Benchmarks and Middleware for Designing Convergent HPC, Big Data and Deep Learning Software Stacks for Exascale Systems

Keynote Talk at Bench ’19 Conference

by

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

Follow us on
https://twitter.com/mvapich
High-End Computing (HEC): PetaFlop to ExaFlop

100 PFlops in 2017

149 PFlops in 2018

1 EFlops in 2020-2021?

Expected to have an ExaFlop system in 2020-2021!
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!!
Can We Run HPC, Big Data and Deep Learning Jobs on Existing HPC Infrastructure?
Can We Run HPC, Big Data and Deep Learning Jobs on Existing HPC Infrastructure?
Can We Run HPC, Big Data and Deep Learning Jobs on Existing HPC Infrastructure?
Can We Run HPC, Big Data and Deep Learning Jobs on Existing HPC Infrastructure?
Presentation Overview

• **MVAPICH Project**
  – MPI and PGAS Library with CUDA-Awareness

• **HiBD Project**
  – High-Performance Big Data Analytics Library

• **HiDL Project**
  – High-Performance Deep Learning

• **Public Cloud Deployment**
  – Microsoft-Azure and Amazon-AWS

• **Conclusions**
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 3,050 organizations in 89 countries
  - More than 614,000 (> 0.6 million) downloads from the OSU site directly
  - Empowering many TOP500 clusters (Nov ‘18 ranking)
    - 3rd, 10,649,600-core (Sunway TaihuLight) at National Supercomputing Center in Wuxi, China
    - 5th, 448, 448 cores (Frontera) at TACC
    - 8th, 391,680 cores (ABCI) in Japan
    - 15th, 570,020 cores (Neurion) in South Korea 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 TACC Frontera System

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

## High Performance Parallel Programming Models

<table>
<thead>
<tr>
<th>Message Passing Interface (MPI)</th>
<th>PGAS (UPC, OpenSHM, CAF, UPC++)</th>
<th>Hybrid --- MPI + X (MPI + PGAS + OpenMP/Cilk)</th>
</tr>
</thead>
</table>

## 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

(InfinitBand, iWARP, RoCE, Omni-Path, Elastic Fabric Adapter)

<table>
<thead>
<tr>
<th>Transport Protocols</th>
<th>Modern Features</th>
<th>Transport Mechanisms</th>
<th>Modern Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC</td>
<td>UMR</td>
<td>Shared Memory</td>
<td>Optane*</td>
</tr>
<tr>
<td>SRD</td>
<td>ODP</td>
<td>CMA</td>
<td>NVLink</td>
</tr>
<tr>
<td>UD</td>
<td>SR-IOV</td>
<td>IVSHMEM</td>
<td>CAPI*</td>
</tr>
<tr>
<td>DC</td>
<td>Multi Rail</td>
<td>XPMEM</td>
<td></td>
</tr>
</tbody>
</table>

### Support for Modern Multi-/Many-core Architectures

(Intel-Xeon, OpenPOWER, Xeon-Phi, ARM, NVIDIA GPGPU)

- **Optane**
- **NVLink**
- **CAPI**

* Upcoming
## MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP, Omni-Path, and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI Features/Support, OSU INAM, PGAS and MPI+PGAS with IB, Omni-Path, and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MPI with IB, RoCE &amp; GPU and Support for Deep Learning</td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>HPC Cloud with MPI &amp; IB</td>
<td>MVAPICH2-Virt</td>
</tr>
<tr>
<td>Energy-aware MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2-EA</td>
</tr>
<tr>
<td>MPI Energy Monitoring Tool</td>
<td>OEMT</td>
</tr>
<tr>
<td>InfiniBand Network Analysis and Monitoring</td>
<td>OSU INAM</td>
</tr>
<tr>
<td>Microbenchmarks for Measuring MPI and PGAS Performance</td>
<td>OMB</td>
</tr>
</tbody>
</table>
Convergent Software Stacks for HPC, Big Data and Deep Learning
Need for Micro-Benchmarks to Design and Evaluate Programming Models

- Message Passing Interface (MPI) is the common programming model in scientific computing
- Has 100’s of APIs and Primitives (Point-to-point, RMA, Collectives, Datatypes, …)
- Multiple challenges for MPI developers, users, managers of HPC centers
  - How to optimize the designs of these APIs on various hardware platforms and configurations?
    - Designers and developers
  - Comparing performance of an MPI library (at the API-level) across various platforms and configurations?
    - Designers, developers and users
  - How to compare the performance of multiple MPI libraries (at the API-level) on a given platform and across platforms?
    - Procurement decision by managers
  - How to correlate the performance from the micro-benchmark level to the overall application level?
    - Application developers and users, also beneficial for co-designs
OSU Micro-Benchmarks (OMB)

- Available since 2004 ([https://mvapich.cse.ohio-state.edu/benchmarks](https://mvapich.cse.ohio-state.edu/benchmarks))
- Suite of microbenchmarks to study communication performance of various programming models
- Benchmarks available for the following programming models
  - Message Passing Interface (MPI)
  - Partitioned Global Address Space (PGAS)
    - Unified Parallel C (UPC)
    - Unified Parallel C++ (UPC++)
    - OpenSHMEM
- Benchmarks available for multiple accelerator based architectures
  - Compute Unified Device Architecture (CUDA)
  - OpenACC Application Program Interface
- Part of various national resource procurement suites like NERSC-8 / Trinity Benchmarks
- Continuing to add support for newer primitives and features
OSU Micro-Benchmarks (MPI): Examples and Capabilities

- **Host-Based**
  - Point-to-point
  - Collectives
    - Blocking and Non-Blocking

- **Job-startup**

- **GPU-Based**
  - CUDA-aware
    - Point-to-point: Device-to-Device (DD), Device-to-Host (DH), Host-to-Device (HD)
    - Collectives
  - Managed Memory
    - Point-to-point: Managed-Device-to-Managed-Device (MD-MD)
One-way Latency: MPI over IB with MVAPICH2

**Small Message Latency**

- TrueScale-QDR - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with IB switch
- ConnectX-3-FDR - 2.8 GHz Deca-core (IvyBridge) Intel PCI Gen3 with IB switch
- ConnectIB-Dual FDR - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with IB switch
- ConnectX-4-EDR - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with IB Switch
- Omni-Path - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with Omni-Path switch
- ConnectX-6 HDR - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with IB Switch

**Large Message Latency**

- TrueScale-QDR
- ConnectX-3-FDR
- ConnectIB-DualFDR
- ConnectX-4-EDR
- Omni-Path
- ConnectX-6 HDR

Network Based Computing Laboratory

Bench ’19

17
Bandwidth: MPI over IB with MVAPIC2

**Unidirectional Bandwidth**

- **Message Size (bytes)**
- **Bandwidth (MBytes/sec)**

**Bidirectional Bandwidth**

- **Message Size (bytes)**
- **Bandwidth (MBytes/sec)**

**TrueScale-QDR - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with IB switch**
- 24,532 MBytes/sec

**ConnectX-3-FDR - 2.8 GHz Deca-core (IvyBridge) Intel PCI Gen3 with IB switch**
- 12,590 MBytes/sec

**ConnectIB-Dual FDR - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with IB switch**
- 12,366 MBytes/sec

**ConnectX-4-EDR - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with IB Switch**
- 12,083 MBytes/sec

**Omni-Path - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with Omni-Path switch**
- 6,356 MBytes/sec

**ConnectX-6-HDR - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with IB Switch**
- 3,373 MBytes/sec
Intra-node Point-to-Point Performance on OpenPOWER

**Intra-Socket Small Message Latency**

Latency (us)

- MVAPICH2-X 2.3.2
- SpectrumMPI-10.3.0.01

Latency: 0.25us

**Intra-Socket Large Message Latency**

Latency (us)

- MVAPICH2-X 2.3.2
- SpectrumMPI-10.3.0.01

**Intra-Socket Bandwidth**

Bandwidth (MB/s)

- MVAPICH2-X 2.3.2
- SpectrumMPI-10.3.0.01

**Intra-Socket Bi-directional Bandwidth**

Bandwidth (MB/s)

- MVAPICH2-X 2.3.2
- SpectrumMPI-10.3.0.01

**Platform:** Two nodes of OpenPOWER (Power9-ppc64le) CPU using Mellanox EDR (MT4121) HCA
Point-to-point: Latency & Bandwidth (Inter-socket) on ARM

---

**Latency - Small Messages**

3.5x better

**Latency - Medium Messages**

8.3x better

**Latency - Large Messages**

---

**Bandwidth - Small Messages**

5x better

**Bandwidth - Medium Messages**

---

**Bandwidth - Large Messages**
OSU Micro-Benchmarks (MPI): Examples and Capabilities

- **Host-Based**
  - Point-to-point
  - **Collectives**
    - Blocking and Non-Blocking

- **Job-startup**

- **GPU-Based**
  - **CUDA-aware**
    - Point-to-point: Device-to-Device (DD), Device-to-Host (DH), Host-to-Device (HD)
    - Collectives
  - Managed Memory
    - Point-to-point: Managed-Device-to-Managed-Device (MD-MD)
For MPI_Allreduce latency with 32K bytes, MVAPICH2-OPT can reduce the latency by 2.4X


Available since MVAPICH2-X 2.3b
Shared Address Space (XPMEM)-based Collectives Design

- “Shared Address Space”-based true zero-copy Reduction collective designs in MVAPICH2
- Offloaded computation/communication to peers ranks in reduction collective operation
- Up to 4X improvement for 4MB Reduce and up to 1.8X improvement for 4M AllReduce


Available since MVAPICH2-X 2.3rc1
Evaluation of SHArP based Non Blocking Allreduce

**MPI_Iallreduce Benchmark**

- Complete offload of Allreduce collective operation to Switch helps to have much higher overlap of communication and computation

*PPN: Processes Per Node

Available since MVAPICH2 2.3a
OSU Micro-Benchmarks (MPI): Examples and Capabilities

- **Host-Based**
  - Point-to-point
  - Collectives
    - Blocking and Non-Blocking

- **Job-startup**

- **GPU-Based**
  - CUDA-aware
    - Point-to-point: Device-to-Device (DD), Device-to-Host (DH), Host-to-Device (HD)
  - Managed Memory
    - Point-to-point: Managed-Device-to-Managed-Device (MD-MD)
Startup Performance on TACC Frontera

**MPI_Init on Frontera**

- **MPI_Init takes 3.9 seconds on 57,344 processes on 1,024 nodes**
- **All numbers reported with 56 processes per node**

New designs available in MVAPICH2-2.3.2
OSU Micro-Benchmarks (MPI): Examples and Capabilities

- **Host-Based**
  - Point-to-point
  - Collectives
    - Blocking and Non-Blocking

- **Job-startup**

- **GPU-Based**
  - CUDA-aware
    - Point-to-point: Device-to-Device (DD), Device-to-Host (DH) and Host-to-Device (HD)
    - Collectives
  - Managed Memory
    - Point-to-point: Managed-Device-to-Managed-Device (MD-MD)
Optimizing MPI Data Movement on GPU Clusters

• Connected as PCIe devices – Flexibility but Complexity

1. Intra-GPU
2. Intra-Socket GPU-GPU
3. Inter-Socket GPU-GPU
4. Inter-Node GPU-GPU
5. Intra-Socket GPU-Host
6. Inter-Socket GPU-Host
7. Inter-Node GPU-Host

8. Inter-Node GPU-GPU with IB adapter on remote socket
and more . . .

• For each path different schemes: Shared_mem, IPC, GPUDirect RDMA, pipeline ...
• Critical for runtimes to optimize data movement while hiding the complexity
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
Optimized MVAPICH2-GDR Design (D-D)

**GPU-GPU Inter-node Latency**

- **MV2-(NO-GDR)**
- **MV2-GDR 2.3**

**GPU-GPU Inter-node Bi-Bandwidth**

- **MV2-(NO-GDR)**
- **MV2-GDR-2.3**

**Latency (μs)**

- **Message Size (Bytes)**
  - 0
  - 1K
  - 2K
  - 4K
  - 8K

**Bandwidth (MB/s)**

- **Message Size (Bytes)**
  - 1
  - 2
  - 4
  - 8
  - 16
  - 32
  - 64
  - 128
  - 256
  - 512
  - 1K
  - 2K
  - 4K

**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**

**Optimized MVAPICH2-GDR Design (D-D)**

- **Latency**: 1.85us, 10x improvement
- **Bandwidth**: 9x improvement
D-to-D Performance on OpenPOWER w/ GDRCopy (NVLink2 + Volta)

Platform: OpenPOWER (POWER9-ppc64le) nodes equipped with a dual-socket CPU, 4 Volta V100 GPUs, and 2port EDR InfiniBand Interconnect

Intra-node Latency: 0.90 us (with GDRCopy)

Intra-node Bandwidth: 62.79 GB/sec for 4MB (via NVLINK2)

Inter-node Latency: 2.04 us (with GDRCopy)

Inter-node Bandwidth: 12.03 GB/sec (2 port EDR)

Available since MVAPICH2-GDR 2.3.2

Network Based Computing Laboratory

Bench ‘19
D-to-H & H-to-D Performance on OpenPOWER w/ GDRCopy (NVLink2 + Volta)

**Platform:** OpenPOWER (POWER9-ppc64le) nodes equipped with a dual-socket CPU, 4 Volta V100 GPUs, and 2port EDR InfiniBand Interconnect

**Intra-node D-H Latency:** 0.49 us (with GDRCopy)

**Intra-node H-D Latency:** 0.49 us (with GDRCopy)

**Intra-node D-H Bandwidth:** 16.70 GB/sec for 2MB (via NVLINK2)

**Intra-node H-D Bandwidth:** 26.09 GB/sec for 2MB (via NVLINK2)

Available since MVAPICH2-GDR 2.3a
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
OSU Micro-Benchmarks (MPI): Examples and Capabilities

- **Host-Based**
  - Point-to-point
  - Collectives
    - Blocking and Non-Blocking
- **Job-startup**
- **GPU-Based**
  - CUDA-aware
    - Point-to-point: Device-to-Device (DD), Device-to-Host (DH) and Host-to-Device (HD)
    - Collectives
  - Managed Memory
    - Point-to-point: Managed-Device-to-Managed-Device (MD-MD)
Managed Memory Performance (Inter-node x86) with MVAPICH2-GDR

Latency MD MD

Bandwidth MD MD

Bi-Bandwidth MD MD

Message Size (Bytes)

Latency (us)

Bandwidth (MBps)

Message Size (Bytes)

Network Based Computing Laboratory
Bench ‘19
Managed Memory Performance (OpenPOWER Intra-node)

Latency MD MD

Bandwidth MD MD

Bi-Bandwidth MD MD
Presentation Overview

- MVAPICH Project
  - MPI and PGAS Library with CUDA-Awareness

- HiBD Project
  - High-Performance Big Data Analytics Library

- HiDL Project
  - High-Performance Deep Learning

- Public Cloud Deployment
  - Microsoft-Azure and Amazon-AWS

- Conclusions
Data Management and Processing on Modern Datacenters

- Substantial impact on designing and utilizing data management and processing systems in multiple tiers
  - Front-end data accessing and serving (Online)
    - Memcached + DB (e.g. MySQL), HBase
  - Back-end data analytics (Offline)
    - HDFS, MapReduce, Spark
Convergent Software Stacks for HPC, Big Data and Deep Learning

MVAPICH2
MVAPICH2-X
MVAPICH2-GDR

HPC
(MPI, RDMA, Lustre, etc.)

Big Data
(Hadoop, Spark, HBase, Memcached, etc.)

Deep Learning
(Caffe, TensorFlow, BigDL, etc.)

RDMA-Hadoop
RDMA-Spark
RDMA-HBase
RDMA-Memcached
RDMA-Kafka
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: 315 organizations from 35 countries
- More than 31,600 downloads from the project site

Available for InfiniBand and RoCE
Also run on Ethernet
Available for x86 and OpenPOWER
Support for Singularity and Docker
Current set of Benchmarks for Big Data

- Hadoop Benchmarks
  - DFSIO, Terasort, Teragen, HiBench, ...
- PUMA
- YCSB
- Spark Benchmarks
- GroupBy, PageRank, K-means, ...
- BigData Bench
Are the Current Benchmarks Sufficient for Big Data?

• The current benchmarks provide some performance behavior

• However, do not provide any information to the designer/developer on:
  – What is happening at the lower-layer?
  – Where the benefits are coming from?
  – Which design is leading to benefits or bottlenecks?
  – Which component in the design needs to be changed and what will be its impact?
  – Can performance gain/loss at the lower-layer be correlated to the performance gain/loss observed at the upper layer?
Challenges in Benchmarking of Optimized Designs

Applications

Big Data Middleware
(HDFS, MapReduce, HBase, Spark and Memcached)

Programming Models
(Sockets)

Networking Technologies
(InfiniBand, 1/10/40/100 GigE and Intelligent NICs)

Big Data Middleware
(HDFS, MapReduce, HBase, Spark and Memcached)

Point-to-Point Communication

Threaded Models and Synchronization

Virtualization (SR-IOV)

I/O and File Systems

QoS & Fault Tolerance

Performance Tuning

Networking Technologies
(InfiniBand, 1/10/40/100 GigE and Intelligent NICs)

Commodity Computing System Architectures
(Multi- and Many-core architectures and accelerators)

Storage Technologies
(HDD, SSD, NVM, and NVMe-SSD)

Current
Benchmarks

Correlation?

No Benchmarks
Iterative Process – Requires Deeper Investigation and Design for Benchmarking Next Generation Big Data Systems and Applications

<table>
<thead>
<tr>
<th>Applications</th>
<th>Benchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Big Data Middleware</td>
<td></td>
</tr>
<tr>
<td>Programming Models</td>
<td>RDMA Protocols</td>
</tr>
<tr>
<td>(Sockets)</td>
<td></td>
</tr>
<tr>
<td>Communication and I/O Library</td>
<td></td>
</tr>
<tr>
<td>Point-to-Point Communication</td>
<td>Threaded Models and Synchronization</td>
</tr>
<tr>
<td>I/O and File Systems</td>
<td>QoS &amp; Fault Tolerance</td>
</tr>
<tr>
<td>Networking Technologies</td>
<td>Virtualization (SR-IOV)</td>
</tr>
<tr>
<td>(InfiniBand, 1/10/40/100 GigE and Intelligent NICs)</td>
<td>Performance Tuning</td>
</tr>
<tr>
<td>Commodity Computing System Architectures</td>
<td>Storage Technologies</td>
</tr>
<tr>
<td>(Multi- and Many-core architectures and accelerators)</td>
<td>(HDD, SSD, NVM, and NVMe-SSD)</td>
</tr>
</tbody>
</table>

Micro-Benchmarks

Applications-Level Benchmarks
OSU HiBD Micro-Benchmark (OHB) Suite - HDFS

• Evaluate the performance of standalone HDFS

• Five different benchmarks
  – Sequential Write Latency (SWL)
  – Sequential or Random Read Latency (SRL or RRL)
  – Sequential Write Throughput (SWT)
  – Sequential Read Throughput (SRT)
  – Sequential Read-Write Throughput (SRWT)


<table>
<thead>
<tr>
<th>Benchmark</th>
<th>File Name</th>
<th>File Size</th>
<th>HDFS Parameter</th>
<th>Readers</th>
<th>Writers</th>
<th>Random/Sequential Read</th>
<th>Seek Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWL</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRL/RRL</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td>√</td>
<td>√ (RRL)</td>
</tr>
<tr>
<td>SWT</td>
<td>√</td>
<td></td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRT</td>
<td>√</td>
<td></td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRWT</td>
<td>√</td>
<td></td>
<td>√</td>
<td></td>
<td></td>
<td>√</td>
<td></td>
</tr>
</tbody>
</table>
OSU HiBD Micro-Benchmark (OHB) Suite - MapReduce

- Evaluate the performance of stand-alone MapReduce
- Does not require or involve HDFS or any other distributed file system
- Models shuffle data patterns in real-workload Hadoop application workloads
- Considers various factors that influence the data shuffling phase
  - underlying network configuration, number of map and reduce tasks, intermediate shuffle data pattern, shuffle data size etc.
- Two different micro-benchmarks based on generic intermediate shuffle patterns
  - **MR-AVG**: intermediate data is evenly distributed (or approx. equal) among reduce tasks
    - **MR-RR** i.e., round-robin distribution and **MR-RAND** i.e., pseudo-random distribution
  - **MR-SKEW**: intermediate data is unevenly distributed among reduce tasks
    - Total number of shuffle key/value pairs, max% per reducer, min% per reducer to configure skew


OSU HiBD Micro-Benchmark (OHB) Suite - RPC

- Two different micro-benchmarks to evaluate the performance of standalone Hadoop RPC
  - Latency: Single Server, Single Client
  - Throughput: Single Server, Multiple Clients
- A simple script framework for job launching and resource monitoring
- Calculates statistics like Min, Max, Average
- Network configuration, Tunable parameters, DataType, CPU Utilization

<table>
<thead>
<tr>
<th>Component</th>
<th>Network Address</th>
<th>Port</th>
<th>Data Type</th>
<th>Min Msg Size</th>
<th>Max Msg Size</th>
<th>No. of Iterations</th>
<th>Handlers</th>
<th>Verbose</th>
</tr>
</thead>
<tbody>
<tr>
<td>lat_client</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>lat_server</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Network Address</th>
<th>Port</th>
<th>Data Type</th>
<th>Min Msg Size</th>
<th>Max Msg Size</th>
<th>No. of Iterations</th>
<th>No. of Clients</th>
<th>Handlers</th>
<th>Verbose</th>
</tr>
</thead>
<tbody>
<tr>
<td>thr_client</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>thr_server</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

OSU HiBD Micro-Benchmark (OHB) Suite - Memcached

- Evaluates the performance of stand-alone Memcached in different modes
- Default API Latency benchmarks for Memcached in-memory mode
  - SET Micro-benchmark: Micro-benchmark for memcached set operations
  - GET Micro-benchmark: Micro-benchmark for memcached get operations
  - MIX Micro-benchmark: Micro-benchmark for a mix of memcached set/get operations
    (Read:Write ratio is 90:10)
- Latency benchmarks for Memcached hybrid-memory mode
- Non-Blocking API Latency Benchmark for Memcached (both in-memory and hybrid-memory mode)
- Calculates average latency of Memcached operations in different modes

**Different Modes of RDMA for Apache Hadoop 2.x**

- **HHH**: Heterogeneous storage devices with hybrid replication schemes are supported in this mode of operation to have better fault-tolerance as well as performance. This mode is enabled by default in the package.

- **HHH-M**: A high-performance in-memory based setup has been introduced in this package that can be utilized to perform all I/O operations in-memory and obtain as much performance benefit as possible.

- **HHH-L**: With parallel file systems integrated, HHH-L mode can take advantage of the Lustre available in the cluster.

- **HHH-L-BB**: This mode deploys a Memcached-based burst buffer system to reduce the bandwidth bottleneck of shared file system access. The burst buffer design is hosted by Memcached servers, each of which has a local SSD.

- **MapReduce over Lustre, with/without local disks**: Besides, HDFS based solutions, this package also provides support to run MapReduce jobs on top of Lustre alone. Here, two different modes are introduced: with local disks and without local disks.

- **Running with Slurm and PBS**: Supports deploying RDMA for Apache Hadoop 2.x with Slurm and PBS in different running modes (HHH, HHH-M, HHH-L, and MapReduce over Lustre).
Using HiBD Packages for Big Data Processing on Existing HPC Infrastructure

Hadoop Job with HiBD
- HHH (-M, -L, -BB-L)
- RDMA-MapReduce (over Lustre)
- HBase, Hive, Pig, etc.

MPI Job

Spark Job

Deep Learning Job
RDMA for Apache Spark Distribution

- High-Performance Design of Spark over RDMA-enabled Interconnects
  - High performance RDMA-enhanced design with native InfiniBand and RoCE support at the verbs-level for Spark
  - RDMA-based data shuffle and SEDA-based shuffle architecture
  - Non-blocking and chunk-based data transfer
  - Off-JVM-heap buffer management
  - Support for OpenPOWER
  - Easily configurable for different protocols (native InfiniBand, RoCE, and IPoIB)

- Current release: 0.9.5
  - Based on Apache Spark 2.1.0
  - Tested with
    - Mellanox InfiniBand adapters (DDR, QDR, FDR, and EDR)
    - RoCE support with Mellanox adapters
    - Various multi-core platforms (x86, POWER)
    - RAM disks, SSDs, and HDD
  - [http://hibd.cse.ohio-state.edu](http://hibd.cse.ohio-state.edu)
Using HiBD Packages for Big Data Processing on Existing HPC Infrastructure

- Spark Job with HiBD
  - RDMA-Spark
  - Integration with HHH
  - Spark SQL, MLlib, etc.
Presentation Overview

• MVAPICH Project
  – MPI and PGAS Library with CUDA-Awareness

• HiBD Project
  – High-Performance Big Data Analytics Library

• **HiDL Project**
  – High-Performance Deep Learning

• Public Cloud Deployment
  – Microsoft-Azure and Amazon-AWS

• Conclusions
Deep Learning: New Challenges for MPI Runtimes

- Deep Learning frameworks are a different game altogether
  - Unusually large message sizes (order of megabytes)
  - Most communication based on GPU buffers
- Existing State-of-the-art
  - cuDNN, cuBLAS, NCCL --> scale-up performance
  - NCCL2, CUDA-Aware MPI --> scale-out performance
    - For small and medium message sizes only!
- Proposed: Can we co-design the MPI runtime (MVAPICH2-GDR) and the DL framework (Caffe) to achieve both?
  - Efficient Overlap of Computation and Communication
  - Efficient Large-Message Communication (Reductions)
  - What application co-designs are needed to exploit communication-runtime co-designs?

Convergent Software Stacks for HPC, Big Data and Deep Learning

HPC (MPI, RDMA, Lustre, etc.)

Big Data (Hadoop, Spark, HBase, Memcached, etc.)

Deep Learning (Caffe, TensorFlow, BigDL, etc.)

MVAPICH2
MVAPICH2-X
MVAPICH2-GDR

RDMA-Hadoop
RDMA-Spark
RDMA-HBase
RDMA-Memcached
RDMA-Kafka
High-Performance Deep Learning

- CPU-based Deep Learning
  - Using MVAPICH2-X

- GPU-based Deep Learning
  - Using MVAPICH2-GDR
Large-Scale Benchmarking of DL Frameworks on Frontera

- TensorFlow, PyTorch, and MXNet are widely used Deep Learning Frameworks
- Optimized by Intel using Math Kernel Library for DNN (MKL-DNN) for Intel processors
- Single Node performance can be improved by running Multiple MPI processes

Impact of Batch Size on Performance for ResNet-50

Performance Improvement using Multiple MPI processes

ResNet-50 using various DL benchmarks on Frontera

- Observed 260K images per sec for ResNet-50 on 2,048 Nodes
- Scaled MVAPICH2-X on 2,048 nodes on Frontera for Distributed Training using TensorFlow
- ResNet-50 can be trained in 7 minutes on 2048 nodes (114,688 cores)

Benchmarking TensorFlow (TF) and PyTorch

- Comprehensive and systematic performance benchmarking
  - tf_cnn_benchmarks (TF)
  - Horovod benchmark (PyTorch)
- TensorFlow is up to 2.5X faster than PyTorch for 128 Nodes.
- TensorFlow: up to 125X speedup for ResNet-152 on 128 nodes
- PyTorch: Scales well but overall lower performance than TensorFlow

Benchmarking HyPar-Flow on Stampede

- CPU based Hybrid-Parallel (Data Parallelism and Model Parallelism) training on Stampede2
- Benchmark developed for various configuration
  - Batch sizes
  - No. of model partitions
  - No. of model replicas
- Evaluation on a very deep model
  - ResNet-1000 (a 1,000-layer model)

110x speedup on 128 Intel Xeon Skylake nodes (TACC Stampede2 Cluster)

High-Performance Deep Learning

- **CPU-based Deep Learning**
  - Using MVAPICH2-X

- **GPU-based Deep Learning**
  - Using MVAPICH2-GDR
Distributed Training with TensorFlow and MVAPICH2-GDR

- ResNet-50 Training using TensorFlow benchmark on SUMMIT -- 1536 Volta GPUs!

- 1,281,167 (1.2 mil.) images

- Time/epoch = 3.6 seconds

- Total Time (90 epochs) = 3.6 x 90 = 332 seconds = 5.5 minutes!

*We observed errors for NCCL2 beyond 96 GPUs

Platform: The Summit Supercomputer (#1 on Top500.org) -- 6 NVIDIA Volta GPUs per node connected with NVLink, CUDA 9.2
New Benchmark for Image Segmentation on Summit

- Near-linear scaling may be achieved by tuning Horovod/MPI
  - Optimizing MPI/Horovod towards large message sizes for high-resolution images
- Develop a generic Image Segmentation benchmark
- Tuned DeepLabV3+ model using the benchmark and Horovod – up to 1.3X better than default

*Anthony et al., “Scaling Semantic Image Segmentation using Tensorflow and MVAPICH2-GDR on HPC Systems” (Submission under review)*
Using HiDL Packages for Deep Learning on Existing HPC Infrastructure

Deep Learning Jobs
- MVAPICH2-X for CPUs
- MVAPICH2-GDR for GPUs
- Both use Horovod for TF and PyTorch

MPI Job

Spark Job

Hadoop Job
Presentation Overview

- MVAPICH Project
  - MPI and PGAS Library with CUDA-Awareness
- HiBD Project
  - High-Performance Big Data Analytics Library
- HiDL Project
  - High-Performance Deep Learning
- Public Cloud Deployment
  - Microsoft-Azure and Amazon-AWS
- Conclusions
• Released on 08/16/2019
• Major Features and Enhancements
  – Based on MVAPICH2-2.3.2
  – Enhanced tuning for point-to-point and collective operations
  – Targeted for Azure HB & HC virtual machine instances
  – Flexibility for 'one-click' deployment
  – Tested with Azure HB & HC VM instances
MVAPICH2-X-AWS 2.3

- Released on 08/12/2019
- Major Features and Enhancements
  - Based on MVAPICH2-X 2.3
  - New design based on Amazon EFA adapter’s Scalable Reliable Datagram (SRD) transport protocol
  - Support for XPMEM based intra-node communication for point-to-point and collectives
  - Enhanced tuning for point-to-point and collective operations
  - Targeted for AWS instances with Amazon Linux 2 AMI and EFA support
  - Tested with c5n.18xlarge instance
Concluding Remarks

• Upcoming Exascale systems need to be designed with a holistic view of HPC, Big Data, Deep Learning, and Cloud
• Presented an overview of designing convergent software stacks
• Presented benchmarks and middleware to enable HPC, Big Data, and Deep Learning communities to take advantage of current and next-generation systems
Commercial Support for MVAPICH2, HiBD, and HiDL Libraries

• Supported through X-ScaleSolutions (http://x-scalesolutions.com)
• Benefits:
  – Help and guidance with installation of the library
  – Platform-specific optimizations and tuning
  – Timely support for operational issues encountered with the library
  – Web portal interface to submit issues and tracking their progress
  – Advanced debugging techniques
  – Application-specific optimizations and tuning
  – Obtaining guidelines on best practices
  – Periodic information on major fixes and updates
  – Information on major releases
  – Help with upgrading to the latest release
  – Flexible Service Level Agreements
• Support provided to Lawrence Livermore National Laboratory (LLNL) for the last two years
Silver ISV Member for the OpenPOWER Consortium + Products

- Has joined the OpenPOWER Consortium as a silver ISV member
- Provides flexibility:
  - To have MVAPICH2, HiDL and HiBD libraries getting integrated into the OpenPOWER software stack
  - A part of the OpenPOWER ecosystem
  - Can participate with different vendors for bidding, installation and deployment process
- Introduced two new integrated products with support for OpenPOWER systems (Presented at the OpenPOWER North America Summit)
  - X-ScaleHPC
  - X-ScaleAI
  - Send an e-mail to contactus@x-scalesolutions.com for free trial!!
Multiple Events at SC ‘19

- Presentations at OSU and X-Scale Booth (#2094)
  - Members of the MVAPICH, HiBD and HiDL members
  - External speakers
- Presentations at SC main program (Tutorials and Workshops)
- Presentation at many other booths and satellite events
- Complete details available at
  [http://mvapich.cse.ohio-state.edu/conference/752/talks/](http://mvapich.cse.ohio-state.edu/conference/752/talks/)
Funding Acknowledgments

Funding Support by

Equipment Support by
Personnel Acknowledgments

**Current Students (Graduate)**
- A. Awan (Ph.D.)
- M. Bayatpour (Ph.D.)
- C.-H. Chu (Ph.D.)
- J. Hashmi (Ph.D.)
- A. Jain (Ph.D.)
- K. S. Kandadi (M.S.)
- 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.)
- S. Chakraborthy (Ph.D.)
- 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. Kulikarni (M.S.)
- R. Kumar (M.S.)
- S. Krishnamoorthy (M.S.)
- K. Kandalla (Ph.D.)
- M. Li (Ph.D.)
- J. Lin
- M. Luo
- E. Mancini
- Q. Zhou (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. Pai (M.S.)
- S. Pottiuri (Ph.D.)

**Past Students**
- D. Banerjee
- X. Besseron
- H.-W. Jin
- 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.)
- S. Chakraborthy (Ph.D.)
- N. Dandapanthula (M.S.)
- V. Dhanraj (M.S.)

**Past Research Scientists**
- K. Hamidouche
- S. Sur
- X. Lu

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

**Past Post-Docs**
- J. Lin
- M. Luo
- E. Mancini
- S. Marcarelli
- J. Vienne
- H. Wang

**Current Research Scientist**
- H. Subramoni
- M. S. Ghazimeersaeed
- A. Ruhela
- K. Manian

**Current Students (Undergraduate)**
- V. Gangal (B.S.)
- N. Sarkauskas (B.S.)
- R. Rajachandrasekar (Ph.D.)
- D. Shankar (Ph.D.)
- G. Santhanaraman (Ph.D.)
- A. Singh (Ph.D.)
- J. Sridhar (M.S.)
- S. Sur (Ph.D.)
- H. Subramoni (Ph.D.)
- K. Vaidyanathan (Ph.D.)
- A. Vishnu (Ph.D.)
- J. Wu (Ph.D.)
- W. Yu (Ph.D.)
- J. Zhang (Ph.D.)
Thank You!

panda@cse.ohio-state.edu