Distributed Training: A Gentle Introduction

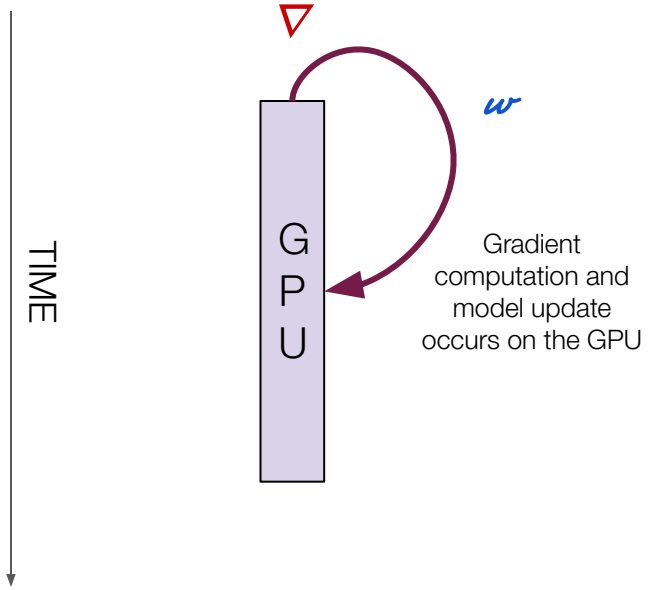
Stephen Balaban, Lambda

Single GPU Training

Computation Happens:
- On one GPU

Gradient transfers:
- N/A

Model transfers:
- N/A



Examples:
- TensorFlow
- PyTorch
- Caffe / Caffe 2
- MXNet
- etc.

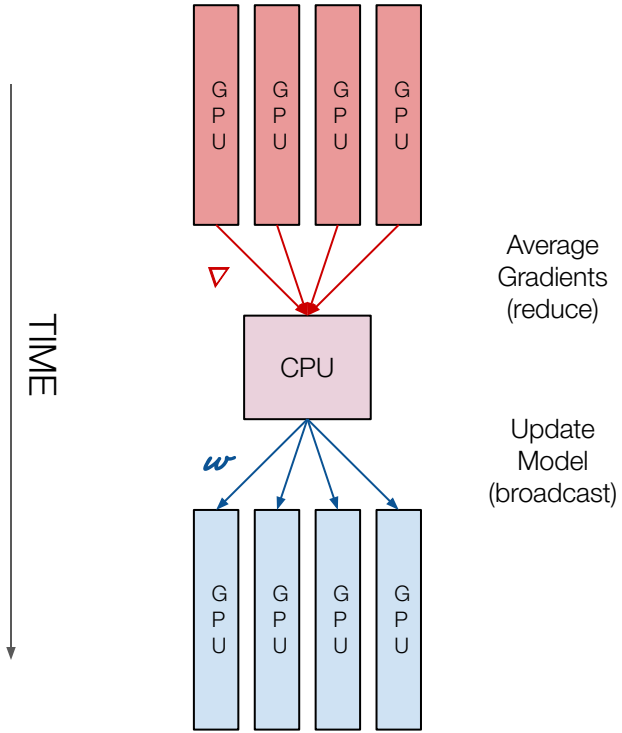
Problems:
- Small batch size => noisier stochastic approximation of the gradient => lower learning rate => slower training.

Multi GPU Training (CPU Parameter Server)

Computation Happens:
- On all GPUs and CPU

Gradient transfers:
- From GPU to CPU (reduce)

Model transfers:
- From CPU to GPU (broadcast)



Examples:
- TensorFlow w/ Graph Replication AKA Parameter Server AKA "Towers"
- PyTorch
- Caffe

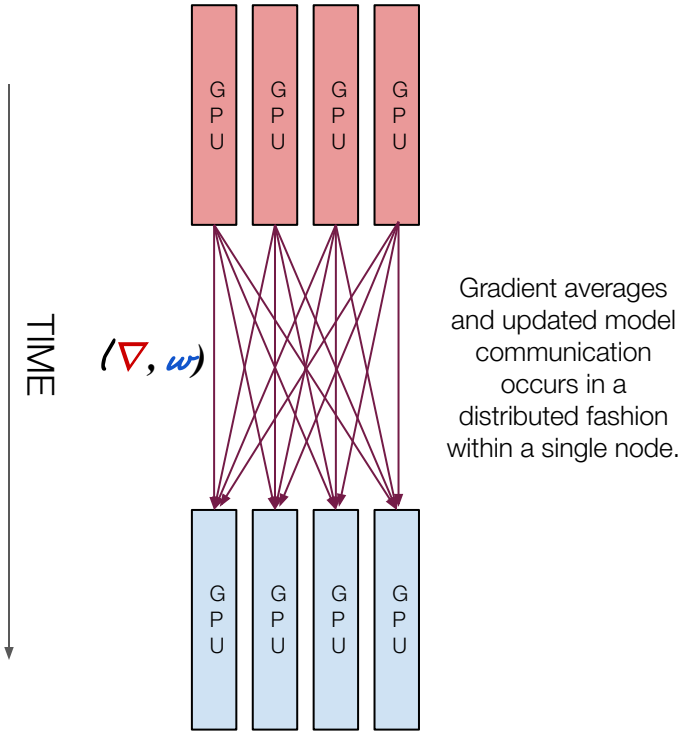
Problems:
- Not good for low arithmetic intensity models.
- Performance highly dependent on PCIe topology.

Multi GPU Training (Multi GPU all-reduce)

Computation Happens:
- On all GPUs

Gradient transfers:
- GPU to GPU during NCCL all-reduce

Model transfers:
- GPU to GPU during NCCL all-reduce



Examples:
- TensorFlow + NCCL
- PyTorch + NCCL

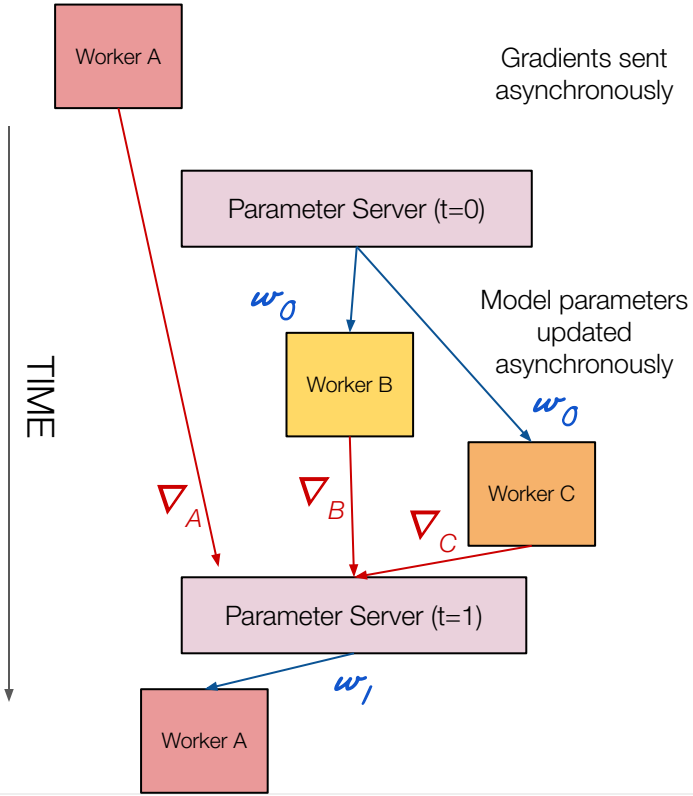
Problems:
- Not good for high arithmetic intensity models.
- Performance highly dependent on PCIe topology.

Asynchronous Distributed SGD

Computation Happens:
- On all workers and parameter servers

Gradient transfers:
- Worker to parameter server (asynchronously)

Model transfers:
- Parameter server to worker (asynchronously)



Examples:
- Hogwild!
- Async SGD is AKA Downpour SGD

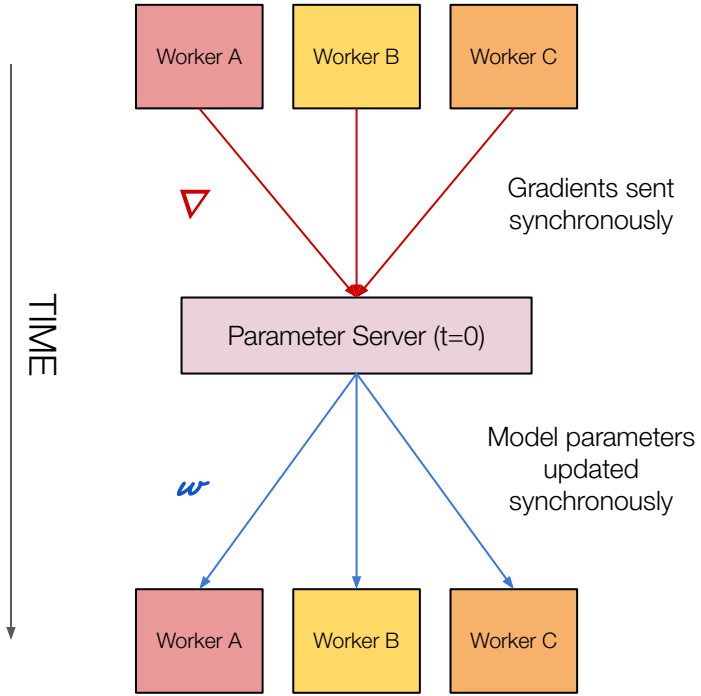
Problems:
- Stale gradients.
- Code that is difficult to write and maintain.
- Difficult to reason about order of operations.

Synchronous Distributed SGD

Computation Happens:
- On all workers and parameter servers

Gradient transfers:
- Worker to parameter server

Model transfers:
- Parameter server to worker



Examples:
- TensorFlow Distributed
- Torch.distributed

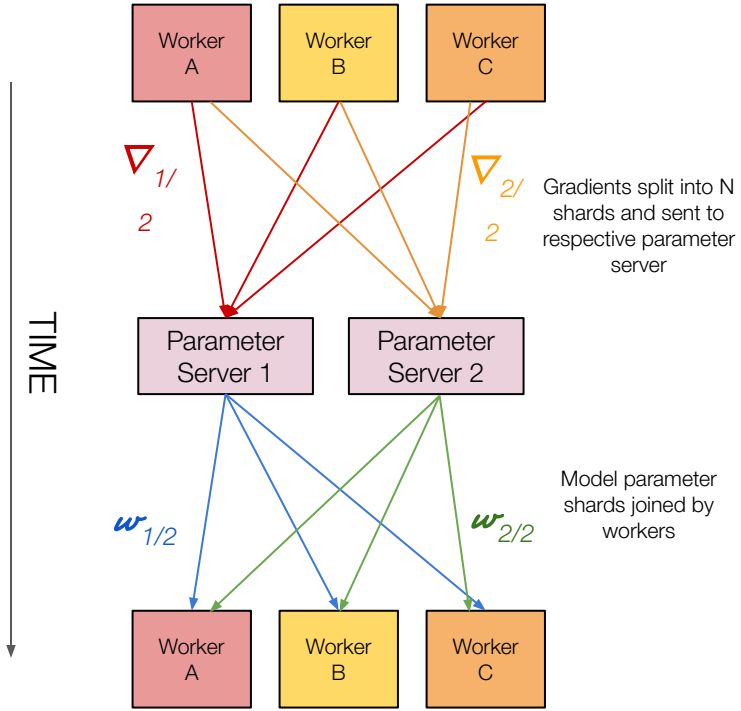
Problems:
- Needs lots of worker to parameter server bandwidth.
- Requires extra code and hardware for parameter server.

Multiple Parameter Servers

Computation Happens:
- On all workers and parameter servers

Gradient transfers:
- Worker gradient shards to parameter servers

Model transfers:
- Parameter server model shards to workers



Examples:
- TensorFlow Distributed
- Paddle Paddle

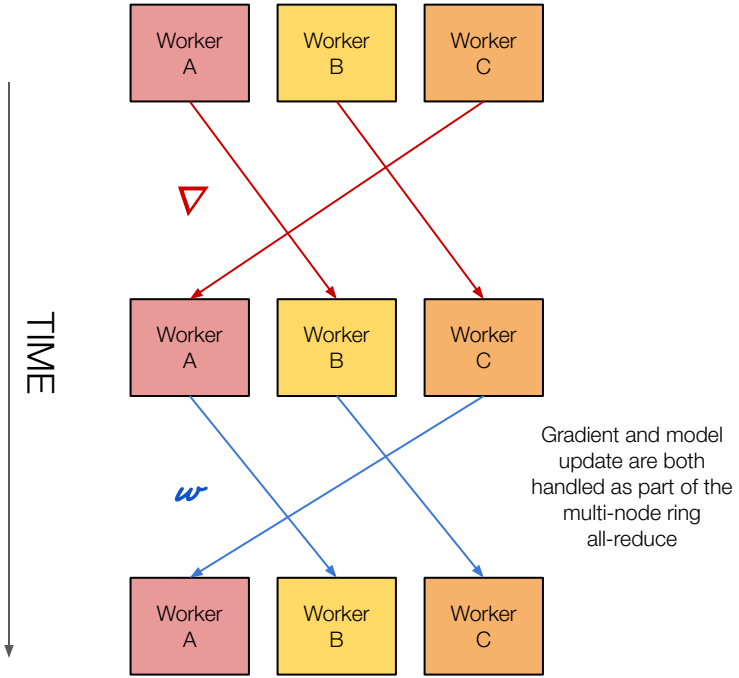
Problems:
- Need to tune ratio of parameter servers to workers.
- Again, even more complicated and difficult to maintain code.

Ring all-reduce Distributed Training

Computation Happens:
- On all workers

Gradient transfers:
- Worker transfers gradient to peers during all-reduce

Model transfers:
- Model "update" happens at the end of multi-node all-reduce operation



Examples:
- Horovod[1]
- tensorflow-allreduce

[1]*Horovod uses NCCL 2.0's implementation of multi-node all-reduce.*

Parameter Servers vs Multi-node Ring all-reduce

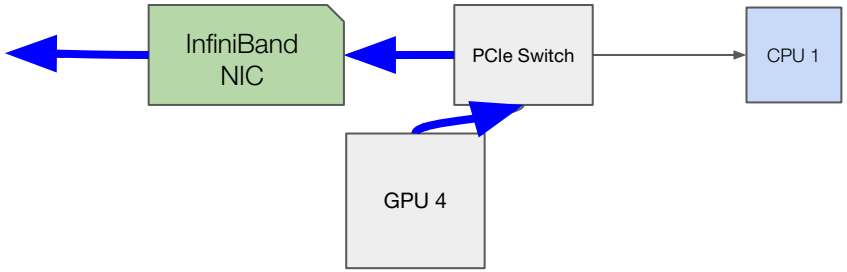
Parameter Servers

- Good for compute intensive workloads. (High arithmetic intensity.)
- High node-to-parameter server communication.

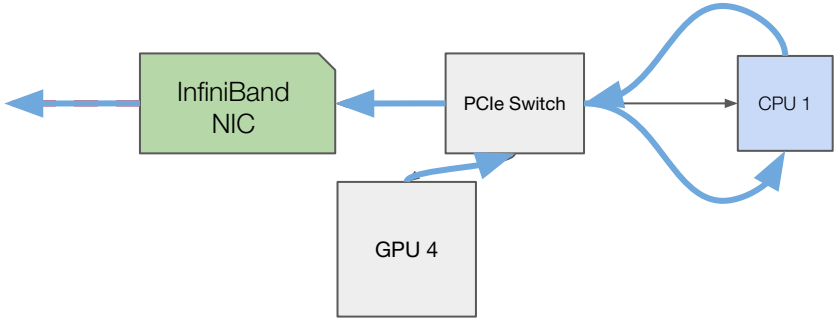
Multi-node Ring all-reduce

- Good for communication intensive workloads. (Low arithmetic intensity.)
- High node-to-node communication.

GPU RDMA over InfiniBand

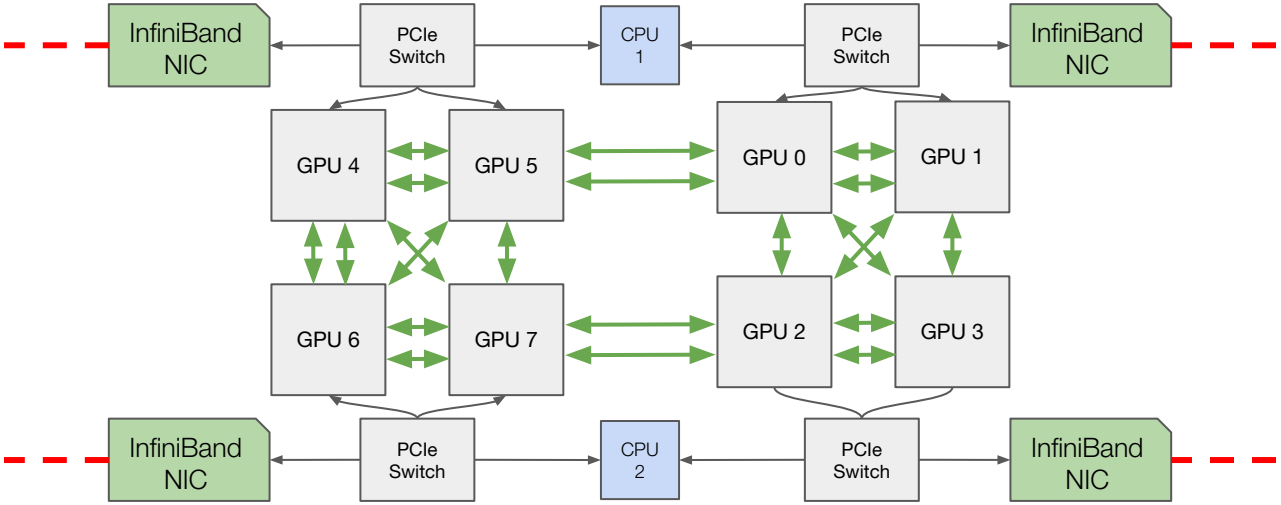


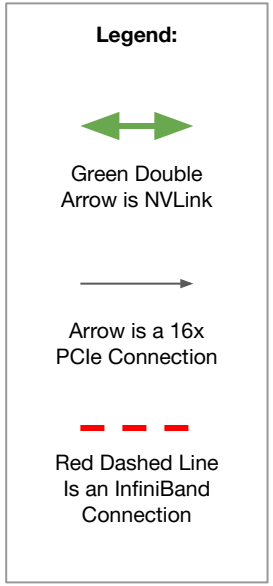
Data pathway with RDMA
(Directly out the door via PCIe switch)



Data pathway without RDMA
(Additional copy to CPU memory)

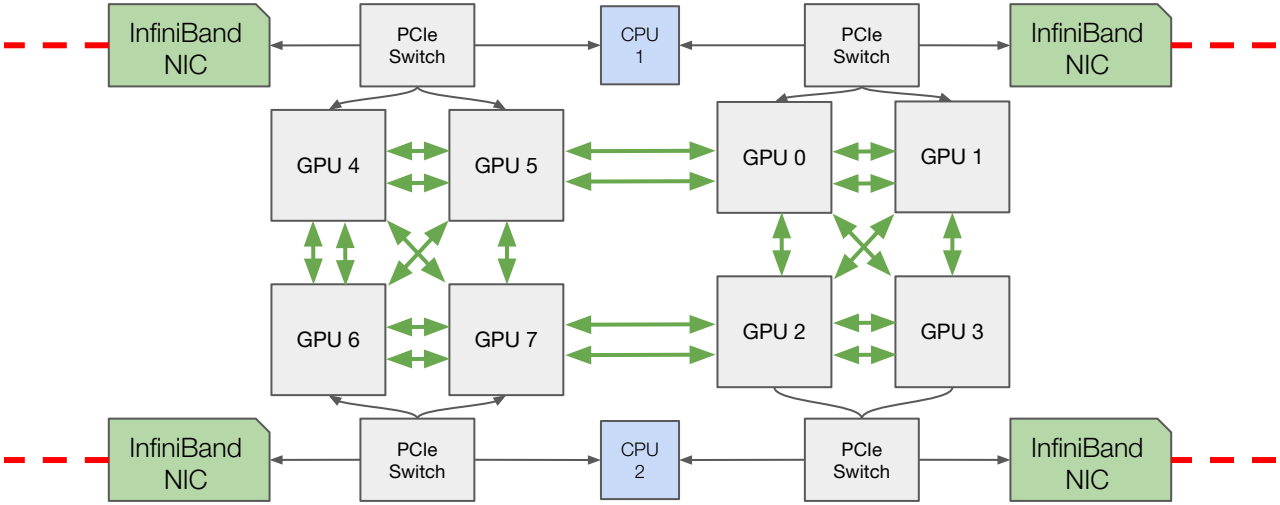
Hardware Configuration for Multi-node all-reduce





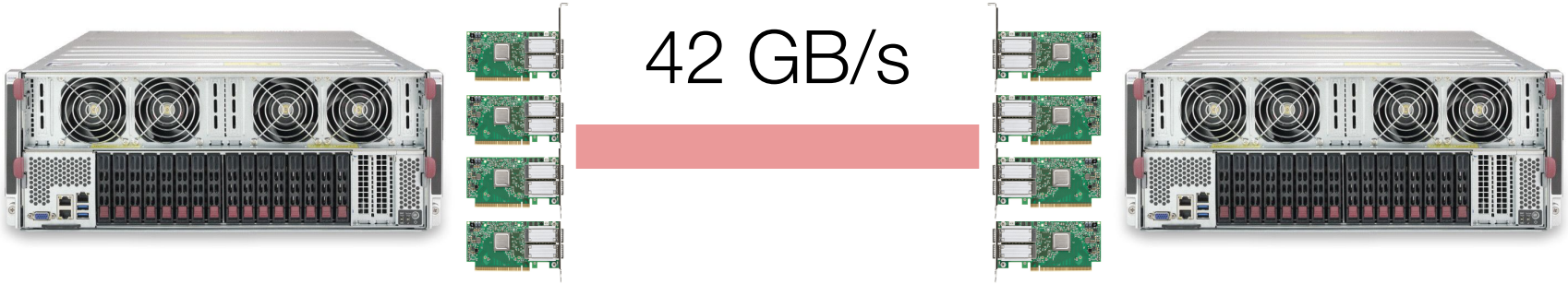
Four InfiniBand NICs are placed alongside the NVLink fabric underneath a PCIe switch. They are used for GPU RDMA during the distributed all-reduce operation.

Four Non-overlapping Pathways



Each GPU has six NVLink connections which allows for four non-overlapping pathways to be drawn through all eight GPUs on the system and out an InfiniBand card, optimizing both GPU to GPU and node to node communication during distributed all-reduce. (See Sylvain Jeaugey's "NCCL 2.0" presentation for more information.)

Inter-node Bandwidth



The GPU RDMA (Remote Direct Memory Access) capabilities of the InfiniBand cards and the V100 GPUs allows for a inter-node memory bandwidth of 42 GB/s. 84% of the 50 GB/s theoretical peak allowed by the four cards. 50 GB / s = 4 cards * 100 Gb/s / (8 bits/byte)

Citations

- Sergeev, Alexander and Mike Del Balso. **Horovod: fast and easy distributed deep learning in TensorFlow.** https://arxiv.org/pdf/1802.05799.pdf

- Pitch Patarasuk and Xin Yuan. **Bandwidth optimal all-reduce algorithms for clusters of workstations.** J. Parallel Distrib. Comput., 69:117–124, 2009. https://www.cs.fsu.edu/~xyuan/paper/09jpdc.pdf

- Jeaugey, Sylvain. **NCCL 2.0.** (2017). http://on-demand.gputechconf.com/gtc/2017/presentation/s7155-jeaugey-nccl.pdf

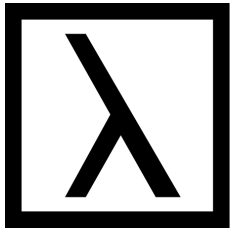
- WikiChip NVLink. https://fuse.wikichip.org/news/1224/a-look-at-nvidias-nvlink-interconnect-and-the-nvswitch/

Additional thanks to Chuan Li and Steve Clarkson.

Lambda Customers

About Me

- CEO of Lambda.
- Started using CNNs for face recognition in 2012.
- First employee at Perceptio. We developed image recognition CNNs that ran locally on the iPhone. Acquired by Apple in 2015.
- Published in SPIE and NeurIPS.



Lambda

enterprise@lambdalabs.com

https://lambdalabs.com