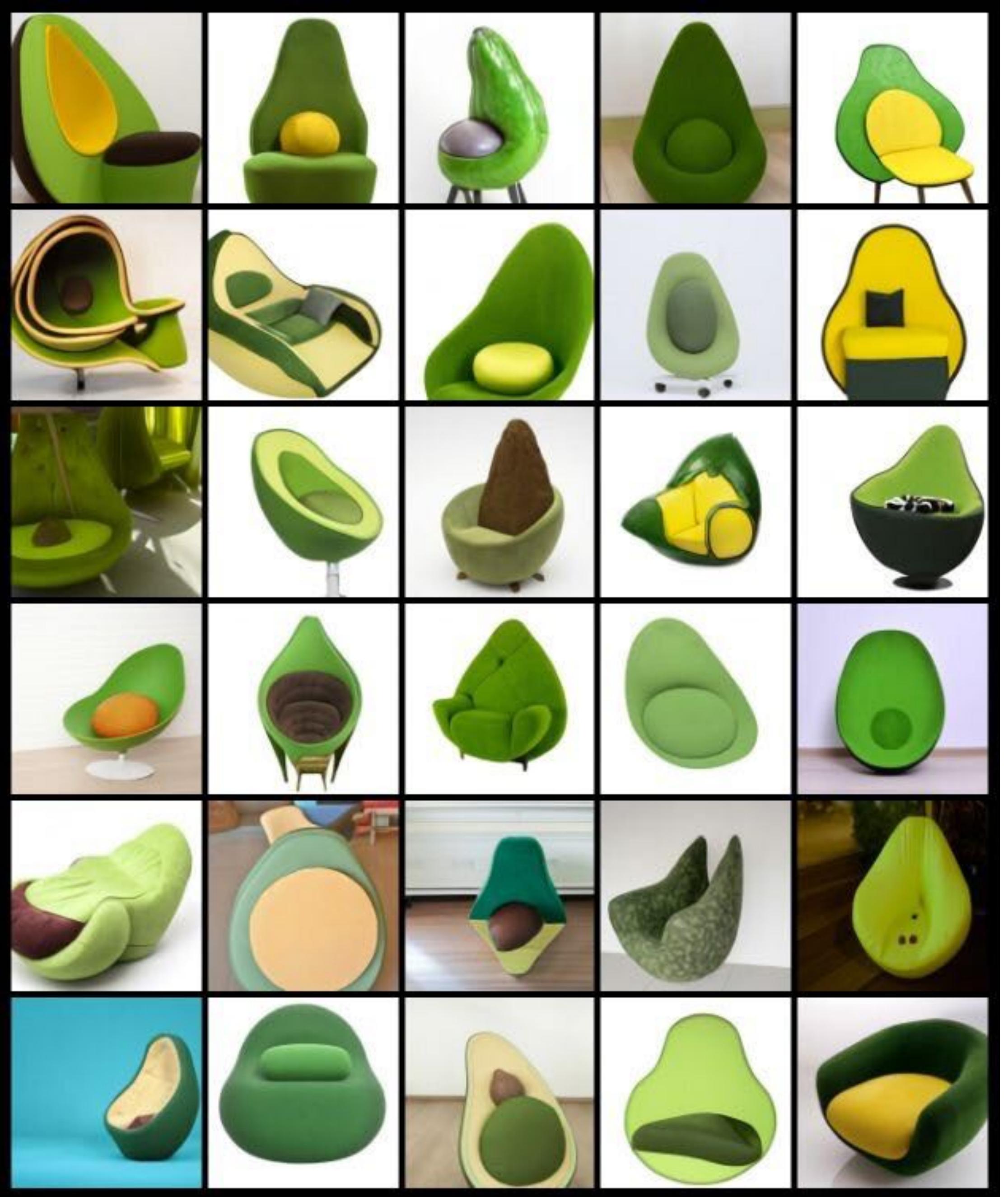


months?



*measured on NVIDIA M40 8-gpus

“An armchair in the shape of an avocado”



Total Compute Used During Training

Training Petaflop/s-days

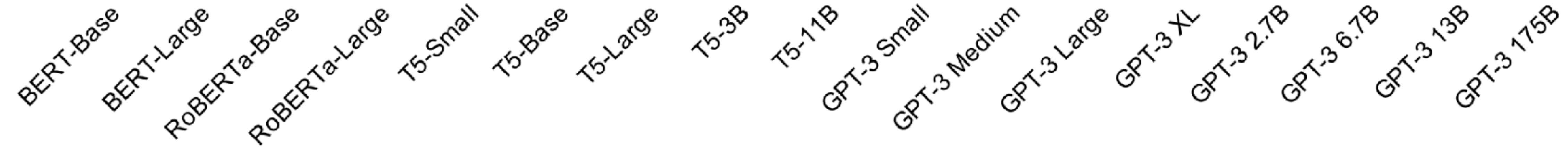
10000

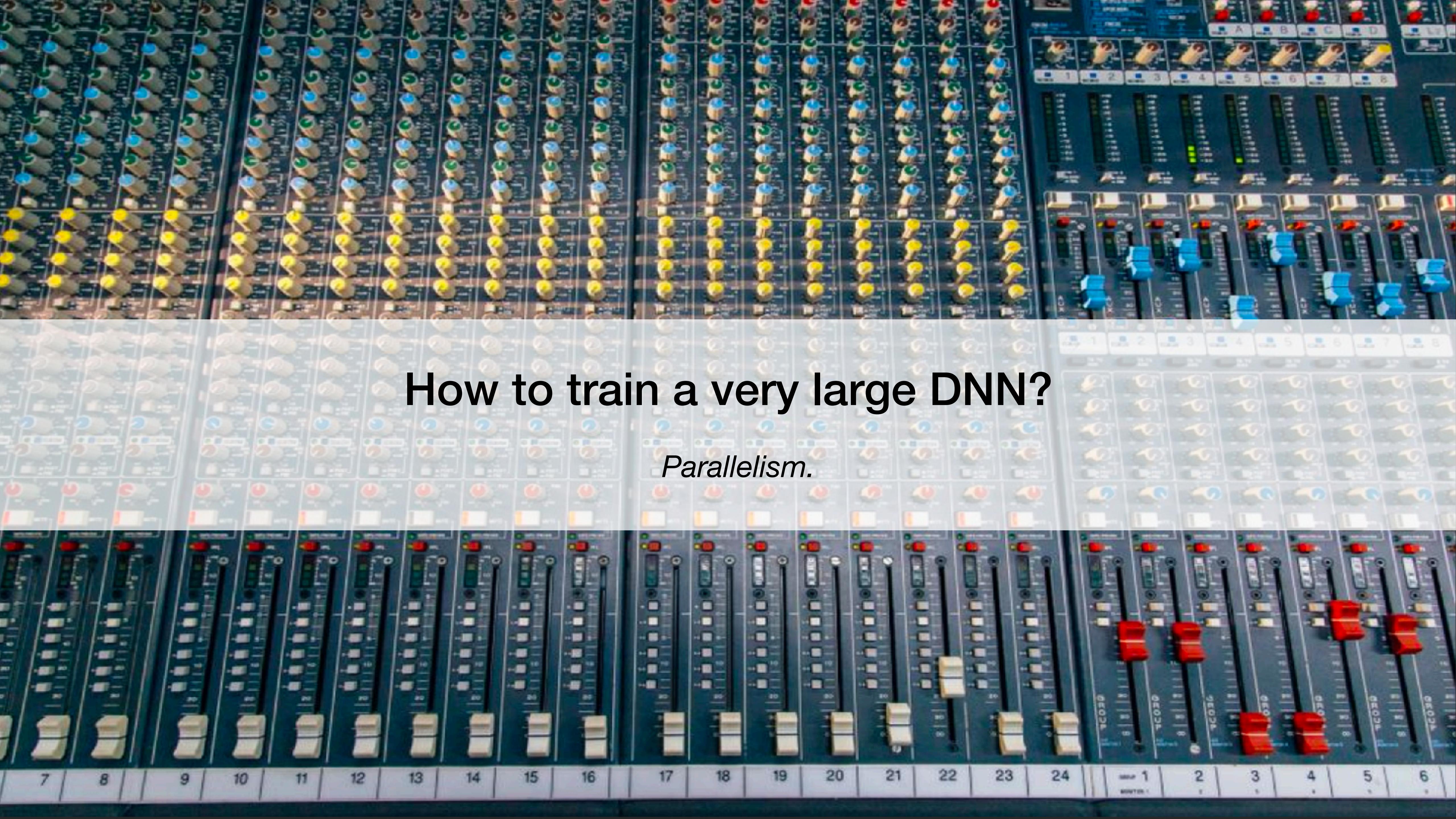
1000

100

10

1

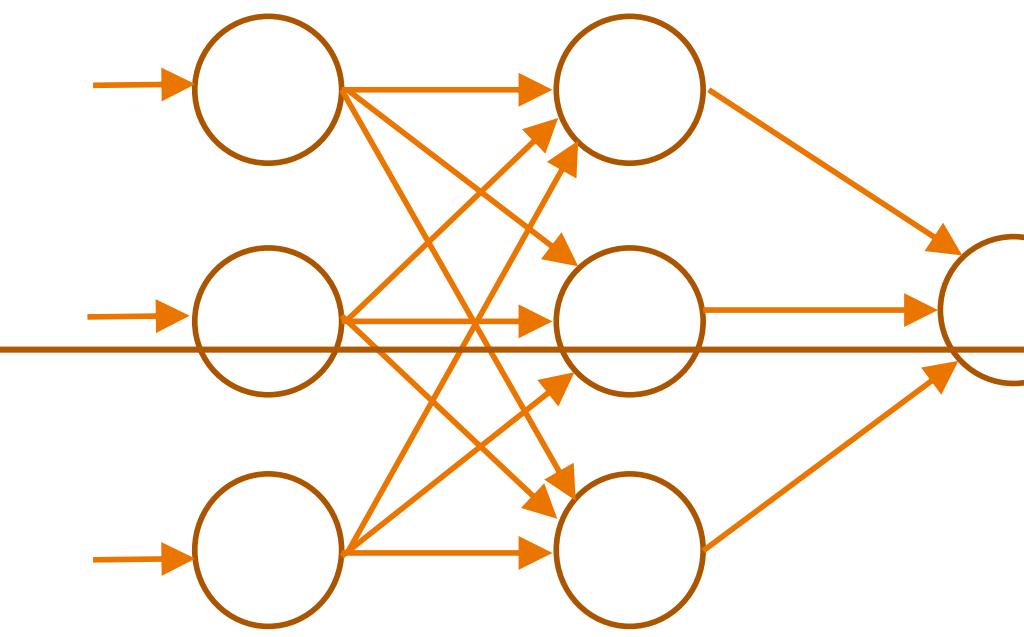
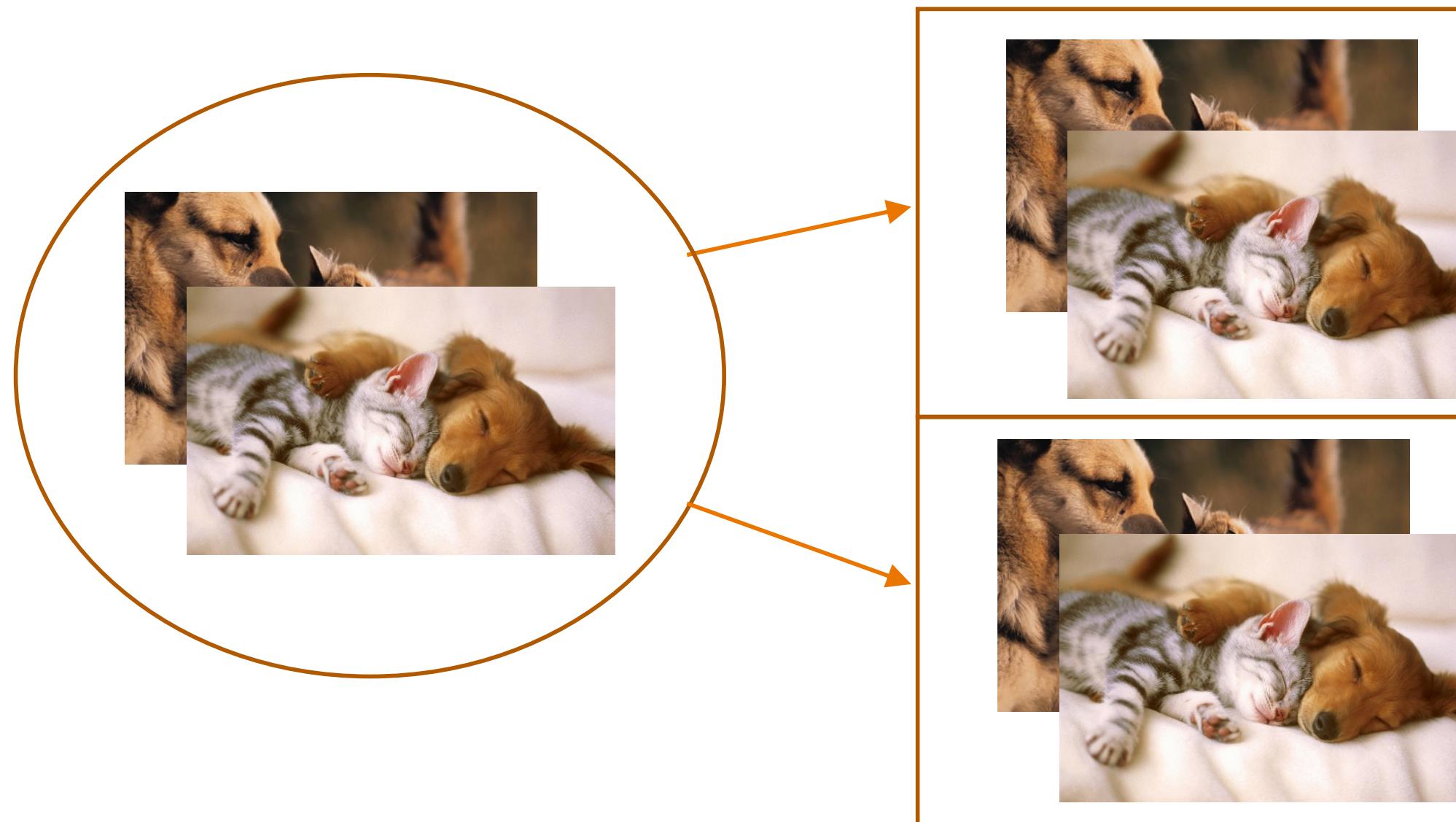




How to train a very large DNN?

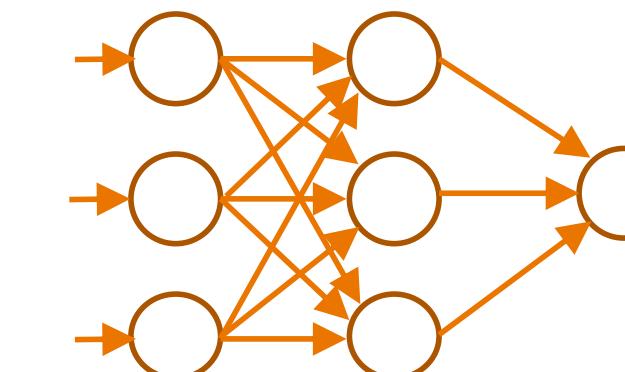
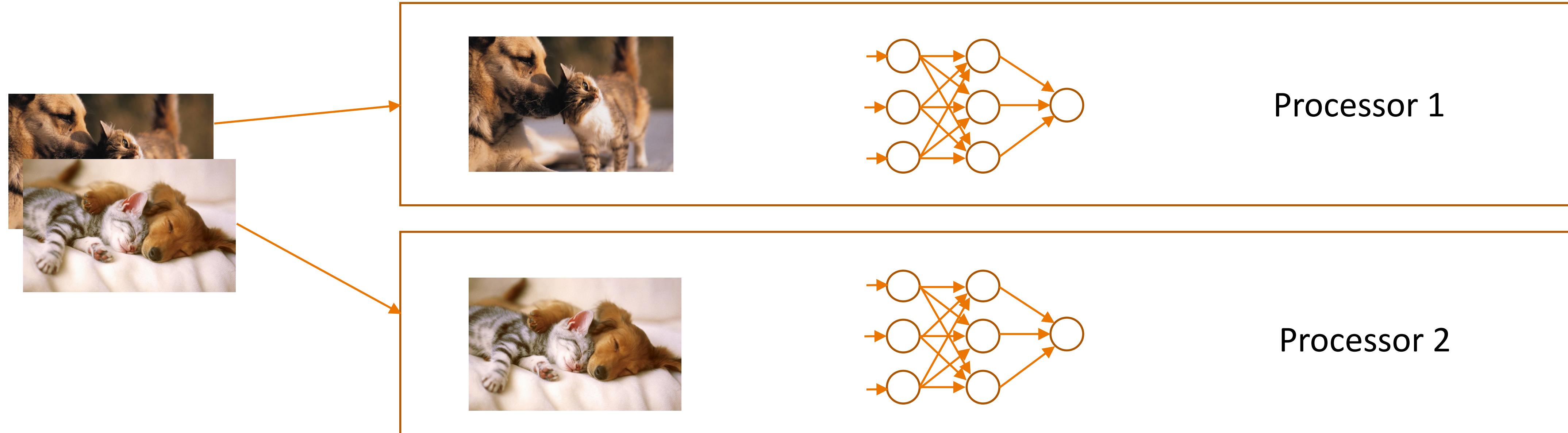
Parallelism.

Model Parallel



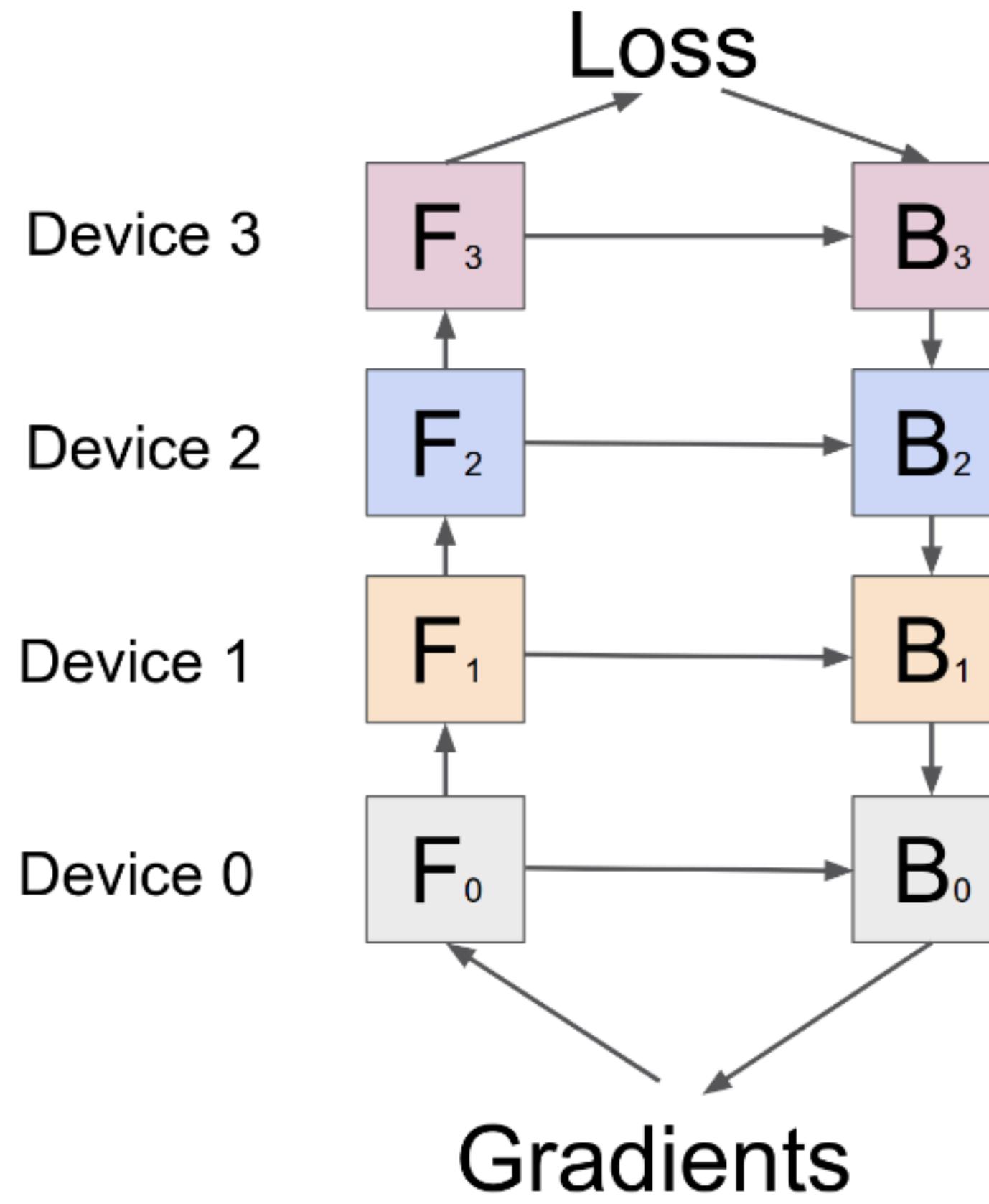
Processor 1
Processor 2

Data Parallel

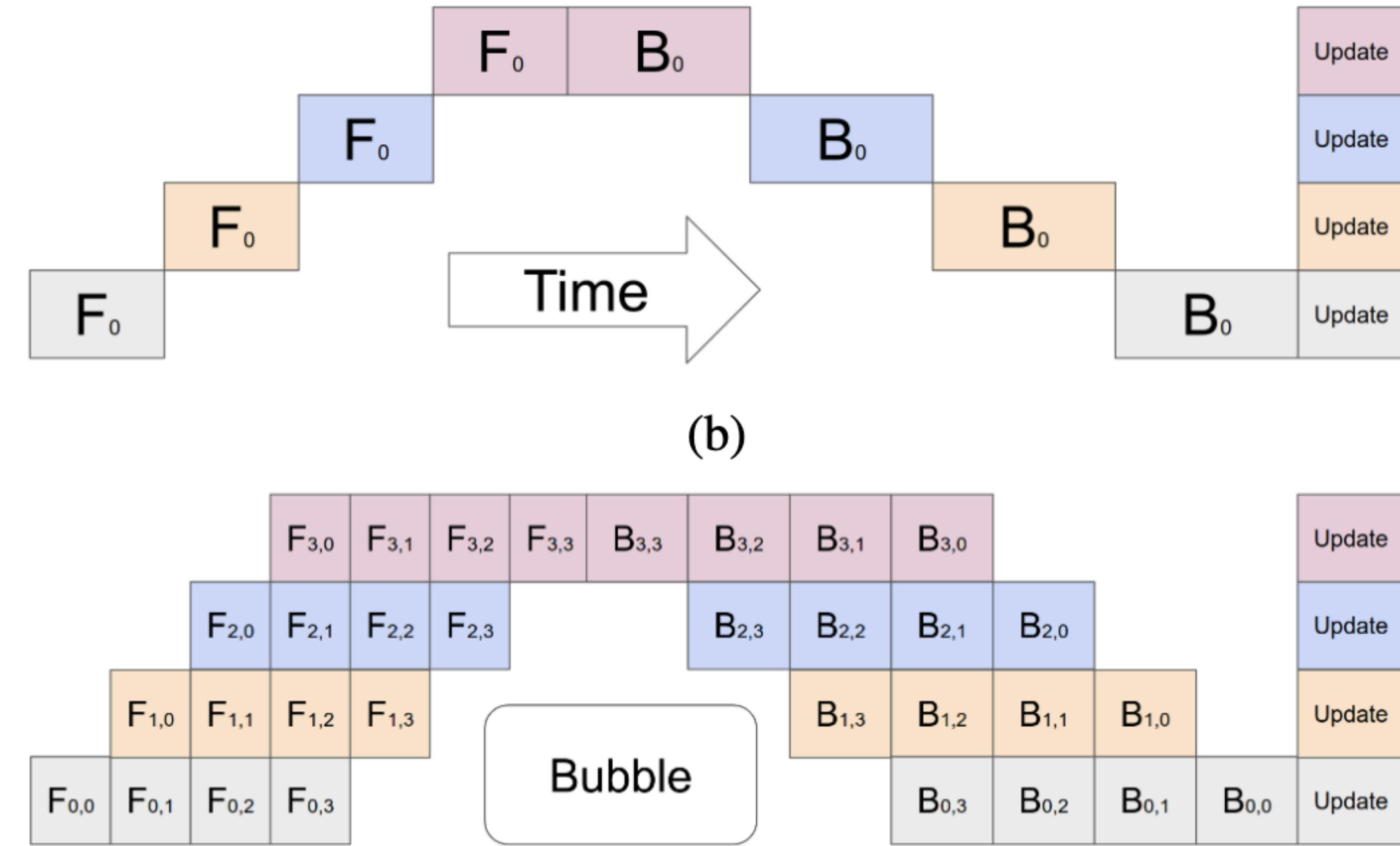


Processor 1
Processor 2

Pipeline Parallelism



(a)



(b)

(c)

Learning: Optimizing the loss function

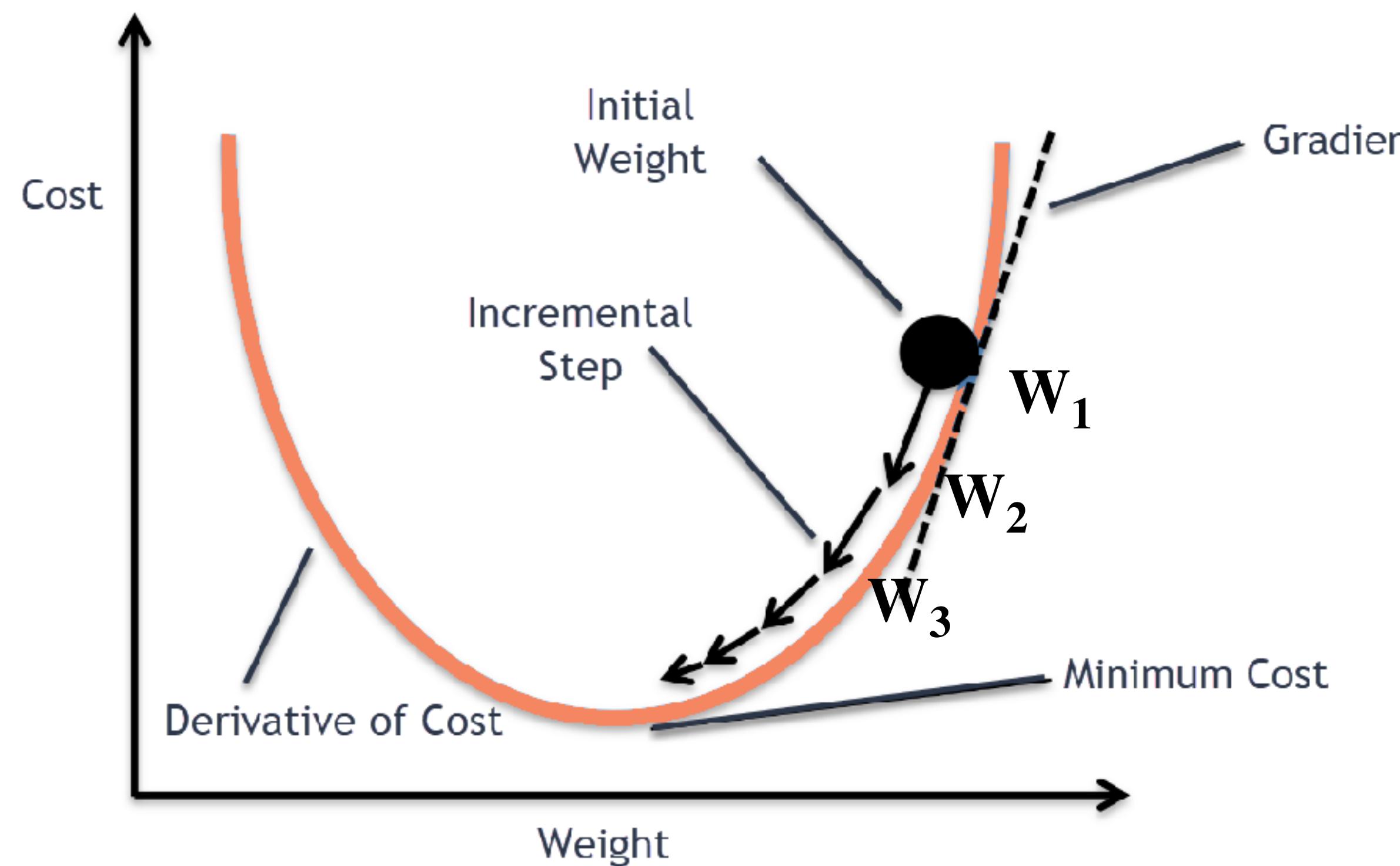
Objective: minimize the **difference** between predicted value the actual value

$$\mathcal{L}(\mathbf{W}, b) = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

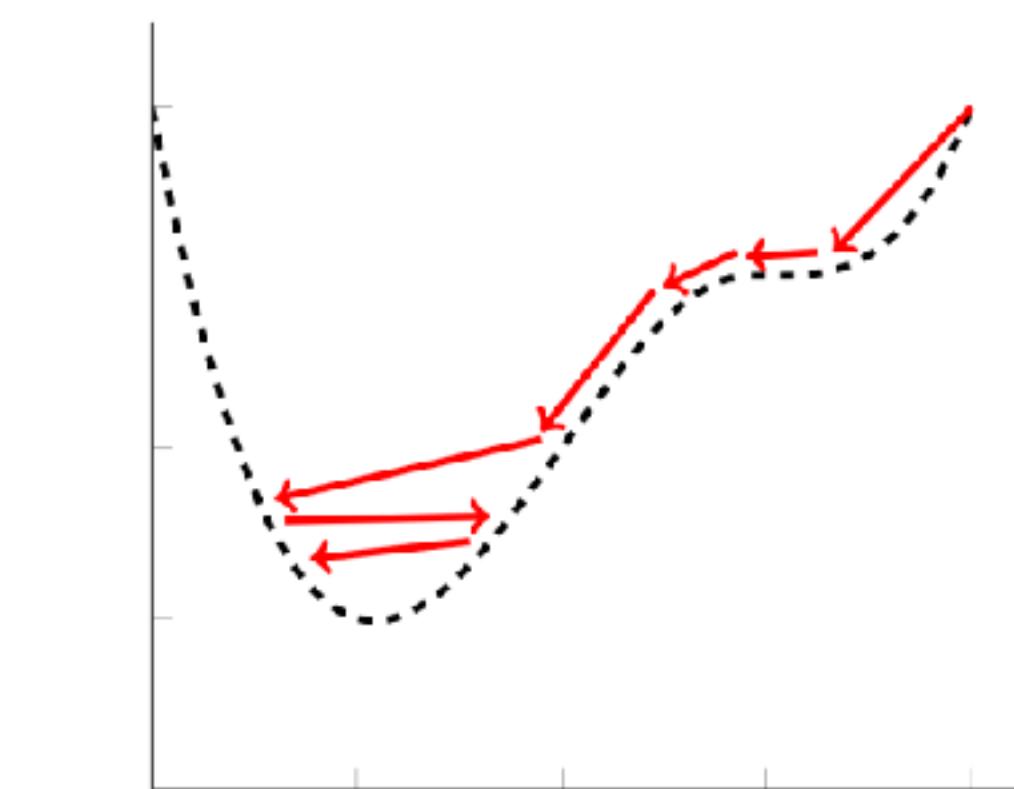
$\nabla \mathcal{L}(\mathbf{W}, b)$

$$\mathbf{W}_{j+1} = \mathbf{W}_j - \alpha \nabla \mathcal{L}(\mathbf{W}_j, b)$$

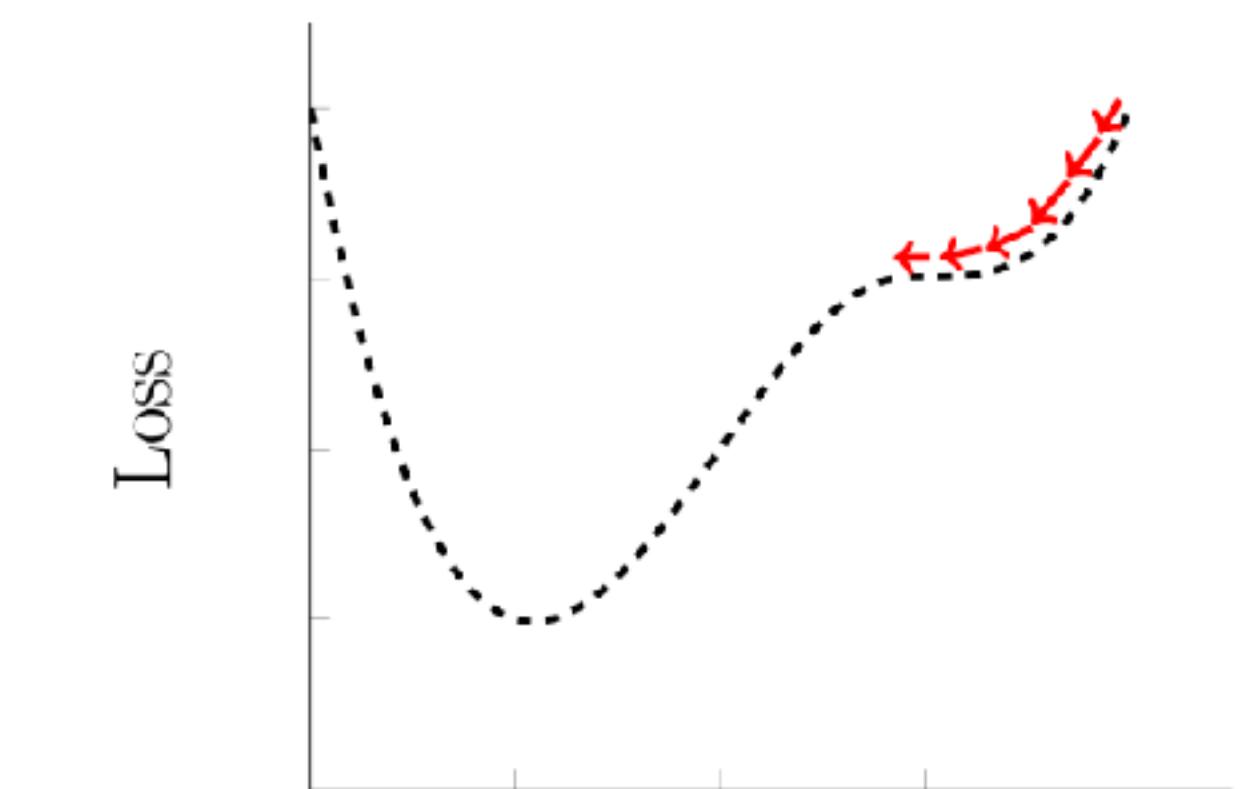
↑
Learning rate



High Learning Rate



Low Learning Rate



Parameter θ

Parameter θ

Parallel Gradient Descent

Single processor

$$\mathcal{L}(\mathbf{W}, b) = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad \mathbf{W}_{j+1} = \mathbf{W}_j - \alpha \nabla \mathcal{L}(\mathbf{W}_j, b)$$

$$g_1 = \nabla \mathcal{L}(\mathbf{W}_j, b)$$

$$g_2 = \nabla \mathcal{L}(\mathbf{W}_j, b)$$

Multiple processors

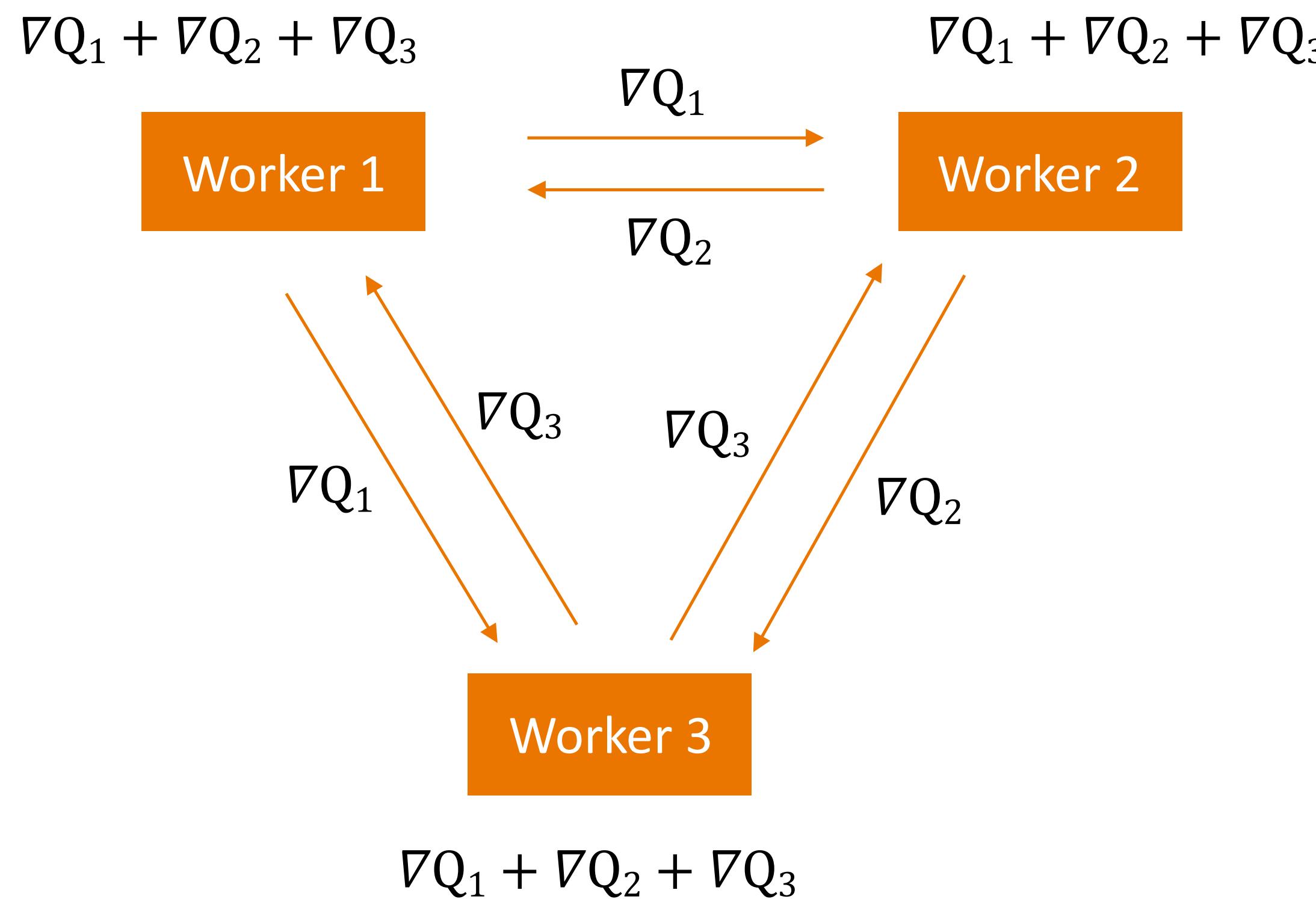
$$g_3 = \nabla \mathcal{L}(\mathbf{W}_j, b)$$

$$g_4 = \nabla \mathcal{L}(\mathbf{W}_j, b)$$

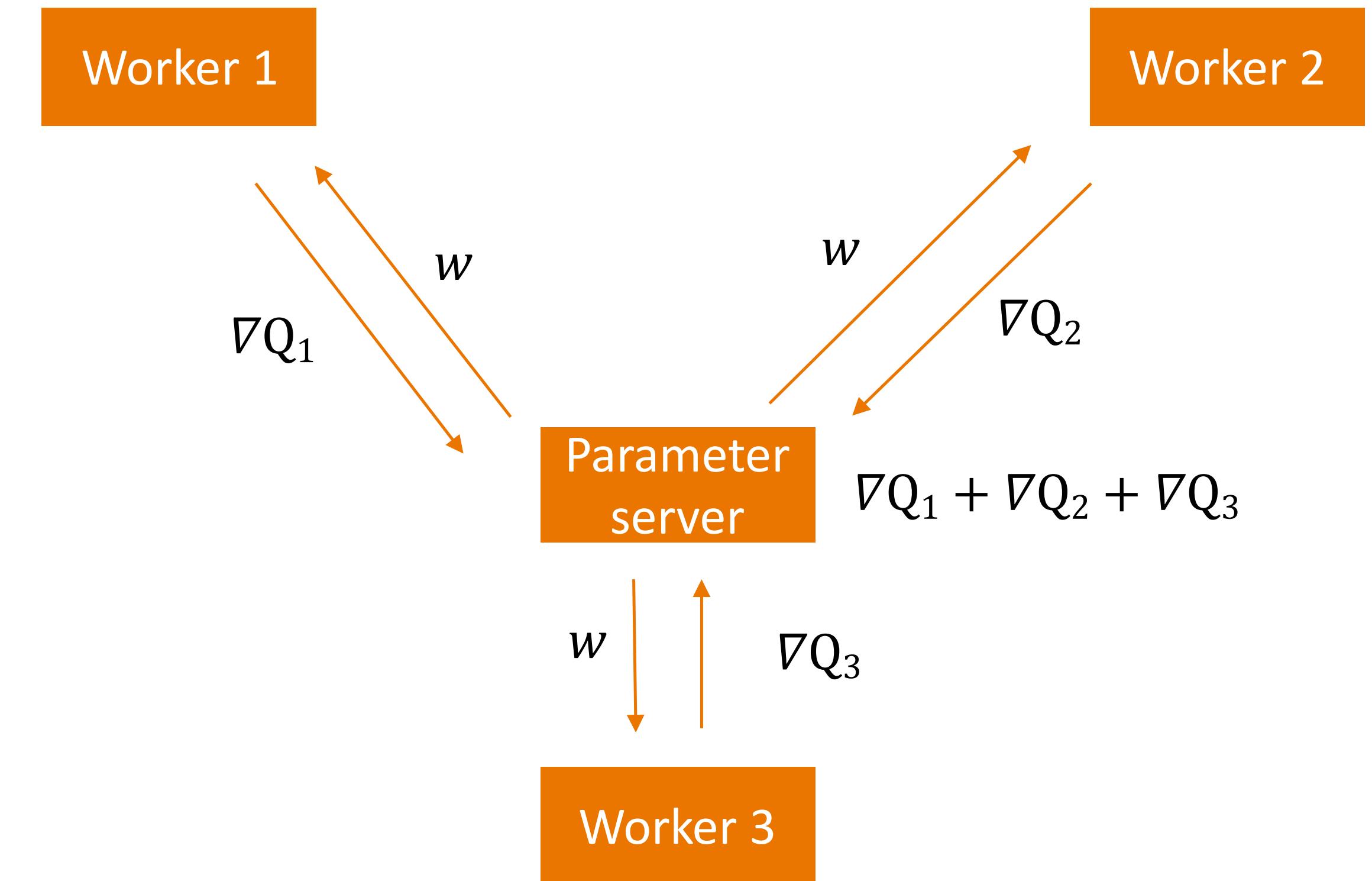
$$\nabla \mathbf{Q} = \sum_1^N g_n$$

$$\mathbf{W}_{j+1} = \mathbf{W}_j - \alpha \nabla \mathbf{Q}$$

Gradient Reduction Topology (sync'ed)



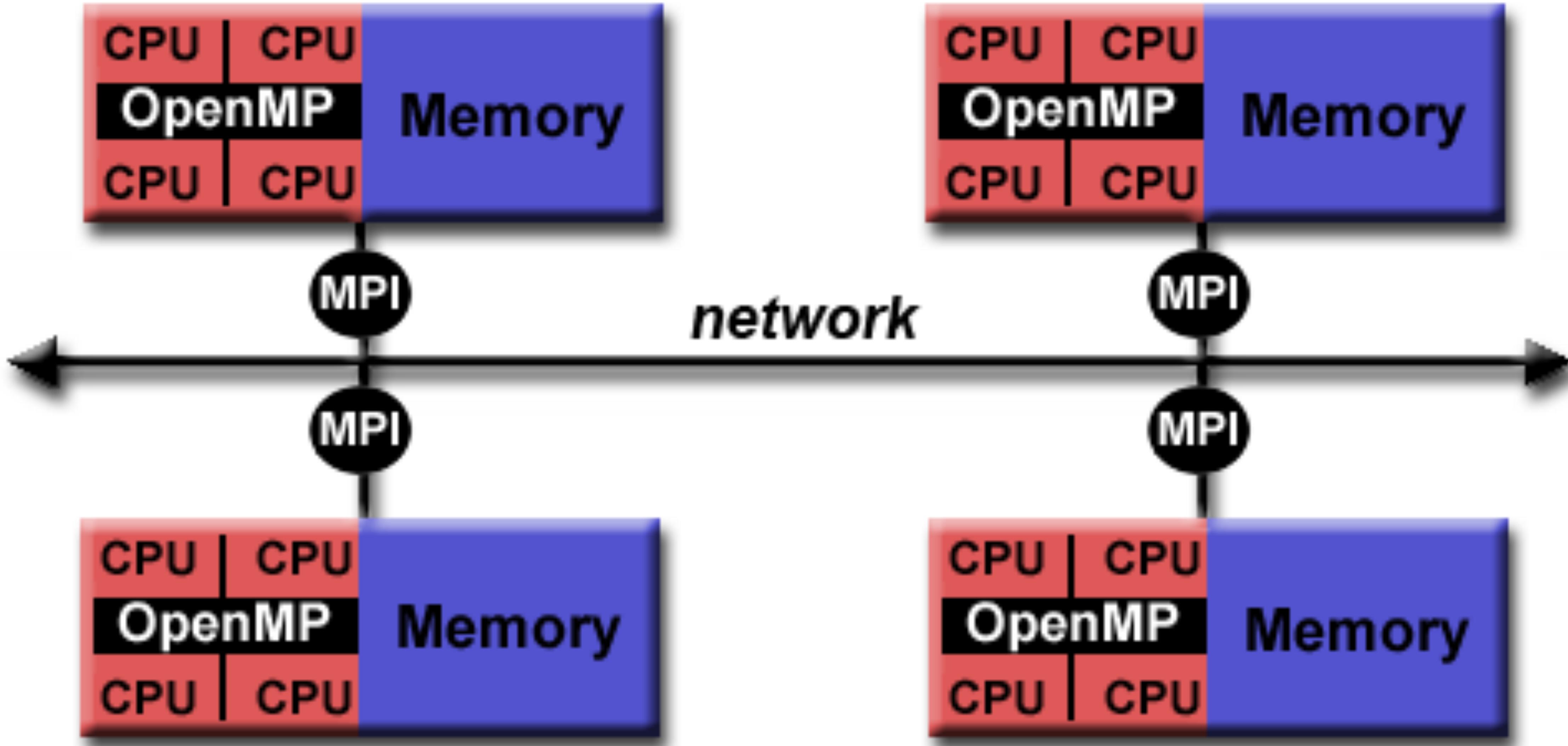
Decentralized



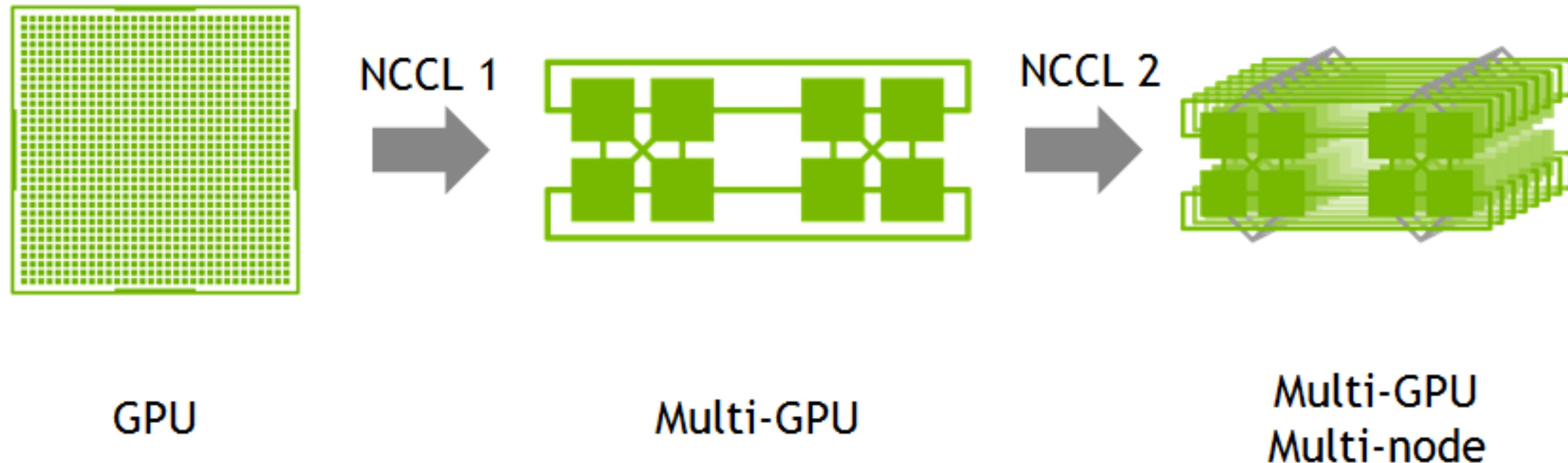
Centralized

How do we send gradients?

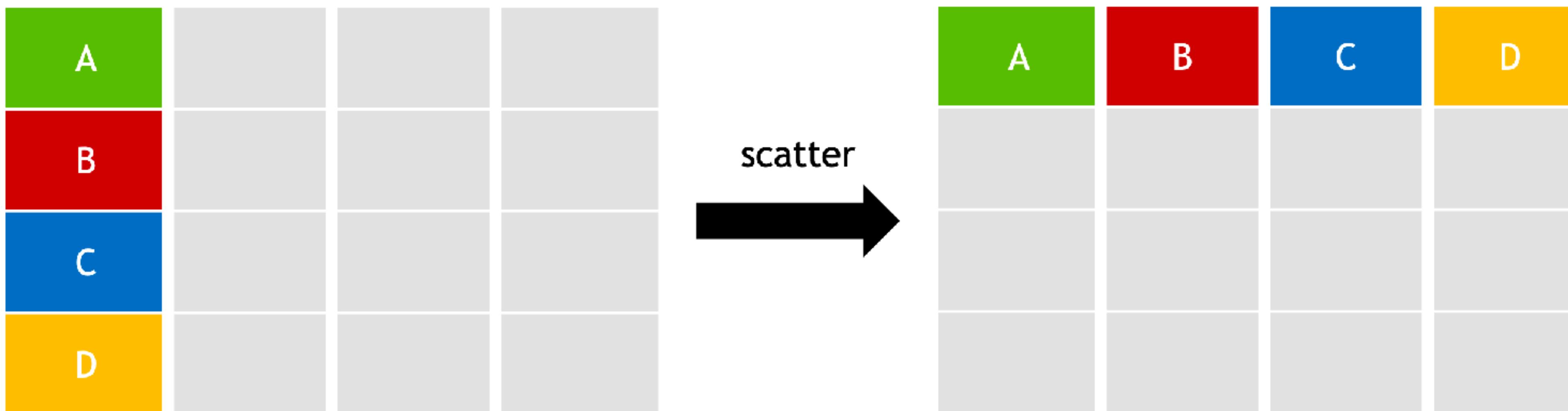
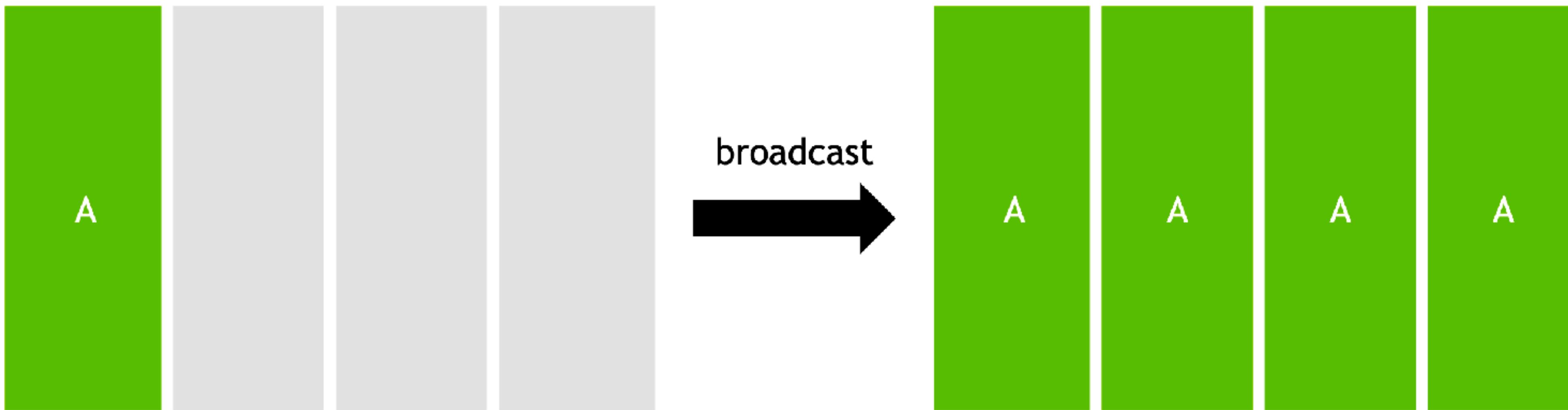
Message Passing Interface (MPI)



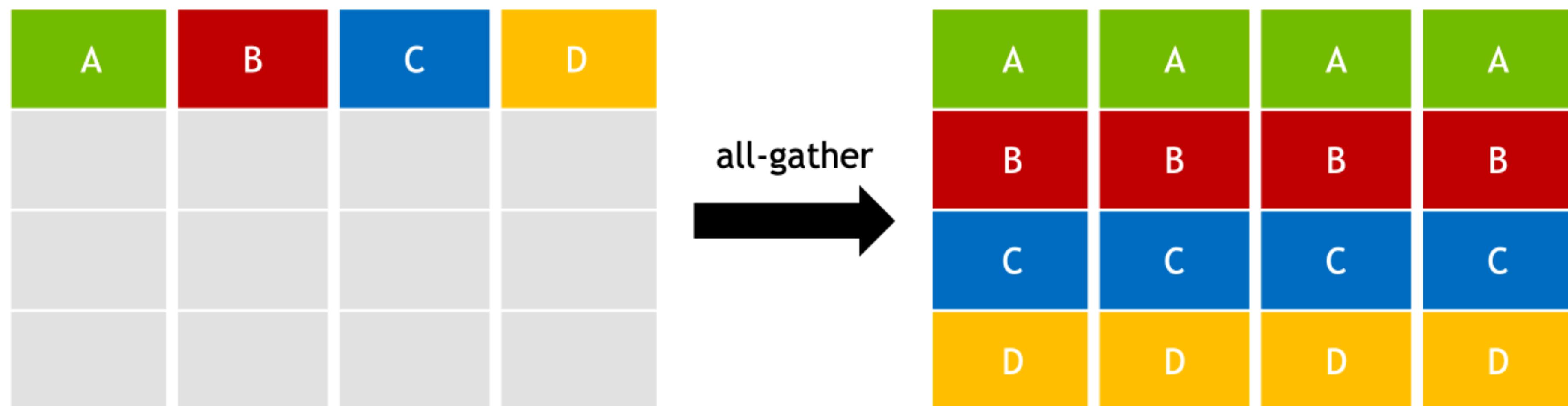
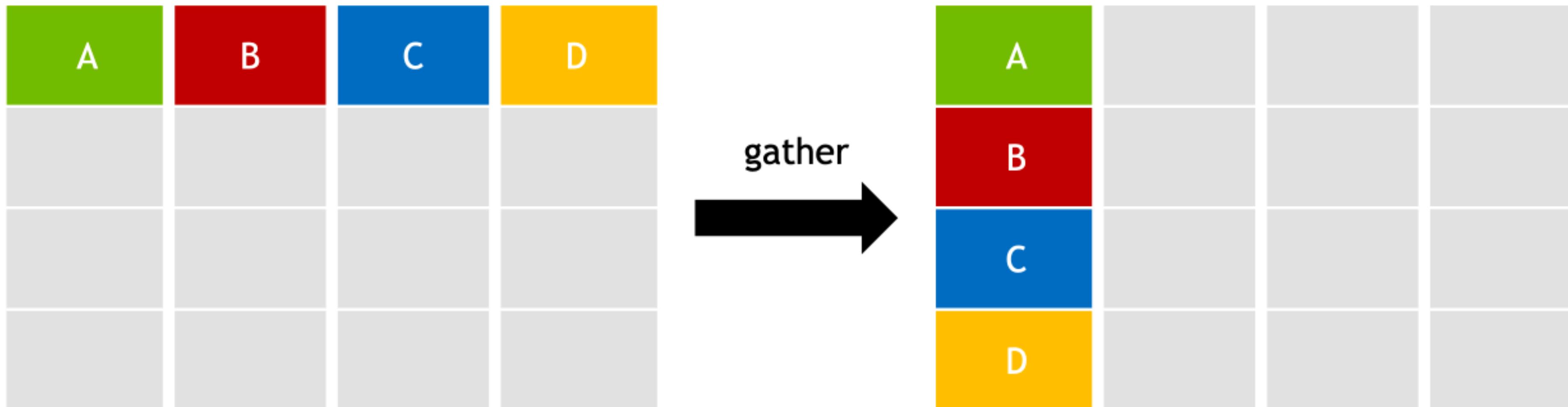
NVIDIA Collective Communication Library (NCCL)



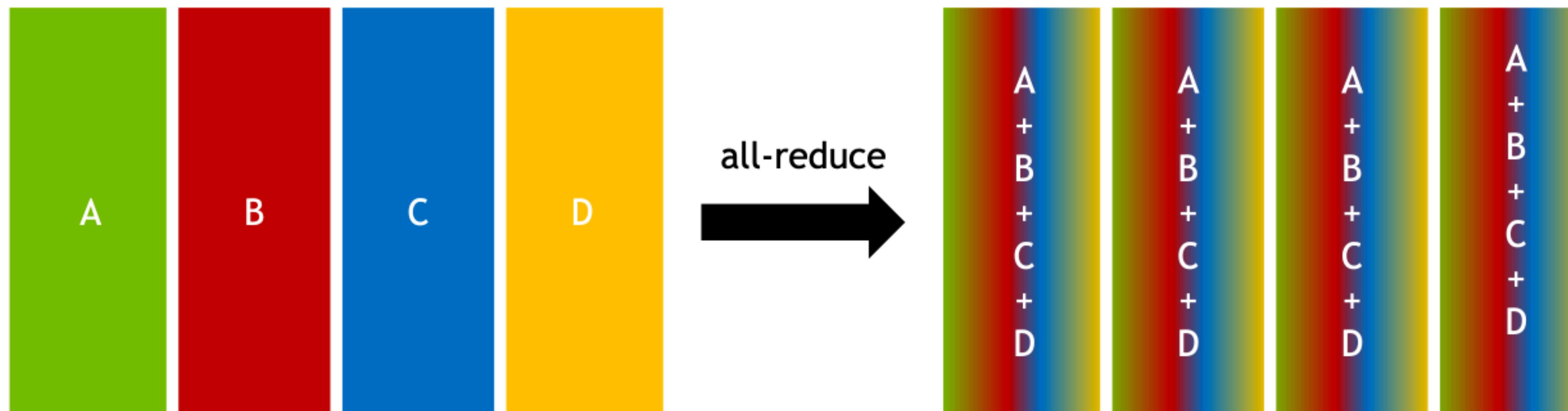
Collective Communication



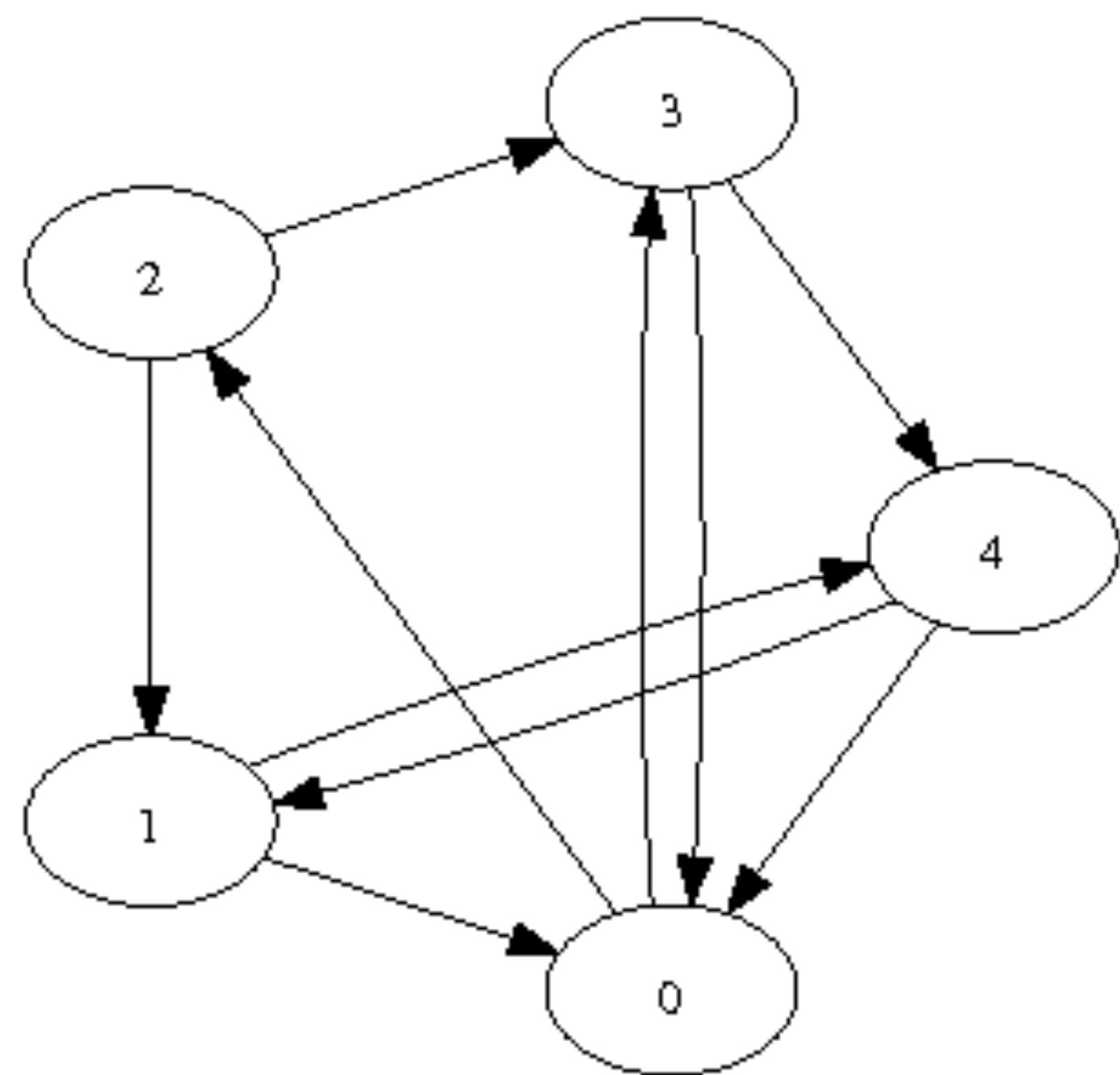
Collective Communication



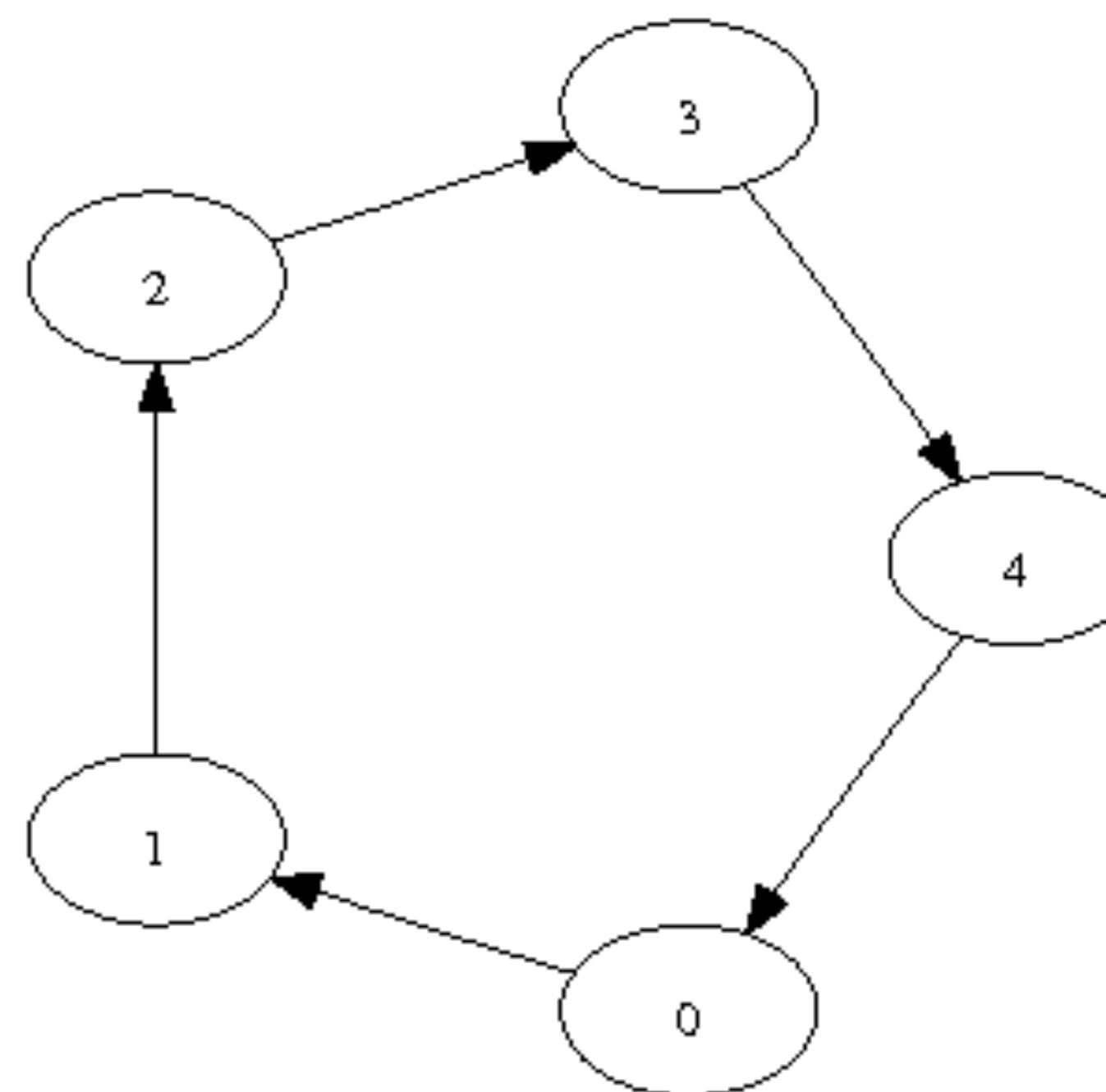
Collective Communication



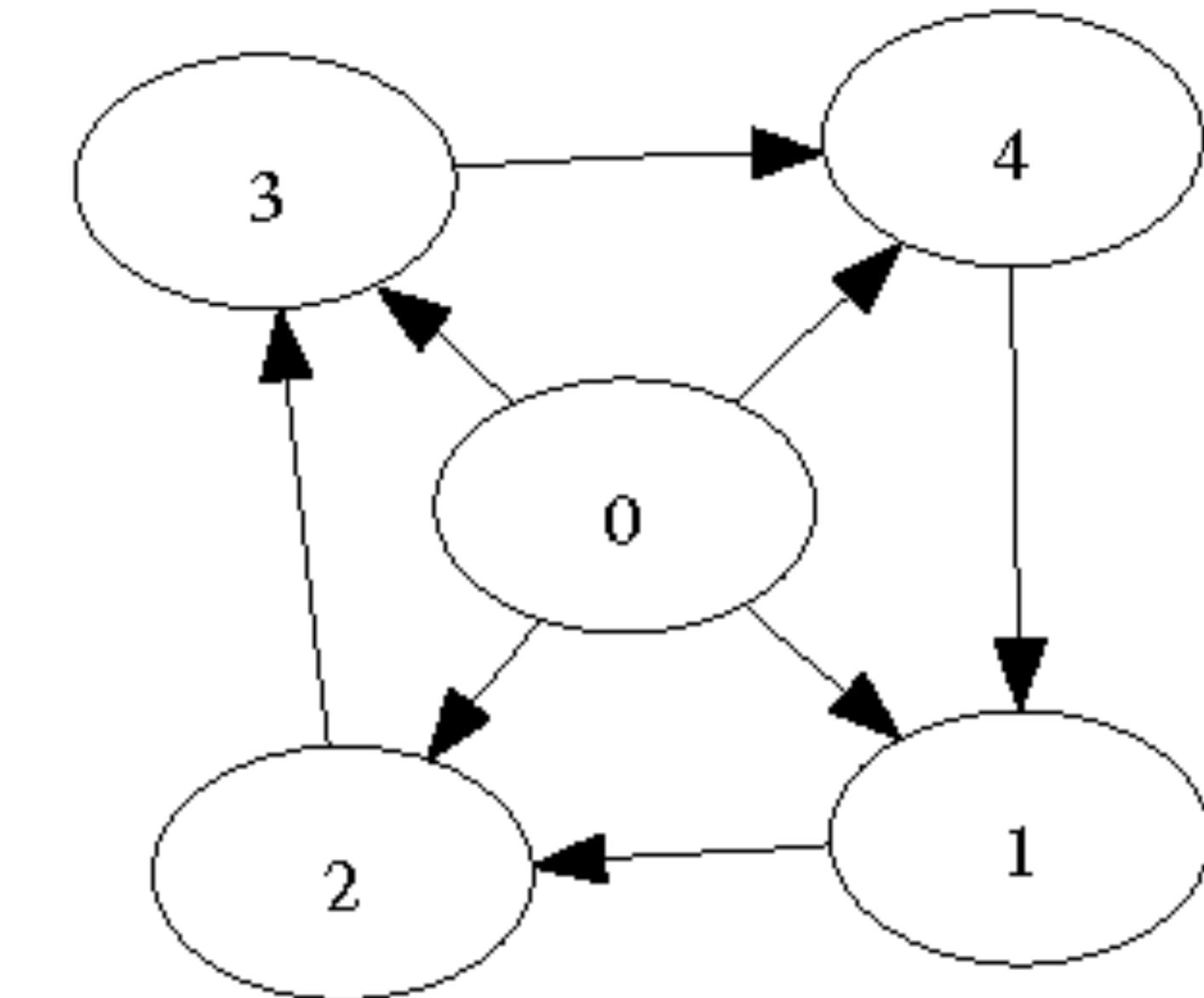
Communication Topology



(a) Random



(b) Ring



(c) Wheel

DL Framework Support



TensorFlow

CPU, GPU, TPU

Multi-GPU on one node with `tf.device`

Multi-node support with `tf.distribute`

cuDNN, MKL-DNN

MPI, NCCL

PyTorch

CPU, GPU

Multi-GPU on one node with `torch.multiprocessing`

Multi-node support with `torch.distributed`

cuDNN, MKL-DNN

Gloo, MPI, NCCL

Or use a framework on the top of DL libraries



- Supports TensorFlow, Keras, PyTorch, MXNet
- Supports MPI and NCCL
- Just a few lines of code changes (in principle)
- Has its own profiling facility (Horovod Timeline)

