Overview of the MVAPICH Project: Latest Status and Future Roadmap

MVAPICH2 User Group (MUG) Meeting

by

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

- Programming models provide abstract machine models
- Models can be mapped on different types of systems
  - e.g. Distributed Shared Memory (DSM), MPI within a node, etc.
- PGAS models and Hybrid MPI+PGAS models are gradually receiving importance
Supporting Programming Models for Multi-Petaflop and Exaflop Systems: Challenges

Application Kernels/Applications

Middleware

Programming Models
MPI, PGAS (UPC, Global Arrays, OpenSHMEM), CUDA, OpenMP, OpenACC, Cilk, Hadoop (MapReduce), Spark (RDD, DAG), etc.

Communication Library or Runtime for Programming Models
Point-to-point Communication
Collective Communication
Energy-Awareness
Synchronization and Locks
I/O and File Systems
Fault Tolerance

Networking Technologies
(InfiniBand, 40/100GigE, Aries, and Omni-Path)

Multi-/Many-core Architectures

Accelerators (GPU and MIC)

Co-Design Opportunities and Challenges across Various Layers
Performance
Scalability
Resilience

Network Based Computing Laboratory

MVAPICH User Group Meeting (MUG) 2017
Designing (MPI+X) for Exascale

- Scalability for million to billion processors
  - Support for highly-efficient inter-node and intra-node communication (both two-sided and one-sided)

- Scalable Collective communication
  - Offloaded
  - Non-blocking
  - Topology-aware

- Balancing intra-node and inter-node communication for next generation multi-/many-core (128-1024 cores/node)
  - Multiple end-points per node

- Support for efficient multi-threading

- Integrated Support for GPGPUs and Accelerators

- Fault-tolerance/resiliency

- QoS support for communication and I/O

- Support for Hybrid MPI+PGAS programming
  - MPI + OpenMP, MPI + UPC, MPI + OpenSHMEM, CAF, MPI + UPC++...

- Virtualization

- Energy-Awareness
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-3.0), 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 2,800 organizations in 85 countries
  - More than 424,000 (> 0.4 million) downloads from the OSU site directly
  - Empowering many TOP500 clusters (June ‘17 ranking)
    - 1st, 10,649,600-core (Sunway TaihuLight) at National Supercomputing Center in Wuxi, China
    - 15th, 241,108-core (Pleiades) at NASA
    - 20th, 462,462-core (Stampede) at TACC
    - 44th, 74,520-core (Tsubame 2.5) at Tokyo Institute of Technology
  - Available with software stacks of many vendors and Linux Distros (RedHat and SuSE)
  - http://mvapich.cse.ohio-state.edu

- Empowering Top500 systems for over a decade
  - System-X from Virginia Tech (3rd in Nov 2003, 2,200 processors, 12.25 TFlops) ->
  - Stampede at TACC (12th in Jun’16, 462,462 cores, 5.168 Plops)
MVAPICH Project Timeline

- MVAPICH2
- MVAPICH2-MIC
- MVAPICH2-GDR
- MVAPICH2-Virt
- MVAPICH2-EA
- OSU-INAM

Timeline:
- Oct-02
- Jan-04
- Nov-04
- Jan-10
- Nov-12
- Aug-14
- Apr-15
- Jul-15
- Aug-15
- Sep-15

EOL

OMB

Network Based Computing Laboratory

MVAPICH User Group Meeting (MUG) 2017
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 (MIC, KNL), NVIDIA GPGPU)

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

* Upcoming
Strong Procedure for Design, Development and Release

- Research is done for exploring new designs
- Designs are first presented to conference/journal publications
- Best performing designs are incorporated into the codebase
- Rigorous Q&A procedure before making a release
  - Exhaustive unit testing
  - Various test procedures on diverse range of platforms and interconnects
  - Performance tuning
  - Applications-based evaluation
  - Evaluation on large-scale systems
- Even alpha and beta versions go through the above testing
# MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI, OSU INAM, PGAS and MPI+PGAS with IB and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MPI with IB &amp; GPU</td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>MPI with IB &amp; MIC</td>
<td>MVAPICH2-MIC</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>
MVAPICH2 2.3b

- Released on 08/10/2017

- Major Features and Enhancements
  - Based on MPICH-3.2
  - Enhance performance of point-to-point operations for CH3-Gen2 (InfiniBand), CH3-PSM, and CH3-PSM2 (Omni-Path) channels
  - Improve performance for MPI-3 RMA operations
  - Introduce support for Cavium ARM (ThunderX) systems
  - Improve support for process to core mapping on many-core systems
    - New environment variable MV2_THREADS_BINDING_POLICY for multi-threaded MPI and MPI+OpenMP applications
    - Support 'linear' and 'compact' placement of threads
    - Warn user if over-subscription of core is detected
  - Improve launch time for large-scale jobs with mpirun_rsh
  - Add support for non-blocking Allreduce using Mellanox SHARP
  - Efficient support for different Intel Knight's Landing (KNL) models
  - Improve performance for Intra- and Inter-node communication for OpenPOWER architecture
  - Improve support for large processes per node and huge pages on SMP systems
  - Enhance collective tuning for many architectures/systems
  - Enhance support for MPI_T PVARs and CVARs
One-way Latency: MPI over IB with MVAPICH2

**Small Message Latency**

- 1.19
- 1.11
- 1.15
- 1.01
- 1.04

**Large 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
Bandwidth: MPI over IB with MVAPICH2

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 IB switch
Omni-Path - 3.1 GHz Deca-core (Haswell) Intel PCI Gen3 with Omni-Path switch
Towards High Performance and Scalable Startup at Exascale

- Near-constant MPI and OpenSHMEM initialization time at any process count
- 10x and 30x improvement in startup time of MPI and OpenSHMEM respectively at 16,384 processes
- Memory consumption reduced for remote endpoint information by $O(\text{processes per node})$
- 1GB Memory saved per node with 1M processes and 16 processes per node

On-demand Connection Management for OpenSHMEM and OpenSHMEM+MPI. S. Chakraborty, H. Subramoni, J. Perkins, A. A. Awan, and D K Panda, 20th International Workshop on High-level Parallel Programming Models and Supportive Environments (HIPS ’15)


**Startup Performance on KNL + Omni-Path**

- **MPI_Init** takes 22 seconds on 229,376 processes on 3,584 KNL nodes (Stampede2 – Full scale)
- 8.8 times faster than Intel MPI at 128K processes (Courtesy: TACC)
- At 64K processes, **MPI_Init** and **Hello World** takes 5.8s and 21s respectively (Oakforest-PACS)
- All numbers reported with 64 processes per node

New designs available since MVAPICH2-2.3a and as patch for SLURM-15.08.8 and SLURM-16.05.1
Advanced Allreduce Collective Designs Using SHArP

**Avg DDOT Allreduce time of HPCG**

<table>
<thead>
<tr>
<th>(Number of Nodes, PPN)</th>
<th>MVAPICH2</th>
<th>MVAPICH2-SHArP</th>
</tr>
</thead>
<tbody>
<tr>
<td>(4,28)</td>
<td>0.16</td>
<td>0.14</td>
</tr>
<tr>
<td>(8,28)</td>
<td>0.22</td>
<td>0.20</td>
</tr>
<tr>
<td>(16,28)</td>
<td>0.28</td>
<td>0.26</td>
</tr>
</tbody>
</table>

SHArP Support is available since MVAPICH2 2.3a
Non-blocking SHArP Support added in MVAPICH2 2.3b

**Mesh Refinement Time of MiniAMR**

<table>
<thead>
<tr>
<th>(Number of Nodes, PPN)</th>
<th>MVAPICH2</th>
<th>MVAPICH2-SHArP</th>
</tr>
</thead>
<tbody>
<tr>
<td>(4,28)</td>
<td>0.08</td>
<td>0.07</td>
</tr>
<tr>
<td>(8,28)</td>
<td>0.12</td>
<td>0.11</td>
</tr>
<tr>
<td>(16,28)</td>
<td>0.16</td>
<td>0.14</td>
</tr>
</tbody>
</table>

M. Bayatpour, S. Chakraborty, H. Subramoni, X. Lu, and D. K. Panda,
Scalable Reduction Collectives with Data Partitioning-based Multi-Leader Design,
Supercomputing '17.
Intra-node Point-to-Point Performance on OpenPower

**Small Message Latency**

**Large Message Latency**

**Bandwidth**

**Bi-directional Bandwidth**

Platform: OpenPOWER (Power8-ppc64le) processor with 160 cores dual-socket CPU. Each socket contains 80 cores.
Inter-node Point-to-Point Performance on OpenPower

Platform: Two nodes of OpenPOWER (Power8-ppc64le) CPU using Mellanox EDR (MT4115) HCA.
Intra-node Point-to-point Performance on ARMv8

**Small Message Latency**

- MVAPICH2: 0.74 us (MVAPICH2)
- OpenMPI-v2.1.1: 0.92 us (OpenMPI)

(19.5% improvement at 4-byte)

**Large Message Latency**

- MVAPICH2: 53% at 4K
- OpenMPI-v2.1.1: 28% at 2M

**Bandwidth**

- MVAPICH2: 38% at 2M
- OpenMPI-v2.1.1

**Bi-directional Bandwidth**

- MVAPICH2: 1.85 X at 2M
- OpenMPI-v2.1.1

---

**Platform:** ARMv8 (aarch64) MIPS processor with 96 cores dual-socket CPU. Each socket contains 48 cores.

Available in MVAPICH2 2.3b

---

**Notes:**

- OpenMPI version 2.2.1 used with "—mca bta self,sm"

---
Performance Engineering Applications using MVAPICH2 and TAU

- Enhance existing support for MPI_T in MVAPICH2 to expose a richer set of performance and control variables
- Get and display MPI Performance Variables (PVARs) made available by the runtime in TAU
- Control the runtime’s behavior via MPI Control Variables (CVARs)
- Introduced support for new MPI_T based CVARs to MVAPICH2
  - MPIR_CVAR_MAX_INLINE_MSG_SZ, MPIR_CVAR_VBUF_POOL_SIZE, MPIR_CVAR_VBUF_SECONDARY_POOL_SIZE
- TAU enhanced with support for setting MPI_T CVARs in a non-interactive mode for uninstrumented applications
- S. Ramesh, A. Maheo, S. Shende, A. Malony, H. Subramoni, and D. K. Panda, *MPI Performance Engineering with the MPI Tool Interface: the Integration of MVAPICH and TAU*, EuroMPI/USA ’17, Best Paper Finalist
- More details in Prof. Malony’s talk today and poster presentations

VBUF usage without CVAR based tuning as displayed by ParaProf

VBUF usage with CVAR based tuning as displayed by ParaProf
MVAPICH2 Upcoming Features

- Dynamic and Adaptive Tag Matching
- Dynamic and Adaptive Communication Protocols
- Optimized Kernel-Assisted collectives
- Enhanced Collectives to exploit MCDRAM
Dynamic and Adaptive Tag Matching

**Challenge**
- Tag matching is a significant overhead for receivers
- Existing solutions are static and do not adapt dynamically to communication patterns
- Do not consider memory overhead

**Solution**
- A new tag matching design
  - Dynamically adapt to communication patterns
  - Use different strategies for different ranks
  - Decisions are based on the number of request object that must be traversed before hitting on the required one

**Results**
- Better performance than other state-of-the-art tag-matching schemes
- Minimum memory consumption
- Will be available in future MVAPICH2 releases

**Graphs**
- **Normalized Total Tag Matching Time at 512 Processes**
  - Normalized to Default (Lower is Better)
- **Normalized Memory Overhead per Process at 512 Processes**
  - Compared to Default (Lower is Better)

Dynamic and Adaptive MPI Point-to-point Communication Protocols

Desired Eager Threshold

<table>
<thead>
<tr>
<th>Process Pair</th>
<th>Eager Threshold (KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 – 4</td>
<td>32</td>
</tr>
<tr>
<td>1 – 5</td>
<td>64</td>
</tr>
<tr>
<td>2 – 6</td>
<td>128</td>
</tr>
<tr>
<td>3 – 7</td>
<td>32</td>
</tr>
</tbody>
</table>

Eager Threshold for Example Communication Pattern with Different Designs

<table>
<thead>
<tr>
<th>Process Pair</th>
<th>Eager Threshold (KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>32</td>
</tr>
<tr>
<td>Manually Tuned</td>
<td>64</td>
</tr>
<tr>
<td>Dynamic + Adaptive</td>
<td>128</td>
</tr>
</tbody>
</table>

Default

- Poor overlap; Low memory requirement
- Low Performance; High Productivity

Manually Tuned

- Good overlap; High memory requirement
- High Performance; Low Productivity

Dynamic + Adaptive

- Good overlap; Optimal memory requirement
- High Performance; High Productivity

Execution Time of Amber

Relative Memory Consumption of Amber

Optimized Kernel-Assisted Collectives for Multi-/Many-Core

- Kernel-Assisted transfers (CMA, LiMIC, KNEM) offers single-copy transfers for large messages
- Scatter/Gather/Bcast etc. can be optimized with direct Kernel-Assisted transfers
- Applicable to Broadwell and OpenPOWER architectures as well


Enhanced Designs will be available in upcoming MVAPICH2 releases
Enhanced Collectives to Exploit MCDRAM

- New designs to exploit high concurrency and MCDRAM of KNL
- Significant improvements for large message sizes
- Benefits seen in varying message size as well as varying MPI processes

Intra-node Broadcast with 64MB Message

16-process Intra-node All-to-All

Very Large Message Bi-directional Bandwidth
Performance Benefits of Enhanced Designs

- Benefits observed on training time of Multi-level Perceptron (MLP) model on MNIST dataset using CNTK Deep Learning Framework

Enhanced Designs will be available in upcoming MVAPICH2 releases
## MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI, OSU INAM, PGAS and MPI+PGAS with IB and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MPI with IB &amp; GPU</td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>MPI with IB &amp; MIC</td>
<td>MVAPICH2-MIC</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>
## MVAPICH2-X for Hybrid MPI + PGAS Applications

<table>
<thead>
<tr>
<th>High Performance Parallel Programming Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MPI</strong></td>
</tr>
<tr>
<td>Message Passing Interface</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>High Performance and Scalable Unified Communication Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diverse APIs and Mechanisms</strong></td>
</tr>
<tr>
<td>Optimized Point-to-point Primitives</td>
</tr>
<tr>
<td>Support for Modern Networking Technologies</td>
</tr>
<tr>
<td>(InfiniBand, iWARP, RoCE, Omni-Path...)</td>
</tr>
</tbody>
</table>

- **Current Model** – Separate Runtimes for OpenSHMEM/UPC/UPC++/CAF and MPI
  - Possible deadlock if both runtimes are not progressed
  - Consumes more network resource
- **Unified communication runtime for MPI, UPC, UPC++, OpenSHMEM, CAF**
  - Available with since 2012 (starting with MVAPICH2-X 1.9)
  - [http://mvapich.cse.ohio-state.edu](http://mvapich.cse.ohio-state.edu)
UPC++ Support in MVAPICH2-X

- Full and native support for hybrid MPI + UPC++ applications
- Better performance compared to IBV and MPI conduits
- OSU Micro-benchmarks (OMB) support for UPC++
- Available since MVAPICH2-X (2.2rc1)

More Details in Student Poster Presentation
Application Level Performance with Graph500 and Sort

Graph500 Execution Time

- Performance of Hybrid (MPI+OpenSHMEM) Graph500 Design
  - 8,192 processes
    - 2.4X improvement over MPI-CSR
    - 7.6X improvement over MPI-Simple
  - 16,384 processes
    - 1.5X improvement over MPI-CSR
    - 13X improvement over MPI-Simple

Sort Execution Time

- Performance of Hybrid (MPI+OpenSHMEM) Sort Application
  - 4,096 processes, 4 TB Input Size
    - MPI – 2408 sec; 0.16 TB/min
    - Hybrid – 1172 sec; 0.36 TB/min
    - 51% improvement over MPI-design


J. Jose, S. Potluri, K. Tomko and D. K. Panda, Designing Scalable Graph500 Benchmark with Hybrid MPI+OpenSHMEM Programming Models, International Supercomputing Conference (ISC’13), June 2013

J. Jose, K. Kandalla, M. Luo and D. K. Panda, Supporting Hybrid MPI and OpenSHMEM over InfiniBand: Design and Performance Evaluation, Int'l Conference on Parallel Processing (ICPP ’12), September 2012
MVAPICH2-X Upcoming Features

- Updating to OpenSHMEM 1.3
- UPC to be updated to 2.42.2
- Optimized OpenSHMEM with AVX and MCDRAM Support
- Implicit On-Demand Paging (ODP) Support
OpenSHMEM Application kernels

- AVX-512 vectorization and MCDRAM based optimizations for MVAPICH2-X
  - Will be available in future MVAPICH2-X release

**Scalable Integer Sort Kernel (ISx)**

- AVX-512 vectorization showed up to 3X improvement over default
- KNL showed on-par performance as Broadwell on Node-by-node comparison

**Single KNL and Single Broadwell node**

+30%

-5%
**Implicit On-Demand Paging (ODP)**

- Introduced by Mellanox to avoid pinning the pages of registered memory regions
- ODP-aware runtime could reduce the size of pin-down buffers while maintaining performance

---

**Applications (256 Processes)**

- **Pin-down**
- **Explicit-ODP**
- **Implicit-ODP**

---

*M. Li, X. Lu, H. Subramoni, and D. K. Panda,* “Designing Registration Caching Free High-Performance MPI Library with Implicit On-Demand Paging (ODP) of InfiniBand”, Under Review
## MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI, OSU INAM, PGAS and MPI+PGAS with IB and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td><strong>MPI with IB &amp; GPU</strong></td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>MPI with IB &amp; MIC</td>
<td>MVAPICH2-MIC</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><strong>MPI Energy Monitoring Tool</strong></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>
CUDA-Aware MPI: MVAPICH2-GDR 1.8-2.2 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
Performance of MVAPICH2-GPU with GPU-Direct RDMA (GDR)

**GPU-GPU internode latency**

- MV2-GDR2.2
- MV2-GDR2.0b
- MV2 w/o GDR

**GPU-GPU Internode Bandwidth**

- MV2-GDR2.2
- MV2-GDR2.0b
- MV2 w/o GDR

**MVAPICH2-GDR-2.2**
- Intel Ivy Bridge (E5-2680 v2) node - 20 cores
- NVIDIA Tesla K40c GPU
- Mellanox Connect-X4 EDR HCA
- CUDA 8.0
- Mellanox OFED 3.0 with GPU-Direct-RDMA

**Results**

- 2.18us latency
- 10x bandwidth increase
- 11x bi-bandwidth increase
Application-Level Evaluation (HOOMD-blue)

**64K Particles**

- Platform: Wilkes (Intel Ivy Bridge + NVIDIA Tesla K20c + Mellanox Connect-IB)
- HoomdBlue Version 1.0.5
  - GDRCOPY enabled: MV2_USE_CUDA=1 MV2_IBA_HCA=mlx5_0 MV2_IBA_EAGER_THRESHOLD=32768 MV2_VBUF_TOTAL_SIZE=32768 MV2_USE_GPUDIRECT_LOOPBACK_LIMIT=32768 MV2_USE_GPUDIRECT_GDRCOPY=1 MV2_USE_GPUDIRECT_GDRCOPY_LIMIT=16384

**256K Particles**

- MV2
- MV2+GDR

- Average Time Steps per second (TPS)
- Number of Processes

- 4 8 16 32
- 0 500 1000 1500 2000 2500 3000 3500
- 4 8 16 32
- 0 500 1000 1500 2000 2500 3000 3500

- 4X 2X
Application-Level Evaluation (Cosmo) and Weather Forecasting in Switzerland

- 2X improvement on 32 GPUs nodes
- 30% improvement on 96 GPU nodes (8 GPUs/node)

On-going collaboration with CSCS and MeteoSwiss (Switzerland) in co-designing MV2-GDR and Cosmo Application


Cosmo model: http://www2.cosmo-model.org/content/tasks/operational/meteoSwiss/
Enhanced Support for GPU Managed Memory

- CUDA Managed => no memory pin down
  - No IPC support for intranode communication
  - No GDR support for Internode communication
- Significant productivity benefits due to abstraction of explicit allocation and `cudaMemcpy()`
- Initial and basic support in MVAPICH2-GDR
  - For both intra- and inter-nodes use “pipeline through” host memory
- Enhance intranode managed memory to use IPC
  - Double buffering pair-wise IPC-based scheme
  - Brings IPC performance to Managed memory
  - High performance and high productivity
  - 2.5 X improvement in bandwidth
- OMB extended to evaluate the performance of point-to-point and collective communications using managed buffers

D. S. Banerjee, K Hamidouche, and D. K Panda, Designing High Performance Communication Runtime for GPUManaged Memory: Early Experiences, GPGPU-9 Workshop, held in conjunction with PPoPP '16
MVAPICH2-GDR Upcoming Features

- Optimized Designs
- Support for OpenPower+NVLINK Platforms
- Optimized Support for Deep Learning (Caffe and CNTK)
- Support for Streaming Applications
- GPU Direct Async (GDS) Support
- CUDA-aware OpenSHMEM
Performance of MVAPICH2-GPU with GPU-Direct RDMA (GDR)

GPU-GPU Inter-node Latency

- MV2-(NO-GDR)
- MV2-GDR-2.3a

1.91us 10x

Message Size (Bytes)

GPU-GPU Inter-node Bandwidth

- MV2-(NO-GDR)
- MV2-GDR-2.3a

9x

Message Size (Bytes)

GPU-GPU Inter-node Bi-Bandwidth

- MV2-(NO-GDR)
- MV2-GDR-2.3a

11X

Message Size (Bytes)

MVAPICH2-GDR-2.3a

Intel Haswell (E5-2687W) node - 20 cores
NVIDIA Pascal P100 GPU
Mellanox Connect-X5 EDR HCA
CUDA 8.0
Mellanox OFED 4.0 with GPU-Direct-RDMA
MVAPICH2-GDR: Performance on OpenPOWER (NVLink + Pascal)

Intra-node Bandwidth: 32 GB/sec (NVLINK)

Intra-node Bandwidth: 6 GB/sec (FDR)

Inter-node Bandwidth: 6 GB/sec (FDR)

Intra-node Latency: 16.8 us (without GPUDirectRDMA)

Inter-node Latency: 22 us (without GPUDirectRDMA)

Platform: OpenPOWER (ppc64le) nodes equipped with a dual-socket CPU, 4 Pascal P100-SXM GPUs, and 4X-FDR InfiniBand Inter-connect

Network Based Computing Laboratory

MVAPICH User Group Meeting (MUG) 2017
Efficient Broadcast for Deep Learning: MVAPICH2-GDR and NCCL

- NCCL has some limitations
  - Only works for a single node, thus, no scale-out on multiple nodes
  - Degradation across IOH (socket) for scale-up (within a node)

- We propose optimized MPI_Bcast
  - Communication of very large GPU buffers (order of megabytes)
  - Scale-out on large number of dense multi-GPU nodes

- Hierarchical Communication that efficiently exploits:
  - CUDA-Aware MPI_Bcast in MV2-GDR
  - NCCL Broadcast primitive


Performance Benefits: Microsoft CNTK DL framework (25% avg. improvement)
Large Message Optimized Collectives for Deep Learning

- MV2-GDR provides optimized collectives for large message sizes
- Optimized Reduce, Allreduce, and Bcast
- Good scaling with large number of GPUs
- Available in MVAPICH2-GDR 2.2GA
OSU-Caffe: Scalable Deep Learning

- 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/

More Details in Poster
Streaming Support (Combining GDR and IB-Mcast)

• Combining MCAST+GDR hardware features for heterogeneous configurations:
  – Source on the Host and destination on Device
  – SL design: Scatter at destination
    • Source: Data and Control on Host
    • Destinations: Data on Device and Control on Host
  – Combines IB MCAST and GDR features at receivers
  – CUDA IPC-based topology-aware intra-node broadcast
  – Minimize use of PCIe resources (Maximizing availability of PCIe Host-Device Resources)

Exploiting GDR+IB-Mcast Design for Deep Learning Applications

- Optimizing MCAST+GDR Broadcast for deep learning:
  - Source and destination buffers are on GPU Device
    - Typically very large messages (>1MB)
  - Pipelining data from Device to Host
    - Avoid GDR read limit
    - Leverage high-performance SL design
  - Combines IB MCAST and GDR features
  - Minimize use of PCIe resources on the receiver side
    - Maximizing availability of PCIe Host-Device Resources

Application Evaluation: Deep Learning Frameworks

- @ RI2 cluster, 16 GPUs, 1 GPU/node
  - Microsoft Cognitive Toolkit (CNTK) [https://github.com/Microsoft/CNTK]

More Details in Poster

<table>
<thead>
<tr>
<th>Model</th>
<th>MV2-GDR-Knomial</th>
<th>MV2-GDR-Ring</th>
<th>MCAST-GDR-Opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td></td>
<td></td>
<td>15%</td>
</tr>
<tr>
<td>VGG</td>
<td></td>
<td></td>
<td>24%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>MV2-GDR-Knomial</th>
<th>MV2-GDR-Ring</th>
<th>MCAST-GDR-Opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG</td>
<td></td>
<td></td>
<td>6%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>15%</td>
</tr>
</tbody>
</table>

Lower is better

- Reduces up to 24% and 15% of latency for AlexNet and VGG models
- Higher improvement can be observed for larger system sizes

Latency Oriented: Able to hide the kernel launch overhead
- 25% improvement at 256 Bytes compared to default behavior

Throughput Oriented: Asynchronously to offload queue the Communication and computation tasks
- 14% improvement at 1KB message size

Intel Sandy Bridge, NVIDIA K20 and Mellanox FDR HCA

Will be available in a public release soon
CUDA-Aware OpenSHMEM: Evaluation with GPULBM and 2DStencil

GPULBM: 64x64x64

- Platform: Wilkes (Intel Ivy Bridge + NVIDIA Tesla K20c + Mellanox Connect-IB)

- New designs achieve 20% and 19% improvements on 32 and 64 GPU nodes

2DStencil 2Kx2K

- Redesign the application
  - CUDA-Aware MPI: Send/Recv => hybrid CUDA-Aware MPI+OpenSHMEM
  - cudaMalloc => shmalloc(size,1);
  - MPI_Send/recv => shmem_put + fence
  - 53% and 45%
  - Degradation is due to small input size
  - Will be available in future MVAPICH2-GDR releases

MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI, OSU INAM, PGAS and MPI+PGAS with IB and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MPI with IB &amp; GPU</td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>MPI with IB &amp; MIC</td>
<td>MVAPICH2-MIC</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>
Can HPC and Virtualization be Combined?

- Virtualization has many benefits
  - Fault-tolerance
  - Job migration
  - Compaction

- Have not been very popular in HPC due to overhead associated with Virtualization

- New SR-IOV (Single Root – IO Virtualization) support available with Mellanox InfiniBand adapters changes the field

- Enhanced MVAPICH2 support for SR-IOV

- MVAPICH2-Virt 2.2 supports:
  - Efficient MPI communication over SR-IOV enabled InfiniBand networks
  - High-performance and locality-aware MPI communication for VMs and containers
  - Automatic communication channel selection for VMs (SR-IOV, IVSHMEM, and CMA/LiMIC2) and containers (IPC-SHM, CMA, and HCA)
  - OpenStack, Docker, and Singularity

J. Zhang, X. Lu, J. Jose, R. Shi and D. K. Panda, Can Inter-VM Shmem Benefit MPI Applications on SR-IOV based Virtualized InfiniBand Clusters? EuroPar’14

J. Zhang, X. Lu, J. Jose, M. Li, R. Shi and D.K. Panda, High Performance MPI Libray over SR-IOV enabled InfiniBand Clusters, HiPC’14

Application-Level Performance on Chameleon (SR-IOV Support)

- 32 VMs, 6 Core/VM
- Compared to Native, 2-5% overhead for Graph500 with 128 Procs
- Compared to Native, 1-9.5% overhead for SPEC MPI2007 with 128 Procs
• 64 Containers across 16 nodes, pining 4 Cores per Container

• Compared to Container-Def, up to 11% and 16% of execution time reduction for NAS and Graph 500

• Compared to Native, less than 9% and 4% overhead for NAS and Graph 500
Application-Level Performance on Singularity with MVAPICH2

- 512 Processes across 32 nodes
- Less than 7% and 6% overhead for NPB and Graph500, respectively

More Details in Tomorrow’s Tutorial/Demo Session
MVAPICH2-Virt Upcoming Features

- Support for SR-IOV Enabled VM Migration
- SR-IOV and IVSHMEM Support in SLURM
- Support for Microsoft Azure Platform
High Performance SR-IOV enabled VM Migration Support in MVAPICH2

- Migration with SR-IOV device has to handle the challenges of detachment/re-attachment of virtualized IB device and IB connection
- Consist of SR-IOV enabled IB Cluster and External Migration Controller
- Multiple parallel libraries to notify MPI applications during migration (detach/reattach SR-IOV/IVShmem, migrate VMs, migration status)
- Handle the IB connection suspending and reactivating
- Propose Progress engine (PE) and migration thread based (MT) design to optimize VM migration and MPI application performance

Performance Evaluation of VM Migration Framework

- 8 VMs in total and 1 VM carries out migration during application running
- Compared with NM, MT-worst and PE incur some overhead compared with NM
- MT-typical allows migration to be completely overlapped with computation
SR-IOV and IVSHMEM in SLURM

• Requirement of managing and isolating virtualized resources of SR-IOV and IVSHMEM

• Such kind of management and isolation is hard to be achieved by MPI library alone, but much easier with SLURM

• Efficient running MPI applications on HPC Clouds needs SLURM to support managing SR-IOV and IVSHMEM
  – Can critical HPC resources be efficiently shared among users by extending SLURM with support for SR-IOV and IVSHMEM based virtualization?
  – Can SR-IOV and IVSHMEM enabled SLURM and MPI library provide bare-metal performance for end applications on HPC Clouds?

## MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI, OSU INAM, PGAS and MPI+PGAS with IB and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MPI with IB &amp; GPU</td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>MPI with IB &amp; MIC</td>
<td>MVAPICH2-MIC</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>
Energy-Aware MVAPICH2 & OSU Energy Management Tool (OEMT)

• MVAPICH2-EA 2.1 (Energy-Aware)
  • A white-box approach
  • New Energy-Efficient communication protocols for pt-pt and collective operations
  • Intelligently apply the appropriate Energy saving techniques
  • Application oblivious energy saving

• OEMT
  • A library utility to measure energy consumption for MPI applications
  • Works with all MPI runtimes
  • PRELOAD option for precompiled applications
  • Does not require ROOT permission:
    • A safe kernel module to read only a subset of MSRs
**MVAPICH2-EA: Application Oblivious Energy-Aware-MPI (EAM)**

- An energy efficient runtime that provides energy savings without application knowledge
- Uses automatically and transparently the best energy lever
- Provides guarantees on maximum degradation with 5-41% savings at <= 5% degradation
- Pessimistic MPI applies energy reduction lever to each MPI call

**A Case for Application-Oblivious Energy-Efficient MPI Runtime**

• MPI_Win_fence dominates application execution time in graph500
• Between 128 and 512 processes, EAM-RMA yields between 31% and 46% savings with no degradation in execution time in comparison with the default optimistic MPI runtime
• MPI-3 RMA Energy-efficient support will be available in upcoming MVAPICH2-EA release
## MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI, OSU INAM, PGAS and MPI+PGAS with IB and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MPI with IB &amp; GPU</td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>MPI with IB &amp; MIC</td>
<td>MVAPICH2-MIC</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>
Overview of OSU INAM

- A network monitoring and analysis tool that is capable of analyzing traffic on the InfiniBand network with inputs from the MPI runtime
  - [http://mvapich.cse.ohio-state.edu/tools/osu-inam/](http://mvapich.cse.ohio-state.edu/tools/osu-inam/)
- Monitors IB clusters in real time by querying various subnet management entities and gathering input from the MPI runtimes
- OSU INAM v0.9.1 released on 05/13/16
- Significant enhancements to user interface to enable scaling to clusters with thousands of nodes
- Improve database insert times by using 'bulk inserts'
- Capability to look up list of nodes communicating through a network link
- Capability to classify data flowing over a network link at job level and process level granularity in conjunction with MVAPICH2-X 2.2rc1
- "Best practices “ guidelines for deploying OSU INAM on different clusters
- Capability to analyze and profile node-level, job-level and process-level activities for MPI communication
  - Point-to-Point, Collectives and RMA
- Ability to filter data based on type of counters using “drop down” list
- Remotely monitor various metrics of MPI processes at user specified granularity
- "Job Page" to display jobs in ascending/descending order of various performance metrics in conjunction with MVAPICH2-X
- Visualize the data transfer happening in a “live” or “historical” fashion for entire network, job or set of nodes
OSU INAM Features

- Show network topology of large clusters
- Visualize traffic pattern on different links
- Quickly identify congested links/links in error state
- See the history unfold – play back historical state of the network

Comet@SDSC --- Clustered View
(1,879 nodes, 212 switches, 4,377 network links)

Finding Routes Between Nodes
OSU INAM Features (Cont.)

- **Job level view**
  - Show different network metrics (load, error, etc.) for any live job
  - Play back historical data for completed jobs to identify bottlenecks

- **Node level view - details per process or per node**
  - CPU utilization for each rank/node
  - Bytes sent/received for MPI operations (pt-to-pt, collective, RMA)
  - Network metrics (e.g. XmitDiscard, RcvError) per rank/node

- **Estimated Link Utilization view**
  - Classify data flowing over a network link at different granularity in conjunction with MVAPICH2-X 2.2rc1
    - Job level and
    - Process level

*More Details in Tutorial/Demo Session Tomorrow*
OSU Microbenchmarks

• Available since 2004
• 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
• Please visit the following link for more information
  – http://mvapich.cse.ohio-state.edu/benchmarks/
Applications-Level Tuning: Compilation of Best Practices

- MPI runtime has many parameters
- Tuning a set of parameters can help you to extract higher performance
- Compiled a list of such contributions through the MVAPICH Website
  - [http://mvapich.cse.ohio-state.edu/best_practices/](http://mvapich.cse.ohio-state.edu/best_practices/)
- Initial list of applications
  - Amber
  - HoomDBlue
  - HPCG
  - Lulesh
  - MILC
  - Neuron
  - SMG2000
- Soliciting additional contributions, send your results to mvapich-help at cse.ohio-state.edu.
- We will link these results with credits to you.
MVAPICH2 – Plans for Exascale

- Performance and Memory scalability toward 1-10M cores
- Hybrid programming (MPI + OpenSHMEM, MPI + UPC, MPI + CAF …)
  - MPI + Task*
- Enhanced Optimization for GPU Support and Accelerators
- Taking advantage of advanced features of Mellanox InfiniBand
  - Tag Matching*
  - Adapter Memory*
- Enhanced communication schemes for upcoming architectures
  - Knights Landing with MCDRAM*
  - NVLINK*
  - CAPI*
- Extended topology-aware collectives
- Extended Energy-aware designs and Virtualization Support
- Extended Support for MPI Tools Interface (as in MPI 3.0)
- Extended FT support
- Support for * features will be available in future MVAPICH2 Releases
Thank You!

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

The MVAPICH2 Project
http://mvapich.cse.ohio-state.edu/