Designing High Performance Scalable Middleware for HPC/AI Exascale Systems and Clouds

Keynote Talk at HPC-AI Advisory Council Stanford Conference (May ‘21)

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

- Expected to have an ExaFlop system in 2021!

100 PetaFlops in 2017

442 PetaFlops in 2020 (Fugaku in Japan with 7.63M cores)

1 ExaFlops
Increasing Usage of HPC, Big Data and Deep/Machine Learning

Convergence of HPC, Big Data, and Deep/Machine Learning!

Increasing Need to Run these applications on the Cloud!!
Presentation Overview

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

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

• HiBD Project
  – Accelerating Data Science Applications with Dask

• Optimizations and Deployments in Public Cloud
  – AWS and Azure

• Commercial Support and Value Added Products

• Conclusions
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 (NVIDIA and AMD)
• 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 3,150 organizations in 89 countries
• More than 1.35 Million downloads from the OSU site directly
• Empowering many TOP500 clusters (Nov ‘20 ranking)
  – 4th, 10,649,600-core (Sunway TaihuLight) at NSC, Wuxi, China
  – 9th, 448,448 cores (Frontera) at TACC
  – 14th, 391,680 cores (ABCI) in Japan
  – 21st, 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 9th ranked TACC Frontera system
• Empowering Top500 systems for more than 16 years
Architecture of MVAPICH2 Software Family for HPC and DL/ML

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, Elastic Fabric Adapter)

- Transport Protocols: RC, SRD, 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: 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/Machine 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>
Startup Performance on TACC Frontera

- MPI_Init takes 31 seconds on 229,376 processes on 4,096 nodes
- All numbers reported with 56 processes per node

New designs available since MVAPICH2-2.3.4
Hardware Multicast-aware MPI_Bcast on TACC Frontera

- MCAST-based designs improve latency of MPI_Bcast by up to **2X** at 2,048 nodes
- Use MV2_USE_MCAST=1 to enable MCAST-based designs

Size=16B, PPN=28

\[ \text{Size}=16B, \text{PPN}=28 \]

\[ 1.8X \]

Size=32kB, PPN=28

\[ \text{Size}=32kB, \text{PPN}=28 \]

\[ 2X \]
Performance of Collectives with SHARP on TACC Frontera

**Optimized SHARP designs in MVAPICH2-X**

*Up to 9X* performance improvement with SHARP over MVAPICH2-X default for 1ppn MPI_Barrier, *6X* for 1ppn MPI_Reduce and *5X* for 1ppn MPI_Allreduce

Support for ARM A64FX with InfiniBand (Ookami)

**Inter-Node Latency**

Latency (us) vs Message Size (Bytes)

- MVAPICH2-X
- OpenMPI

**Inter-Node Bidirectional Bandwidth**

Bandwidth (MB/s) vs Message Size (Bytes)

- MVAPICH2-X
- OpenMPI

**Intra-Node Latency**

Latency (us) vs Message Size (Bytes)

- MVAPICH2-X
- OpenMPI

**MPI_Bcast (8 nodes 48 PPN)**

Latency (us) vs Message Size (Bytes)

- MVAPICH2-X
- OpenMPI
Performance of Neuroscience Mini-Application with MVAPICH2-X

Comparison of Execution Time (s) on 9,920 cores

<table>
<thead>
<tr>
<th>Neighbors Selection</th>
<th>HPE-MPI</th>
<th>MVAPICH2-X</th>
<th>% Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>188.07</td>
<td>120.22</td>
<td>36.07</td>
</tr>
<tr>
<td>Consecutive</td>
<td>124.46</td>
<td>116.34</td>
<td>6.52</td>
</tr>
</tbody>
</table>

- EPFL Mini-Application is a benchmark where each process sends data to a selected set of neighbors.
- Up to 36% benefits over HPE-MPI on Neuroscience Mini-Application at 496 nodes and 20 PPN

Available in upcoming version of MVAPICH2-X
Optimized MVAPICH2-GDR with CUDA-Aware MPI Support

**GPU-GPU Inter-node Latency**

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

**GPU-GPU Inter-node Bandwidth**

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

**GPU-GPU Inter-node Bi-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

---

**Network Based Computing Laboratory**

HPC-AI-Switzerland (May ‘21)
On-the-fly GPU Compression (Benefits with AWP-ODC)

- Weak-Scaling of HPC application **AWP-ODC** on Lassen cluster (V100 nodes)
- MPC-OPT achieves up to **+18%** GPU computing flops, **-15%** runtime per timestep
- ZFP-OPT achieves up to **+35%** GPU computing flops, **-26%** runtime per timestep (rate=8, compression ratio=4)

On-the-fly GPU Compression (Benefits with DASK)

- Data science framework **DASK** on RI2 cluster (V100 nodes)
- Dask benchmark creates cuPy array and distributes its chunks across Dask workers
- ZFP-OPT achieves up to **1.56x** throughput, **-37%** runtime (rate=8, compression ratio=4)

![Graph showing aggregate throughput and execution time for different number of Dask workers and compression rates.](image)

MVAPICH2-GDR ROCm Support for AMD GPUs

Intra-Node Point-to-Point Latency

- **Allreduce – 128 GPUs (16 nodes, 8 GPUs Per Node)**
  - MVAPICH2-GDR
  - OpenMPI 4.1.0 + UCX 1.10

  **Latency (us)**
  - MVAPICH2-GDR: 1.73 us
  - OpenMPI 4.1.0 + UCX 1.10: 1.88 us

Inter-Node Point-to-Point Latency

- **Bcast – 128 GPUs (16 nodes, 8 GPUs Per Node)**
  - MVAPICH2-GDR
  - OpenMPI 4.1.0 + UCX 1.10

  **Latency (us)**
  - MVAPICH2-GDR: 3.5 us
  - OpenMPI 4.1.0 + UCX 1.10: 4.01 us

Corona Cluster - ROCm-4.1.0 (mi50 AMD GPUs)

Available with MVAPICH2-GDR 2.3.5
Exploiting BlueField-2 DPU for Offloading MPI Functions: Accelerating P3DFFT

M. Bayatpour, N. Sarkauskas, H. Subramoni, J. M. Hashmi, and D. K. Panda, BluesMPI: Efficient MPI Non-blocking Alltoall offloading Designs on Modern BlueField Smart NICs, Int'l Supercomputing Conference (ISC '21), Accepted to be Presented.
Presentation Overview

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

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

• **HiBD Project**
  – Accelerating Data Science Applications with Dask

• **Optimizations and Deployments in Public Cloud**
  – AWS and Azure

• **Commercial Support and Value Added Products**

• **Conclusions**
MVAPICH2 (MPI)-driven Infrastructure for ML/DL Training

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

- MPI-driven Deep Learning with Data Parallelism
- Out-of-core DNN training
- Exploiting Hybrid (Data and Model) Parallelism
- Use-Case: AI-Driven Digital Pathology
- Accelerating CuML Applications
Distributed TensorFlow on TACC Frontera (2,048 CPU nodes with 114,688 cores)

- Scaled TensorFlow to 2048 nodes on Frontera using MVAPICH2
- MVAPICH2 and IntelMPI give similar performance for DNN training
- Report a peak of 260,000 images/sec on 2,048 nodes
- On 2048 nodes, ResNet-50 can be trained in 7 minutes!

MVAPICH2-GDR vs. NCCL2 – Allreduce Operation (DGX-2)

- Optimized designs in 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 10.1

MVAPICH2-GDR: MPI_Allreduce at Scale (ORNL Summit)

- Optimized designs in 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 TensorFlow on ORNL Summit (1,536 GPUs)

- ResNet-50 Training using TensorFlow benchmark on SUMMIT -- 1536 Volta GPUs!
- 1,281,167 (1.2 mil.) images
- Time/epoch = 3 seconds
- Total Time (90 epochs) = 3 x 90 = 270 seconds = 4.5 minutes!

*We observed issues for NCCL2 beyond 384 GPUs

Platform: The Summit Supercomputer (#2 on Top500.org) – 6 NVIDIA Volta GPUs per node connected with NVLink, CUDA 10.1

ImageNet-1k has 1.2 million images
MVAPICH2-GDR reaching ~0.42 million images per second for ImageNet-1k!
PyTorch at Scale: Training ResNet-50 on 256 V100 GPUs

- Training performance for 256 V100 GPUs on LLNL Lassen
  - ~10,000 Images/sec faster than NCCL training!

<table>
<thead>
<tr>
<th>Distributed Framework</th>
<th>Torch.distributed</th>
<th>Horovod</th>
<th>DeepSpeed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Images/sec on 256 GPUs</td>
<td>61,794</td>
<td>74,063</td>
<td>80,217</td>
</tr>
<tr>
<td></td>
<td>72,120</td>
<td>84,659</td>
<td>88,873</td>
</tr>
<tr>
<td>Communication Backend</td>
<td>NCCL 2.7</td>
<td>NCCL 2.7</td>
<td>NCCL 2.7</td>
</tr>
<tr>
<td></td>
<td>MVAPICH2-GDR</td>
<td>MVAPICH2-GDR</td>
<td>MVAPICH2-GDR</td>
</tr>
</tbody>
</table>
Multiple Approaches taken up by OSU

- MPI-driven Deep Learning with Data Parallelism
- Out-of-core DNN training
- Exploiting Hybrid (Data and Model) Parallelism
- Use-Case: AI-Driven Digital Pathology
- Accelerating CuML Applications
Out-of-Core Training and Hybrid Parallelism (HyPar-Flow)

- Why Hybrid parallelism?
  - Data Parallel training has limits!
- We propose HyPar-Flow
  - An easy-to-use Hybrid parallel training framework
    - Hybrid = Data + Model
  - Supports Keras models and exploits TF 2.0 Eager Execution
  - Exploits MPI for Point-to-point and Collectives

Benchmarking large-models lead to better insights and ability to develop new approaches!

HyPar-Flow at Scale (512 nodes on TACC Frontera)

- ResNet-1001 with variable batch size
- Approach:
  - 48 model-partitions for 56 cores
  - 512 model-replicas for 512 nodes
  - Total cores: 56 x 512 = 28,672
- Speedup
  - **253X** on 256 nodes
  - **481X** on 512 nodes
- Scaling Efficiency
  - **98%** up to 256 nodes
  - **93.9%** for 512 nodes

Why do we need Memory aware designs?
- Data and Model Parallel training has limitation!
- Maximum Batch Size depends on the memory
- Basic Model Parallelism suffers from underutilization of memory and compute

Memory requirement increases with the increase in image size!

GEMS at Scale (1,024 V100 GPUs on LLNL Lassen)

- Two Approaches:
  - Memory Aware Synchronized Training (MAST)
  - Memory Aware Synchronized Training with Enhanced Replications (MASTER)
- Setup
  - ResNet-1k on 512 X 512 images
  - 128 Replications on 1024 GPUs
- Scaling Efficiency
  - 97.32% on 1024 nodes

SUPER: **Sub-Graph Parallelism for Transform**ER**s**

Sub-Graph Parallelism

- Exploits inherent parallelism in modern DNN architectures
- Improves the Performance of multi-branch DNN architectures
- Can be used to accelerate the training of state-of-the-art Transformer models
- Provides better performance than Data-Parallelism for in-core models

**Simple example of a multi-branch DNN architecture**

4-way **Sub-Graph Parallelism combined with Data-Parallelism (D&SP)**

A. Jain, T. moon, T. Benson, H. Subramoni, S. Jacobs, D. Panda, B. Essen, “SUPER: Sub-Graph Parallelism for Transform**E**Rs”, IPDPS ’21, to be presented
Accelerating Transformers using SUPER

- We propose sub-graph parallelism integrated with data parallelism to accelerate the training of Transformers.

- Approach
  - Data and Sub-Graph Parallelism (D&SP)
    - #-way D&SP (#: number of sub-graphs)

- Setup
  - T5-Large-Mod on WMT Dataset
  - 1,024 NVIDIA V100 GPUs

- Speedup
  - Up to 3.05X over Data Parallelism (DP)

---

Multiple Approaches taken up by OSU

- MPI-driven Deep Learning with Data Parallelism
- Out-of-core DNN training
- Exploiting Hybrid (Data and Model) Parallelism
- Use-Case: AI-Driven Digital Pathology
- Accelerating CuML Applications
Digital Pathology

• Whole Slide Images (WSI)
  – Replacing the glass slide for diagnostic purposes
  – Typically, 100,000 X 100,000 pixels in size

A whole slide image (WSI)
A Hematoxylin and Eosin stained whole slide image labeled as Tall Cell Variant (TCV) of the papillary thyroid cancer (PTC).

A tile at 10x magnification level
A 1024x1024 image tile at 10 magnification level shows histologic feature of elongated follicles arranged in parallel cords or tram tracks.

A tile at 20x magnification level
A 1024x1024 image tile at 20 magnification level shows cellular features of tall cells.
AI-Driven Digital Pathology

- Traditionally, a pathologist reviews a slide and carries out diagnosis based on prior knowledge and training
- Experience matters
- Can Deep Learning to be used to train thousands of slides and
  - Let the computer carry out the diagnosis
  - Narrow down the diagnosis and help a pathologist to make a final decision
- Significant benefits in
  - Reducing diagnosis time
  - Making pathologists productive
  - Reducing health care cost
Exploiting Model Parallelism in AI-Driven Digital Pathology

- Pathology whole slide image (WSI)
  - Each WSI = 100,000 x 100,000 pixels
  - Can not fit in a single GPU memory
  - Tiles are extracted to make training possible

- Two main problems with tiles
  - Restricted tile size because of GPU memory limitation
  - Smaller tiles loose structural information

- Reduced training time significantly
  - GEMS-Basic: 7.25 hours (1 node, 4 GPUs)
  - GEMS-MAST: 6.28 hours (1 node, 4 GPUs)
  - GEMS-MASTER: 4.21 hours (1 node, 4 GPUs)
  - GEMS-Hybrid: 0.46 hours (32 nodes, 128 GPUs)
  - Overall 15x reduction in training time!!!!


Scaling ResNet110 v2 on 1024×1024 image tiles using histopathology data

Throughput Speedup (images per sec) vs Number of GPUs

0 5 10 15 20 25
4 8 16 32 64 128
1x 1.9x 3.6x 7x 12x 22x
Multiple Approaches taken up by OSU

- MPI-driven Deep Learning with Data Parallelism
- Out-of-core DNN training
- Exploiting Hybrid (Data and Model) Parallelism
- Use-Case: AI-Driven Digital Pathology
- Accelerating CuML Applications
Accelerating cuML with MVAPICH2-GDR

- Utilize MVAPICH2-GDR (with mpi4py) as communication backend during the training phase (the fit() function) for Multi-node Multi-GPU (MNMG) setting over cluster of GPUs

- Communication primitives:
  - Allreduce
  - Reduce
  - Broadcast

- Exploit optimized collectives
MPI4cuML 0.1 Release

- MPI4cuML 0.1 was released in Feb ‘21 adding support for high-performance MPI communication to cuML:
  - Can be downloaded from: http://hidl.cse.ohio-state.edu

- Features:
  - Built with Python 3.7, CUDA 10.1, 10.2 or 11.0
  - Optimized support at MPI-level for machine learning workloads
    - Efficient large-message and small-message collectives (e.g. Allreduce and Bcast) on GPUs
    - GPU-Direct Algorithms for all collective operations (e.g. Allgather and Alltoall)
    - Support for fork safety
  - Exploits efficient large-message and small-message collectives in MVAPICH2-GDR
  - Tested with
    - Mellanox InfiniBand adapters (FDR and HDR)
    - NVIDIA GPU P100 and V100
    - Various x86-based multi-core platforms (AMD and Intel)
Presentation Overview

- MVAPICH Project
  - MPI and PGAS (MVAPICH) Library with CUDA-Awareness
- HiDL Project
  - High-Performance Deep Learning
  - High-Performance Machine Learning
- **HiBD Project**
  - Accelerating Data Science Applications with Dask
- Optimizations and Deployments in Public Cloud
  - AWS and Azure
- Commercial Support and Value Added Products
- Conclusions
Dask Architecture

- Dask Bag
- Dask Array
- Dask DataFrame
- Delayed
- Future

Task Graph

Dask-MPI
Dask-CUDA
Dask-Jobqueue

Distributed
- Scheduler
- Worker
- Client

Comm Layer
- tcp.py
- ucx.py
- MPI4Dask
- UCX-Py (Cython wrappers)
- mpi4py
- MVAPICH2-GDR

Laptops/Desktops

High Performance Computing Hardware
MPI4Dask Release

- MPI4Dask 0.2 was released in Mar ‘21 adding support for high-performance MPI communication to Dask:
  - Can be downloaded from: http://hibd.cse.ohio-state.edu

- Features:
  - Based on Dask Distributed 2021.01.0
  - Compliant with user-level Dask APIs and packages
  - Support for MPI-based communication in Dask for cluster of GPUs
  - Implements point-to-point communication co-routines
  - Efficient chunking mechanism implemented for large messages
  - (NEW) Built on top of mpi4py over the MVAPICH2, MVAPICH2-X, and MVAPICH2-GDR libraries
  - (NEW) Support for MPI-based communication for CPU-based Dask applications
  - Supports starting execution of Dask programs using Dask-MPI
  - Tested with
    - (NEW) CPU-based Dask applications using numPy and Pandas data frames
    - (NEW) GPU-based Dask applications using cuPy and cuDF
    - Mellanox InfiniBand adapters (FDR and EDR)
    - Various multi-core platforms
    - NVIDIA V100 and Quadro RTX 5000 GPUs
Benchmark #1: Sum of cuPy Array and its Transpose (RI2)

3.47x better on average

6.92x better on average


MPI4Dask 0.2 release
(http://hibd.cse.ohio-state.edu)
Benchmark #2: cuDF Merge (TACC Frontera GPU Subsystem)

2.91x better on average

2.90x better on average


MPI4Dask 0.2 release
(http://hibd.cse.ohio-state.edu)
Presentation Overview

- **MVAPICH Project**
  - MPI and PGAS (MVAPICH) Library with CUDA-Awareness

- **HiDL Project**
  - High-Performance Deep Learning
  - High-Performance Machine Learning

- **HiBD Project**
  - Accelerating Data Science Applications with Dask

- **Optimizations and Deployments in Public Cloud**
  - AWS and Azure

- **Commercial Support and Value Added Products**

- **Conclusions**
MVAPICH2-Azure Deployment

- Released on 05/20/2020
- Integrated Azure CentOS HPC Images
  - [https://github.com/Azure/azhpc-images/releases/tag/centos-7.6-hpc-20200417](https://github.com/Azure/azhpc-images/releases/tag/centos-7.6-hpc-20200417)
- MVAPICH2 2.3.3
  - CentOS Images (7.6, 7.7 and 8.1)
  - Tested with multiple VM instances
- MVAPICH2-X 2.3.RC3
  - CentOS Images (7.6, 7.7 and 8.1)
  - Tested with multiple VM instances
- More details from Azure Blog Post
WRF Application Results on HBv2 (AMD Rome)

- Performance of WRF with MVAPICH2 and MVAPICH2-X-XPMEM

- WRF 3.6
  - [https://github.com/hanschen/WRFV3](https://github.com/hanschen/WRFV3)

- Benchmark: 12km resolution case over the Continental U.S. (CONUS) domain
  - [https://www2.mmm.ucar.edu/wrf/WG2/benchv3/#_Toc212961288](https://www2.mmm.ucar.edu/wrf/WG2/benchv3/#_Toc212961288)

- Update io_form_history in namelist.input to 102
  - [https://www2.mmm.ucar.edu/wrf/users/namelist_best_prac_wrf.html#io_form_history](https://www2.mmm.ucar.edu/wrf/users/namelist_best_prac_wrf.html#io_form_history)

MVAPICH2-X-XPMEM is able to deliver better performance and scalability
MVAPICH2-X-AWS 2.3

• Released on 09/24/2020

• Major Features and Enhancements
  – Based on MVAPICH2-X 2.3
  – Improved inter-node latency and bandwidth performance for large messages
  – Optimized collectives
  – Support for dynamic run-time XPMEM module detection
  – Support for currently available basic OS types on AWS EC2 including: Amazon Linux 1/2, CentOS 6/7, Ubuntu 16.04/18.04
WRF Application Results

- Performance of WRF with Open MPI 4.0.3 vs Intel MPI 2019.7.217 vs MVAPICH2-X-AWS v2.3
- Run on c5n.18xlarge instances with libfabric 1.10
Presentation Overview

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

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

• HiBD Project
  – Accelerating Data Science Applications with Dask

• Optimizations and Deployments in Public Cloud
  – AWS and Azure

• Commercial Support and Value Added Products

• Conclusions
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 being provided to National Laboratories and International Supercomputing Centers
X-ScaleAI Product and Features

- **Aim**: High-performance solution for distributed training for your complex AI problems on modern HPC platforms

- **Features**:
  - Powered by MVAPICH2 libraries
  - Great performance and scalability as delivered by MVAPICH2 libraries
  - Integrated packaging to run various Deep Learning Frameworks (TensorFlow, PyTorch, MXNet, and others)
  - Targeted for both CPU-based and GPU-based Deep Learning Training
  - **Integrated profiling and introspection support for Deep Learning Applications across the stacks (DeepIntrospect)**
    - Provides cross-stack performance analysis in a visual manner and help users to optimize their DL applications and harness higher performance and scalability
  - **Out-of-the-box optimal performance**
    - Tuned for various CPU- and GPU-based HPC systems
  - **One-click deployment and execution**
    - Do not need to struggle for many hours
  - Support for x86 and OpenPOWER platforms
  - Support for InfiniBand, RoCE and NVLink Interconnects
Concluding Remarks

• Upcoming Exascale systems and Cloud need to be designed with a holistic view of HPC, Big Data, and Deep/Machine Learning

• Presented an overview of designing convergent software stacks for HPC and Deep/Machine Learning

• Presented solutions which enable HPC and Deep/Machine Learning communities to take advantage of current and next-generation systems

• Next-generation Exascale and Zetascale systems will need continuous innovations in designing converged software architectures .....
Funding Acknowledgments

Funding Support by

[Logos of various companies]

Equipment Support by

[Logos of various companies]
Acknowledgments to all the Heroes (Past/Current Students and Staffs)

**Current Students (Graduate)**
- Q. Anthony (Ph.D.)
- M. Bayatpour (Ph.D.)
- C.-C. Chun (Ph.D.)
- A. Jain (Ph.D.)
- K. S. Khorassani (Ph.D.)
- P. Kousha (Ph.D.)
- N. Sarkauskas (Ph.D.)

**Current Research Scientists**
- A. Shafi
- H. Subramoni

**Current Software Engineers**
- N. Shineman

**Current Research Specialist**
- J. Smith

**Past Students**
- A. Awan (Ph.D.)
- 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. Chakraborty (Ph.D.)
- N. Dandapanthula (M.S.)
- V. Dhanraj (M.S.)
- C.-H. Chu (Ph.D.)
- T. Gangadhara (M.S.)
- K. Gopakrishnan (M.S.)
- J. Hashmi (Ph.D.)
- W. Huang (Ph.D.)
- W. Jiang (M.S.)
- J. Jose (Ph.D.)
- M. Kedia (M.S.)
- 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. Pai (M.S.)
- S. Potluri (Ph.D.)
- K. Raj (M.S.)
- R. Rajachandrasekar (Ph.D.)
- D. Shankar (Ph.D.)
- G. Santanaraman (Ph.D.)
- N. Sarkauskas (B.S.)
- 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.)

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

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

**Past Research Specialist**
- M. Arnold

**Past Post-Docs**
- D. Banerjee
- X. Besseron
- M. S. Ghazimeersaedd
- H.-W. Jin
- J. Lin
- M. Luo
- E. Mancini
- K. Manian
- M. Luo
- A. Ruhela
- J. Vienne
- H. Wang

Network Based Computing Laboratory
HPC-AI-Switzerland (May ‘21)
Thank You!

panda@cse.ohio-state.edu

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

Follow us on
https://twitter.com/mvapich

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/