NV-Group: Link-Efficient Reductions for Distributed Deep Learning on Modern Dense GPU Systems

Ching-Hsiang Chu, Pouya Kousha, Ammar A. Awan, Kawthar Shafie Khorassani, Hari Subramoni, and Dhabaleswar K. (DK) Panda
{chu.368, kousha.2, awan.10, shafiekhorassani.1}@osu.edu
{subramon, panda}@cse.ohio-state.edu

Network-based Computing Laboratory
The Ohio State University
Outline

• **Introduction**
  - Trend in Modern HPC systems
  - All-reduce for Distributed Deep Learning on Dense-GPU systems

• Research Challenge

• Proposed Designs: NV-Group Allreduce

• Performance Evaluation

• Concluding Remarks
Trends in Modern HPC Architecture: Heterogeneous

- Multi/Many-core Processors
- High Performance Interconnects: InfiniBand, Omni-Path, EFA (<1usec latency, 200Gbps+ Bandwidth)
- Accelerators: high compute density, high performance/watt
- SSD, NVMe-SSD, NVRAM Node local storage

Top 500 Supercomputers:

1. **Fugaku**
   - (158,976 nodes with A64FX ARM CPU, a GPU-like processor)

2. **Summit**
   - (27,648 GPUs)

3. **Sierra**
   - (17,280 GPUs)

4. **Lassen**
   - (2,664 GPUs)

5. **HPC5**
   - (7,280 GPUs)

6. **Selene**
   - (2,240 GPUs)

7. **NVIDIA DGX SuperPOD**

More information: [https://www.top500.org/]
Trends in Modern Large-scale Dense-GPU Systems

- **Scale-up** (up to 150 GB/s)
  - PCIe, NVLink/NVSwitch
  - Infinity Fabric, Xe Link

- **Scale-out** (up to 25 GB/s)
  - InfiniBand, Omni-path, Ethernet
  - Cray Slingshot
GPU-enabled Distributed Deep Learning

- Easy-to-use and high-performance frameworks
- Wide range of applications
  - Image Classification
  - Speech Recognition
  - Self-driving Car
  - Healthcare
  - Climate Analytic

999 PetaFlop/s sustained, and 1.13 ExaFlop/s peak FP 16 performance over 4560 nodes (27,360 GPUs)

• Distributed deep learning training with data parallelism
  – Using Allreduce operations to exchange and update gradients, weights...etc.

• State-of-the-art Ring-based Allreduce for GPUs*
  – Pros: Contention-free
  – Cons: not scalable

Reduction Operations for Distributed Deep Learning

Agent 1

\[
\begin{align*}
W^{(0)} & \quad \text{All-Reduce} \quad \text{All-Reduce} \\
W^{(1)} & \quad \ldots \\
\end{align*}
\]

Agent m

\[
\begin{align*}
W^{(1)} & \quad \ldots \\
W^{(T)} & \quad \ldots \\
\end{align*}
\]

Time


https://www.oreilly.com/ideas/distributed-tensorflow

*Please refer to the paper for the analysis of more algorithms
Motivation

- Ring-based Allreduce cannot efficiently utilize NVLinks

* Profiling tool: P. Kousha et al., Designing a Profiling and Visualization Tool for Scalable and In-Depth Analysis of High-Performance GPU Clusters, HiPC 2019.
Outline

• Introduction

• **Research Challenge**

• Proposed Designs: NV-Group

• Performance Evaluation

• Concluding Remarks
How to design a link-efficient Allreduce algorithm that can maximize the utilization of available hardware communication channels to boost the performance for distributed DL training on emerging dense GPU systems?
Outline

• Introduction
• Research Challenge

• Proposed Designs: NV-Group Allreduce

• Performance Evaluation
• Concluding Remarks
Overview of the Proposed NVGroup Allreduce

1. Forming NV-Groups
   – Treat multiple GPUs as one

2. Cooperative reduction kernel within NV-Group
   – Persistent GPU kernels
   – Exploit load-store primitives over NVLinks
   – High-occupancy kernel

3. Communication across NV-Groups
   – Contention-free over slowest IB networks
Forming NV-Groups

- Topology detection and GPU grouping
  - Discover which GPUs are **fully connected** by NVLink; using tools such as hwloc[1] and NVML[2]
  - Create logical GPU groups, e.g., MPI Group or Communicator
Cooperative Reduction Kernel within NV-Group

- **CPU creates work queue for each Cooperative Thread Array (CTA or block)**

- **Persistent GPU Kernel**
  1. **Poll** the individual work queue
  2. **Reduce** the data chunks
     - Reduce-scatter among GPUs
     - Direct Load-Store over NVLink
  3. **Signal** CPU upon completion
     - Implicit synchronization\[1\]

\[1\] Ching-Hsiang Chu et al. “Designing High-Performance In-Memory Key-Value Operations with Persistent GPU Kernels and OpenSHMEM,” OpenSHMEM 2018.
Cooperative Reduction Kernel - Efficiency

- **High-Occupancy kernel** with low register pressure*
  - CPU coordinates the topology and communication paths
  - Enable all threads for reduction operations
- **Free resources for applications**
  - Low SM consumption
    - Low scheduling overhead
  - Enable overlap opportunity

Group-Wise Communication – CPU-GPU Cooperation

• CPU processes
  – Inter-group communication
  – Ring-based Reduce-Scatter + Allgather *over IB or X-BUS*
  – Offload reduction to NV-Group

• GPUs (NV-Group):
  – Processing operations requested by CPU
    • Direct Reduce-scatter or Allgather *over NVLink*

* Please check out the paper for more optimization techniques.
Outline

• Introduction
• Research Challenge
• Proposed Designs: NV-Group
• **Performance Evaluation**
• Concluding Remarks
## Experimental Environments

<table>
<thead>
<tr>
<th></th>
<th>#1 Summit</th>
<th>#10 Lassen*</th>
<th>NVIDIA DGX-2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU Model</strong></td>
<td>IBM POWER9 AC922</td>
<td></td>
<td>Intel Skylake</td>
</tr>
<tr>
<td><strong>System memory</strong></td>
<td>512 GB</td>
<td>256 GB</td>
<td>1.5 TB</td>
</tr>
<tr>
<td><strong>GPU Model</strong></td>
<td>NVIDIA Volta V100 x 6</td>
<td>NVIDIA Volta V100 x 4</td>
<td>NVIDIA Volta V100 x 16</td>
</tr>
<tr>
<td><strong>Interconnects between CPU &amp; GPU</strong></td>
<td>2-lane NVLink</td>
<td>3-lane NVLink</td>
<td>PCIe Gen3</td>
</tr>
<tr>
<td><strong>Interconnects between GPUs</strong></td>
<td></td>
<td></td>
<td>6-lane NVLink &amp; NVSwitch</td>
</tr>
<tr>
<td><strong>Interconnects between nodes</strong></td>
<td>Dual-rail Mellanox IB EDR</td>
<td></td>
<td>Mellanox IB EDR x 8 (Unused)</td>
</tr>
<tr>
<td><strong>NVIDIA driver &amp; CUDA versions</strong></td>
<td>418.116 &amp; 10.1.243</td>
<td></td>
<td>410.48 &amp; 10.1.243</td>
</tr>
</tbody>
</table>

- **Libraries:** SpectrumMPI v10.3.1, OpenMPI v4.0.3+UCX v1.8.0, MVAPICH2-GDR v2.3, NCCL v2.6
- **Benchmarks:** OSU Micro-Benchmark (OMB) & modified nccl-test
- **Applications:** Horovod v0.19 with TensorFlow v1.14 & PyTorch v1.5

*Please refer to the paper for the thorough performance comparison*
Overview of the MVAPICH2 Project

- High Performance open-source MPI Library
- Support for multiple interconnects
  - InfiniBand, Omni-Path, Ethernet/iWARP, RDMA over Converged Ethernet (RoCE), and AWS EFA
- Support for multiple platforms
  - x86, OpenPOWER, ARM, Xeon-Phi, GPGPUs
- Started in 2001, first open-source version demonstrated at SC ’02
- Supports the latest MPI-3.1 standard
- http://mvapich.cse.ohio-state.edu
- Additional optimized versions for different systems/environments:
  - MVAPICH2-X (Advanced MPI + PGAS), since 2011
  - MVAPICH2-GDR with support for NVIDIA GPGPUs, since 2014
  - MVAPICH2-MIC with support for Intel Xeon-Phi, since 2014
  - MVAPICH2-Virt with virtualization support, since 2015
  - MVAPICH2-EA with support for Energy-Awareness, since 2015
  - MVAPICH2-Azure for Azure HPC IB instances, since 2019
  - MVAPICH2-X-AWS for AWS HPC+EFA instances, since 2019
- Tools:
  - OSU MPI Micro-Benchmarks (OMB), since 2003
  - OSU InfiniBand Network Analysis and Monitoring (INAM), since 2015
- Used by more than 3,100 organizations in 89 countries
- More than 772,000 (> 0.7 million) downloads from the OSU site directly
- Empowering many TOP500 clusters (June 2020 ranking)
  - 4th, 10,649,600-core (Sunway TaihuLight) at NSC, Wuxi, China
  - 8th, 448, 448 cores (Frontera) at TACC
  - 12th, 391,680 cores (ABCI) in Japan
  - 18th, 570,020 cores (Nurion) in South Korea and many others
- Available with software stacks of many vendors and Linux Distros (RedHat, SuSE, OpenHPC, and Spack)
- Partner in the 8th ranked TACC Frontera system
- Empowering Top500 systems for more than 15 years
Evaluation of Link Utilization on Summit

Throughput of Allreduce operation on IBM AC922 machine

- SpectrumMPI-10.3
- OpenMPI-4.0.3
- MV2-GDR-2.3
- NCCL-2.6
- Proposed

Throughput (GB/s)

Throughput of Allreduce operation on IBM AC922 machine using various versions of MPI and communication libraries shows a significant improvement in performance, especially for the proposed method, which utilizes all NVLinks.

- NVGroup design utilizes all NVLinks
Allreduce Benchmarking on Summit

- NVGroup yields 40% and 30% better latency and bandwidth than NCCL
- NVGroup scale significantly better than production libraries on Summit system

*Please refer to the paper for the thorough comparison
Distributed Deep Learning Training on DGX-2 Machine

- ResNet-50 Training using Horovod with TensorFlow
  - Synthetic ImageNet dataset with batch size 64 and 100 batches
- NVGroup achieves higher throughput due to high-occupancy reduction kernel
Distributed Deep Learning Training on Summit

- ResNet-50 Training using Horovod with TensorFlow and PyTorch
  - Synthetic ImageNet dataset with batch size 64 and 100 batches

![Graph showing images per second vs number of GPUs for TensorFlow v1.14 and PyTorch v1.5. TensorFlow v1.14 has 5% higher images per second compared to PyTorch v1.5 at 192 GPUs. PyTorch v1.5 has 1.48X higher images per second compared to TensorFlow v1.14 at 384 GPUs.](image-url)
Outline

• Introduction
• Research Challenge
• Proposed Designs
• Performance Evaluation

• Concluding Remarks
Concluding Remarks

• Allreduce operations dominate the performance of distributed deep learning training with data parallelism

• State-of-the-art ring-based Allreduce failed to efficiently utilize interconnects on the modern Dense GPU systems

• Proposed NVGroup Allreduce can
  – Maximize the NVLink utilization for the dense-GPU systems
  – Perform cooperative reduction on GPUs
  – Achieve faster distributed DL training on dense-GPU systems such as #1 Summit

• Publicly available since MVAPICH2-GDR 2.3.2 release!
  – http://mvapich.cse.ohio-state.edu/
Thank You!

{chu.368, kousha.2, awan.10, shafiekhorassani.1}@osu.edu
{subramon, panda}@cse.ohio-state.edu

Network-Based Computing Laboratory
http://nowlab.cse.ohio-state.edu/

The High-Performance MPI/PGAS Project
http://mvapich.cse.ohio-state.edu/

The High-Performance Big Data Project
http://hibd.cse.ohio-state.edu/

The High-Performance Deep Learning Project
http://hidl.cse.ohio-state.edu/