National Institute of Advanced Industrial Science and Technology (AIST), Japan

Large-Scale Design of Distributed Deep Learning

Truong Thao Nguyen

Digital Architecture Research Center
Continuum Computing Architecture Research Team

AIST-Tokyo Tech Real World Big-Data Computation Open Innovation Laboratory (RWBC-OIL)

29th July 2021

Introduction

Self Introduction

Personal information

- TRUONG THAO NGUYEN
- Email: nguyen.truong@aist.go.jp, nguyen.88.tt@gmail.com
- Web: https://researchmap.jp/nguyentt?lang=en

Research expertise:

Interconnection network, HPC System, Distributed computing, High-Performance Large-scale Distributed Deep Learning

- Design Interconnection Network for High Performance Computing System.
  - Master at SoICT/HUST, VietNam (2012-2014)
  - PhD at SOKENDAI / NII, Japan (Apr 2015 – Mar 2018)

- High-Performance Large-scale Distributed Deep Learning.
  - Postdoc at RWBC-OIL, AIST (Apr 2018 – Sept 2019)
  - Researcher at AIST (Oct 2019 – present)

- https://www.aist.go.jp/
- The National Institute of Advanced Industrial Science and Technology (AIST), one of the largest public research organizations in Japan, focuses on the creation and practical realization of technologies useful to Japanese industry and society, and on **“bridging” the gap between innovative technological seeds and commercialization**.
- AIST is organized into **5 departments and 2 centers** that bring together core technologies to exert its comprehensive strength.
- AIST is also actively building a global network by, for example, signing memorandums of understanding for comprehensive research cooperation (**MOUs**) with major research institutes around the world.

Information Technology and Human Factors

From IT Infrastructure to application technology: make, connect, use

Research Unit

- Human Informatics and Interaction Research Institute
- Cyber Physical Security Research Center
- Human Augmentation Research Center
- Human-Centered Mobility Research Center
- Artificial Intelligence Research Center
- Industrial Cyber-Physical Systems Research Center
- Digital Architecture Research Center

Open Innovation Laboratory

- AIST- Tokyo Tech Real World Big-Data Computation Open Innovation Laboratory (RWBC-OIL)

https://www.aist.go.jp/aist_e/dept/en_dithf.html

The rapid growth of devices (smartphone, tablets, robots, wearables, etc.) and the Internet has increased the amount of information that is being produced and accessed by society. In order to better utilize the data produced from millions of devices and systems, we are conducting research and development in a wide range of fields at the interface between information technologies and human factors. Our mission is to engage and enrich the public through the **research and development of intelligent systems combining computational and physical capabilities** for human use.

ABCI - The World #12 Supercomputer

Artificial Intelligence Cloud Research Team(~ 2021/03/31)
https://www.airc.aist.go.jp/en/teams/

AI Bridging Cloud Infrastructure (ABCI) is the world's first large-scale Open AI Computing Infrastructure

ABCI consists of; 120 Compute Nodes (A) that form in total 960 NVIDIA A100 GPU accelerators, 1,088 Compute Nodes (V) that form in total 4,352 NVIDIA GPU V100 accelerators, shared file systems that provide in total 35PB capacity, Cloud Storage,

https://www.digiarc.aist.go.jp/en/org/

Digital Architecture Research Center

The Digital Architecture Research Center was established to develop and implement technologies that promote the digital revolution as a contribution to solving social issues and innovative innovation. New services and value for all individuals and organizations by appropriately linking various, diverse, and large amounts of data generated by individuals, devices, factories, etc., and highly integrating cyberspace and physical space while utilizing AI technology. We aim to realize an advanced society that continuously provides new services. Coordinating research units inside and outside the domain to advance the data-driven digital society, it acts as a control tower for architectural design, necessary technology development, standardization, software implementation and dissemination activities.

Continuum Computing Architecture Research Team

Leader

Yusuke Tanimura

Continuum Computing Infrastructure Research Team

Leader

Takahiro Hirofuchi

This Talk - Overview

Motivations

- Explosion of Deep Learning (DL)
  - Effectiveness in a variety of applications
- Long training time limits the development of new DL applications
  - Training GPT-3 model takes 355 years on a V100 GPU server and cost $4.6M
- Parallel training on Supercomputer / HPC systems
- ➔ Our target: **large-scale** training, e.g., 100s-1000s of GPUs

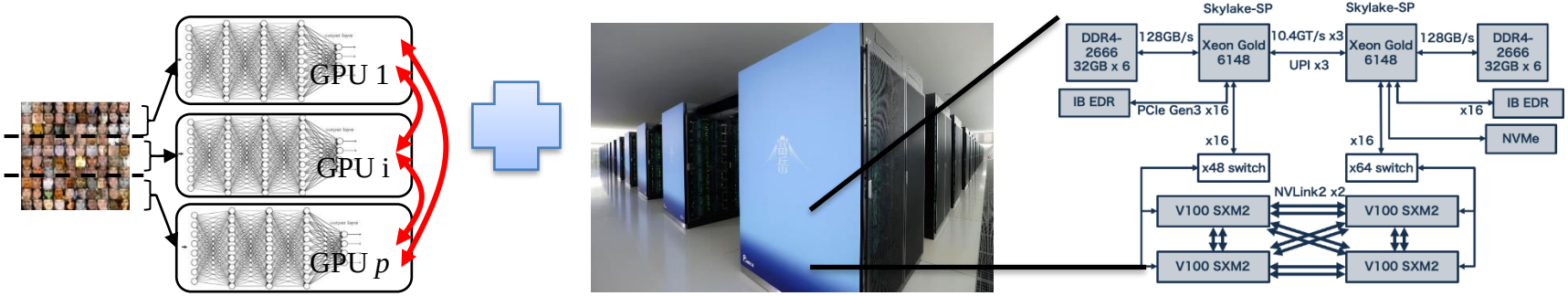


Illustration of data parallelism with Supercomputer

Long Training Time Limits the Development of New Deep Learning Applications

Training at supercomputer /datacenter

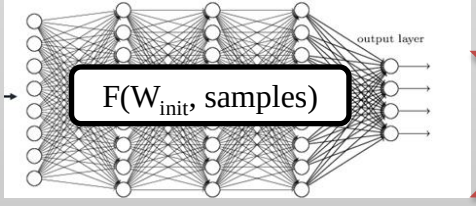
over hours/days/weeks

Input: Lots of Labeled data

Nguyen

Millions/billions samples, e.g., images

A Deep Neural Network (DNN) model



Find parameter W to minimize the loss function L(W)

$$W = W - \frac{\varphi}{B}\sum_{i=1}^{B}\nabla L(W)$$

Output: trained model

F(W)

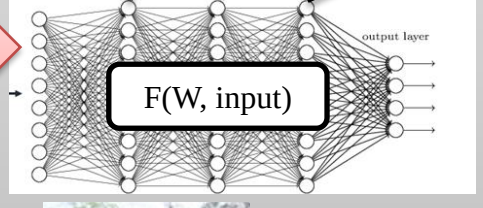
Inference at endpoint (mobile phone)

in a second

New input from Camera

local contrast face features face

Trained Deep Learning Model



97% Nguyen
3% Mr. A

Output: face recognition

Challenges of Training DNN in New Domains

Input: lots of Labeled data

A Deep Neural Network (DNN) model

Output: trained model

Recent well-known application

100 **KB**
(100x100 pixel)

RESNET50
(~**60M** param.)

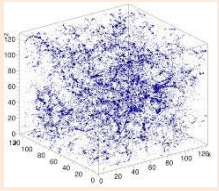
Inceptionv3
(~**40M** param.)

Bigger input (dataset/samples)

Deeper, bigger DNN model

New domain's application

High-resolution image ~**MBs**
(2K x 2K pixel)



Multi-dimensions , e.g., **2 GB**/ sample in cosmology [1]

RESNET1001
(> **x0M** param.)

AmoebaNet @Google
(~ **600M** param.)

Megatron-ML (**8.3B** param.), Turing-NLG (**17B** param.), GPT-3 (**175B**) from Natural Language Processing

Scalability Problems

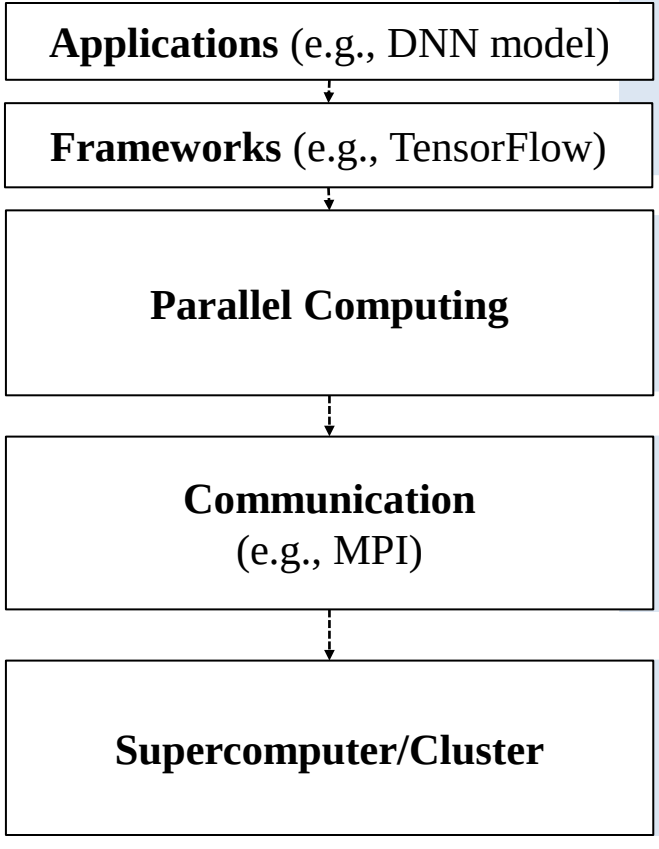
1. Huge communication /data movement overhead
2. Lack of memory capacity
3. Maintain the application accuracy

HOW - Research Goal and Approach

Goal: enable training DNNs on **large-scale** HPC system in the magnitude of hours.

Technical Goal: (1) Less Memory (2) Faster (3) Accuracy

Stack for Distributed DNN Training



Question: Design high-performance optimization/ method
Goal: (2) (3)
Topics: Shuffling, 2nd order method, pruning model

Question: What is the best parallelization strategy?
Goal: (1) (2) (3)
Topics: Data parallelism ➔ New parallelism approach

Question: How to reduce the communication time?
Goal: (2) (3)
Topics: Topology-aware, data compression, overlapping

Question: How do the network architectures effect the per.
Goal: (2)
Topics: Network topology, inter-node link bandwidth

High Performance I/O

HOW - Research Goal and Approach

Goal: enable training DNNs on **large-scale** HPC system in the magnitude of hours.

Technical Goal: (1) Less Memory (2) Faster (3) Accuracy

Stack for Distributed DNN Training

Applications (e.g., DNN model)

Frameworks (e.g., TensorFlow)

Parallel Computing

Communication (e.g., MPI)

Supercomputer/Cluster

Question: Design high-performance optimization/ method
Goal: (2) (3)
Topics: Shuffling, 2nd order method, pruning model

Question: What is the best parallelization strategy?
Goal: (1) (2) (3)
Topics: Data parallelism ➔ New parallelism approach

Question: How to reduce the communication time?
Goal: (2) (3)
Topics: Topology-aware, data compression, overlapping

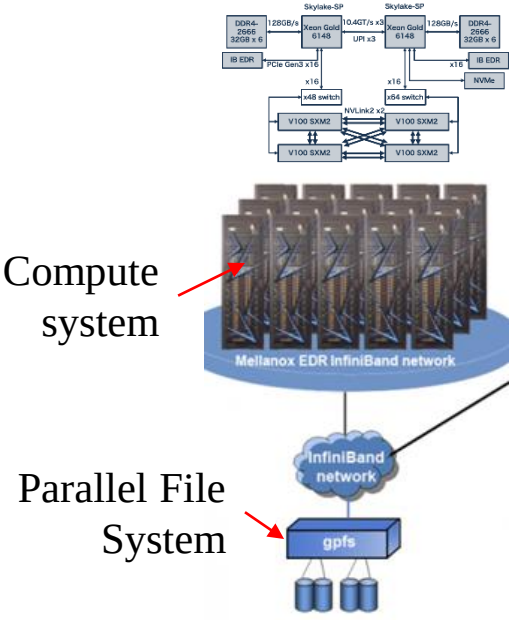
Question: How do the network architectures effect the per.
Goal: (2)
Topics: Network topology, inter-node link bandwidth

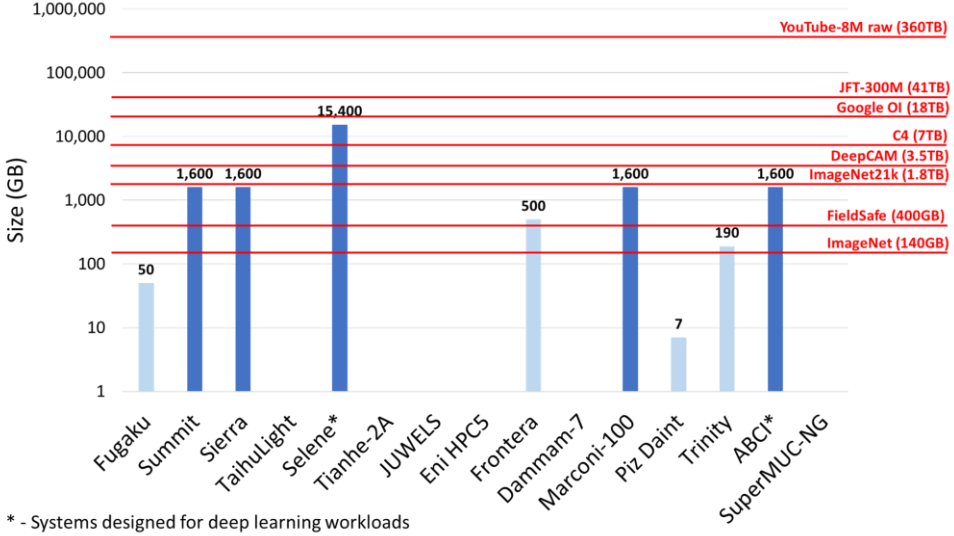
Bigger Dataset - I/O Challenges

- Random access with Global Shuffling each epoch
- Number of input files is large (millions - billions)
  - File size is also big, e.g., Cosmoflow
- Load the whole dataset into local storage before training

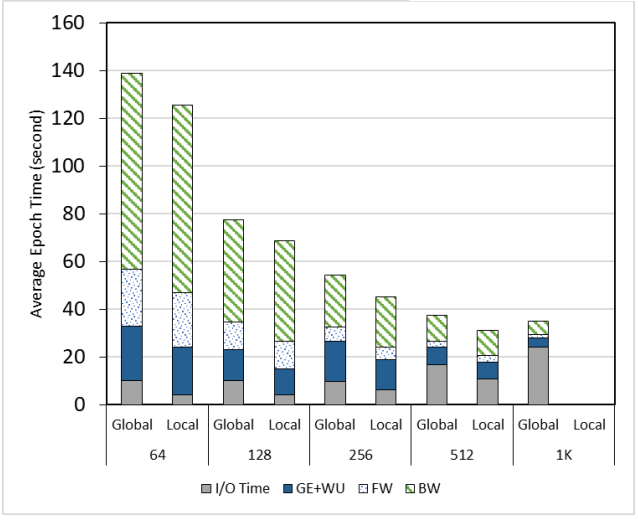
Problems: Dataset size is bigger than local storage (e.g., SSD)

- Bad performance with PFS
- Related work: Local Shuffling





* - Systems designed for deep learning workloads



High Performance Communication

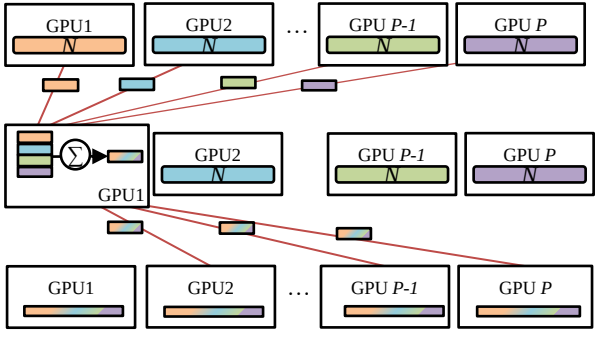
Targeted Communication Model

- Synchronous: communication start when the computation are finished at all the computing nodes.

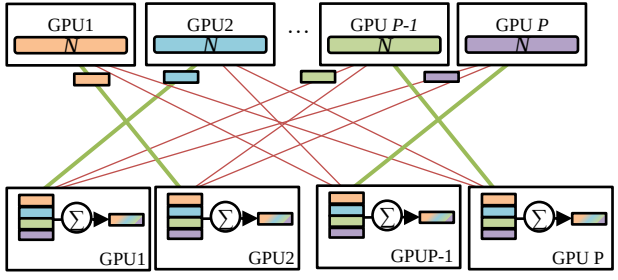
1. Reduce G_i to Parameter Server

$$G = \sum_{i=1}^{P} Gi$$

2. Broadcast the global update G



Parameter Server model
(Reduce + Broadcast)



Distributed model (our target)
(Allreduce)

Distributed Communication becomes practical

Limitation of Data Parallelism in Future

Recent well-known application

RESNET50 (~**60M** param.)

Inceptionv3 (~**40M** param.)

Larger message size

New domain's application

RESNET1001 (> **x0M** param.)

AmoebaNet @Google (~ **600M** param.)

Megatron-ML (**8.3B** param.), Turing-NLG (**17B** param.), GPT-3 (**175B**) from Natural Language Processing

Larger scale (number of GPUs)

From 10s GPUs to **1000s** GPUs

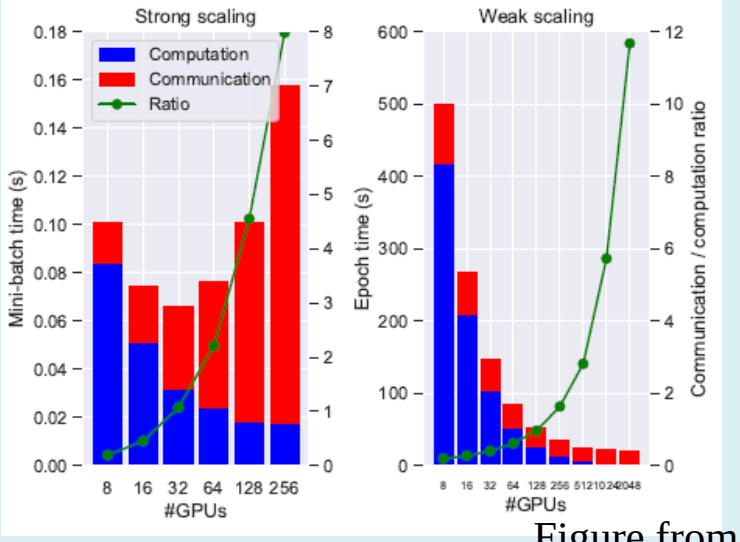
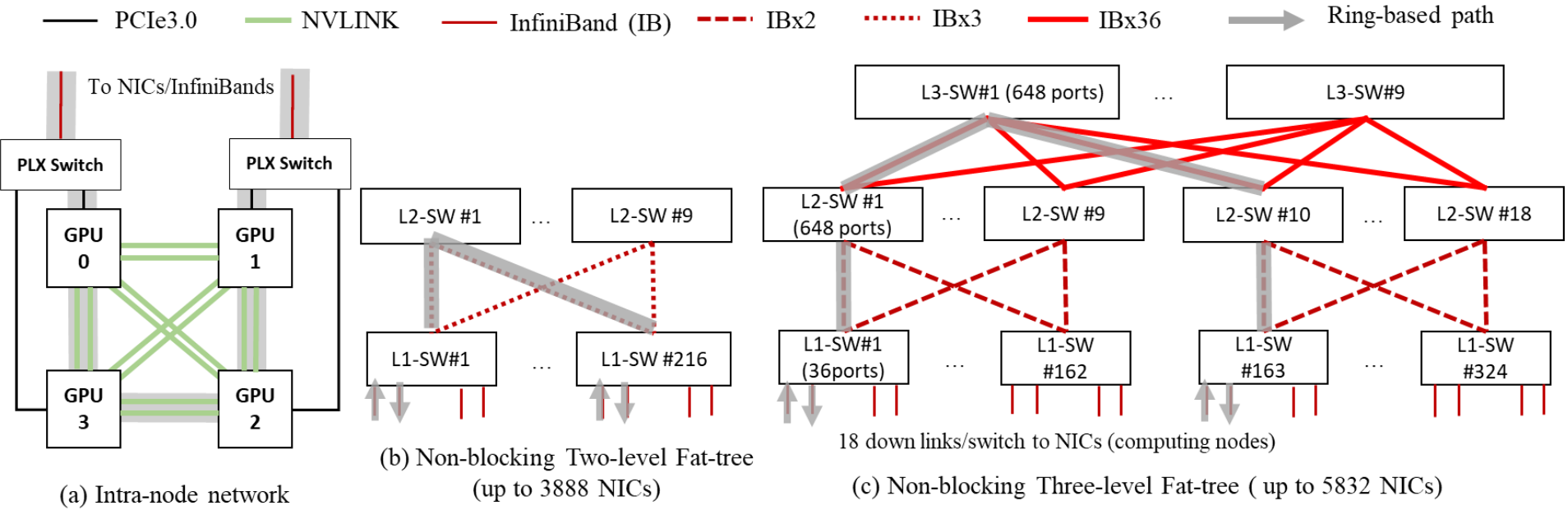


Figure from [*]

Communication becomes bottleneck

[*] Dryden, Nikoli, et al. "Aluminum: An asynchronous, GPU-aware communication library optimized for large-scale training of deep neural networks on HPC systems", MLHPC 2018.

Large-Scale Distributed Deep Learning



(a) Intra-node network

(b) Non-blocking Two-level Fat-tree (up to 3888 NICs)

(c) Non-blocking Three-level Fat-tree (up to 5832 NICs)

Latency factor

$$2(P-1)\,\alpha_{inter}$$

Bandwidth factor

$$\frac{2(P-1)N}{P}\,\beta_{inter}$$

Network	Latency factor (α)	Bandwidth factor (β)
IB switch and links	90 – 400 ns per switch	8×10^{-11} **(12.5 GB/s)**
PLX switch + PCIe	110 ns per switch	6.25×10^{-11} (16 GB/s)
NVLINK	~ 9 us	2×10^{-11} (50 GB/s)

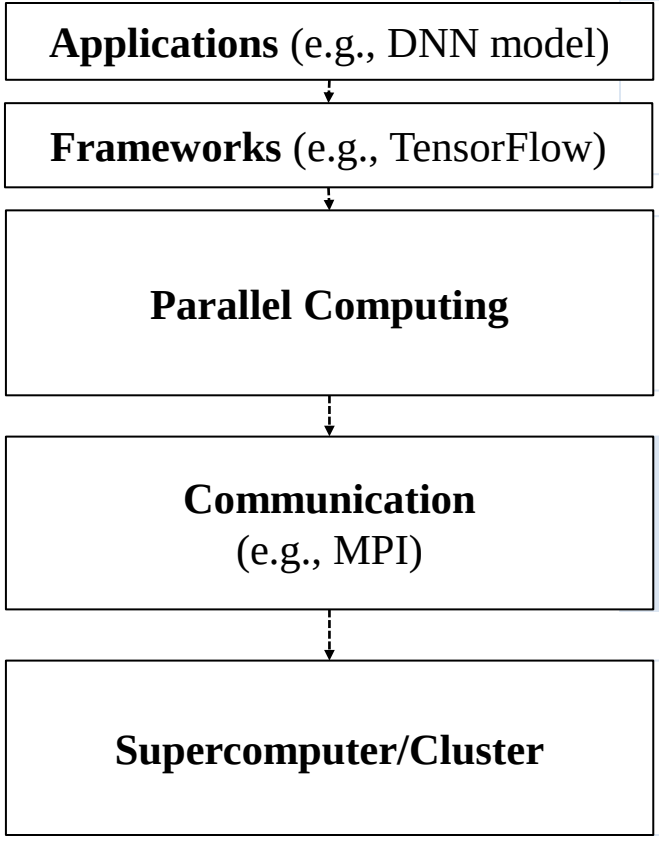
Inter-node communication becomes bottlenecks

Approach 1: Communication Algorithm

Goal: enable training DNNs on **large-scale** HPC system in the magnitude of hours.

Technical Goal: (1) Less Memory (2) Faster (3) Accuracy

Stack for Distributed DNN Training



Question: Design high-performance optimization/ method
Goal: (2) (3)
Topics: Shuffling, 2nd order method, pruning model

Question: What is the best parallelization strategy?
Goal: (1) (2) (3)
Topics: Data parallelism ➔ New parallelism approach

Question: How to reduce the communication time?
Goal: (2) (3)
Topics: Topology-aware, data compression, overlapping

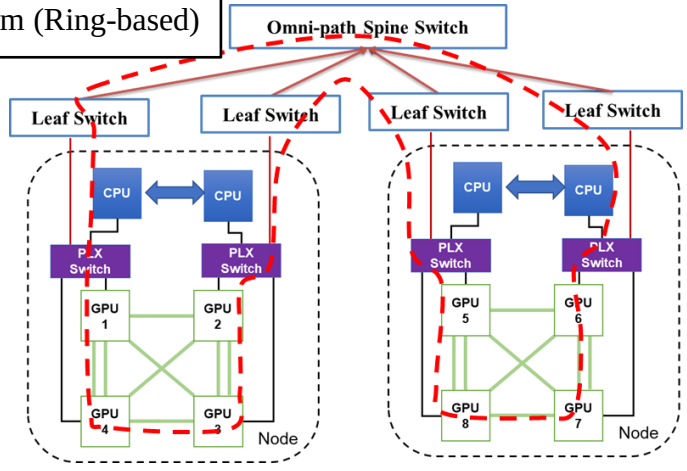
Question: How do the network architectures effect the per.
Goal: (2)
Topics: Network topology, inter-node link bandwidth

Optimizing Message Passing Algorithms

Problem: High communication time due to (1) large message size and (2) larger scale

Task: Study the method to reduce the communication time while maintain the accuracy

Bandwidth-Optimal Algorithm (Ring-based)

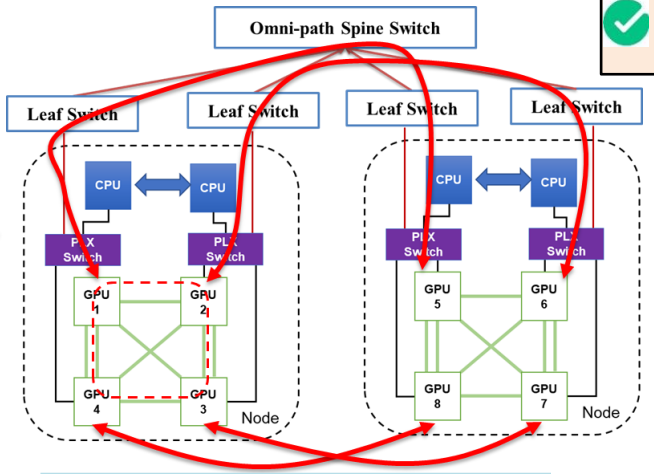


Latency factor

$$2(P-1)\,\alpha_{inter}$$

Bandwidth factor

$$\frac{2(P-1)N}{P}\beta_{inter}$$



Latency factor

$$2(k-1)\alpha_{intra}+2(\frac{P}{k}-1)\alpha_{inter}$$

Bandwidth factor

$$\frac{2(k-1)}{k}N\beta_{intra}+\frac{2(P-k)}{P}\frac{N}{k}\beta_{inter}$$

Architecture-aware algorithms

Our proposal

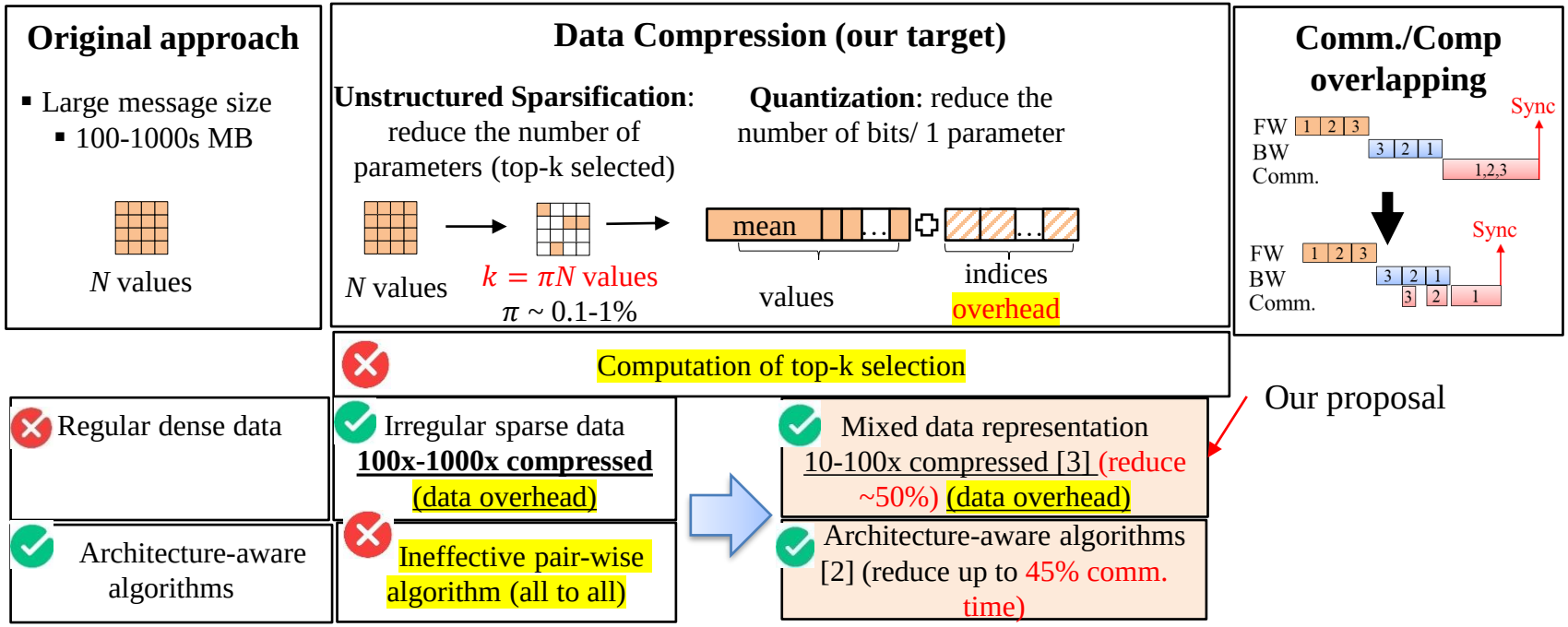
[2] N. T. Truong, M. Wahib, and R. Takano, "Efficient MPI-Allreduce for Large-Scale Deep Learning on GPU-Clusters", in Concurrency and Computation Practice and Experience, Special Issue on Parallel and Distributed Computing and Networking (CCPE)

Data Compression

Sub-issues of sparcification:

(1) Ineffective pair-wise communication algorithm

(2) Computation overhead of top-k select algorithm



[3] N.T. Truong, "Topology-aware Sparse Allreduce for Large-scale Deep Learning", 38th IEEE International Performance Computing and Communications Conference (IPCCC), Oct 2019

Sparcification with Mixed Data Representation

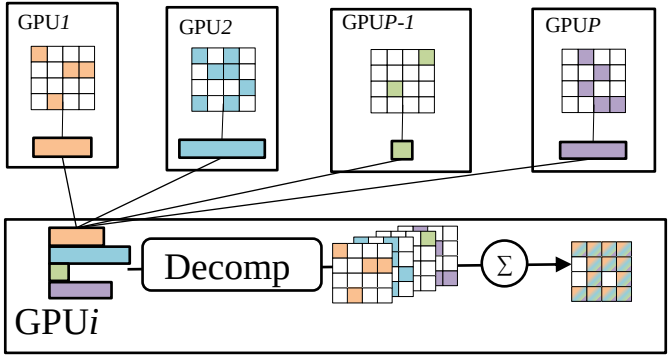
Target: **Reduce the communicated message size with Data Compression**

Idea: Compress the message

Communication Challenges: Unknown overlapping of the subsets between GPUs

Baseline

- Can not use ring-based Reduce
- Use pair-wise communication



Cost Send all data in *O(P)* steps

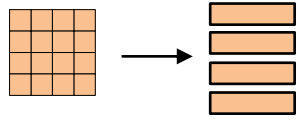
$$2(P-1)\,\alpha \quad + 2(P-1)N_2\beta$$

Proposed Method

- Dynamically switch between sparse data format and dense data format

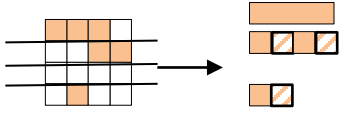
Dense data format

$N_1 = N \times se$



Sparse data format

$N_2 = N\pi\ (se + si)$



- Fundamental operations: SPLIT, SUM, COPY

SMALL EXTRA COMPUTATION γ (<10%)

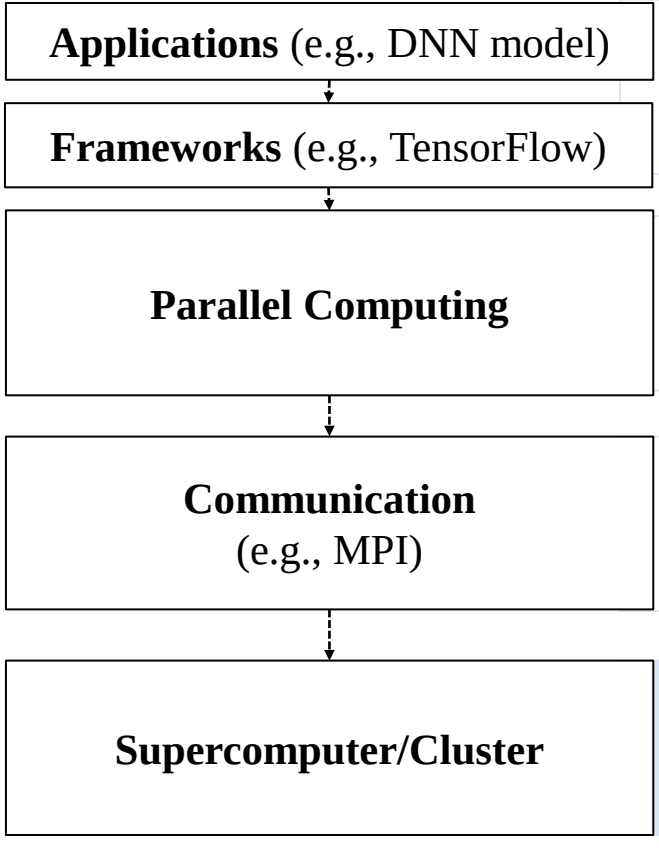
$$\leq 2(P-1)\,\alpha \quad + \frac{2(P-1)N_1}{P}\beta + \gamma$$

Approach 2: Algorithm-Architecture Codesign

Goal: enable training DNNs on **large-scale** HPC system in the magnitude of hours.

Technical Goal: (1) Less Memory (2) Faster (3) Accuracy

Stack for Distributed DNN Training



Question: Design high-performance optimization/ method
Goal: (2) (3)
Topics: Shuffling, 2nd order method, pruning model

Question: What is the best parallelization strategy?
Goal: (1) (2) (3)
Topics: Data parallelism ➔ New parallelism approach

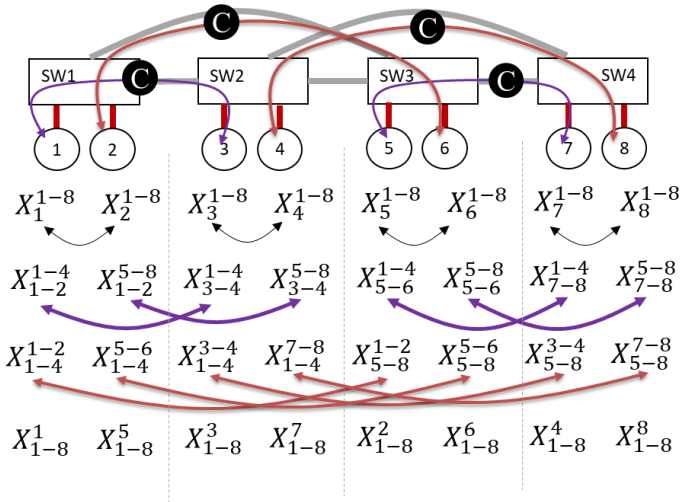
Question: How to reduce the communication time?
Goal: (2) (3)
Topics: Topology-aware, data compression, overlapping

Question: How do the network architectures effect the per.
Goal: (2)
Topics: Network topology, inter-node link bandwidth

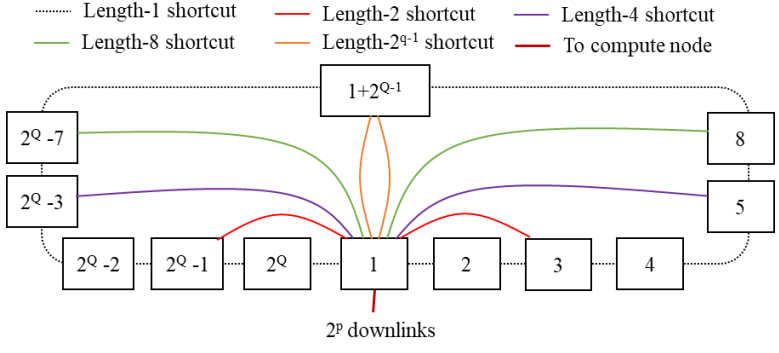
Network Congestion

Problem: Number of involved GPUs (P) increase

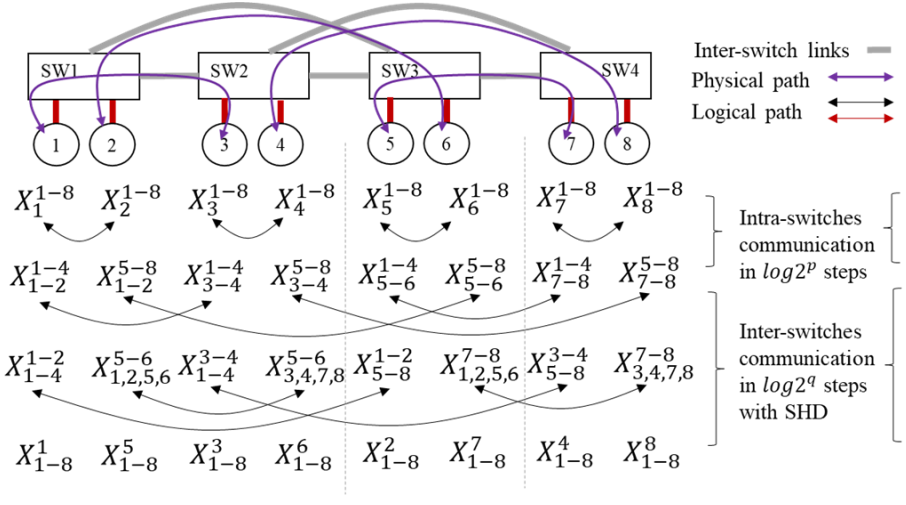
- Ring-based algorithm O(P)α: latency factor increase.
- Halving-Doubling algorithms O(log$P$)α: network congestion



Proposal: Distributed Loop Network topology



Shilfted Halving – Doubling Algorithm



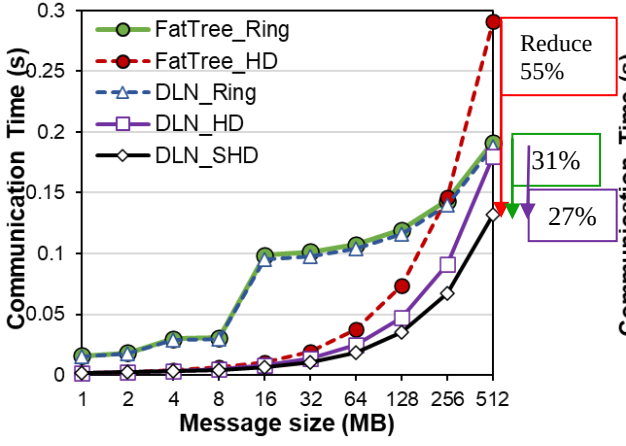
[4] **N.T. Truong,** et. al. “An Algorithm and System Co-design for Large-Scale Training of Distributed Deep Learning”, CCGRID 2021

Micro-benchmark Result

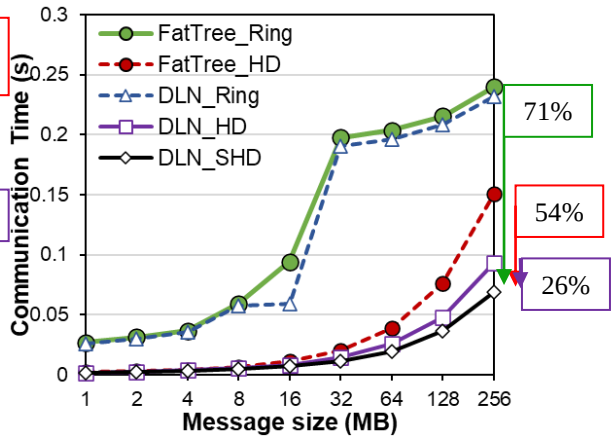
↓ Better

Simulation result on ABCI

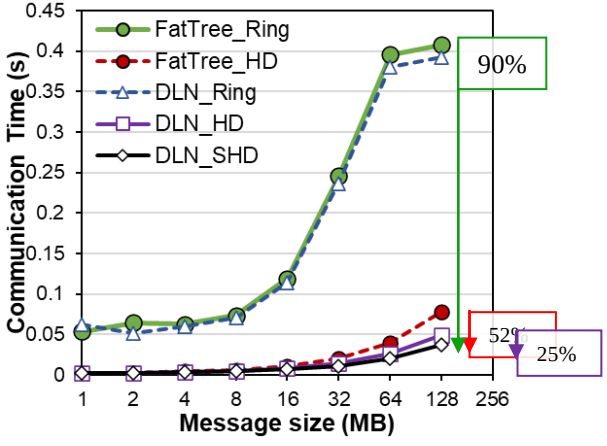
- ✓ DLN_Ring slightly better FatTree_Ring 2-5%
- ✓ DLN_SHD significantly better than FatTree_Ring: 31%, 71%, and 90%
- ✓ DLN_SHD better than FatTree_HD: ~ 50%
- ✓ DLN_SHD better than DLN_HD: ~ 25%



256 GPUs



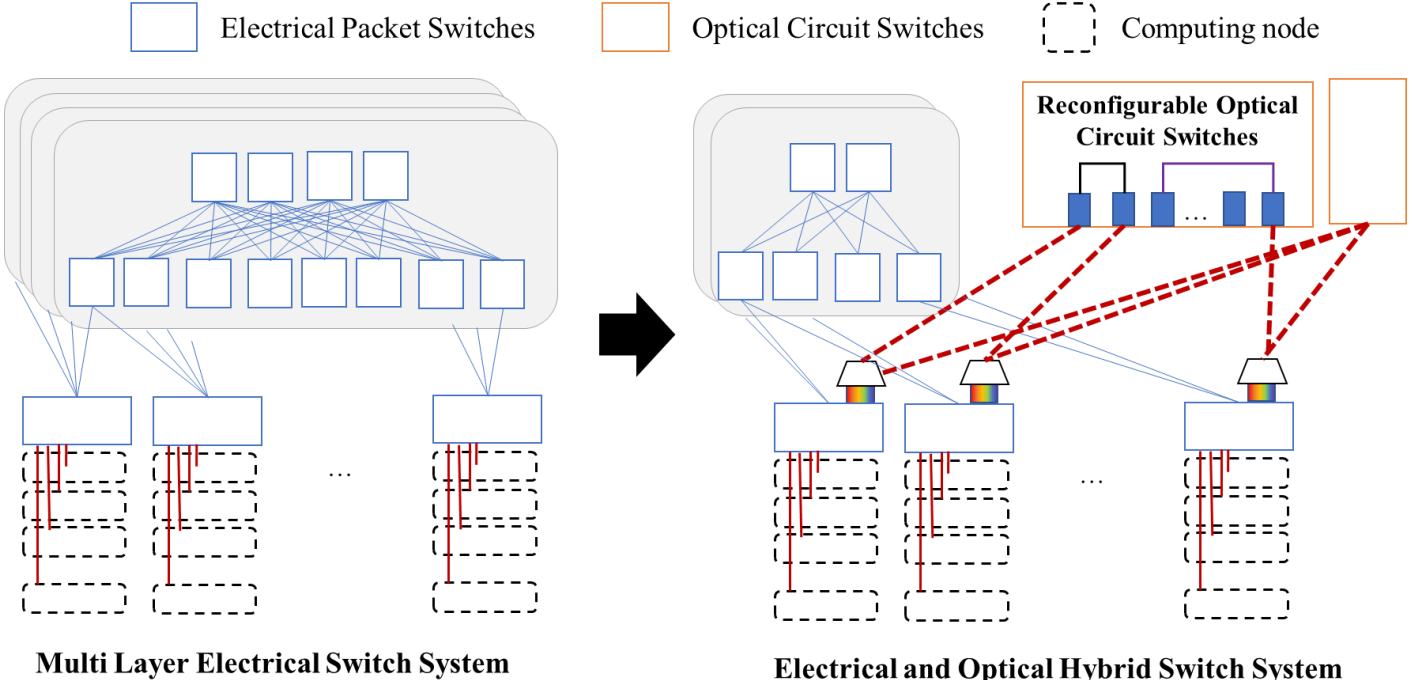
512 GPUs



1K GPUs

High-bandwidth Link: Hybrid Switching System for DL

- Small flows use EPS (electric packet switching)
- Big flows use OPS (optical circuit switching) such as data of DL
  - Logical-ring algorithm never change the pattern
  - No-need of reconfiguration



[5] **N.T. Truong,** et. al. "On the Feasibility of Hybrid Electrical/Optical Switch Architecture for Large-Scale Training of Distributed Deep Learning ",PHOTONICS, 2019

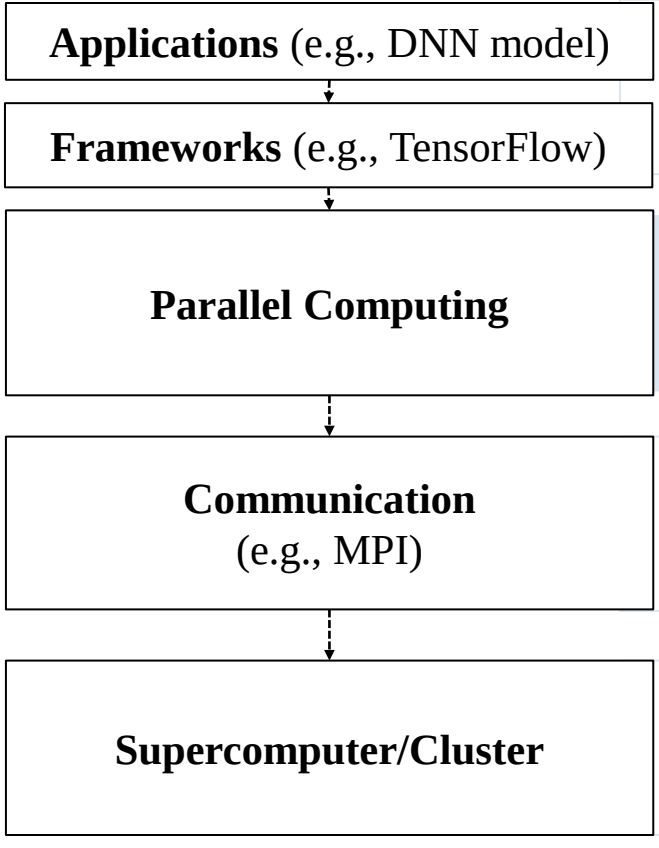
High Performance Parallelism Model

HOW - Research Goal and Approach

Goal: enable training DNNs on **large-scale** HPC system in the magnitude of hours.

Technical Goal: (1) Less Memory (2) Faster (3) Accuracy

Stack for Distributed DNN Training



Question: Design high-performance optimization/ method
Goal: (2) (3)
Topics: Shuffling, 2nd order method, pruning model

Question: What is the best parallelization strategy?
Goal: (1) (2) (3)
Topics: Data parallelism ➔ New parallelism approach

Question: How to reduce the communication time?
Goal: (2) (3)
Topics: Topology-aware, data compression, overlapping

Question: How do the network architectures effect the per.
Goal: (2)
Topics: Network topology, inter-node link bandwidth

Memory Really a Limiting Factor?

- Yes!
- Examples:
  - Larger models:
    - Megatron-LM: 8.3B parameters **O(10) GBs**
    - ZeRO-2: 170B parameters **O(100) GBs**
  - Larger Samples: Common in scientific datasets
    - CosmoFlow: ~2GB/sample **O(100) GBs for act. 1st layer**
  - When Increasing mini-batch size
    - Can go up to 127K with methods such as K-FAC

Solutions?

Use More Memory/ Node or GPUs!

In-efficient use of resources

Out-Of-Core Methods

- More sensitive to co-located jobs
- Tricky when no local DRAM/SSD

-ve

- Simplicity of data parallelism
- No changes to model/framework
- No messing with Batch Norm (BN)

+ve

Explicit Model-parallelism

- Constructing a cost model that is accurate
- High implementation cost

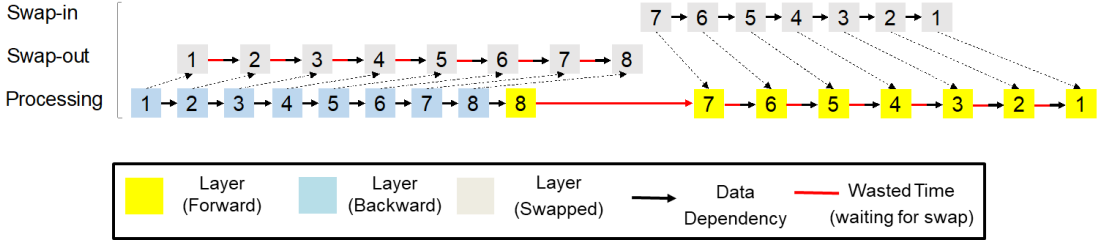
-ve

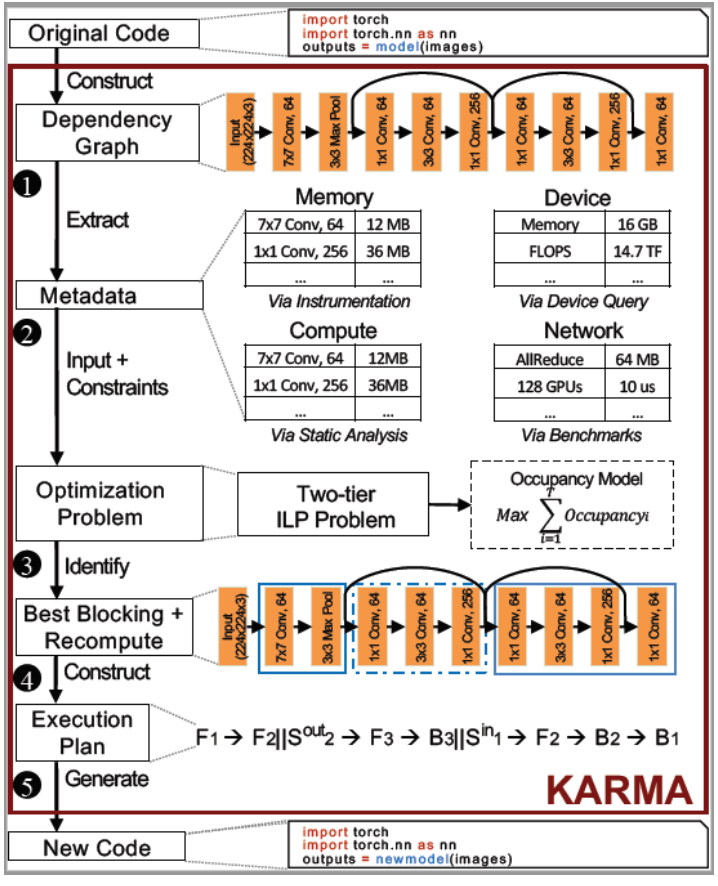
- Analytical: deterministic
- Low overhead

+ve

Out-of-core to Overcome Memory Limitation

- **Problem**: Model size bigger than GPU's memory
  - 100s TB as in GPT-3 vs. 32GB HBM2
- **Proposal**: OOC in DNN is intuitively swapping layer
  - Move finished layer to CPU memory
  - Redundantly recompute in backward pass





[6] Mohamed Wahib, Haoyu Zhang, **Truong Thao Nguyen**, Aleksandr Drozd, Jens Domke, Lingqi Zhang, Ryousei Takano, Satoshi Matsuoka,"Scaling Distributed Deep Learning Workloads beyond the Memory Capacity with KARMA", Supercomputing 2020

Solutions (2)

Use More Memory/ Node or GPUs!

In-efficient use of resources

Out-Of-Core Methods

- More sensitive to co-located jobs
- Tricky when no local DRAM/SSD

-ve

- Simplicity of data parallelism
- No changes to model/framework
- No messing with Batch Norm (BN)

+ve

Explicit Model-parallelism

- Constructing a cost model that is accurate
- High implementation cost

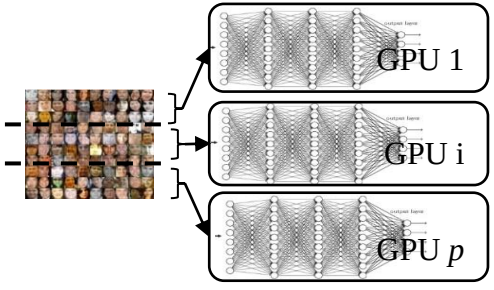
-ve

- Analytical: deterministic
- Low overhead

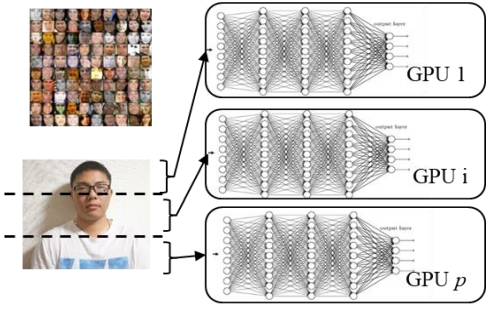
+ve

Strategies for Distributed Training

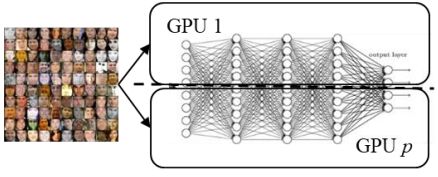
✓ Formally defines possible pure parallelism strategies for Distributed Deep Learning:
- Data, Spatial, Layer, Channel, Filter, Hybrid (combine of data with others)



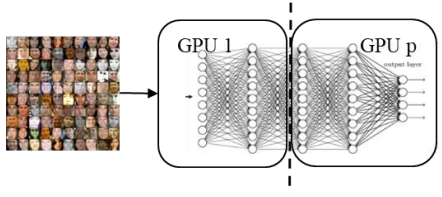
Data parallelism
split the dataset (input)



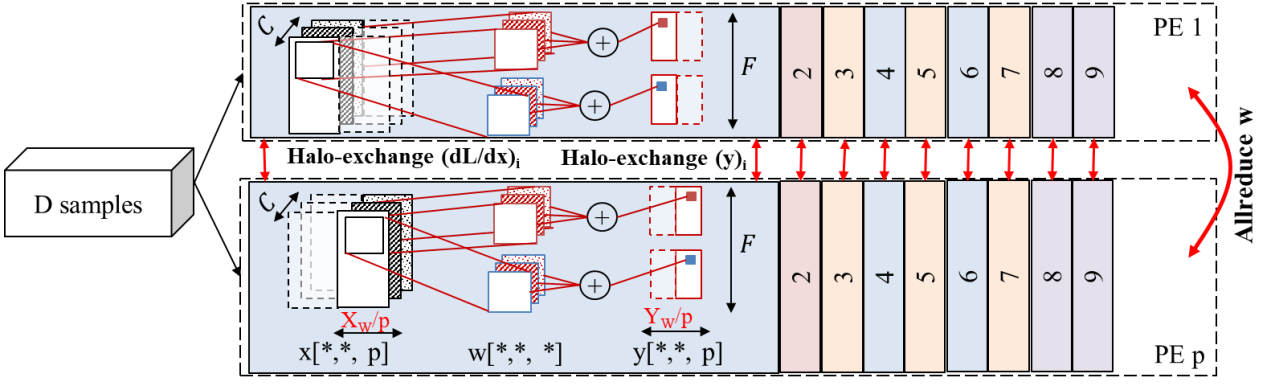
Spatial parallelism
split a sample (input)



Channel/Filter parallelism:
vertically split the DNN model

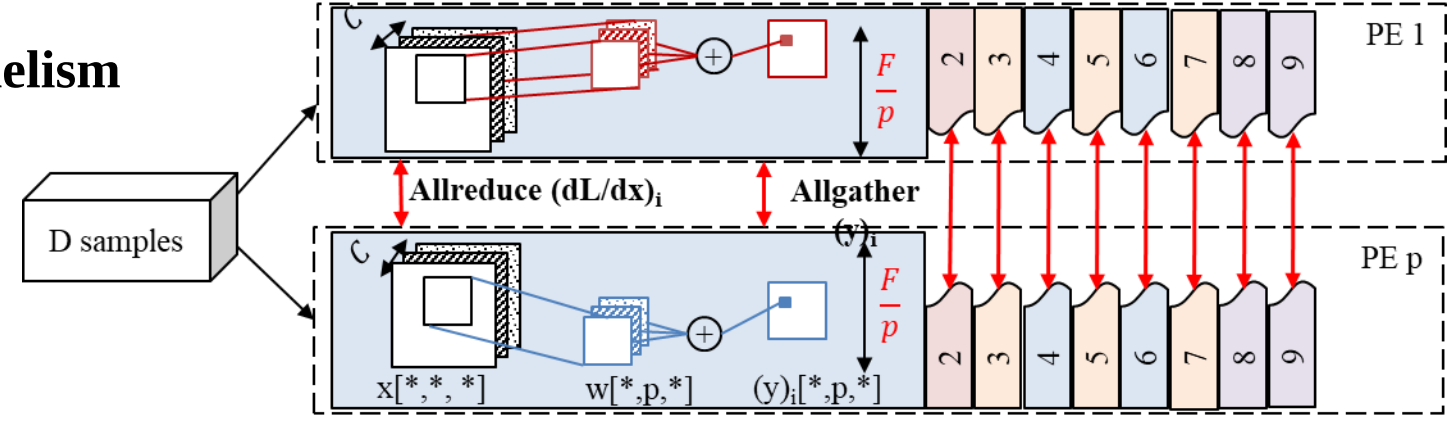


Layer (Pipe) parallelism:
horizontally split the DNN model

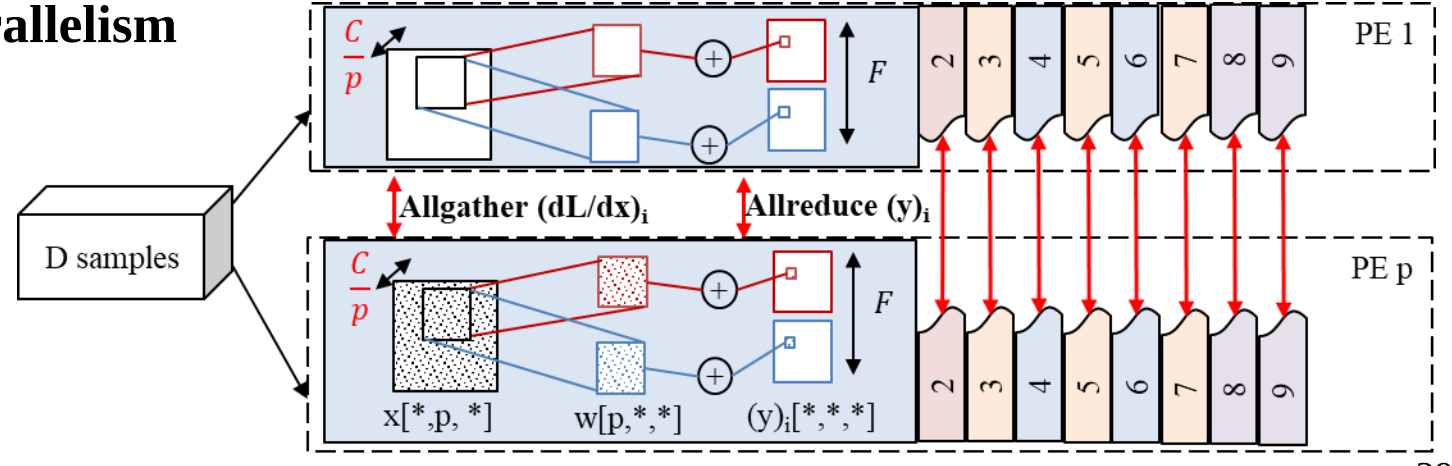


Model Parallelism for CNN

Filter Parallelism

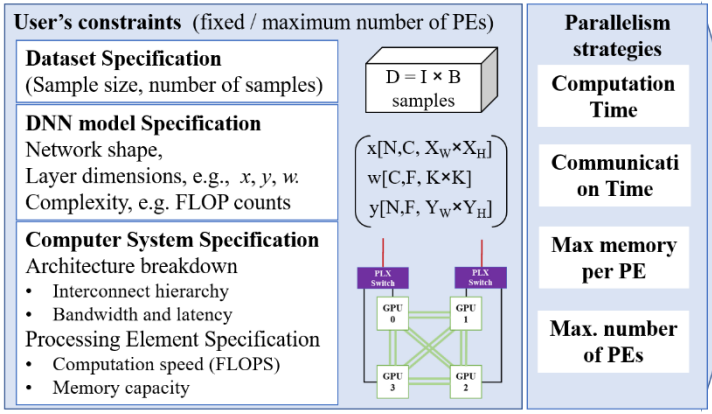


Channel Parallelism



Performance and Memory Projection

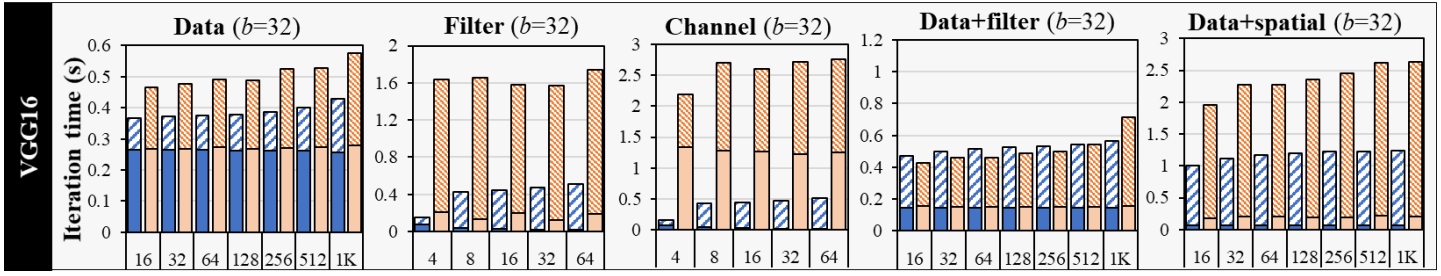
- ✓ Proposed an initial analysis/estimation model
  - Support many CNN, transformer (GEMM)
  - Based on an **Ideal Parameterization**



	Computation Time T_{comp}	Communication Time T_{comm}	Maximum Memory Per PE	Number of PEs p
Serial	$D\sum_{l=1}^{G}\left(FW_l+BW_l\right)+\frac{D}{B}\sum_{l=1}^{G}(WU_l)$	0	$2\gamma\delta\sum_{l=1}^{G}\left(B(\lvert x_l\rvert+\lvert y_l\rvert)+\lvert w_l\rvert\right)$	$p=1$
Data	$\frac{D}{p}\sum_{l=1}^{G}(FW_l+BW_l)+\frac{D}{B}\sum_{l=1}^{G}(WU_l)$	$2\frac{D}{B}(p-1)\left(\alpha+\frac{\sum_{l=1}^{G}\lvert w_l\rvert}{p}\delta\beta\right)$	$2\gamma\delta\sum_{l=1}^{G}\left(\frac{B}{p}(\lvert x_l\rvert+\lvert y_l\rvert)+\lvert w_l\rvert\right)$	$p\leq B$
Spatial	$\frac{D}{p}\sum_{l=1}^{G}\left(FW_l+BW_l\right)+\frac{D}{B}\sum_{l=1}^{G}(WU_l)$	$2\frac{D}{B}\Big((p-1)(\alpha+\frac{\sum_{l=1}^{G}\lvert w_l\rvert}{p}\delta\beta)+\sum_{l=1}^{G}\left(2\alpha+B(\text{halo}(\lvert x_l\rvert)+\text{halo}(\lvert\frac{dL}{dy_l}\rvert))\delta\beta\right)\Big)$	$2\gamma\delta\sum_{l=1}^{G}\left(B\frac{(\lvert x_l\rvert+\lvert y_l\rvert)}{p}+\lvert w_l\rvert\right)$	$p=pw\times ph\leq\min_{l=1}^{G}(W_l\times H_l)$
Layer (Pipeline)	$\frac{D(p+S-1)}{S}\Big(\max_{i=1}^{p}(FW_{G_i})+\max_{i=1}^{p}(BW_{G_i})\Big)+\max_{i=1}^{p}(WU_{G_i})\Big)$	$2\frac{D(p+S-2)}{B}\left(\max_{i=1}^{p-1}\left(\alpha+\frac{B}{S}\lvert y_{G_i}\rvert\delta\beta\right)\right)$	$2\gamma\delta\max_{i=1}^{p}\left(\sum_{l=1}^{G_i}\left(B(\lvert x_l\rvert+\lvert y_l\rvert)+\lvert w_l\rvert\right)\right)$	$p\leq G$
Filter	$\frac{D}{p}\sum_{l=1}^{G}\left(FW_l+BW_l\right)+\frac{D}{Bp}\sum_{l=1}^{G}(WU_l)$	$3\frac{D}{B}(p-1)\sum_{l=1}^{G-1}(\alpha+\frac{B\lvert y_l\rvert}{p}\delta\beta)$	$2\gamma\delta\sum_{l=1}^{G}\left(B(\lvert x_l\rvert+\lvert y_l\rvert)+\frac{\lvert w_l\rvert}{p}\right)$	$p\leq\min_{l=1}^{G}(F_l)$
Channel	$\frac{D}{p}\sum_{l=1}^{G}\left(FW_l+BW_l\right)+\frac{D}{Bp}\sum_{l=1}^{G}(WU_l)$	$3\frac{D}{B}(p-1)\sum_{l=1}^{G-1}(\alpha+\frac{B\lvert y_l\rvert}{p}\delta\beta)$	$2\gamma\delta\sum_{l=1}^{G}\left(B(\lvert x_l\rvert+\lvert y_l\rvert)+\frac{\lvert w_l\rvert}{p}\right)$	$p\leq\min_{l=1}^{G}(C_l)$
Data + Filter	$\frac{D}{p}\sum_{l=1}^{G}\left(FW_l+BW_l\right)+\frac{D}{Bp2}\sum_{l=1}^{G}(WU_l)$	$3\frac{D}{B}(p2-1)\sum_{l=1}^{G-1}(\alpha+\frac{B\lvert y_l\rvert}{p}\delta\beta)+2\frac{D}{B}(p1-1)(\alpha+\frac{\sum_{l=1}^{G}\lvert w_l\rvert}{p}\delta\beta)$	$2\gamma\delta\sum_{l=1}^{G}\left(\frac{B(\lvert x_l\rvert+\lvert y_l\rvert)}{p1}+\frac{\lvert w_l\rvert}{p2}\right)$	$p=p1\times p2\leq B\times\min_{l=1}^{G}(F_l)$

[7] **Nguyen T. Truong,** et. al. ""An Oracle for Guiding Large-Scale Model/Hybrid Training of Convolutional Neural Networks", HPDC 2021

Improving the Estimation Accuracy?



Estimation accuracy is not good (average ~60%), especially in communication

Different latency α and bandwidth factors β → **Empirical Parameterization**

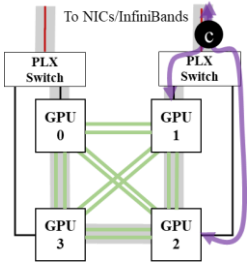
- Computation parameters (FW_l, BW_l, and WU_l)
- Communication parameter (α and β): use NCCL-test, OSU benchmarks

Network contention → **Self-contention modeling**

- Introduce contention penalty coefficient ϕ,
- Using dynamic contention graphs [1] to detect contention

Network congestion→ **Detach it in the empirical result**

- Report minimum communication time
- Congestion impact factor using benchmark [2]



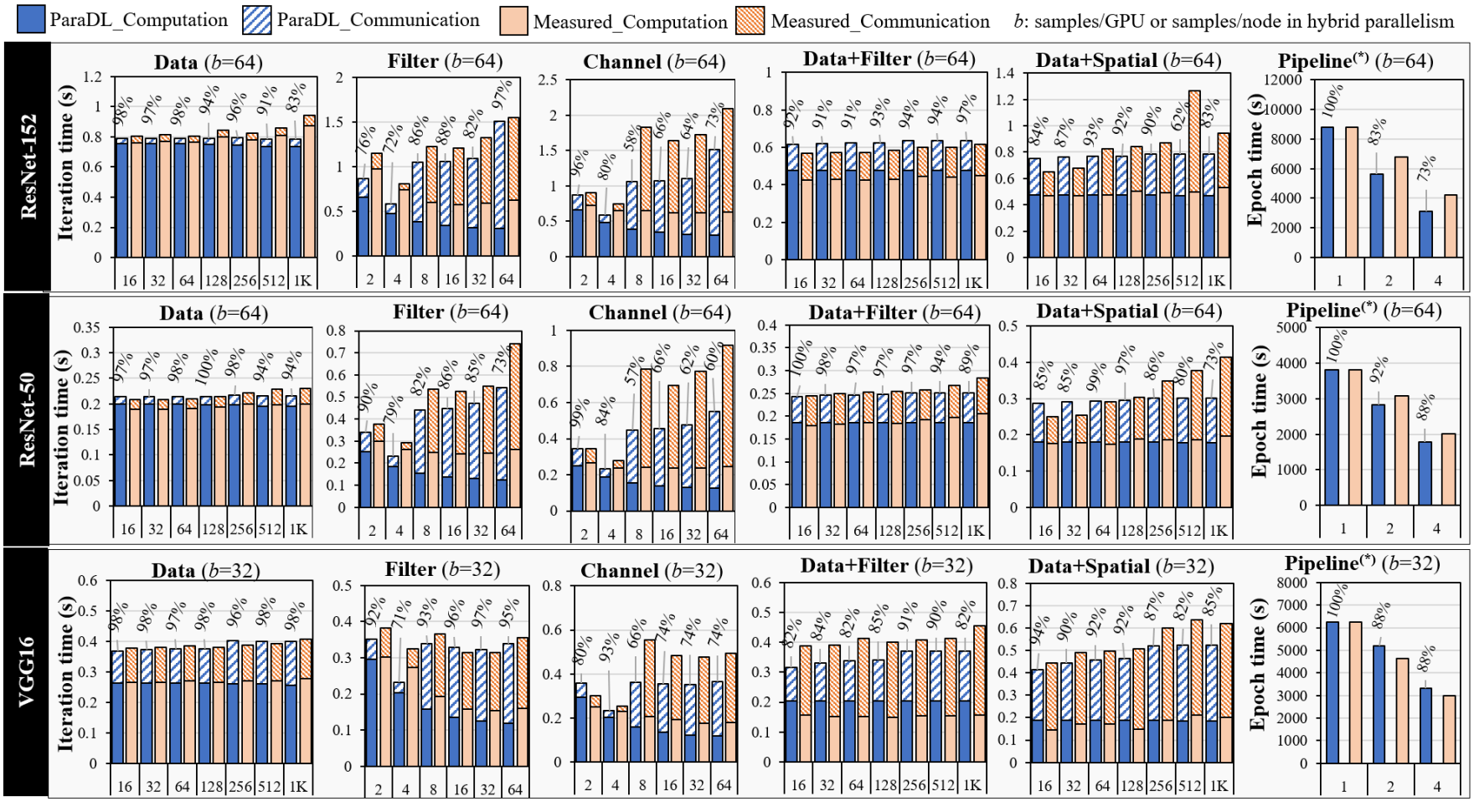
Self-contention in hybrid parallelism with $\phi = 2$

[1] Maxim Martinasso et al. 2011. A ``Contention-Aware Performance Model for HPC-Based Networks: A Case Study of the InfiniBand Network. In Euro-Par 2011 Parallel Processing."

[2] Sudheer Chunduri et al. 2019. GPCNeT: Designing a Benchmark Suite for Inducing and Measuring Contention in HPC Networks (SC '19)

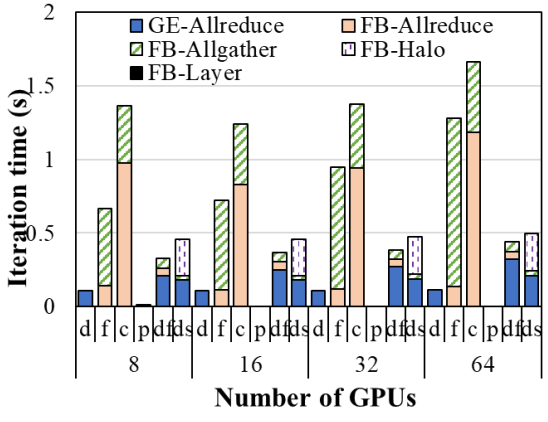
[**HPDC**] **Truong Thao Nguyen**, et al. "An Oracle for Characterizing and Guiding Large-Scale Training of Deep Neural Networks," ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC2021) (**to be appeared**)

Final Projection Result

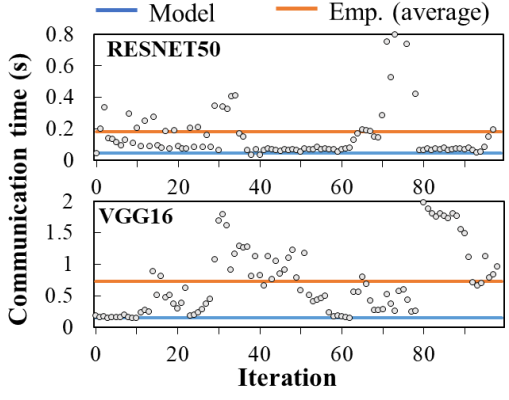


Limits/Bottlenecks of Communication

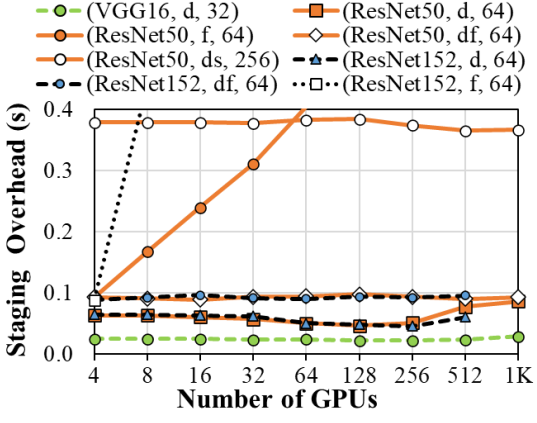
Different communication patterns of parallelism strategies



Network congestion of inter-rack links is the bottle-neck



I/O and stagging overhead become non-trivial for model parallelisms



Optimizing message passing algorithms for Deep Learning [1,2]

New network architecture that helps to reduce congestion [3]

New I/O method: Decentralized storing dataset

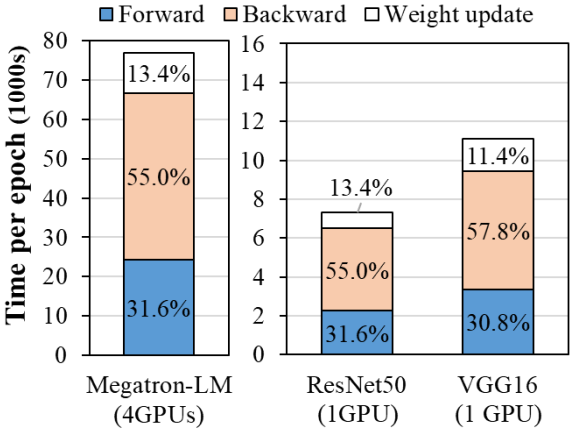
[2] N. T. Truong, M. Wahib, and R. Takano, "Efficient MPI-Allreduce for Large-Scale Deep Learning on GPU-Clusters", in Concurrency and Computation Practice and Experience, Special Issue on Parallel and Distributed Computing and Networking (CCPE)

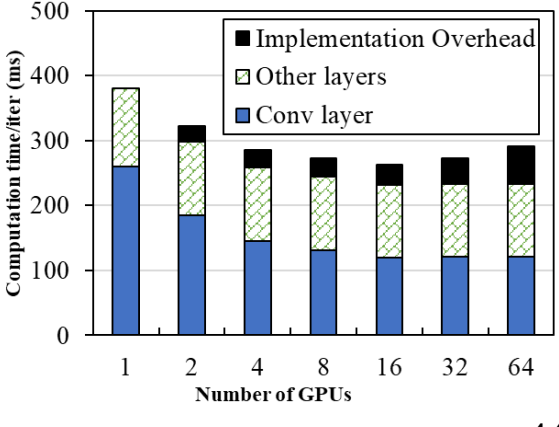
[3]N.T. Truong, "Topology-aware Sparse Allreduce for Large-scale Deep Learning", 38th IEEE International Performance Computing and Communications Conference (IPCCC), Oct 2019

[4] N.T. Truong, et. al. "An Algorithm and System Co-design for Large-Scale Training of Distributed Deep Learning", submitted to IEEE-CCGRID 2021

Limits/Bottlenecks of Computation

- Weight update
  - Weight update forms a significant portion of compute time
- Workload balancing
  - Pipeline can outperform data parallelism because of P2P rather than collective.
  - To achieve this, all stages must balance
- Computation redundancy
  - Mostly caused by frameworks
  - Convolutional layer doesn't scale as expected
  - Computation overhead in split/concat

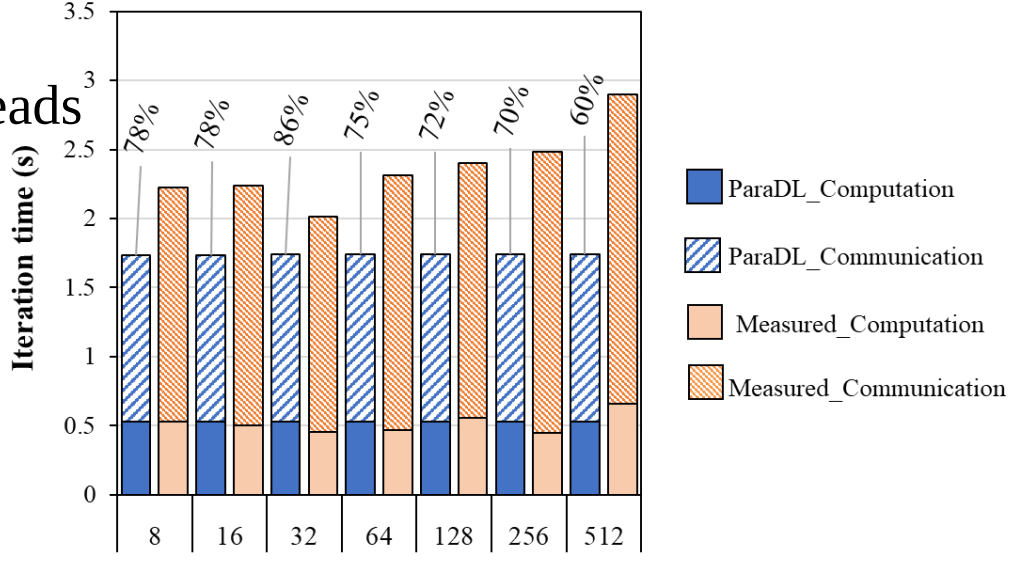




Memory Capacity Limits/Bottlenecks

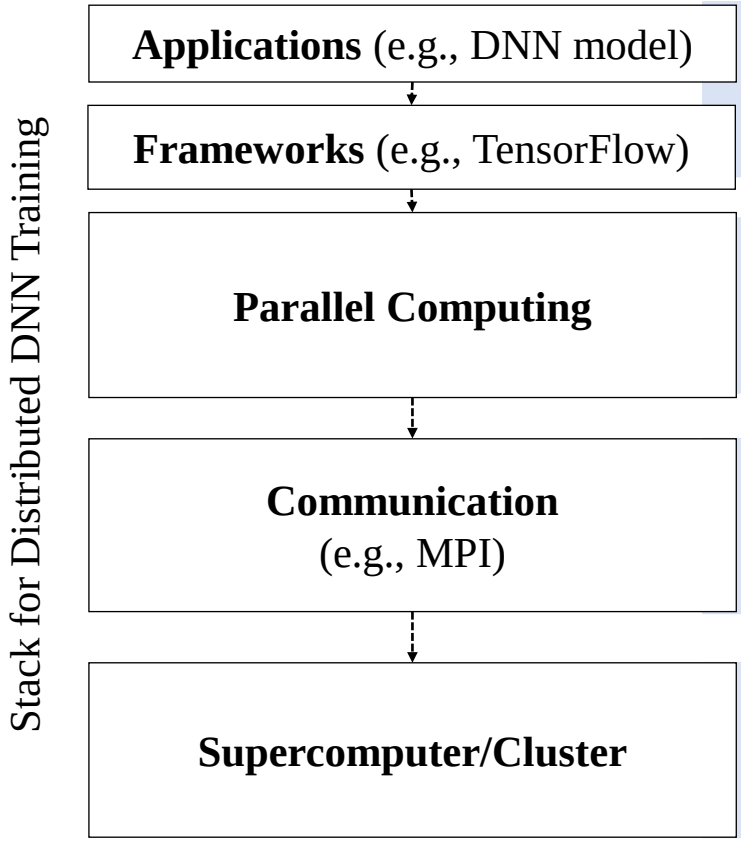
- Redundancy in Memory
  - Activations are distributed but weights are not

- Memory Manager
  - Kernels waiting for memory leads to performance degradation (Cosmoflow – data+spatial)



Collaboration

Research Interest



Question: Design high-performance optimization/ method
Goal: 2 3
Topics: **Pruning model**

Question: What is the best parallelization strategy?
Goal: 1 2 3
Topics: **Fine-grained model parallelism (GNN, Trans.)**

Question: How to reduce the communication time?
Goal: 2 3
Topics: **Data compression**

Question: How do the network architectures effect the per.
Goal: 2
Topics: **FPGA network / unidirectional network**

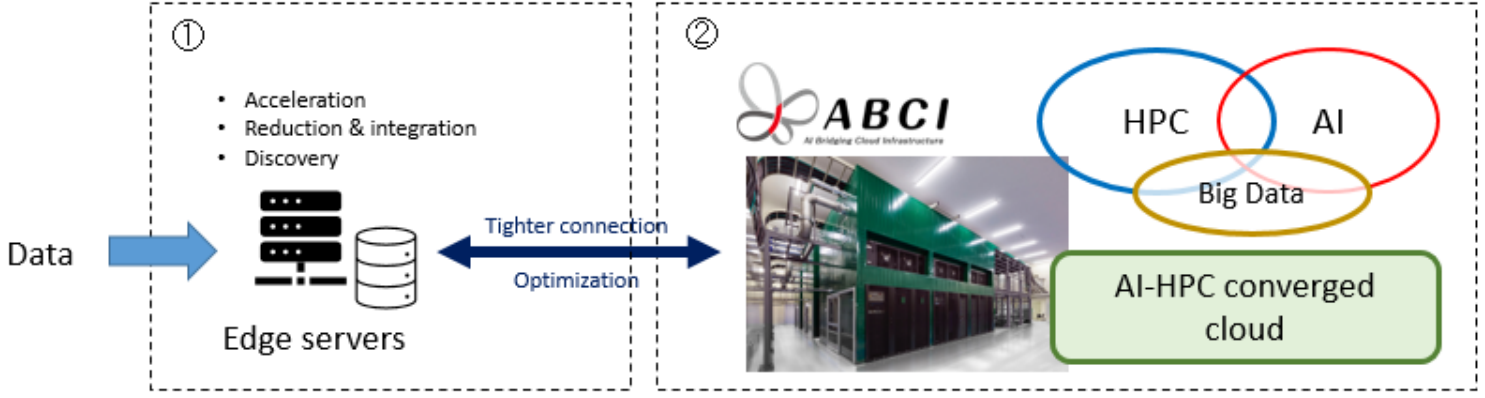
Team Introduction

We do middleware research for edge-to-cloud services and infrastructure (DigiARC calls as Continuum Computing core), specially we fucus on

① Acceleration technologies at edge, for advanced and high-performance CC-applications

② Cloud technologies, which can provide huge AI computing and data processing powers with end services, through tighter integration of edge infrastructure

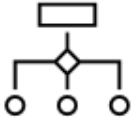
and contribute to the realization of various digital services considered in Society 5.0.



Research Topics (add or remove?)

Data management and use in CC

- Offloading and data reduction at edge for scalability and energy saving, data pipeline optimization
- DataOps automation, balanced with effective human interaction
- Data traceability in sharing and integration



Management of CC-application services

- Develop and build methods of high-performance and robust CC-applications for geo-distributed, unstable and heterogeneous environments
- Zero-touch service deployment, dynamic QoS control, offloading decision, mobility support
- Low-latency service-to-service interaction, autonomous service orchestration

High performance cloud technologies tightly connecting with edge

- Large-scale computing with accelerators
- High performance AI, AI resource hub
- Low-latency and scalable connection services with edge, efficient resource management

TAKEAWAYS

- Large-scale training, e.g., 100-1000GPUs is challenging
  - Inference on HPC system
- New problems when training with different (Dataset, DL Model and computer System)
  - Weight update
  - I/O Stagging
  - Network congestion
- The rise of hybrid parallelism (data + model parallelism)