Efficient Communication Library for Large-Scale Deep Learning

Mar 26, 2018

Minsik Cho (minsikcho@us.ibm.com)
Deep Learning changing Our Life

Automotive/transportation

Security/public safety

Medicine and Biology

Media and Entertainment

Consumer Web, Mobile, Retail
Deep Learning Workflow

Latency to model: Typically days to train complex models
Limited by training compute throughput

Conversion/Retraining
Needed if training/inference precisions differ

Latency to action: Typically ms to complete full inference workflow
Limited by latency of batching (to enable efficient inference) + inference compute + resultant action

This is my focus

Training Data (grouped in large minibatches)

Input Data

Batching
Smaller, varied batch size: Application-dependent

Individual
E.g., from microservices

Training
Forward
Backward

Next Minibatch
Next Epoch

Trained Model

Forward

Error

Forward

Inference

Action
Application-dependent
Advance in Computation for Deep Learning

[P. Goldsborough]

- 10-100 TFLOPS
- Very good scaling for last 15 years

[MichaelGalloy.com]
Motivation: Ok, ever-fast computation. Is this enough?

- **ImageNet1K**: 1.2M images, 1K classes, Resnet101
  - Batch-size = 32 (limited by GPU memory)
  - Iteration time = 300ms
  - #iterations per epoch = 38000
  - Total training time for 100 epoch = 13.2 days
- **ImageNet22K**: 7.5M images, 22K classes, Resnet101
  - Total training time for 100 epoch = 35.2 days

- **No, it is NOT**
  - 1.2M samples are still at toy scale
  - Computation scaling is not fast enough
    - the data explosion/model complexity
    - Innovation will take too long, or even stop at some point
  - I cannot wait for days to get my model trained!
Faster Training Time with Distributed Deep Learning

What will you do?
Iterate more and create more accurate models?
Create more models?
Both?

Learning runs with Power 9
100x

Learning runs with Power 8
54x

Days
9
Distributed Deep Learning

Model parallelism (complex partitioning)

Data parallelism: Parm-Server

Data parallelism: Allreduce

Gradient/weight (10MB-1GB)
• In weak-scaling
  – Computation cost remains constant
  – Communication cost increases with more learners/GPUs
• Computation /Communication is the key for large-scale deep learning
  – Increase Computation
  – Faster Communication
Still scaling, but not fast enough
- Computation is still ahead
- Data perhaps grows much faster
• Model/Application
  • Deeper/wider model to increase compute time
  • Smaller gradient count to reduce communication time

• System
  • Balance network and computing resources
  • Select mini-batch size to adjust the ratio
    • Larger mini-batch size to lower the ratio
    • Too big mini-batch size can hurt convergence and accuracy
  • Network-topology aware communication
IBM PowerAI DDL (Distributed Deep Learning Library)

• Collaborative communication library for Distributed Deep Learning
  – MPI-like interface for easy-integration
  – Enables deep learning software to scale to 100s servers with CPU/GPUs
  – Works across variety of system sizes
  – Works with variety of network types, switch topologies

• DDL orchestrates the data communication
  – Plan efficient communication pattern on a hierarchal network environment
  – Actual point-point data transfer via NCCL or MPI

• Currently integrated into
  – Supported: Caffe, Tensorflow, Chainer, Torch
  – Ongoing: Caffe2, PyTorch, Keras (TF-backend)

• Currently US patent-pending
**DDL: Topology-aware communication**

- Example
  - A, B, C, D broadcast to all others
  - Suffers from congestion
  - Suffers from lower BW

Max bandwidth: 10 Gbytes/sec
Max sustained bandwidth: 100 Gbytes/sec
DDL : Topology-aware communication

- It’s a mapping problem
  - System-specific network
  - Application-specific traffic
- DDL does differently
  - To minimize bus contention
  - To minimize crossing lower BW

```
A -> B  B -> C  C -> D
B -> A  A -> D  D -> C
C -> D  D -> A  A -> B
D -> C  C -> B  B -> A
```

```
suboptimal

```
Optimal

```
• Assumption
  – network topology with various bandwidths

• Problem Definition
  – min-cost multi-commodity flow problem
  – NP-hard problem but can be solved easily if graph size is small (i.e., 4 vertices)

• DDL solves a typical case/topology offline
  – if the cluster/cloud has provided such topology, it performs very well
DDL : How well it performs on Caffe2

- 48 IBM S822LC with PPC64LE RHEL
  - 3 racks and 16 hosts on each, connected through 10GB/s IB
  - Each host has 4 P100-SXM2 with CUDA8, CUDNN5
- Comparing algorithms on Resnet50 + Imagenet1K (preloaded to RAMDisk) mbs=32
  - MPI_Allreduce
  - Ring (all-reduce from Baidu in Feb 2017)
  - GLOO (from Facebook) : NCCL+ib_verb
Comparison with NCCL 2.1.x Allreduce (POWER)

- IBM P9 Newell Systems (NVLink) with V100s
- 100Gbps InfiniBand

Exploiting in-system topology

Exploiting in/cross-system topology
Comparison with NCCL 2.1.x Allreduce (X86)

- X86 Systems (PCIe) with P100s
- 10Gbps Ethernet

**NO** in-system topology

Exploiting cross-system topology
Conclusion

• DDL is a topology-aware communication library in PowerAI
• DDL delivers the industry-best performance with
  – Network hierarchy
  – Multi-tier bandwidth
• DDL is suitable for common distributed training on cloud environment