Demystify Communication Behavior in Training Deep Learning Recommendation Models

Ching-Hsiang Chu
Research Scientist
Meta
chchu@fb.com

P2S2 Workshop Aug. 29, 2022

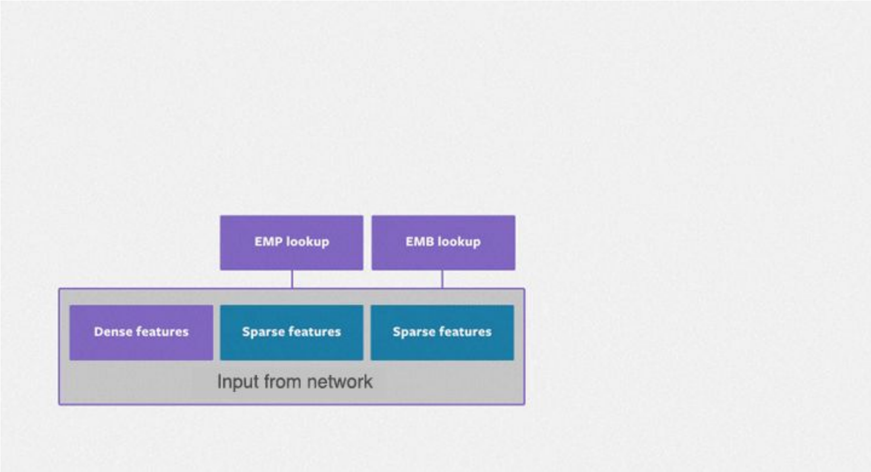
Meta

Agenda

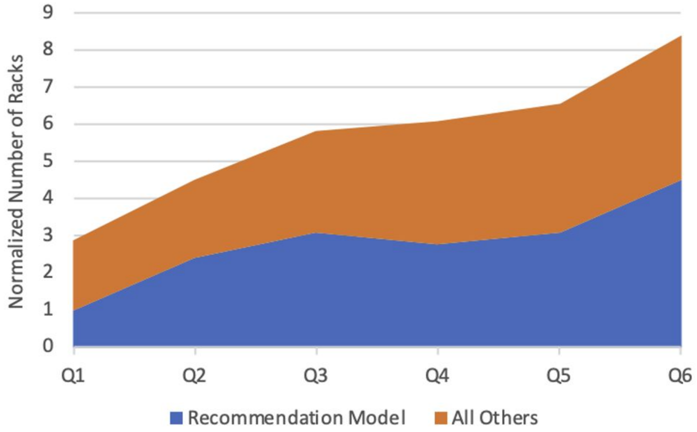
- **Introduction**
- Demystify Communication Behavior in Training DLRM
  - With a real-world example
- Communication Reproduction
- Summary

Deep learning recommendation models (DLRMs)

- DLRMs are used extensively in many companies for building recommendation systems
- MLPerf training and inference benchmarks (https://mlcommons.org/en/)



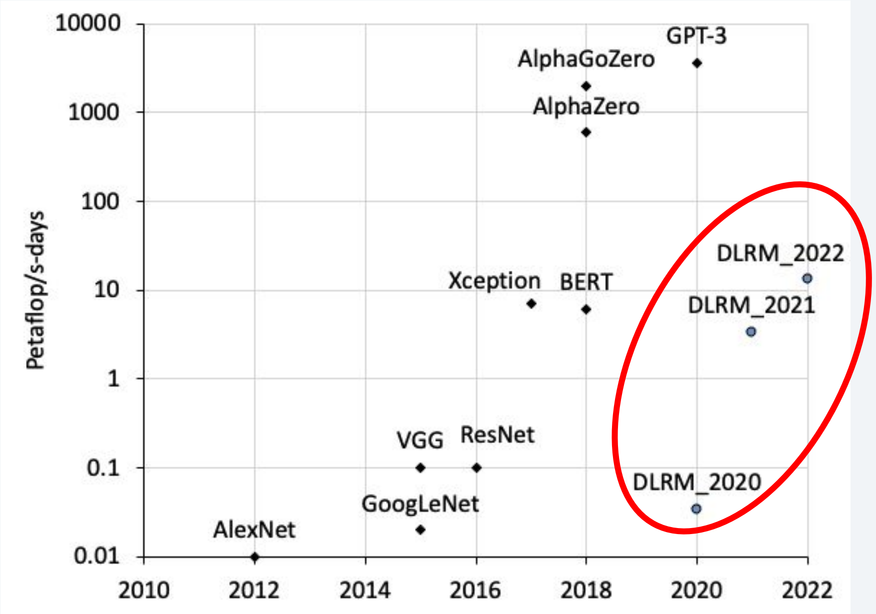
https://ai.facebook.com/blog/dlrm-an-advanced-open-source-deep-learning-recommendation-model/



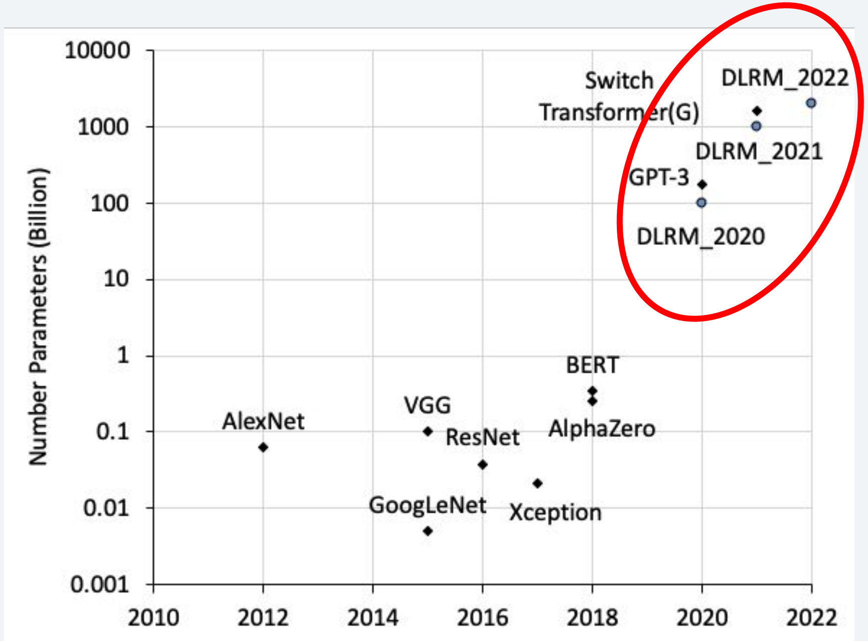
M. Naumov, J. Kim, D. Mudigere, et al. Deep Learning Training in Facebook Data Centers: Design of Scale-up and Scale-out Systems, ArXiv, 2020

Recommendation models are different !

- **Lower compute intensity**



- **Larger sizes**



Dheevatsa Mudigere et al., “Software-hardware co-design for fast and scalable training of deep learning recommendation models. “ ISCA '22

Model parallelism ↔ Data parallelism

→ **Alltoall communication (fwd + bwd)**

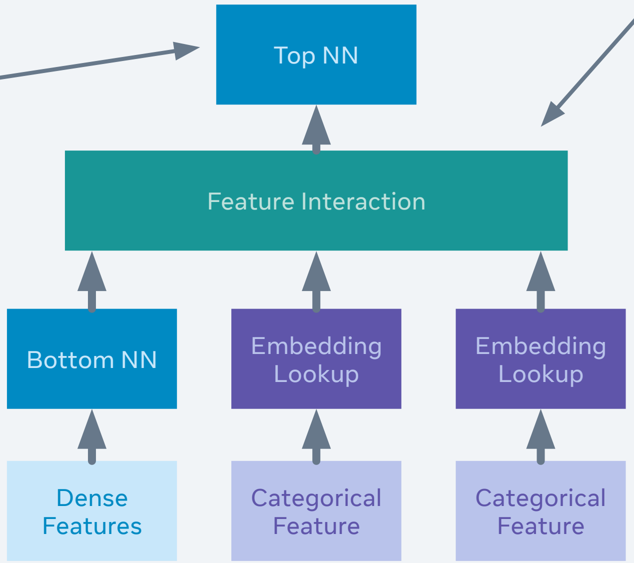
→ **Network bisection BW**

Dense compute (MLP, ResNet…)

→ **Compute-intensive**

→ **Data-parallel, AllReduce comms**

→ **Network injection BW**



1000s of embedding tables, multi-hot

Varying size - a few MBs to TBs

Trillions parameters !

→ **Mem capacity and BW intensity**

→ **Model-parallel**

10s PBs of data via network FS

High data ingestion rate

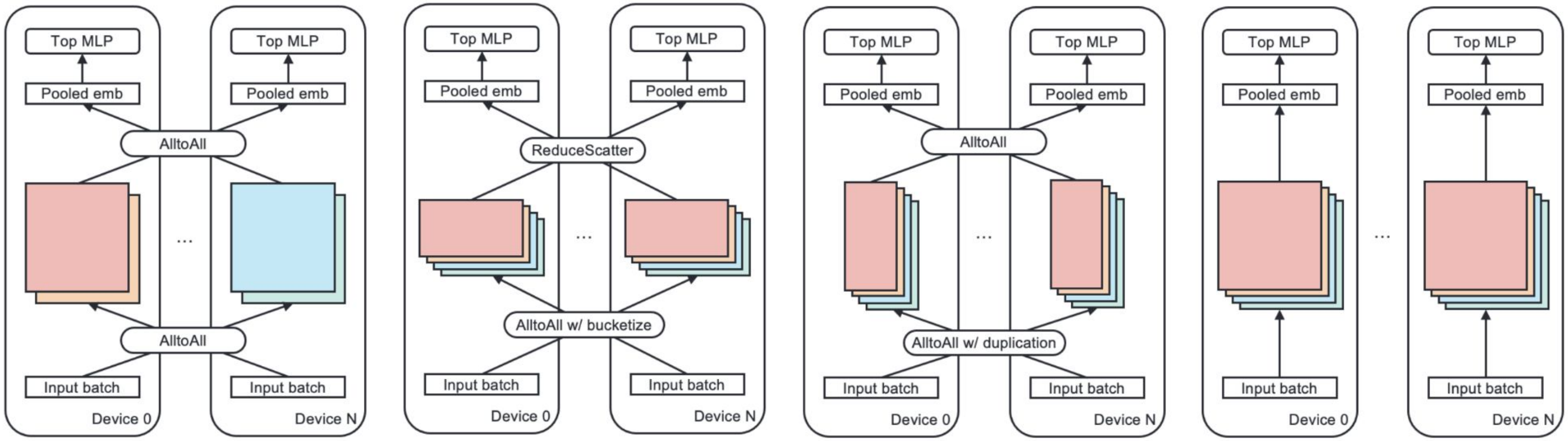
→ **I/O intensive**

Dheevatsa Mudigere et al., "Software-hardware co-design for fast and scalable training of deep learning recommendation models. " ISCA '22

Parallelism - all (possible) ways

Flexible 4D Model parallelism for embedding tables

- Sharding across tables, rows, columns and data
- Hierarchical sharding combining multiple strategies



(a) Table-wise Parallelism **(b)** Row-wise Parallelism **(c)** Column-wise Parallelism **(d)** Data Parallelism

Dheevatsa Mudigere et al., “Software-hardware co-design for fast and scalable training of deep learning recommendation models. “ ISCA '22

Agenda

- Introduction
- **Demystify Communication Behavior in Training DLRM**
  - **With a real-world workload**
- Reproduction of Communication
  - Collect Communication trace
  - Replay communication trace
- Summary

Communication in DLRM-2020 workloads

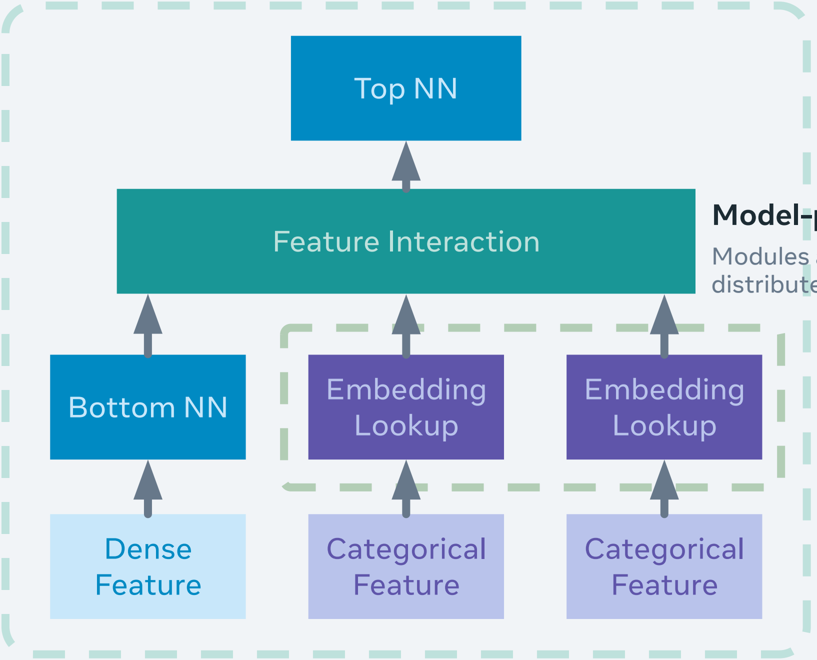
- Testbed*
  - 128 Nvidia V100 GPUs, 8 GPUs per node
  - 8 *200G RDMA NICs per node
- SW stack
  - DLRM (https://github.com/facebookresearch/dlrm)
  - PyTorch (https://pytorch.org/)
  - NCCL (https://github.com/NVIDIA/nccl)

Model	model-A
Num parameters	793B
MFLOPS per sample	638
Num of emb tables	≈ 1000s
Embedding table dims (range [min, max], avg)	[4, 384] avg: 93
Avg pooling size	15
Num MLP layers	20
Avg MLP size	3375
Target local batch size	512
Achieved QPS	1.2M

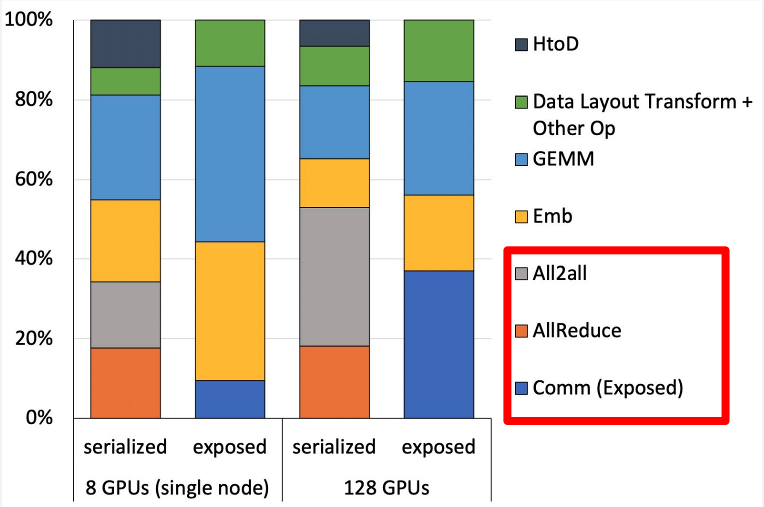
* More details in: Dheevatsa Mudigere et al., “Software-hardware co-design for fast and scalable training of deep learning recommendation models. “ ISCA '22

Distributed Training of DLRMs

- Communication patterns depend on parallelism strategy



Model-parallel training
Modules are sharded and distributed to each rank

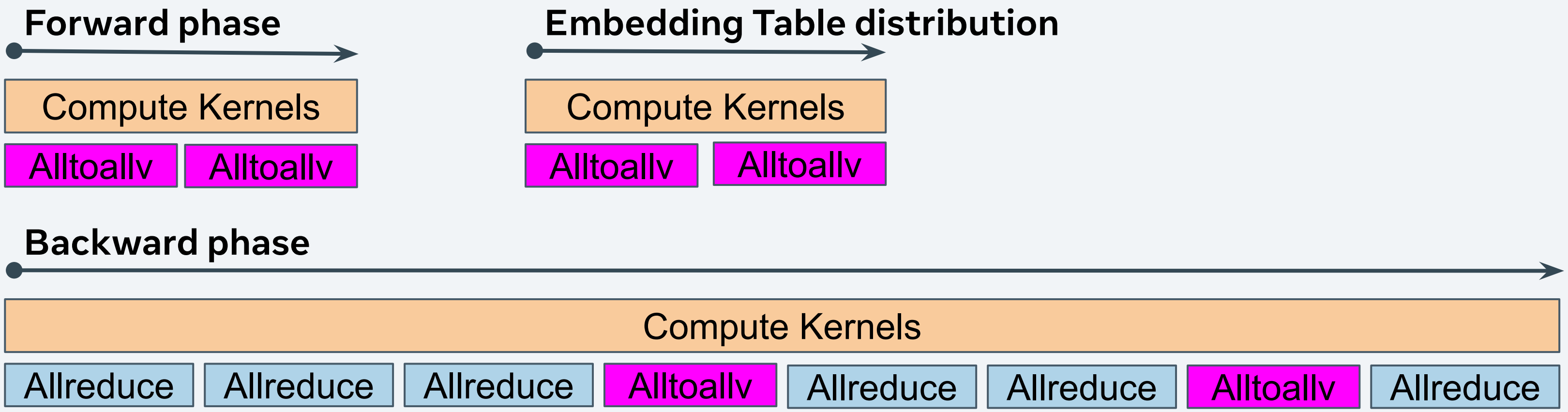


Data-parallel training – Modules are replicated on each rank

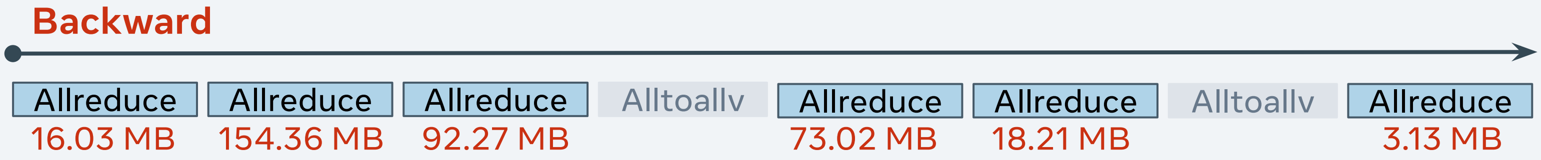
Dheevatsa Mudigere et al., "Software-hardware co-design for fast and scalable training of deep learning recommendation models. " ISCA '22

Communication in DLRM-2020 workloads

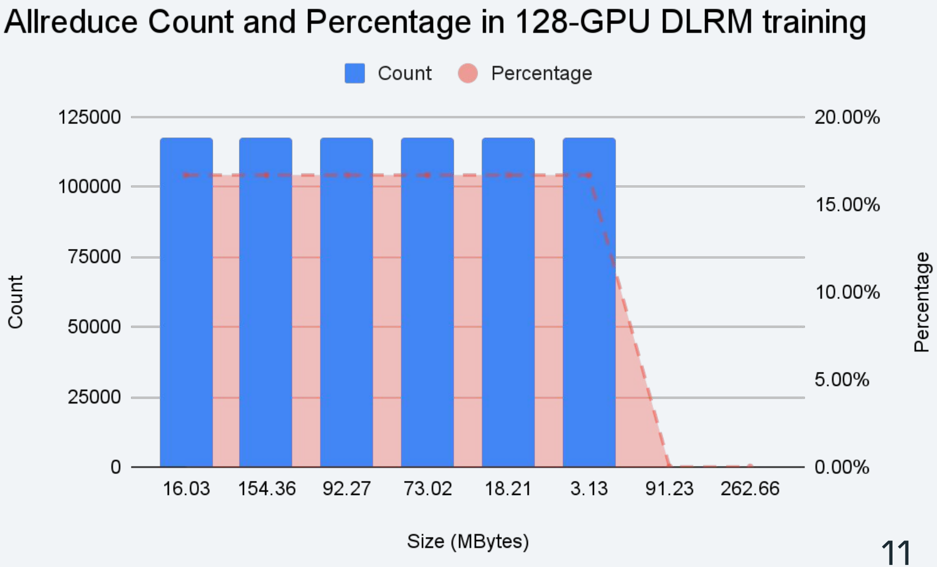
- **Key communication patterns**
  - **Allreduce** operations during backward phase → **data parallelism**
  - **Alltoallv** operations → **model parallelism & table distribution**
- **Message sizes and patterns are varied for different parallelisms**
  - Column-wise as an example



Allreduce Communication in DLRM-2020



- **Variations***
  - Batch size
  - Model parallelism
    - Column-wise
  - PyTorch parallelism
    - [DDP]()
      - Bucket size
    - [FSDP]()



Dheevatsa Mudigere et al., “Software-hardware co-design for fast and scalable training of deep learning recommendation models. “ ISCA '22

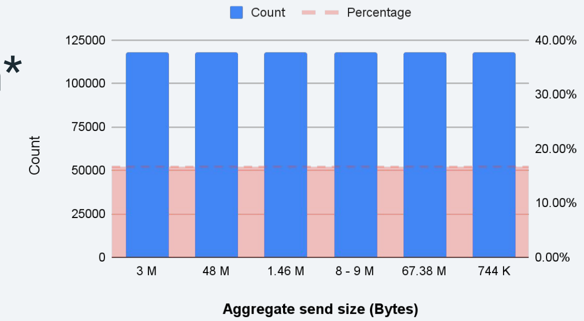
Alltoallv Communication in DLRM-2020

	FWD		Emb. Table Dist.		BWD	
	Alltoallv	Alltoallv	Alltoallv	Alltoallv	Alltoallv	Alltoallv
Send:	3 MB	48 MB	1.46 MB	8.75 MB	67.38 MB	0.73 MB
Receive:	0.73 MB	67.38 MB	0.5 MB	0 B	48 MB	3 MB

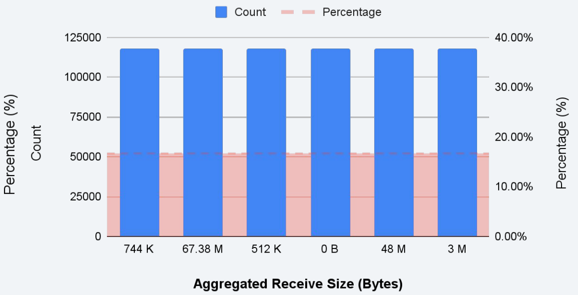
These sizes are captured from rank-0, it is varied across ranks

- **Variations**
  - Model Parallelisms
  - Comms Quantization*

Alltoallv send sizes from Rank-0



Alltoall receive sizes from Rank-0

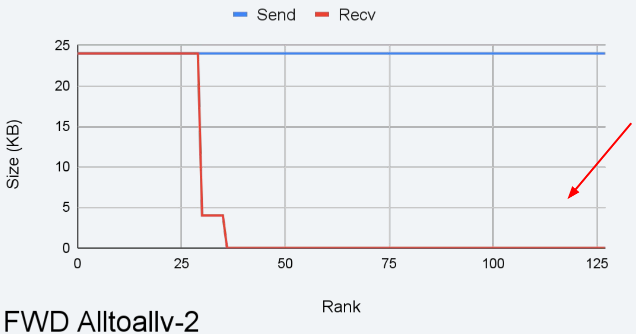


*Yang JA, Park J, Sridharan S, Tang PT. “Training deep learning recommendation model with quantized collective communications. “ DLP-KDD’20

Alltoallv Communication in DLRM-2020 (cont.)

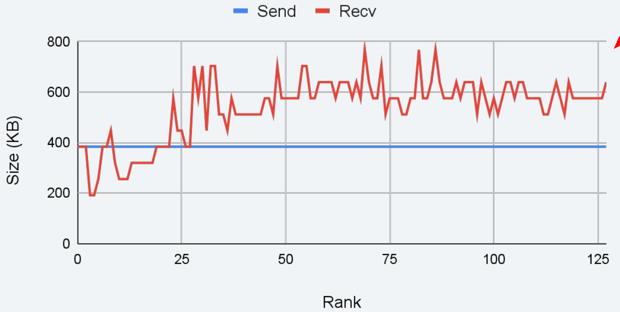
- **Imbalanced communication**

FWD Alltoallv-1



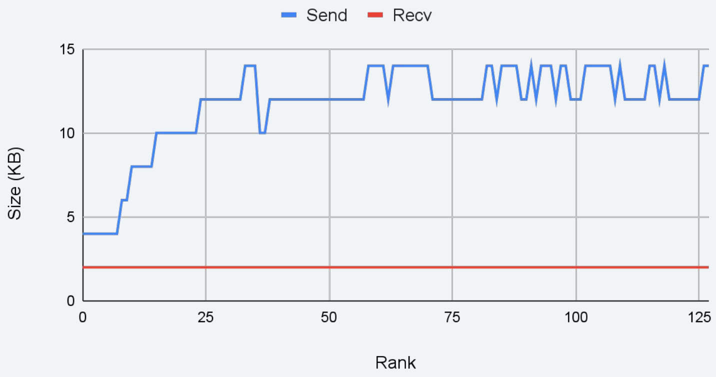
Zero data from most ranks

FWD Alltoallv-2

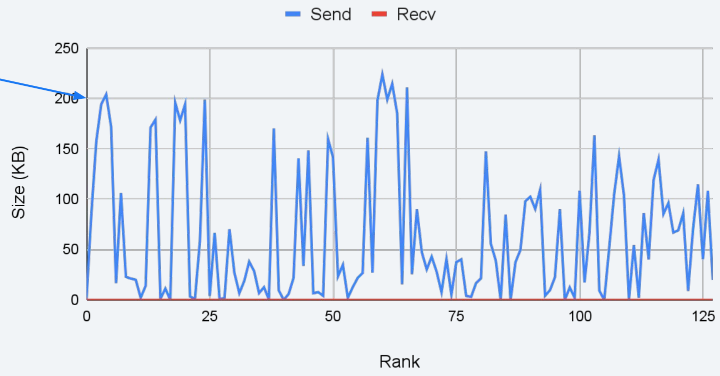


High skewness among ranks

Emb-Dist Alltoallv-1



Emb-Dist Alltoallv-2

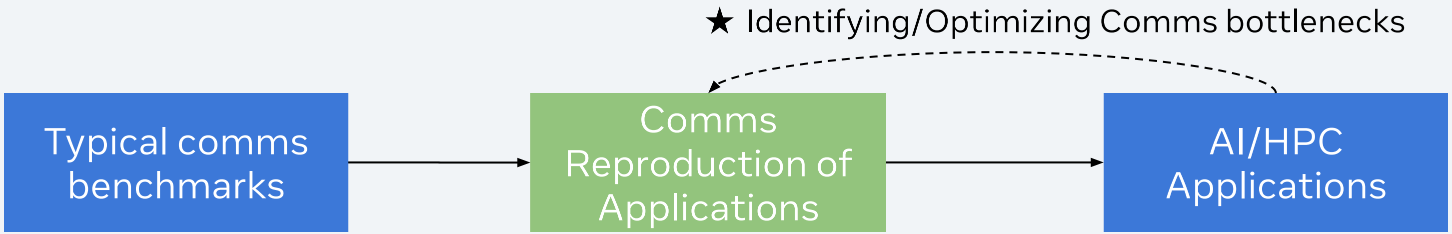


Agenda

- Introduction
- Demystify Communication Behavior in Training DLRM
- **Communication Reproduction**
  - Collect Communication trace
  - Replay communication trace
- Summary

Reproduction of Communication Behavior

- Why?
  - No need to run entire training workloads
  - Focus on understanding and optimizing communication patterns
  - No interference from computation
  - Cross-platform competitively analysis
    - E.g., Explore new SW/HW at scale for existing models



- Evaluate specific communication
- Comms. performance tuning

- Deep dive of existing or new comms patterns

- Full training process
- Optimize compute and comms-comp overlap

How to Replay Communication Trace

- Collecting communication traces
  - Captured from real-world production workloads
- Replaying communication traces using PARAM benchmark
  - https://github.com/facebookresearch/param
  - PyTorch-based communication benchmark suite
  - Multi-backend support
    - NCCL, UCC, MPI, Gloo
  - Multi-device support
    - CPU, GPU (Nvidia & AMD), TPU

Summary

- Communication patterns in training DLRMs are complex, but predictable in general
  - Various parallelism methods, typically **~40%** time spent on communication
  - Large-sized Allreduce operations overlapped with compute kernels
  - Imbalanced Alltoallv operations are common
- Communication reproduction is important for SW/HW optimization
  - Open-source trace-replay benchmark (https://github.com/facebookresearch/param)
  - Need more communication traces of real-world workloads
- **Ongoing work**
  - Collecting, analyzing and sharing more communication traces of real-world DLRM workloads, e.g., various scales, various parallelism

Meta