Scalable and Distributed DNN Training on Modern HPC Systems: Challenges and Solutions

Keynote Talk at HPML ‘19

by

Dhabaleswar K. (DK) Panda
The Ohio State University
E-mail: panda@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~panda
Understanding the Deep Learning Resurgence

- Deep Learning is a sub-set of Machine Learning
  - But, it is perhaps the most radical and revolutionary subset
  - Automatic feature extraction vs. hand-crafted features

- Deep Learning
  - A renewed interest and a lot of hype!
  - Key success: Deep Neural Networks (DNNs)
  - Everything was there since the late 80s except the "computability of DNNs"

Courtesy: http://www.deeplearningbook.org/contents/intro.html
Deep Learning Use Cases and Growth Trends

1.1 Artificial Intelligence Revenue, World Markets: 2016-2025

1.2 Artificial Intelligence Revenue, Top 10 Use Cases, World Markets: 2025

Contract analysis
Object detection and classification - avoidance, navigation
Object identification, detection, classification, tracking from geospatial images
Automated geophysical feature detection
Text query of images
Content distribution on social media
Predictive maintenance
Efficient, scalable processing of patient data
Static image recognition, classification, and tagging
Algorithmic trading strategy performance improvement

Increasing Usage of HPC, Big Data and Deep Learning

Convergence of HPC, Big Data, and Deep Learning!

Increasing Need to Run these applications on the Cloud!!
Newer Workflows - Deep Learning over Big Data (DLoBD)

- Deep Learning over Big Data (DLoBD) is one of the most efficient analyzing paradigms
- More and more deep learning tools or libraries (e.g., Caffe, TensorFlow) start running over big data stacks, such as Apache Hadoop and Spark
- **Benefits** of the DLoBD approach
  - Easily build a powerful data analytics pipeline
  - Better data locality
  - Efficient resource sharing and cost effective
Drivers of Modern HPC Cluster Architectures

- Multi-core/many-core technologies
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
- Solid State Drives (SSDs), Non-Volatile Random-Access Memory (NVRAM), NVMe-SSD
- Accelerators (NVIDIA GPGPUs and Intel Xeon Phi)
- Available on HPC Clouds, e.g., Amazon EC2, NSF Chameleon, Microsoft Azure, etc.
Key Phases of Deep Learning

• Deep Learning has two major tasks
  1. Training of the Deep Neural Network
  2. Inference (or deployment) that uses a trained DNN

• DNN Training
  – Training is a compute/communication intensive process – can take days to weeks
  – Faster training is necessary!

• Faster training can be achieved by
  – Using Newer and Faster Hardware – But, there is a limit!
  – Can we use more GPUs or nodes?
    • The need for Parallel and Distributed Training
Scale-up and Scale-out

- **Scale-up**: Intra-node Communication
  - Many improvements like:
    - NVIDIA cuDNN, cuBLAS, NCCL, etc.
    - CUDA 9 Co-operative Groups

- **Scale-out**: Inter-node Communication
  - DL Frameworks – most are optimized for single-node only
  - Distributed (Parallel) Training is an emerging trend
    - OSU-Caffe – MPI-based
    - Microsoft CNTK – MPI/NCCL2
    - Google TensorFlow – gRPC-based/MPI/NCCL2
    - Facebook Caffe2 – Hybrid (NCCL2/Gloo/MPI)
Holistic Evaluation is Important!!

- My framework is faster than your framework!
- This needs to be understood in a holistic way.
- Performance depends on the entire execution environment (the full stack)
- Isolated view of performance is not helpful

How to efficiently scale-out a Deep Learning (DL) framework and take advantage of heterogeneous High Performance Computing (HPC) resources?
Research Challenges to Exploit HPC Technologies

1. What are the fundamental issues in designing DL frameworks?
   - Memory Requirements
   - Computation Requirements
   - Communication Overhead

2. Why do we need to support distributed training?
   - To overcome the limits of single-node training
   - To better utilize hundreds of existing HPC Clusters
Research Challenges to Exploit HPC Technologies (Cont’d)

3. What are the new design challenges brought forward by DL frameworks for Communication runtimes?
   - Large Message Collective Communication and Reductions
   - GPU Buffers (CUDA-Awareness)

4. Can a Co-design approach help in achieving Scale-up and Scale-out efficiently?
   - Co-Design the support at Runtime level and Exploit it at the DL Framework level
   - What performance benefits can be observed?
   - What needs to be fixed at the communication runtime layer?
Multiple Approaches taken up by OSU

• MPI-driven Deep Learning
  – CPU-based Deep Learning
  – GPU-based Deep Learning
• Co-designing Deep Learning Stacks with High-Performance MPI
• Out-of-core DNN training
  • Accelerating TensorFlow on HPC Systems
  • Accelerating Big Data Stacks
  • Efficient Deep Learning over Big Data
Data Parallel Deep Learning and MPI Collectives

- **Major MPI Collectives** involved in Designing distributed frameworks
- **MPI_Bcast** – required for DNN parameter exchange
- **MPI_Reduce** – needed for gradient accumulation from multiple solvers
- **MPI_Allreduce** – use just one Allreduce instead of Reduce and Broadcast

Overview of the MVAPICH2 Project

- High Performance open-source MPI Library for InfiniBand, Omni-Path, Ethernet/iWARP, and RDMA over Converged Ethernet (RoCE)
  - MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.1), Started in 2001, First version available in 2002
  - MVAPICH2-X (MPI + PGAS), Available since 2011
  - Support for GPGPUs (MVAPICH2-GDR) and MIC (MVAPICH2-MIC), Available since 2014
  - Support for Virtualization (MVAPICH2-Virt), Available since 2015
  - Support for Energy-Awareness (MVAPICH2-EA), Available since 2015
  - Support for InfiniBand Network Analysis and Monitoring (OSU INAM) since 2015
  - Used by more than 3,000 organizations in 88 countries
  - More than 539,000 (> 0.5 million) downloads from the OSU site directly
  - Empowering many TOP500 clusters (Nov ‘18 ranking)
    - 3rd ranked 10,649,640-core cluster (Sunway TaihuLight) at NSC, Wuxi, China
    - 14th, 556,104 cores (Oakforest-PACS) in Japan
    - 17th, 367,024 cores (Stampede2) at TACC
    - 27th, 241,108-core (Pleiades) at NASA and many others
  - Available with software stacks of many vendors and Linux Distros (RedHat, SuSE, and OpenHPC)
  - [http://mvapich.cse.ohio-state.edu](http://mvapich.cse.ohio-state.edu) Partner in the upcoming TACC Frontera System

- Empowering Top500 systems for over a decade
Architecture of MVAPICH2 Software Family

High Performance Parallel Programming Models

- Message Passing Interface (MPI)
- PGAS (UPC, OpenSHMEM, CAF, UPC++)
- Hybrid --- MPI + X (MPI + PGAS + OpenMP/Cilk)

High Performance and Scalable Communication Runtime

Diverse APIs and Mechanisms
- Point-to-point Primitives
- Collectives Algorithms
- Job Startup
- Energy-Awareness
- Remote Memory Access
- I/O and File Systems
- Fault Tolerance
- Virtualization
- Active Messages
- Introspection & Analysis

Support for Modern Networking Technology
(InfiniBand, iWARP, RoCE, Omni-Path)

- Transport Protocols: RC, XRC, UD, DC
- Modern Features: UMR, ODP, SR-IOV, Multi Rail

Support for Modern Multi-/Many-core Architectures
(Intel-Xeon, OpenPower, Xeon-Phi, ARM, NVIDIA GPGPU)

- Transport Mechanisms: Shared Memory, CMA, IVSHMEM, XPMEM*
- Modern Features: MCDRAM*, NVLink*, CAPI*

* Upcoming

Network Based Computing Laboratory
HPML (May ‘19)
**MVAPICH2 Software Family (CPU-Based Deep Learning)**

<table>
<thead>
<tr>
<th>High-Performance Parallel Programming Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>MVAPICH2</td>
</tr>
<tr>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>MVAPICH2-Virt</td>
</tr>
<tr>
<td>MVAPICH2-EA</td>
</tr>
<tr>
<td>MVAPICH2-MIC</td>
</tr>
</tbody>
</table>

**Microbenchmarks**

| OMB | Microbenchmarks suite to evaluate MPI and PGAS (OpenSHMEM, UPC, and UPC++) libraries for CPUs and GPUs |

**Tools**

<table>
<thead>
<tr>
<th>OSU INAM</th>
<th>Network monitoring, profiling, and analysis for clusters with MPI and scheduler integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>OEMT</td>
<td>Utility to measure the energy consumption of MPI applications</td>
</tr>
</tbody>
</table>
Performance of CNTK with MVAPICH2-X on CPU-based Deep Learning

- CPU-based training of AlexNet neural network using ImageNet ILSVRC2012 dataset
- Advanced XPMEM-based designs show up to 20% benefits over Intel MPI (IMPI) for CNTK DNN training using All_Reduce
- The proposed designs show good scalability with increasing system size

Available since MVAPICH2-X 2.3rc1 release

Designing Efficient Shared Address Space Reduction Collectives for Multi-/Many-cores, J. Hashmi, S. Chakraborty, M. Bayatpour, H. Subramoni, and DK Panda, 32nd IEEE International Parallel & Distributed Processing Symposium (IPDPS ‘18), May 2018
Performance of TensorFlow with MVAPICH2-X on CPU

- CPU-based distributed TensorFlow (TF) benchmark
  - `tf_cnn_benchmark` tests
- AlexNet model training
  - ImageNet ILSVRC2012 dataset
- Advanced SALaR and XPMEM based designs in MVAPICH-X showed good scalability
- Up to 15% and 35% improvements in number of images per second at 448 and 896 processes, respectively.

Will be available in future MVAPICH2-X releases

## MVAPICH2 Software Family (GPU-Based Deep Learning)

### High-Performance Parallel Programming Libraries

<table>
<thead>
<tr>
<th>MVAPICH2</th>
<th>Support for InfiniBand, Omni-Path, Ethernet/iWARP, and RoCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MVAPICH2-X</td>
<td>Advanced MPI features, OSU INAM, PGAS (OpenSHMEM, UPC, UPC++, and CAF), and MPI+PGAS programming models with unified communication runtime</td>
</tr>
<tr>
<td>MVAPICH2-GDR</td>
<td>Optimized MPI for clusters with NVIDIA GPUs and for GPU-enabled Deep Learning Applications</td>
</tr>
<tr>
<td>MVAPICH2-Virt</td>
<td>High-performance and scalable MPI for hypervisor and container based HPC cloud</td>
</tr>
<tr>
<td>MVAPICH2-EA</td>
<td>Energy aware and High-performance MPI</td>
</tr>
<tr>
<td>MVAPICH2-MIC</td>
<td>Optimized MPI for clusters with Intel KNC</td>
</tr>
</tbody>
</table>

### Microbenchmarks

| OMB            | Microbenchmarks suite to evaluate MPI and PGAS (OpenSHMEM, UPC, and UPC++) libraries for CPUs and GPUs |

### Tools

| OSU INAM       | Network monitoring, profiling, and analysis for clusters with MPI and scheduler integration |
| OEMT           | Utility to measure the energy consumption of MPI applications |
MPI + CUDA - Naive

• Data movement in applications with standard MPI and CUDA interfaces

At Sender:

cudA::Memcpy(s_hostbuf, s_devbuf, . . .);
MPI_Send(s_hostbuf, size, . . .);

At Receiver:

MPI_Recv(r_hostbuf, size, . . .);
cudA::Memcpy(r_devbuf, r_hostbuf, . . .);

High Productivity and Low Performance
**MPI + CUDA - Advanced**

- Pipelining at user level with non-blocking MPI and CUDA interfaces

---

**At Sender:**
```
for (j = 0; j < pipeline_len; j++)
    cudaMemcpyAsync(s_hostbuf + j * blk, s_devbuf + j * blksz, ...);
for (j = 0; j < pipeline_len; j++) {
    while (result != cudaSuccess) {
        result = cudaStreamQuery(...);
        if(j > 0) MPI_Test(...);
    }
    MPI_Isend(s_hostbuf + j * block_sz, blksz . . .);
}    MPI_Waitall();
```

<<<Similar at receiver>>>

---

**Low Productivity and High Performance**
GPU-Aware (CUDA-Aware) MPI Library: MVAPICH2-GPU

- Standard MPI interfaces used for unified data movement
- Takes advantage of Unified Virtual Addressing (>= CUDA 4.0)
- Overlaps data movement from GPU with RDMA transfers

**At Sender:**

```
MPI_Send(s_devbuf, size, ...);
```

**At Receiver:**

```
MPI_Recv(r_devbuf, size, ...);
```

*High Performance and High Productivity*
CUDA-Aware MPI: MVAPICH2-GDR 1.8-2.3.1 Releases

- Support for MPI communication from NVIDIA GPU device memory
- High performance RDMA-based inter-node point-to-point communication (GPU-GPU, GPU-Host and Host-GPU)
- High performance intra-node point-to-point communication for multi-GPU adapters/node (GPU-GPU, GPU-Host and Host-GPU)
- Taking advantage of CUDA IPC (available since CUDA 4.1) in intra-node communication for multiple GPU adapters/node
- Optimized and tuned collectives for GPU device buffers
- MPI datatype support for point-to-point and collective communication from GPU device buffers
- Unified memory
MVAPICH2-GDR 2.3.1

- Released on 03/16/2018
- Major Features and Enhancements
  - Based on MVAPICH2 2.3.1
  - Enhanced intra-node and inter-node point-to-point performance for DGX-2 and IBM POWER8 and IBM POWER9 systems
  - Enhanced Allreduce performance for DGX-2 and IBM POWER8/POWER9 systems
  - Enhanced small message performance for CUDA-Aware MPI_Put and MPI_Get
  - Support for PGI 18.10
  - Flexible support for running TensorFlow (Horovod) jobs
  - Add support for Volta (V100) GPU
  - Support for OpenPOWER with NVLink
  - Efficient Multiple CUDA stream-based IPC communication for multi-GPU systems with and without NVLink
  - Leverage Linux Cross Memory Attach (CMA) feature for enhanced host-based communication
  - InfiniBand Multicast (IB-MCAST) based designs for GPU-based broadcast and streaming applications
  - Efficient broadcast designs for Deep Learning applications
Optimized MVAPICH2-GDR Design

**GPU-GPU Inter-node Latency**
- MV2-(NO-GDR)
- MV2-GDR 2.3

**GPU-GPU Inter-node Bandwidth**
- MV2-(NO-GDR)
- MV2-GDR-2.3

Intel Haswell (E5-2687W @ 3.10 GHz) node - 20 cores
NVIDIA Volta V100 GPU
Mellanox Connect-X4 EDR HCA
CUDA 9.0
Mellanox OFED 4.0 with GPU-Direct-RDMA
Distributed Training using TensorFlow (TF)

- TensorFlow is the most popular DL framework
- gRPC is the official distributed training runtime
  - Many problems for HPC use-cases
- Community efforts - Baidu and Uber’s Horovod have added MPI support to TF across nodes
- Need to understand several options currently available →

To be Presented on May 17th (Session 7 B)

Scalable TensorFlow using Horovod, MPI, and NCCL

- Efficient Allreduce is crucial for Horovod’s overall training performance
  - Both MPI and NCCL designs are available
- We have evaluated Horovod extensively and compared across a wide range of designs using gRPC and gRPC extensions
- MVAPICH2-GDR achieved up to 90% scaling efficiency for ResNet-50 Training on 64 Pascal GPUs

**MVAPICH2-GDR: Allreduce Comparison with Baidu and OpenMPI**

- 16 GPUs (4 nodes) MVAPICH2-GDR vs. Baidu-Allreduce and OpenMPI 3.0

---

*Available since MVAPICH2-GDR 2.3a*
MVAPICH2-GDR vs. NCCL2 – Allreduce Operation

- Optimized designs in MVAPICH2-GDR 2.3 offer better/comparable performance for most cases
- MPI_Allreduce (MVAPICH2-GDR) vs. ncclAllreduce (NCCL2) on 16 GPUs

Platform: Intel Xeon (Broadwell) nodes equipped with a dual-socket CPU, 1 K-80 GPUs, and EDR InfiniBand Inter-connect
MVAPICH2-GDR vs. NCCL2 – Allreduce Operation (DGX-2)

- Optimized designs in upcoming MVAPICH2-GDR offer better/comparable performance for most cases
- MPI_Allreduce (MVAPICH2-GDR) vs. ncclAllreduce (NCCL2) on 1 DGX-2 node (16 Volta GPUs)

Platform: Nvidia DGX-2 system (16 Nvidia Volta GPUs connected with NVSwitch), CUDA 9.2
MVAPICH2-GDR: Enhanced MPI_Allreduce at Scale

- Optimized designs in upcoming MVAPICH2-GDR offer better performance for most cases
- MPI_Allreduce (MVAPICH2-GDR) vs. ncclAllreduce (NCCL2) up to 1,536 GPUs

Platform: Dual-socket IBM POWER9 CPU, 6 NVIDIA Volta V100 GPUs, and 2-port InfiniBand EDR Interconnect
Distributed Training with TensorFlow and MVAPICH2-GDR

- ResNet-50 Training using TensorFlow benchmark on 1 DGX-2 node (8 Volta GPUs)

Platform: Nvidia DGX-2 system (16 Nvidia Volta GPUs connected with NVSwitch), CUDA 9.2
MVAPICH2-GDR: Pre-requisites for OpenPOWER & x86 Systems

• MVAPICH2-GDR 2.3.1 requires the following software to be installed on your system:
  1. Mellanox OFED 3.2 and later
  2. NVIDIA Driver 367.48 or later
  3. NVIDIA CUDA Toolkit 7.5 and later
  4. NVIDIA Peer Memory (nv_peer_mem) module to enable GPUDirect RDMA (GDR) support

• Strongly Recommended for Best Performance
  5. GDRCOPY Library by NVIDIA: https://github.com/NVIDIA/gdrcopy

• Comprehensive Instructions can be seen from the MVAPICH2-GDR User Guide:
  – http://mvapich.cse.ohio-state.edu/userguide/gdr/
Simple Installation steps for both systems

Pick the right MVAPICH2-GDR RPM from Downloads page:
  - [http://mvapich.cse.ohio-state.edu/downloads/](http://mvapich.cse.ohio-state.edu/downloads/)
  - e.g. [http://mvapich.cse.ohio-state.edu/download/mvapich/gdr/2.3/mofed4.5/mvapich2-gdr-mcast.cuda10.0.mofed4.5.gnu4.8.5-2.3-1.el7.x86_64.rpm](http://mvapich.cse.ohio-state.edu/download/mvapich/gdr/2.3/mofed4.5/mvapich2-gdr-mcast.cuda10.0.mofed4.5.gnu4.8.5-2.3-1.el7.x86_64.rpm) (== <mv2-gdr-rpm-name>.rpm)

$ wget http://mvapich.cse.ohio-state.edu/download/mvapich/gdr/2.3/<mv2-gdr-rpm-name>.rpm

**Root Users:**

$ rpm -Uvh --nodeps <mv2-gdr-rpm-name>.rpm

**Non-Root Users:**

$ rpm2cpio <mv2-gdr-rpm-name>.rpm | cpio – id

Contact MVAPICH help list with any questions related to the package

mvapich-help@cse.ohio-state.edu
OSU-Caffe: Scalable Deep Learning

• Caffe: A flexible and layered Deep Learning framework.

• Benefits and Weaknesses
  – Multi-GPU Training within a single node
  – Performance degradation for GPUs across different sockets
  – Limited Scale-out

• OSU-Caffe: MPI-based Parallel Training
  – Enable Scale-up (within a node) and Scale-out (across multi-GPU nodes)
  – Scale-out on 64 GPUs for training CIFAR-10 network on CIFAR-10 dataset
  – Scale-out on 128 GPUs for training GoogLeNet network on ImageNet dataset

OSU-Caffe publicly available from http://hidl.cse.ohio-state.edu/
Training Large (Out-of-core) Models

- Large DNNs cannot be trained on GPUs due to memory limitation!
  - ResNet-50 for Image Recognition but current frameworks can only go up to a small batch size of 45
  - Next generation models like Neural Machine Translation (NMT) are ridiculously large, consists of billions of parameters, and require even more memory
  - Can we design Out-of-core DNN training support using new software features in CUDA 8/9 and hardware mechanisms in Pascal/Volta GPUs?

- General intuition is that managed allocations “will be” slow!
  - The proposed framework called OC-Caffe (Out-of-Core Caffe) shows the potential of managed memory designs that can provide performance with negligible/no overhead.

- OC-Caffe-Opt: up to 80% better than Intel-optimized CPU Caffe for ResNet-50 training on the Volta V100 GPU with CUDA9 and CUDNN7

A. Awan et al., OC-DNN: Exploiting Advanced Unified Memory Capabilities in CUDA 9 and Volta GPUs for Out-of-Core DNN Training, HiPC ’18
Multiple Approaches taken up by OSU

• MPI-driven Deep Learning
• Co-designing Deep Learning Stacks with High-Performance MPI
• Out-of-core DNN training
• Accelerating TensorFlow on HPC Systems
• Accelerating Big Data Stacks
• Efficient Deep Learning over Big Data
Architecture Overview of gRPC

Key Features:
• Simple service definition
• Works across languages and platforms
  • C++, Java, Python, Android Java etc
  • Linux, Mac, Windows.
• Start quickly and scale
• Bi-directional streaming and integrated authentication
• Used by Google (several of Google’s cloud products and Google externally facing APIs, TensorFlow), Netflix, Docker, Cisco, Juniper Networks etc.
• Uses sockets for communication!

Large-scale distributed systems composed of micro services

Source: http://www.grpc.io/
### Performance Benefits for RDMA-gRPC with Micro-Benchmark

#### RDMA-gRPC RPC Latency

<table>
<thead>
<tr>
<th>Payload (Bytes)</th>
<th>Default gRPC</th>
<th>OSU RDMA gRPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
<td></td>
</tr>
<tr>
<td>512</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2K</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8K</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **gRPC-RDMA Latency on SDSC-Comet-FDR**
  - Up to 2.7x performance speedup over IPoIB for Latency for small messages
  - Up to 2.8x performance speedup over IPoIB for Latency for medium messages
  - Up to 2.5x performance speedup over IPoIB for Latency for large messages

---

Performance Benefit for RDMA-TensorFlow (Inception3)

- TensorFlow Inception3 performance evaluation on an IB EDR cluster
  - Up to 20% performance speedup over Default gRPC (IPoIB) for 8 GPUs
  - Up to 34% performance speedup over Default gRPC (IPoIB) for 16 GPUs
  - Up to 37% performance speedup over Default gRPC (IPoIB) for 24 GPUs

R. Biswas, X. Lu, and D. K. Panda, Accelerating TensorFlow with Adaptive RDMA-based gRPC. HiPC ‘18
High-Performance Design of TensorFlow over RDMA-enabled Interconnects

- High performance RDMA-enhanced design with native InfiniBand support at the verbs-level for gRPC and TensorFlow
- RDMA-based data communication
- Adaptive communication protocols
- Dynamic message chunking and accumulation
- Support for RDMA device selection
- Easily configurable for different protocols (native InfiniBand and IPoIB)

Current release: 0.9.1

- Based on Google TensorFlow 1.3.0
- Tested with
  - Mellanox InfiniBand adapters (e.g., EDR)
  - NVIDIA GPGPU K80
  - Tested with CUDA 8.0 and CUDNN 5.0

http://hidl.cse.ohio-state.edu
Multiple Approaches taken up by OSU

- MPI-driven Deep Learning
- Co-designing Deep Learning Stacks with High-Performance MPI
- Out-of-core DNN Training
- Accelerating TensorFlow on HPC Systems
- Accelerating Big Data Stacks
- Efficient Deep Learning over Big Data
The High-Performance Big Data (HiBD) Project

- RDMA for Apache Spark
- RDMA for Apache Hadoop 3.x (RDMA-Hadoop-3.x)
- RDMA for Apache Hadoop 2.x (RDMA-Hadoop-2.x)
  - Plugins for Apache, Hortonworks (HDP) and Cloudera (CDH) Hadoop distributions
- RDMA for Apache Kafka
- RDMA for Apache HBase
- RDMA for Memcached (RDMA-Memcached)
- RDMA for Apache Hadoop 1.x (RDMA-Hadoop)
- OSU HiBD-Benchmarks (OHB)
  - HDFS, Memcached, HBase, and Spark Micro-benchmarks
- [http://hibd.cse.ohio-state.edu](http://hibd.cse.ohio-state.edu)
- Users Base: 310 organizations from 35 countries
- More than 29,950 downloads from the project site

Available for InfiniBand and RoCE
Also run on Ethernet

Available for x86 and OpenPOWER

Support for Singularity and Docker
Performance Numbers of RDMA for Apache Hadoop 2.x – RandomWriter & TeraGen in OSU-RI2 (EDR)

Cluster with 8 Nodes with a total of 64 maps

- **RandomWriter**
  - **3x** improvement over IPoIB for 80-160 GB file size

- **TeraGen**
  - **4x** improvement over IPoIB for 80-240 GB file size
32 Worker Nodes, 768 cores, PageRank Total Time  
64 Worker Nodes, 1536 cores, PageRank Total Time

- InfiniBand FDR, SSD, 32/64 Worker Nodes, 768/1536 Cores, (768/1536M 768/1536R)
- RDMA-based design for Spark 1.5.1
- RDMA vs. IPoIB with 768/1536 concurrent tasks, single SSD per node.
  - 32 nodes/768 cores: Total time reduced by 37% over IPoIB (56Gbps)
  - 64 nodes/1536 cores: Total time reduced by 43% over IPoIB (56Gbps)
High-Performance Deep Learning over Big Data (DLoBD) Stacks

- **Challenges** of Deep Learning over Big Data (DLoBD)
  - Can RDMA-based designs in DLoBD stacks improve performance, scalability, and resource utilization on high-performance interconnects, GPUs, and multi-core CPUs?
  - What are the performance characteristics of representative DLoBD stacks on RDMA networks?
- **Characterization** on DLoBD Stacks
  - CaffeOnSpark, TensorFlowOnSpark, and BigDL
  - IPoIB vs. RDMA; In-band communication vs. Out-of-band communication; CPU vs. GPU; etc.
  - Performance, accuracy, scalability, and resource utilization
  - RDMA-based DLoBD stacks (e.g., BigDL over RDMA-Spark) can achieve 2.6x speedup compared to the IPoIB based scheme, while maintain similar accuracy

Conclusions

• Scalable distributed training is getting important
• Requires high-performance middleware designs while exploiting modern interconnects
• Provided a set of different solutions to achieve scalable distributed training
  – Optimized collectives for CPU-based training
  – CUDA-aware MPI with optimized collectives for GPU-based training
  – TensorFlow-gRPC with RDMA support
  – Efficient DL support over Big Data
• Will continue to enable the DL community to achieve scalability and high-performance for their distributed training
More details in Two Tutorials at CCGrid ‘19

  - May 15th (Session 1C)

- **Accelerating Big Data Processing on Modern HPC Clusters**
  - May 16th (Session 4C)
Funding Acknowledgments

**Funding Support by**

[Logos of various funding sponsors]

**Equipment Support by**

[Logos of various equipment sponsors]
## Personnel Acknowledgments

### Current Students (Graduate)
- A. Awan (Ph.D.)
- M. Bayatpour (Ph.D.)
- S. Chakraborty (Ph.D.)
- C.-H. Chu (Ph.D.)
- J. Hashmi (Ph.D.)
- A. Jain (Ph.D.)
- K. S. Kandadi (M.S.)

### Current Students (Undergraduate)
- K. S. Khorassani (Ph.D.)
- P. Kousha (Ph.D.)
- A. Quentin (Ph.D.)
- B. Ramesh (M.S.)
- D. Shankar (Ph.D.)
- S. Xu (M.S.)
- Q. Zhou (Ph.D.)

### Current Research Asst. Professor
- X. Lu
- A. Yeretzian (B.S.)

### Current Research Scientist
- H. Subramoni

### Past Students
- A. Augustine (M.S.)
- P. Balaji (Ph.D.)
- R. Biswas (M.S.)
- S. Bhagvat (M.S.)
- A. Bhat (M.S.)
- D. Buntinas (Ph.D.)
- L. Chai (Ph.D.)
- B. Chandrasekharan (M.S.)
- N. Dandapanthula (M.S.)
- V. Dhanraj (M.S.)
- T. Gangadharappa (M.S.)
- K. Gopalakrishnan (M.S.)
- W. Huang (Ph.D.)
- W. Jiang (M.S.)
- J. Jose (Ph.D.)
- S. Kini (M.S.)
- M. Koop (Ph.D.)
- K. Kulkarni (M.S.)
- R. Kumar (M.S.)
- S. Krishnamoorthy (M.S.)
- K. Kandalla (Ph.D.)
- M. Li (Ph.D.)
- P. Lai (M.S.)
- J. Liu (Ph.D.)
- M. Luo (Ph.D.)
- A. Mamidala (Ph.D.)
- G. Marsh (M.S.)
- V. Meshram (M.S.)
- A. Moody (M.S.)
- S. Naravula (Ph.D.)
- R. Noronha (Ph.D.)
- X. Ouyang (Ph.D.)
- S. Potluri (Ph.D.)
- J. Liu (Ph.D.)
- M. Luo (Ph.D.)
- A. Mamidala (Ph.D.)
- G. Marsh (M.S.)
- V. Meshram (M.S.)
- A. Moody (M.S.)
- S. Naravula (Ph.D.)
- R. Noronha (Ph.D.)
- X. Ouyang (Ph.D.)
- S. Potluri (Ph.D.)
- J. Liu (Ph.D.)
- M. Luo (Ph.D.)
- A. Mamidala (Ph.D.)
- G. Marsh (M.S.)
- V. Meshram (M.S.)
- A. Moody (M.S.)
- S. Naravula (Ph.D.)
- R. Noronha (Ph.D.)
- X. Ouyang (Ph.D.)
- S. Potluri (Ph.D.)

### Past Research Scientist
- K. Hamidouche
- S. Sur

### Past Post-Docs
- D. Banerjee
- X. Besseron
- H.-W. Jin
- J. Lin
- M. Luo
- E. Mancini
- S. Marcarelli
- J. Vienne
- H. Wang

### Past Programmers
- D. Bureddy
- J. Perkins

### Current Research Scientist
- H. Subramoni

### Current Research Specialist
- J. Smith

### Current Post-doc
- M. S. Ghazimeersaeed
- A. Ruhela
- K. Manian

### Past Research Specialist
- M. Arnold
Thank You!

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/