Evaluation of Volta-based DGX-1 System Using DNN Workloads

Saiful A. Mojumder
Marcia S Louis, Yifan Sun, Amir Kavyan Ziabari, José L. Abellán, John Kim, David Kaeli, Ajay Joshi
msam@bu.edu

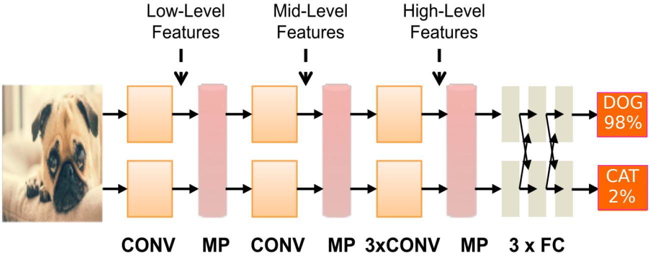
BARC 2019

This work was supported in part by NSF CNS-1525474 and MINECO TIN2016-78799-P.

Motivation

Deep Learning is Popular!

- Achieves high accuracy!
- Solves complex problems!



Motivation

Training of Deep Neural Networks is Time Consuming!

- Efficient hardware and software are needed
- GPU and Multi-GPU System accelerate training

Retrieved from https://www.2work.com.br

Objective

Understand the Characteristics of DNN Workloads

- Training of DNNs
- Compute– and communication–intensiveness

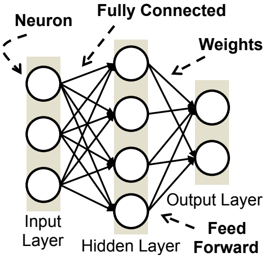
Identify the Factors Affecting the Training of DNNs

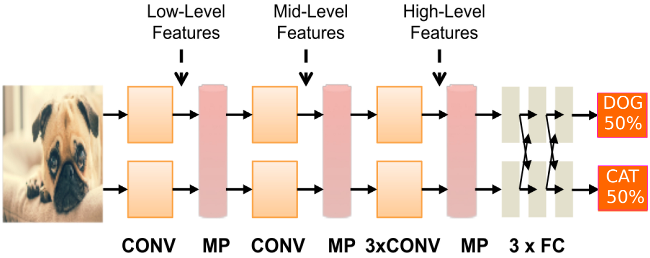
- Hardware-level limitations
- Software-level limitations

Background: DNN

What is a DNN?

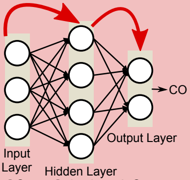
- Multiple layers of neurons
- Two neighboring layers connected via weights





Background: Training Stages of a DNN

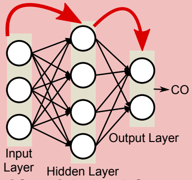
- **Forward Propagation (FP)**



CO = Calculated Output

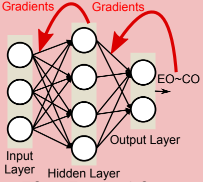
Background: Training Stages of a DNN

- **Forward Propagation (FP)**



CO = Calculated Output

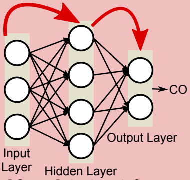
- **Backward Propagation (BP)**



EO = Expected Output

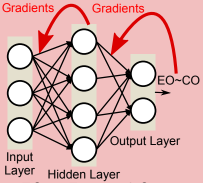
Background: Training Stages of a DNN

- **Forward Propagation (FP)**



CO = Calculated Output

- **Backward Propagation (BP)**

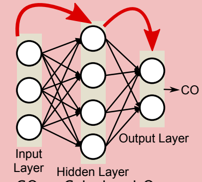


EO = Expected Output

- **Weight Update (WU)**
  - $N_W = O_W + \alpha \times f(G)$
    $N_W \rightarrow$ New Weight
    $O_W \rightarrow$ Old Weight
    $\alpha \rightarrow$ Constant
    $f(G) \rightarrow$ Averaged Gradients

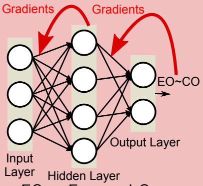
Background: Training Stages of a DNN

- **Forward Propagation (FP)**



CO = Calculated Output

- **Backward Propagation (BP)**



EO = Expected Output

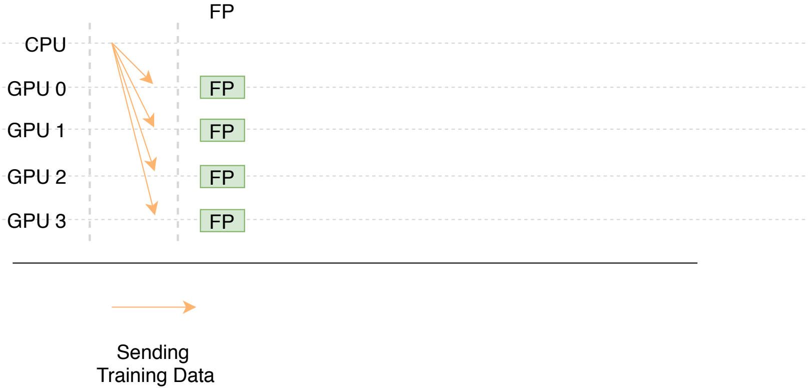
- **Weight Update (WU)**
  - $N_W = O_W + \alpha \times f(G)$
    $N_W \rightarrow$ New Weight
    $O_W \rightarrow$ Old Weight
    $\alpha \rightarrow$ Constant
    $f(G) \rightarrow$ Averaged Gradients
- **Metric Evaluation (ME)**
  - Forward propagation
  - Simple arithmetic operation

Background: Multi-GPU DNN Training

SGD Algorithm

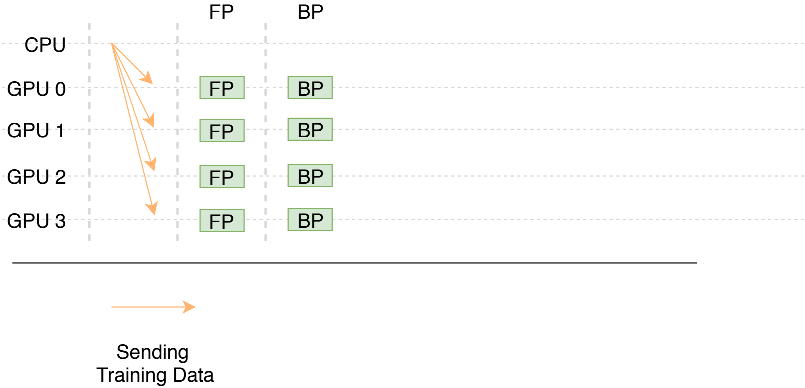
Background: Multi-GPU DNN Training

SGD Algorithm



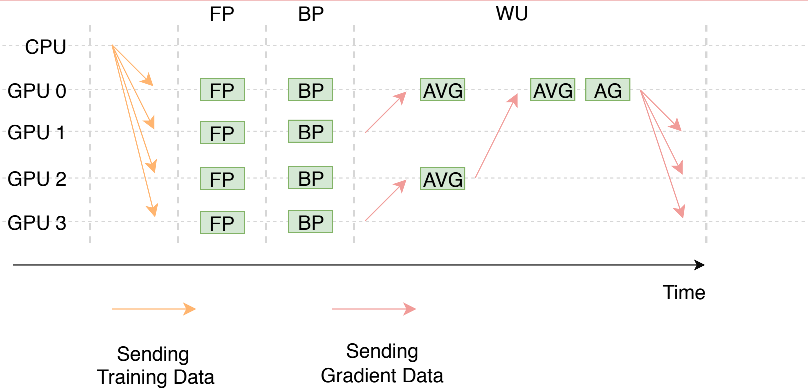
Background: Multi-GPU DNN Training

SGD Algorithm



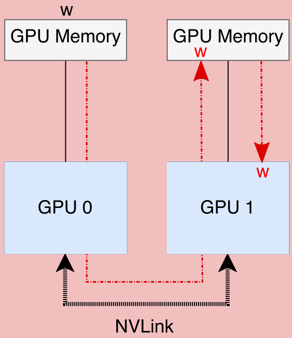
Background: Multi-GPU DNN Training

SGD Algorithm



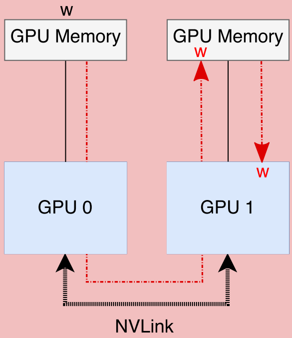
Background: Inter-GPU Communication

Peer-to-Peer (P2P) Memcpy

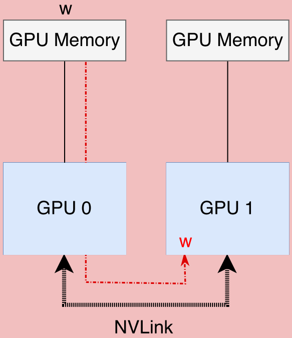


Background: Inter-GPU Communication

Peer-to-Peer (P2P) Memcpy

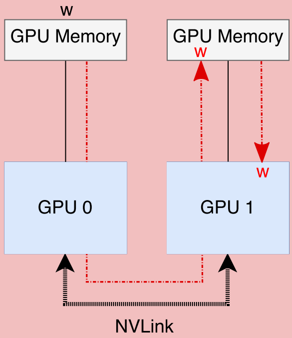


P2P Direct Transfer

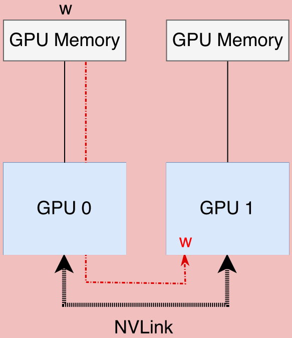


Background: Inter-GPU Communication

Peer-to-Peer (P2P) Memcpy



P2P Direct Transfer



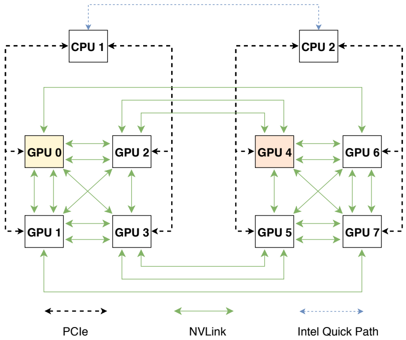
NVIDIA Collective Communication Library (NCCL)

- `Broadcast` and `AllReduce`
- P2P direct transfer

Methodology: Evaluation Platform

DGX-1 System with 8 Tesla V100 GPUs

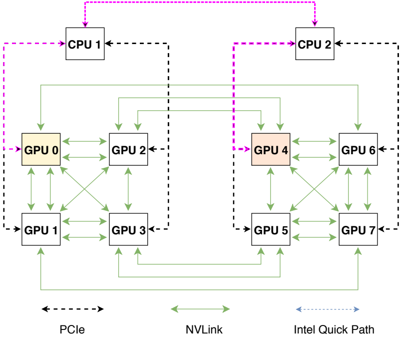
- Asymmetric interconnect
- Lack of direct NVLink connectivity between all GPUs
- PCIe or Two-hop communication



Methodology: Evaluation Platform

DGX-1 System with 8 Tesla V100 GPUs

- Asymmetric interconnect
- Lack of direct NVLink connectivity between all GPUs
- PCIe or Two-hop communication

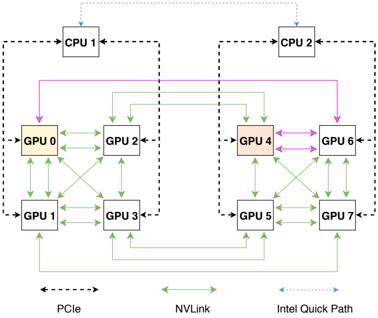


Path 1

Methodology: Evaluation Platform

DGX-1 System with 8 Tesla V100 GPUs

- Asymmetric interconnect
- Lack of direct NVLink connectivity between all GPUs
- PCIe or Two-hop communication

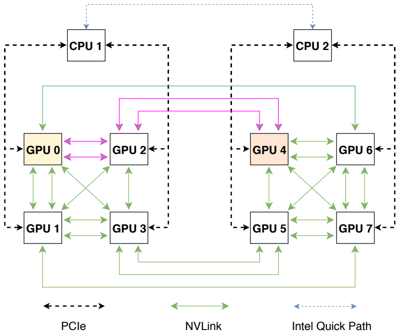


Path 2

Methodology: Evaluation Platform

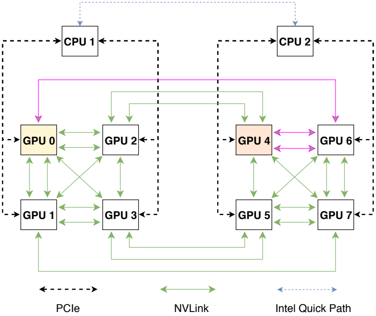
DGX-1 System with 8 Tesla V100 GPUs

- Asymmetric interconnect
- Lack of direct NVLink connectivity between all GPUs
- PCIe or Two-hop communication

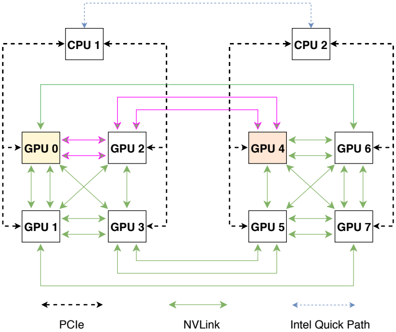


Path 3

Methodology: Evaluation Platform



Path 2



Path 3

Methodology: Workloads and Datasets

DNNs

- 5 different DNNs: LeNet, AlexNet, GoogLeNet, Inception-v3, and ResNet
- We perform training for one epoch

Network	Layers	Conv Layers	Incep Layers	FC Layers	Weights
LeNet	5	2	0	2	60K
AlexNet	8	5	0	3	60M
GoogLeNet	22	3	9	1	4M
Inception-v3	48	7	11	1	24M
ResNet	110	107	0	1	55M

Methodology: Workloads and Datasets

Scaling

- Strong Scaling
  - Increased GPU count
  - Fixed dataset
- Weak Scaling
  - Increased GPU count
  - Increased dataset

Dataset

- A subset of images from the Imagenet dataset
- Strong scaling– 256K images
- Weak scaling– 256K, 512K, 1M and 2M images for 1, 2, 4, and 8 GPUs, respectively
- Batch sizes– 16, 32, and 64

Questions We Address

- Do the workloads scale as GPU count increases?
- Does P2P always perform worse than NCCL?
- What is the impact of network size on training time?
- What is the impact of batch size on training time?
- How do different stages in the training process scale with GPU count, batch size and network size?
- What is the impact of GPU memory on training?
- How does weak scaling correlate with strong scaling?

Do the Workloads Scale with GPU Count?

- Not linearly!
- How well do they scale?
    - Depends!
        - DNN
        - Communication Method

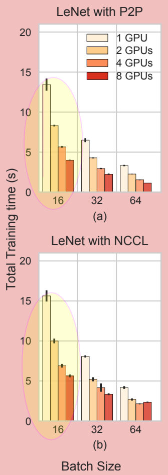
Do the Workloads Scale with GPU Count?

LeNet: Batch Size of 16

- P2P: 1.62×, 2.37×, and 3.36× for 2, 4, and 8 GPUs, respectively
- NCCL: 1.56×, 2.27×, and 2.77× for 2, 4, and 8 GPUs, respectively

LeNet Does Not Scale Well!

- Why? Small number of layers!



Does NCCL Always Outperform P2P?

LeNet: Batch Size of 16

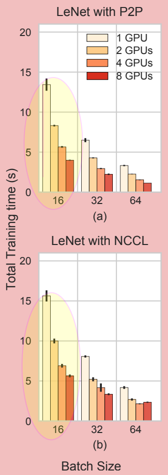
- P2P: 1.62×, 2.37×, and 3.36× for 2, 4, and 8 GPUs, respectively
- NCCL: 1.56×, 2.27×, and 2.77× for 2, 4, and 8 GPUs, respectively

LeNet Does Not Scale Well!

- Why? Small number of layers!

P2P Outperforms NCCL!

- Why? NCCL overhead!



What is the Impact of Network Size on Training Time?

GoogLeNet, Inception-v3 and ResNet: Batch size of 16

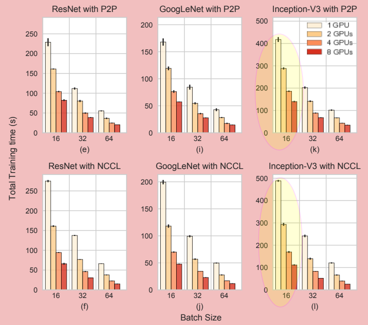
- P2P: $<1.5\times$, $<2.3\times$, $<3\times$ for 2, 4, and 8 GPUs, respectively
- NCCL: $<1.8\times$, $<2.9\times$, $<4.4\times$ for 2, 4, and 8 GPUs, respectively

Scale Better Than LeNet!

- Why? Significantly larger!

NCCL Outperforms P2P!

- Why? Amortization of Overhead!



How Much is the NCCL Overhead?

Measurement

- From 16% to 32% additional overhead for NCCL compared to P2P
- Smaller workload → More overhead

Source of NCCL Overhead

- Different source codes from P2P
- Different data transfer mechanism from P2P
- Different CUDA API from P2P

Network	Batch Size	(%) NCCL Overhead
LeNet	16	16.4
	32	24
	64	26.7
AlexNet	16	21.8
	32	21.8
	64	31.8
ResNet	16	20.1
	32	22.9
	64	19.3
GoogLeNet	16	18.7
	32	17.5
	64	16.2
Inception-v3	16	16.9
	32	19.4
	64	18.9

What is the Impact of Batch Size on Training Time?

For Both P2P and NCCL

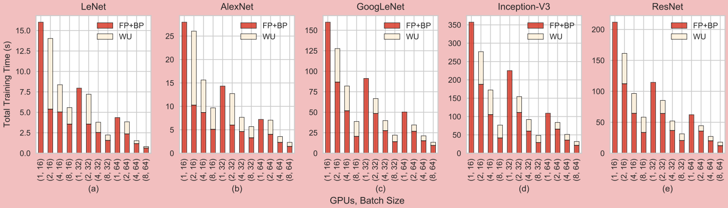
- Linear reduction in training time!
- True for all GPU counts!
- Why?
  - Fewer batches per GPU
  - More computation per batch
  - Fewer data transfers
  - Constant amount of data per batch

How Do Different Stages in the Training Process Scale?

FP+BP and WU Breakdown

- FP+BP
    - Compute-intensive
    - Only computation and no GPU-to-GPU data transfer
- WU
    - Communication-intensive
    - Transfer of gradients and weights
    - Negligible amount of computation

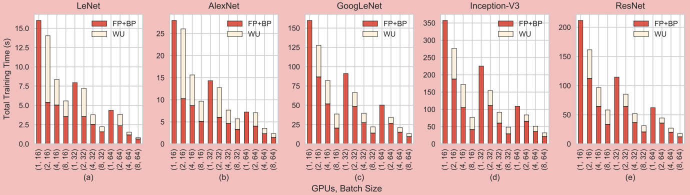
What Is the Impact of GPU Count on FP+BP and WU?



Impact on LeNet and AlexNet

- GPU count 1 $\rightarrow$ 2: $>2\times$ improvement in FP+BP time
- GPU count 2 $\rightarrow$ 4$\rightarrow$ 8: Non-linear decrease in the FP+BP time
  - Why?
    - Low GPU compute utilization!
- $\sim$Linear decrease in WU time!
  - Why?
    - Decrease in batches each GPU processes.

What Is the Impact of Network Size on FP+BP and WU?



Impact on Larger Workloads

- Near linear speedup of FP+BP stages
  - Why?
    - Increased GPU compute utilization!
- Better speedup in WU!
  - Why?
    - More weights per layer
    - Better NVLink BW utilization!

Memory Usage

- $\leq$5% difference between P2P and NCCL
- GPU0 consumes additional memory!
- Pre-training Memory Usage $\approx$ Memory for Network Model
- Training Memory Usage $\approx$ Memory for Network Model + Memory for outputs

What Is the Impact of Batch Size and Network Size on Memory Usage?

Impact of Batch Size

- Negligible increase in pre-training memory usage
- A limit on the maximum batch size
  - Inception-v3: No more than 64!
  - ResNet: No more than 128!

Impact of Network Size

- Larger network $\rightarrow$ More memory

Accelerating DNN Training

Hardware-Level Improvements

- More powerful GPUs!
- More efficient interconnect network!
- More memory capacity!

Software-Level Improvements

- Reduction in overhead
- Development of better scheduling mechanism
- Improvement of high level frameworks (such as MXNet)
- Efficient distribution of data
- Improvement in algorithm

Summary

Contributions

- Comparison between two different multi-GPU communication methods for training DNNs
- Breakdown of training time into computation– and communication–intensive portion
- Demonstration of the impact of GPU memory
- Evaluation of strong and weak scaling
- Guidelines for designing future hardware and software

Evaluation of Volta-based DGX-1 System Using DNN Workloads

Saiful A. Mojumder
Marcia S Louis, Yifan Sun, Amir Kavyan Ziabari, José L. Abellán, John Kim, David Kaeli, Ajay Joshi
msam@bu.edu

BARC 2019

This work was supported in part by NSF CNS-1525474 and MINECO TIN2016-78799-P.

Methodology: Framework and Tools

Framework and Libraries

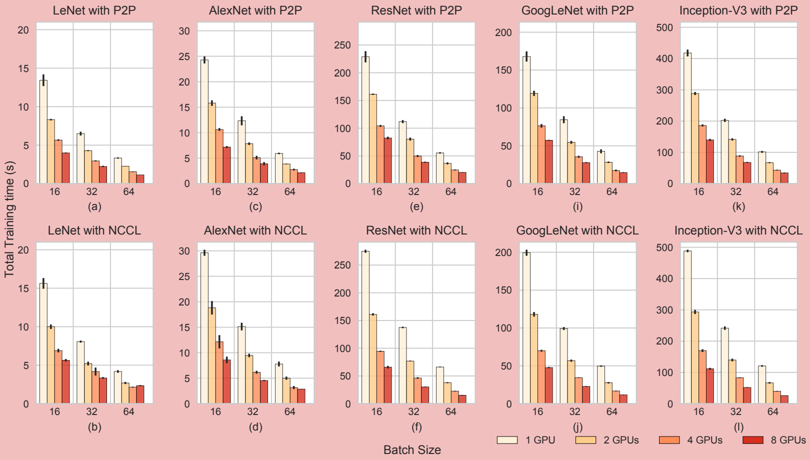
- NVIDIA container image of MXNet, release 18.04
- CUDA 9.0.176
- cuBLAS 9.0.333
- NCCL 2.1.15

Profiler and Tools

- `nvprof`
- `nvidia-smi`

Evaluation: P2P vs. NCCL

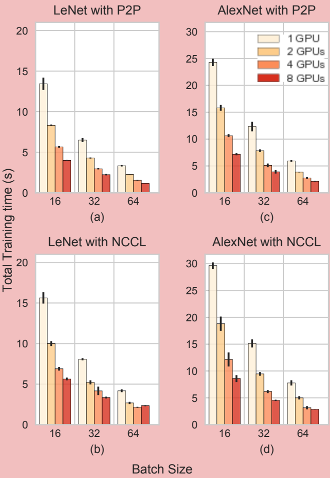
Average Run Time



P2P vs. NCCL: Impact of GPU Count

AlexNet: Batch Size of 16

- Achieves speedup similar to LeNet!
  - Why?
  - Still small number of layers (5 convolution layers)!



FP+BP and WU Breakdown: Batch Size

Increase in Batch Size

- More computation per batch
- Fewer synchronizations
- Less time needed for WU

Memory Usage

Network	Batch Size	Pre-training GPUz (GB)	Training GPU0 (GB)	Training GPUx (GB)	Additional Mem. Usage in GPU0 w.r.t. GPUx (%)	Increase in Mem. Usage w.r.t. the Batch Size of 16 (%)
LeNet	16	1.37	2.76	1.96	41.1	–
LeNet	32	1.38	2.84	2.04	39.4	3.0
LeNet	64	1.40	2.89	2.36	22.7	4.8
AlexNet	16	1.24	2.15	1.55	39.2	–
AlexNet	32	1.25	2.36	1.76	34.5	9.9
AlexNet	64	1.27	2.97	2.37	25.6	38.2
ResNet	16	1.08	3.62	3.29	10.1	–
ResNet	32	1.11	5.66	5.63	6.2	56.1
ResNet	64	1.13	9.48	9.15	3.5	161.5
GoogLeNet	16	0.92	2.35	2.24	4.7	–
GoogLeNet	32	0.94	3.64	3.55	2.5	55.2
GoogLeNet	64	0.97	6.17	6.07	1.6	162.8
Inception-v3	16	1.04	3.89	3.60	7.9	–
Inception-v3	32	1.06	6.70	6.06	10.5	72.3
Inception-v3	64	1.09	11.01	10.78	2.4	183.3

Memory Usage

- $\leq$5% difference between P2P and NCCL
- GPU0 consumes additional memory as it updates the weights and broadcasts to all other GPUs
- Pre-training memory usage depends on the network model

Memory Usage: Impact of Batch Size and Network Size

Impact of Batch Size

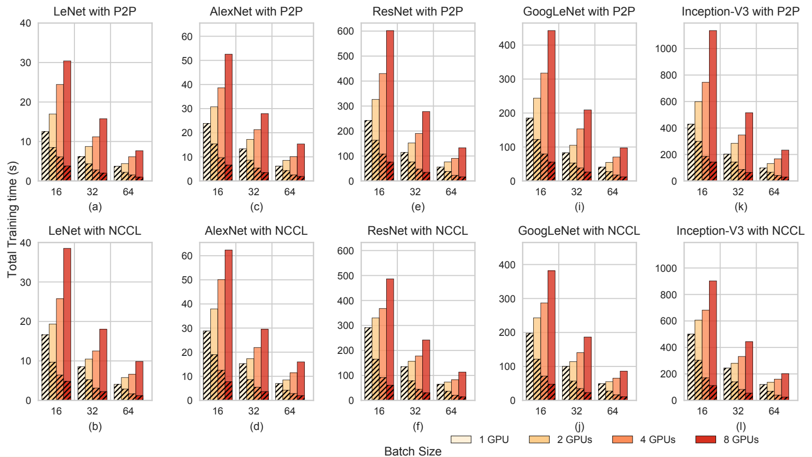
- Increase in batch size does not increase pre-training memory usage
- Increase in batch size increases the memory usage for larger DNNs
- Memory usage poses a limit on the maximum batch size that can be used to train a DNN
  - We could not train Inception-v3 and ResNet with a batch size larger than 64 and ResNet with a batch size larger than 128

Impact of Network Size

- As the network size increases (i.e. increased number of layers and neurons), the memory usage increases

Evaluation: Weak Scaling

Run Time for Weak Scaling



Weak Scaling

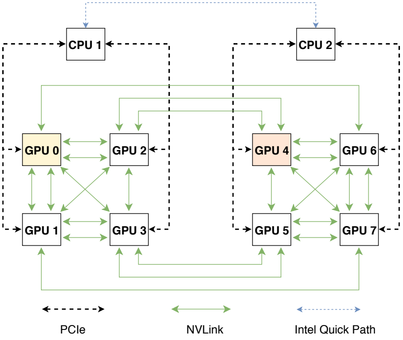
Weak Scaling vs. Strong Scaling

- Smaller workloads (i.e. LeNet and AlexNet) achieve less than 12% training time for weak scaling for all batch sizes and GPU counts
  - Why? Some API overheads associated with CUDA streams get amortized
- For larger workloads (i.e. ResNet, GoogLeNet, and Inception-v3), the speedup for weak scaling is less than 17% for all batch sizes and GPU counts
  - Why? Increased amount of communication leads to further amortization in NCCL overhead

Why Does P2P Perform Worse Than NCCL for 8 GPU Cases?

P2P Performs Poorly!

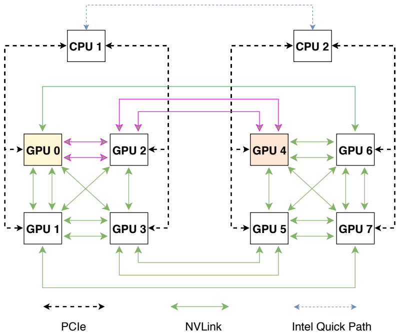
- **Asymmetric link distribution**
- 2-hop data copy
- Copy + Fetch



Why Does P2P Perform Worse Than NCCL for 8 GPU Cases?

P2P Performs Poorly!

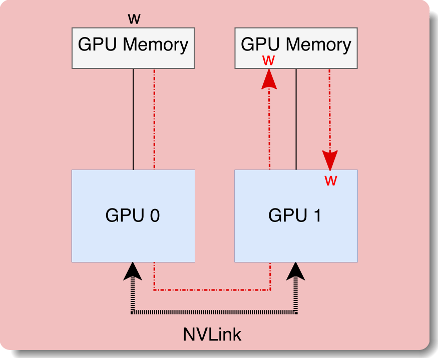
- Asymmetric link distribution
- **2-hop data copy**
- Copy + Fetch



Why Does P2P Perform Worse Than NCCL for 8 GPU Cases?

P2P Performs Poorly!

- Asymmetric link distribution
- 2-hop data copy
- **Copy + Fetch**



How Does Weak Scaling Correlate with Strong Scaling?

LeNet and AlexNet

- $\leq$12% reduction in training time for weak scaling compared to strong scaling
  - Why?

    Some amortization of API overheads

GoogleNet, Inception-v3, and ResNet

- $\leq$17% reduction in training time for weak scaling compared to strong scaling
  - Why?
    - Increased amount of communication $\rightarrow$ further amortization of NCCL overhead