MVAPICH

MPI, PGAS and Hybrid MPI+PGAS Library

Extracting Peak Performance for your Applications on Frontera with MVAPICH2 Libraries

A Talk at Frontera User Meeting (Jan'21)

by

Hari Subramoni

The Ohio State University

E-mail: subramon@cse.ohio-state.edu

http://www.cse.ohio-state.edu/~subramon

Follow us on

https://twitter.com/mvapich

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 3,125 organizations in 89 countries**
- **More than 1.2 Million downloads from the OSU site directly**
- Empowering many TOP500 clusters (Nov '20 ranking)
  - 4^{th} , 10,649,600-core (Sunway TaihuLight) at NSC, Wuxi, China
  - 9^{th}, 448, 448 cores (Frontera) at TACC
  - 14^{th}, 391,680 cores (ABCI) in Japan
  - 21^{th}, 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 9^{th} ranked TACC Frontera system
- **Empowering Top500 systems for more than 16 years**

Architecture of MVAPICH2 Software Family (for HPC and DL)

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: SHARP2*, ODP, SR-IOV, Multi Rail

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

Transport Mechanisms: Shared Memory, CMA, IVSHMEM, XPMEM

Modern Features: MCDRAM*, NVLink, CAPI*

* Upcoming

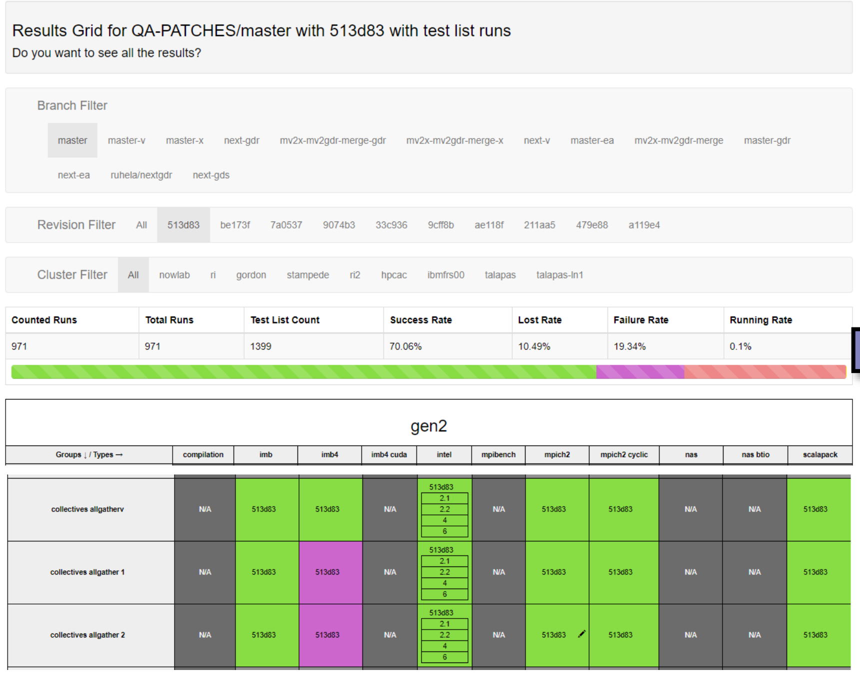
Production Quality Software Design, Development and Release

- Rigorous Q&A procedure before making a release
  - Exhaustive unit testing
  - Various test procedures on diverse range of platforms and interconnects
  - Test 19 different benchmarks and applications including, but not limited to
    - OMB, IMB, MPICH Test Suite, Intel Test Suite, NAS, Scalapak, and SPEC
  - Spend about 18,000 core hours per commit
  - Performance regression and tuning
  - Applications-based evaluation
  - Evaluation on large-scale systems
- All versions (alpha, beta, RC1 and RC2) go through the above testing

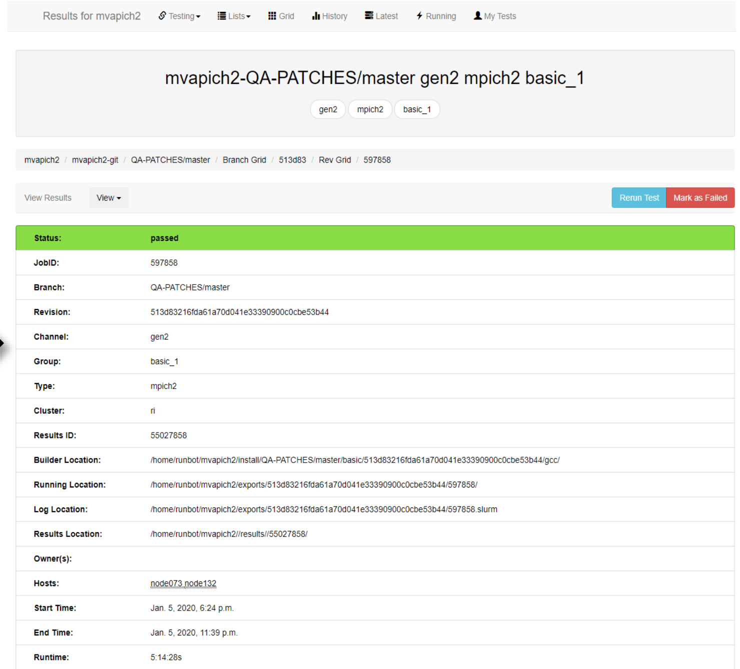
Automated Procedure for Testing Functionality

- Test OMB, IMB, MPICH Test Suite, Intel Test Suite, NAS, Scalapak, and SPEC
- Tests done for each build done build “buildbot”
- Test done for various different combinations of *environment variables* meant to trigger different communication paths in MVAPICH2

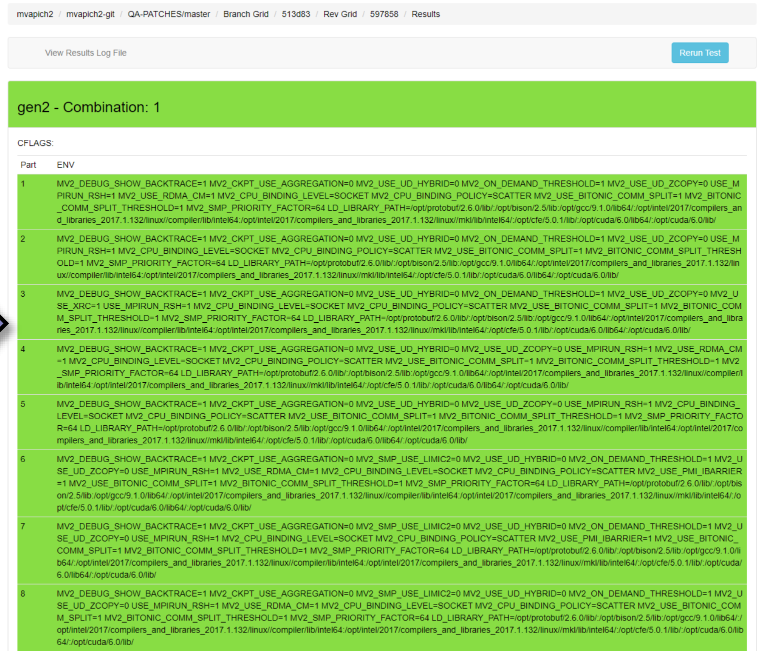
Summary of all tests for one commit



Summary of an individual test

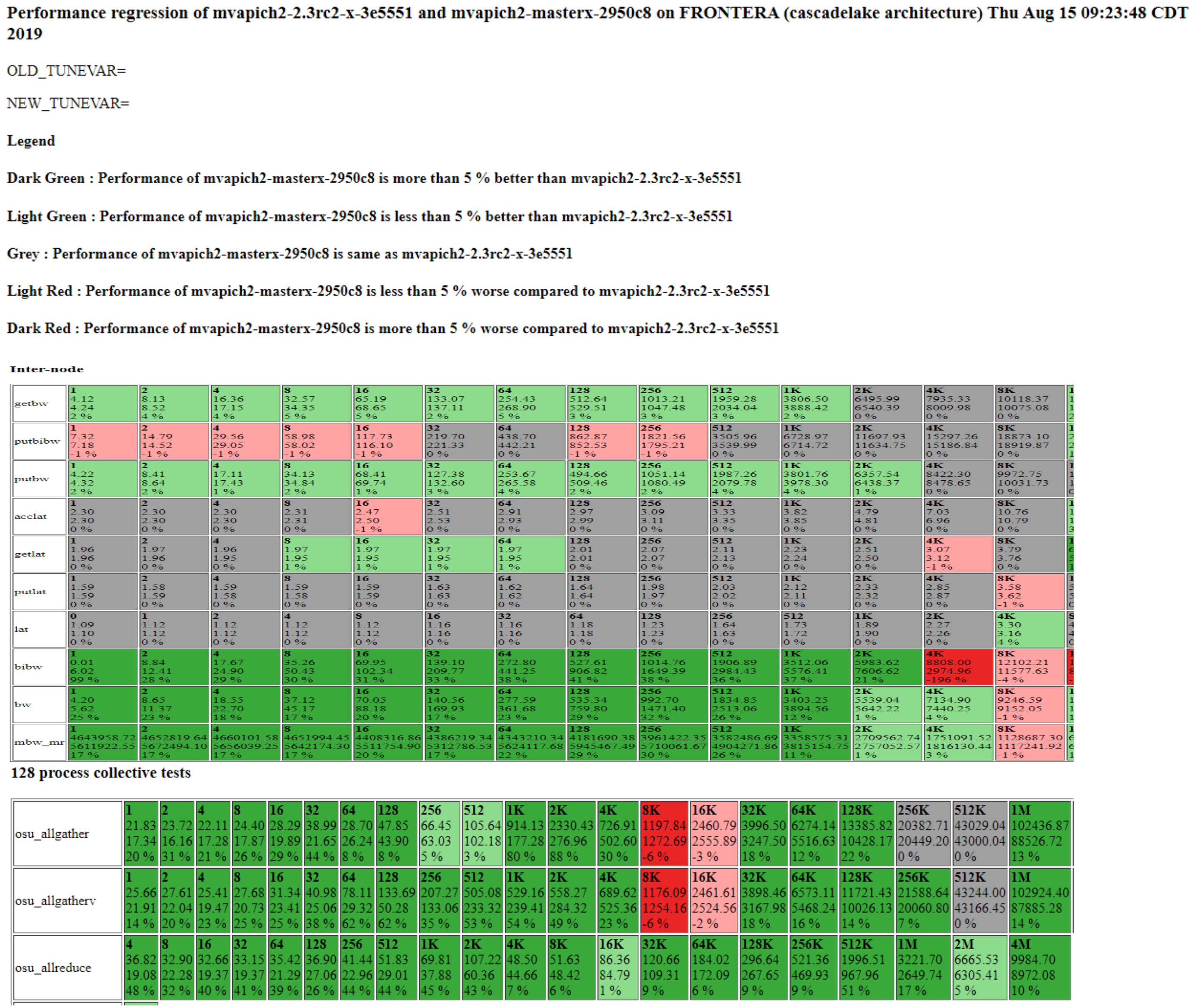


Details of individual combinations in one test



Scripts to Determine Performance Regression

- Automated method to identify performance regression between different commits
- Tests different MPI primitives
  - Point-to-point; Collectives; RMA
- Works with different
  - Job Launchers/Schedulers
    - SLURM, PBS/Torque, JSM
  - Works with different interconnects
- Works on multiple HPC systems
- Works on CPU-based and GPU-based systems



Performance regression of mvapich2-2.3rc2-x-3e5551 and mvapich2-masterx-2950c8 on FRONTERA (cascadelake architecture) Thu Aug 15 09:23:48 CDT 2019

OLD_TUNEVAR=

NEW_TUNEVAR=

Legend

Dark Green : Performance of mvapich2-masterx-2950c8 is more than 5 % better than mvapich2-2.3rc2-x-3e5551

Light Green : Performance of mvapich2-masterx-2950c8 is less than 5 % better than mvapich2-2.3rc2-x-3e5551

Grey : Performance of mvapich2-masterx-2950c8 is same as mvapich2-2.3rc2-x-3e5551

Light Red : Performance of mvapich2-masterx-2950c8 is less than 5 % worse compared to mvapich2-2.3rc2-x-3e5551

Dark Red : Performance of mvapich2-masterx-2950c8 is more than 5 % worse compared to mvapich2-2.3rc2-x-3e5551

Inter-node

getbw	1 4.12 4.24 2 %	2 8.13 8.52 4 %	4 16.36 17.15 4 %	8 32.57 34.35 5 %	16 65.19 68.65 5 %	32 133.07 137.11 2 %	64 254.43 268.90 5 %	128 512.64 529.51 3 %	256 1013.21 1047.48 3 %	512 1959.28 2034.04 3 %	1K 3806.50 3888.42 2 %	2K 6495.99 6540.39 0 %	4K 7935.33 8009.98 0 %	8K 10118.37 10075.08 0 %
putbibw	1 7.32 7.18 -1 %	2 14.79 14.52 -1 %	4 29.56 29.05 -1 %	8 58.98 58.02 -1 %	16 117.73 116.10 -1 %	32 219.70 221.33 0 %	64 438.70 442.21 0 %	128 862.87 852.53 -1 %	256 1821.56 1795.21 -1 %	512 3505.96 3539.99 0 %	1K 6728.97 6714.72 0 %	2K 11697.93 11634.75 0 %	4K 15297.26 15186.84 0 %	8K 18873.10 18919.87 0 %
putbw	1 4.22 4.32 2 %	2 8.41 8.64 2 %	4 17.11 17.43 1 %	8 34.13 34.84 2 %	16 68.41 69.74 1 %	32 127.38 132.60 3 %	64 253.67 265.58 4 %	128 494.66 509.46 2 %	256 1051.14 1080.49 2 %	512 1987.26 2079.78 4 %	1K 3801.76 3978.30 4 %	2K 6357.54 6438.37 1 %	4K 8422.30 8478.65 0 %	8K 9972.75 10031.73 0 %
acclat	1 2.30 2.30 0 %	2 2.30 2.30 0 %	4 2.30 2.30 0 %	8 2.31 2.31 0 %	16 2.47 2.50 -1 %	32 2.51 2.53 0 %	64 2.91 2.93 0 %	128 2.97 2.99 0 %	256 3.09 3.11 0 %	512 3.33 3.35 0 %	1K 3.82 3.85 0 %	2K 4.79 4.81 0 %	4K 7.03 6.96 0 %	8K 10.76 10.79 0 %
getlat	1 1.96 1.96 0 %	2 1.97 1.96 0 %	4 1.96 1.95 0 %	8 1.97 1.95 1 %	16 1.97 1.95 1 %	32 1.97 1.95 1 %	64 1.97 1.95 1 %	128 2.01 2.01 0 %	256 2.07 2.07 0 %	512 2.11 2.13 0 %	1K 2.23 2.24 0 %	2K 2.51 2.50 0 %	4K 3.07 3.12 -1 %	8K 3.79 3.76 0 %
putlat	1 1.59 1.59 0 %	2 1.58 1.59 0 %	4 1.59 1.58 0 %	8 1.59 1.58 0 %	16 1.59 1.59 0 %	32 1.63 1.63 0 %	64 1.62 1.62 0 %	128 1.64 1.64 0 %	256 1.98 1.97 0 %	512 2.03 2.02 0 %	1K 2.12 2.11 0 %	2K 2.33 2.32 0 %	4K 2.85 2.87 0 %	8K 3.58 3.62 -1 %
lat	0 1.09 1.10 0 %	1 1.12 1.12 0 %	2 1.12 1.12 0 %	4 1.12 1.12 0 %	8 1.12 1.12 0 %	16 1.16 1.16 0 %	32 1.16 1.16 0 %	64 1.18 1.18 0 %	128 1.23 1.23 0 %	256 1.64 1.63 0 %	512 1.73 1.72 0 %	1K 1.89 1.90 0 %	2K 2.27 2.26 0 %	4K 3.30 3.16 4 %
bibw	1 0.01 6.02 99 %	2 8.84 12.41 28 %	4 17.67 24.90 29 %	8 35.26 50.43 30 %	16 69.95 102.34 31 %	32 139.10 209.77 33 %	64 272.80 441.25 38 %	128 527.61 906.82 41 %	256 1014.76 1649.39 38 %	512 1906.89 2984.43 36 %	1K 3512.06 5576.41 37 %	2K 5983.62 7606.62 21 %	4K 8808.00 2974.96 -196 %	8K 12102.21 11577.63 -4 %
bw	1 4.20 5.62 25 %	2 8.65 11.37 23 %	4 18.55 22.70 18 %	8 37.12 45.17 17 %	16 70.05 88.18 20 %	32 140.56 169.93 17 %	64 277.59 361.68 23 %	128 535.34 759.80 29 %	256 992.70 1471.40 32 %	512 1834.85 2513.06 26 %	1K 3403.25 3894.56 12 %	2K 5539.04 5642.22 1 %	4K 7134.90 7440.25 4 %	8K 9246.59 9152.05 -1 %
mbw_mr	1 4643958.72 5611922.55 17 %	2 4652819.64 5672494.10 17 %	4 4660101.58 5656039.25 17 %	8 4651994.45 5642174.30 17 %	16 4408316.86 5511754.90 20 %	32 4386219.34 5312786.53 17 %	64 4343210.34 5624117.68 22 %	128 4181690.38 5945467.49 29 %	256 3961422.35 5710061.67 30 %	512 3582486.69 4904271.86 26 %	1K 3358575.31 3815154.75 11 %	2K 2709562.74 2757052.57 1 %	4K 1751091.52 1816130.44 3 %	8K 1128687.30 1117241.92 -1 %

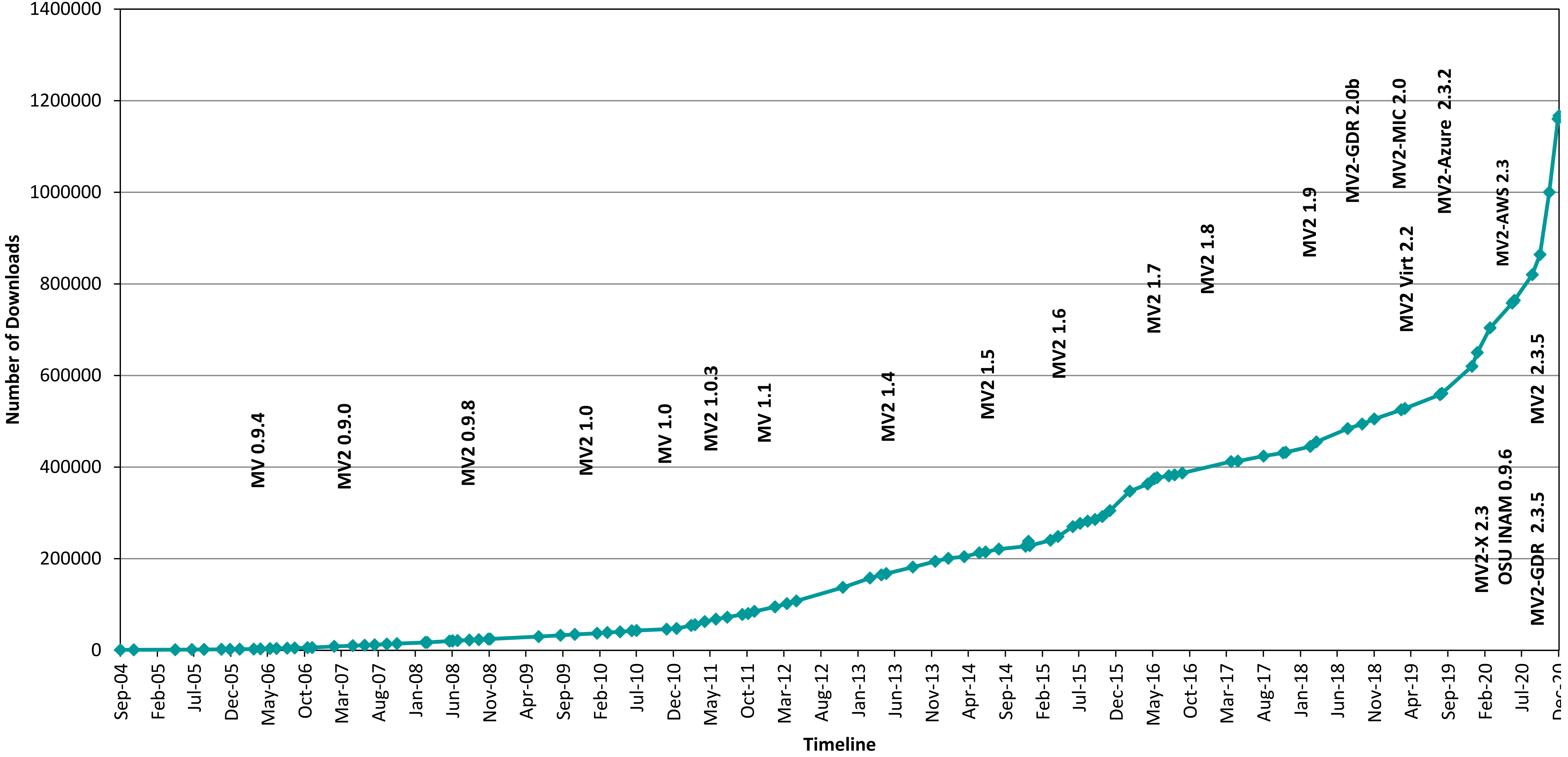
128 process collective tests

osu_allgather	1 21.83 17.34 20 %	2 23.72 16.16 31 %	4 22.11 17.28 21 %	8 24.40 17.87 26 %	16 28.29 19.89 29 %	32 38.99 21.65 44 %	64 28.70 26.24 8 %	128 47.85 43.90 8 %	256 66.45 63.03 5 %	512 105.64 102.18 3 %	1K 914.13 177.28 80 %	2K 2330.43 276.96 88 %	4K 726.91 502.60 30 %	8K 1197.84 1272.69 -6 %	16K 2460.79 2555.89 -3 %	32K 3996.50 3247.50 18 %	64K 6274.14 5516.63 12 %	128K 13385.82 10428.17 22 %	256K 20382.71 20449.20 0 %	512K 43029.04 43000.04 0 %	1M 102436.87 88526.72 13 %
osu_allgatherv	1 25.66 21.91 14 %	2 27.61 22.04 20 %	4 25.41 19.47 23 %	8 27.68 20.73 25 %	16 31.34 23.41 25 %	32 40.98 25.06 38 %	64 78.11 29.32 62 %	128 133.69 50.28 62 %	256 207.27 133.06 35 %	512 505.08 233.32 53 %	1K 529.16 239.41 54 %	2K 558.27 284.32 49 %	4K 689.62 525.36 23 %	8K 1176.09 1254.16 -6 %	16K 2461.61 2524.56 -2 %	32K 3898.46 3167.98 18 %	64K 6573.11 5468.24 16 %	128K 11721.43 10026.13 14 %	256K 21588.64 20060.80 7 %	512K 43244.00 43166.45 0 %	1M 102924.40 87885.28 14 %
osu_allreduce	4 36.82 19.08 48 %	8 32.90 22.28 32 %	16 32.66 19.37 40 %	32 33.15 19.37 41 %	64 35.42 21.29 39 %	128 36.90 27.06 26 %	256 41.44 22.96 44 %	512 51.83 29.01 44 %	1K 69.81 37.88 45 %	2K 107.22 60.36 43 %	4K 48.50 44.66 7 %	8K 51.63 48.42 6 %	16K 86.36 84.79 1 %	32K 120.66 109.31 9 %	64K 184.02 172.09 6 %	128K 296.64 267.65 9 %	256K 521.36 469.93 9 %	512K 1996.51 967.96 51 %	1M 3221.70 2649.74 17 %	2M 6665.53 6305.41 5 %	4M 9984.70 8972.08 10 %

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

MVAPICH2 Release Timeline and Downloads



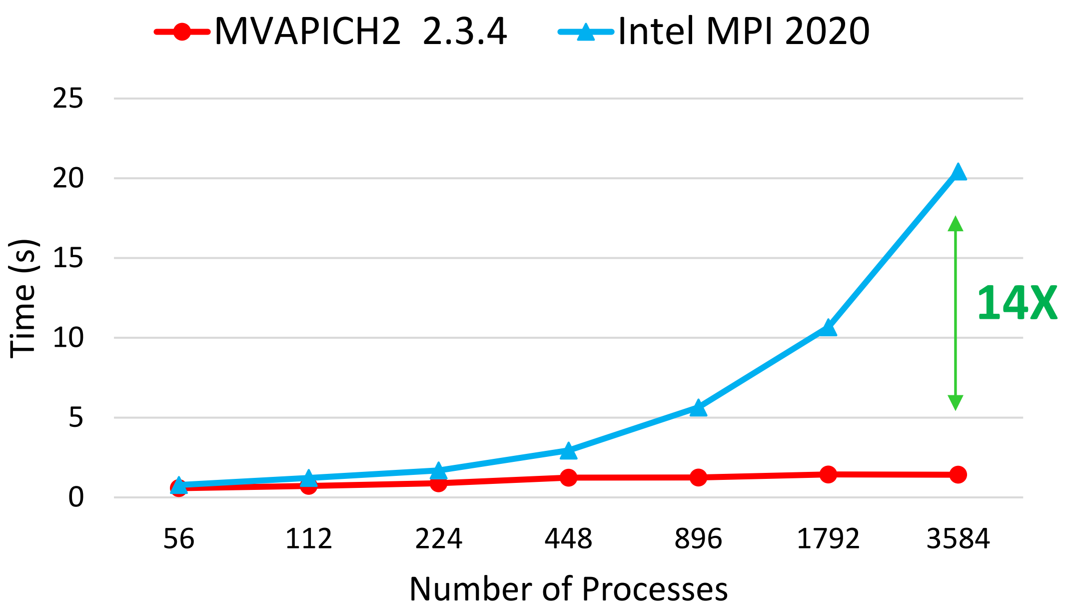
MVAPICH2 Software Family

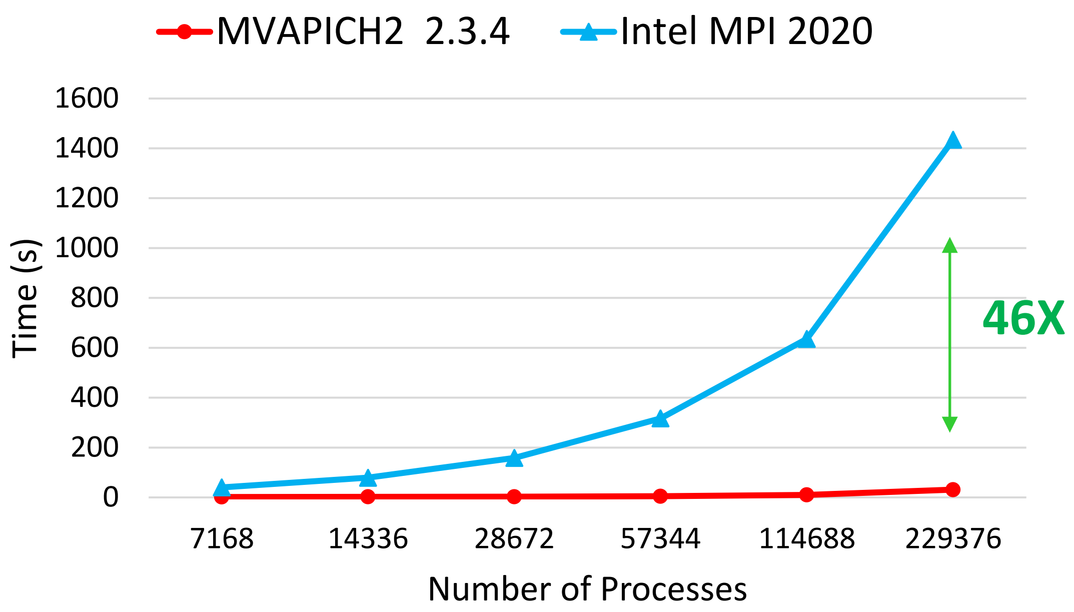
Requirements	Library
MPI with Support for InfiniBand, Omni-Path, Ethernet/iWARP and, RoCE (v1/v2)	MVAPICH2
Optimized Support for Microsoft Azure Platform with InfiniBand	MVAPICH2-Azure
Advanced MPI features/support (UMR, ODP, DC, Core-Direct, SHArP, XPMEM), OSU INAM (InfiniBand Network Monitoring and Analysis),	MVAPICH2-X
Advanced MPI features (SRD and XPMEM) with support for Amazon Elastic Fabric Adapter (EFA)	MVAPICH2-X-AWS
Optimized MPI for clusters with NVIDIA GPUs and for GPU-enabled Deep Learning Applications	MVAPICH2-GDR
Energy-aware MPI with Support for InfiniBand, Omni-Path, Ethernet/iWARP and, RoCE (v1/v2)	MVAPICH2-EA
MPI Energy Monitoring Tool	OEMT
InfiniBand Network Analysis and Monitoring	OSU INAM
Microbenchmarks for Measuring MPI and PGAS Performance	OMB

Overview of MVAPICH2 Features

- Job start-up
- Transport Type Selection
- Collectives
- Support for MPI Tools (MPI_T) Interface
- Solutions for NVIDIA/AMD GPU-enabled Systems
- MPI-based Deep Learning for CPUs and GPUs
- Accelerating Data Science Applications
- Application Specific Tuning

Startup Performance on TACC Frontera

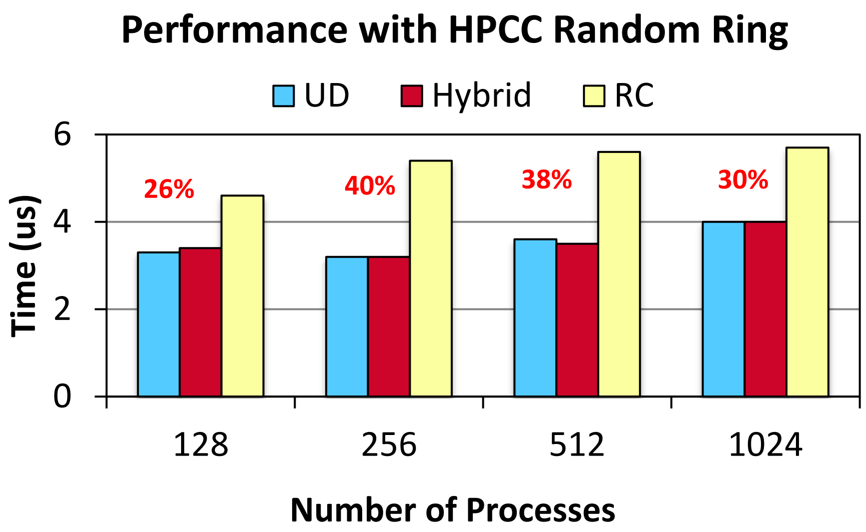




- MPI_Init takes 31 seconds on 229,376 processes on 4,096 nodes
- All numbers reported with 56 processes per node

New designs available from MVAPICH2-2.3.4

Transport Protocol Selection in MVAPICH2

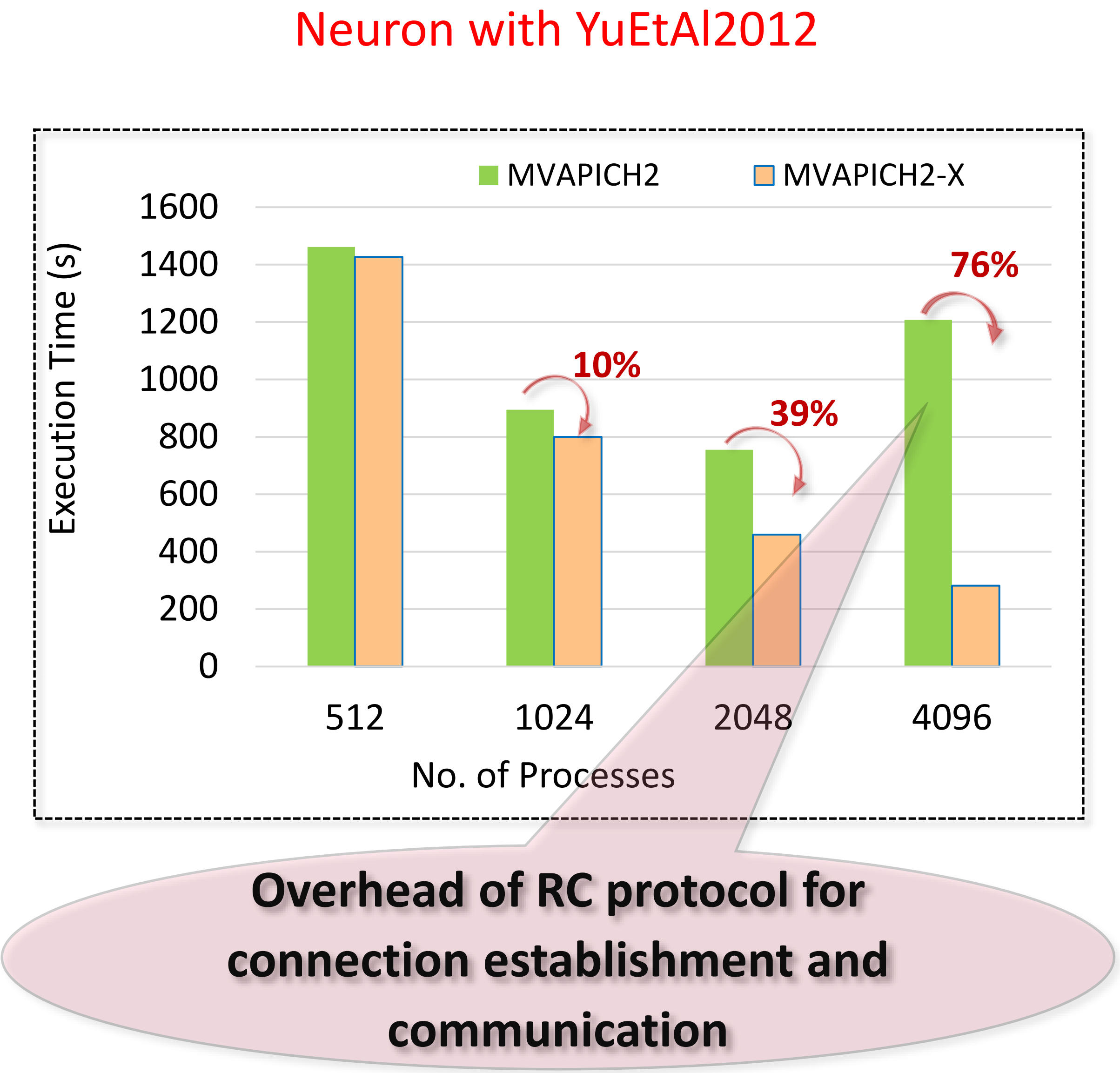


- Both UD and RC/XRC have benefits
  - Hybrid for the best of both
- Enabled by configuring MVAPICH2 with the –enable-hybrid
- Available since MVAPICH2 1.7 as integrated interface

Parameter	Significance	Default	Notes
MV2_USE_UD_HYBRID	• Enable / Disable use of UD transport in Hybrid mode	Enabled	• Always Enable
MV2_HYBRID_ENABLE_THRESHOLD_SIZE	• Job size in number of processes beyond which hybrid mode will be enabled	1024	• Uses RC/XRC connection until job size < threshold
MV2_HYBRID_MAX_RC_CONN	• Maximum number of RC or XRC connections created per process • Limits the amount of connection memory	64	• Prevents HCA QP cache thrashing

- **Refer to Running with Hybrid UD-RC/XRC section of MVAPICH2 user guide for more information**

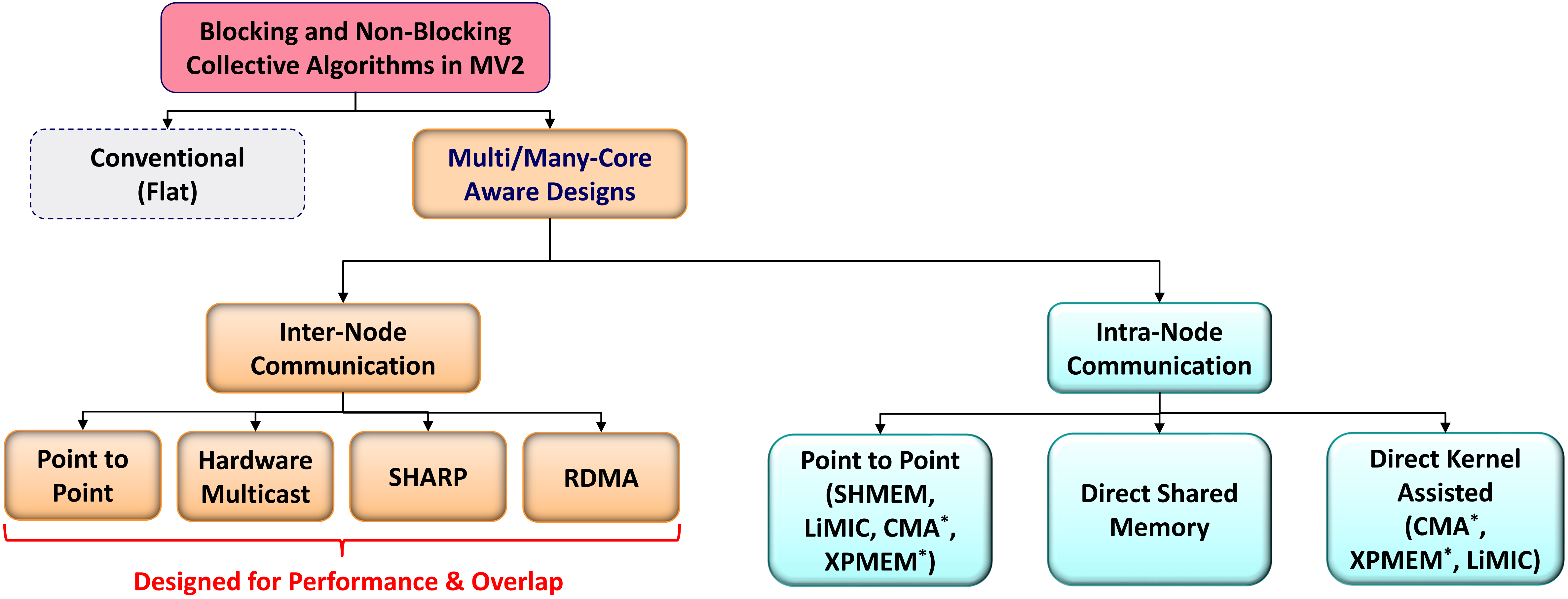
Impact of DC Transport Protocol on Neuron



- Up to 76% benefits over MVAPICH2 for Neuron using Direct Connected transport protocol at scale
  - VERSION 7.6.2 master (f5a1284) 2018-08-15
- Numbers taken on bbpv2.epfl.ch
  - Knights Landing nodes with 64 ppn
  - ./x86_64/special -mpi -c stop_time=2000 -c is_split=1 parinit.hoc
  - Used “runtime” reported by execution to measure performance
- Environment variables used
  - MV2_USE_DC=1
  - MV2_NUM_DC_TGT=64
  - MV2_SMALL_MSG_DC_POOL=96
  - MV2_LARGE_MSG_DC_POOL=96
  - MV2_USE_RDMA_CM=0

Available from MVAPICH2-X 2.3rc2 onwards

Collective Communication in MVAPICH2



Run-time flags:

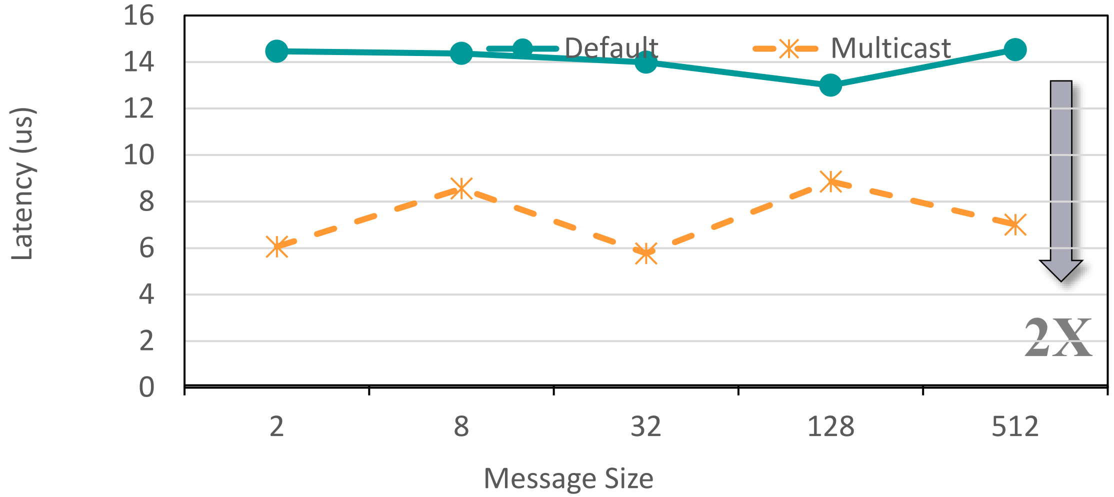
All shared-memory based collectives : MV2_USE_SHMEM_COLL (Default: ON)

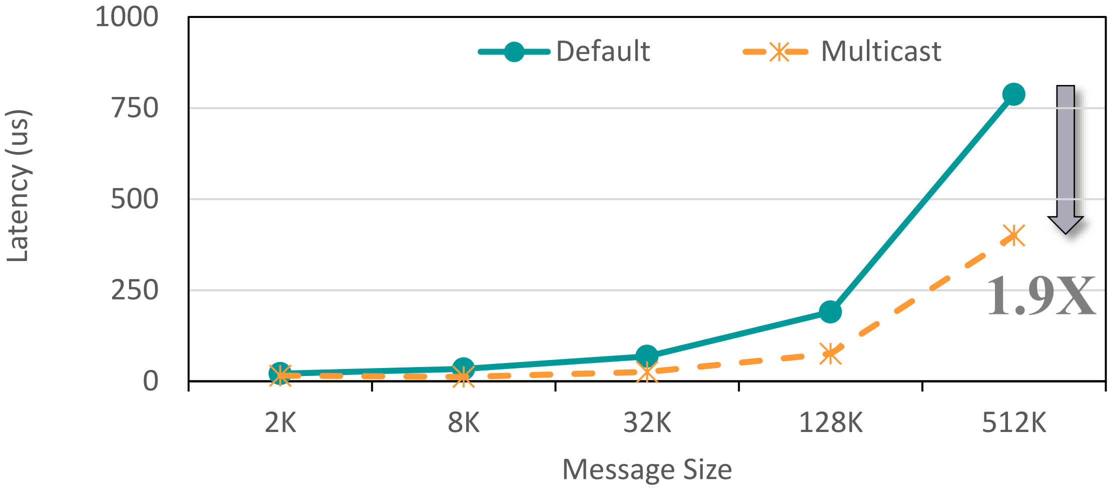
Hardware Mcast-based collectives : MV2_USE_MCAST (Default : OFF)

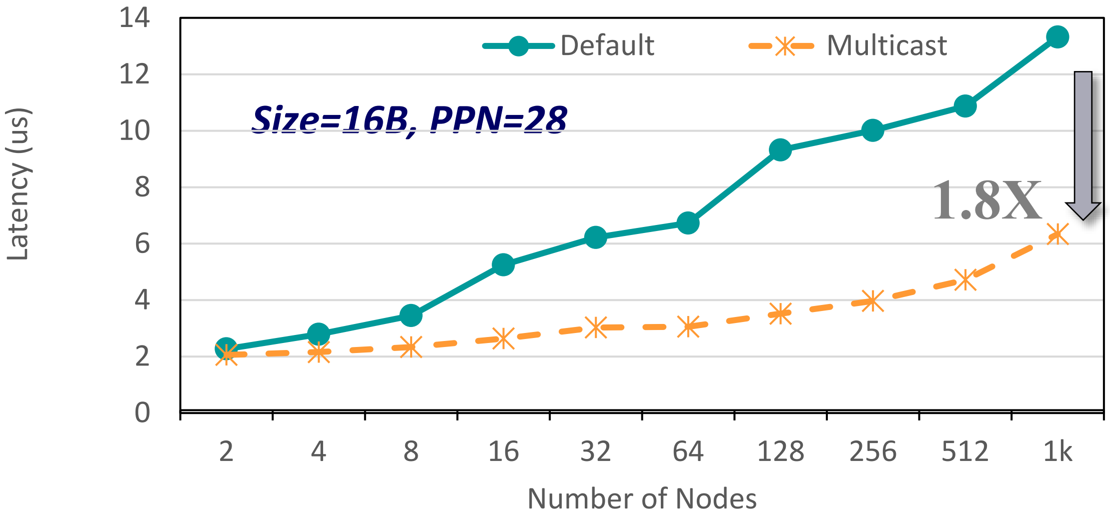
CMA and XPMEM-based collectives are in MVAPICH2-X

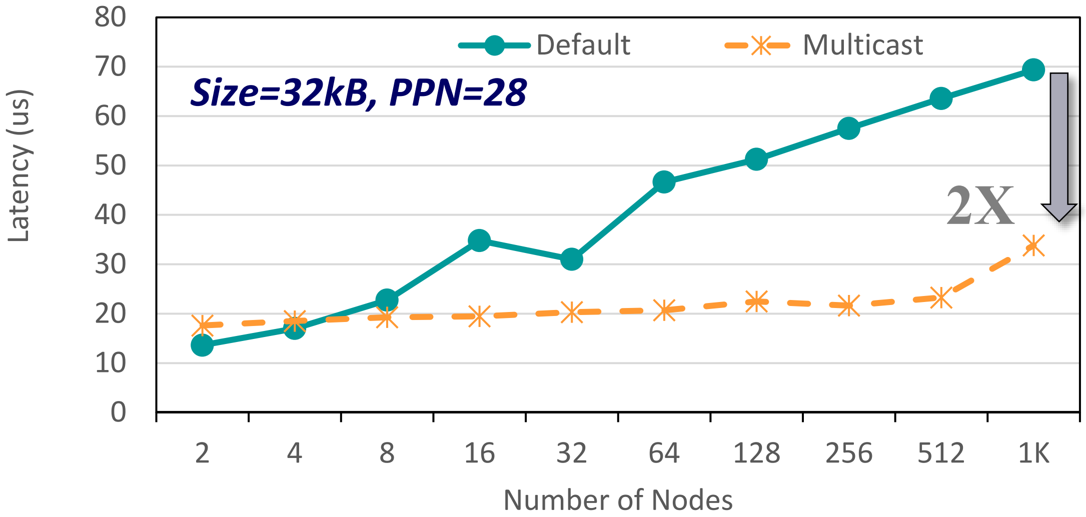
Hardware Multicast-aware MPI_Bcast on TACC Frontera

(Nodes=2K, PPN=28)







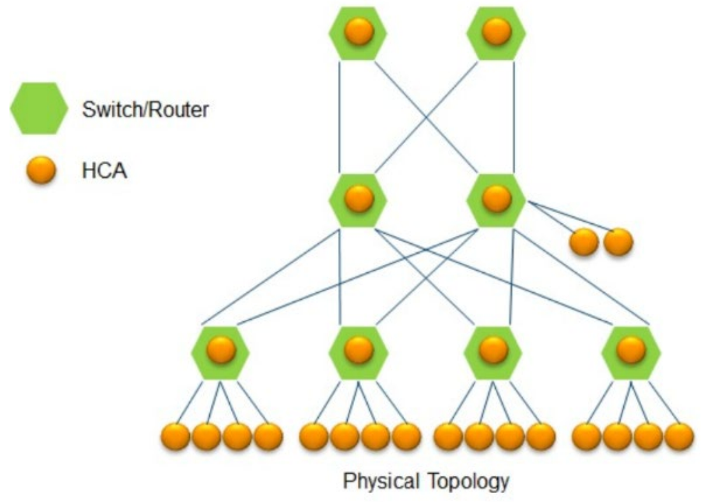


- MCAST-based designs improve latency of MPI_Bcast by up to **2X at 2,048 nodes**
- Use MV2_USE_MCAST=1 to enable MCAST-based designs

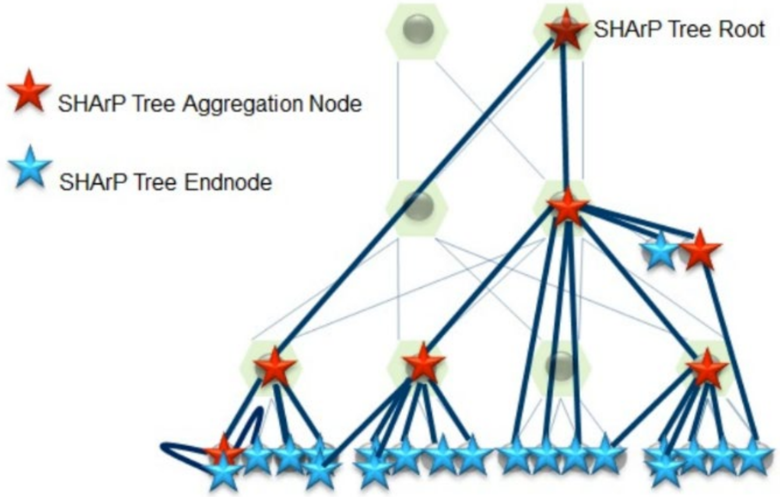
Offloading with Scalable Hierarchical Aggregation Protocol (SHArP)

- Management and execution of MPI operations in the network by using SHArP
  - Manipulation of data while it is being transferred in the switch network
- SHArP provides an abstraction to realize the reduction operation
  - Defines Aggregation Nodes (AN), Aggregation Tree, and Aggregation Groups
  - AN logic is implemented as an InfiniBand Target Channel Adapter (TCA) integrated into the switch ASIC *
  - Uses RC for communication between ANs and between AN and hosts in the Aggregation Tree *

More details in the tutorial "SHARPv2: In-Network Scalable Streaming Hierarchical Aggregation and Reduction Protocol" by Devendar Bureddy (NVIDIA/Mellanox)



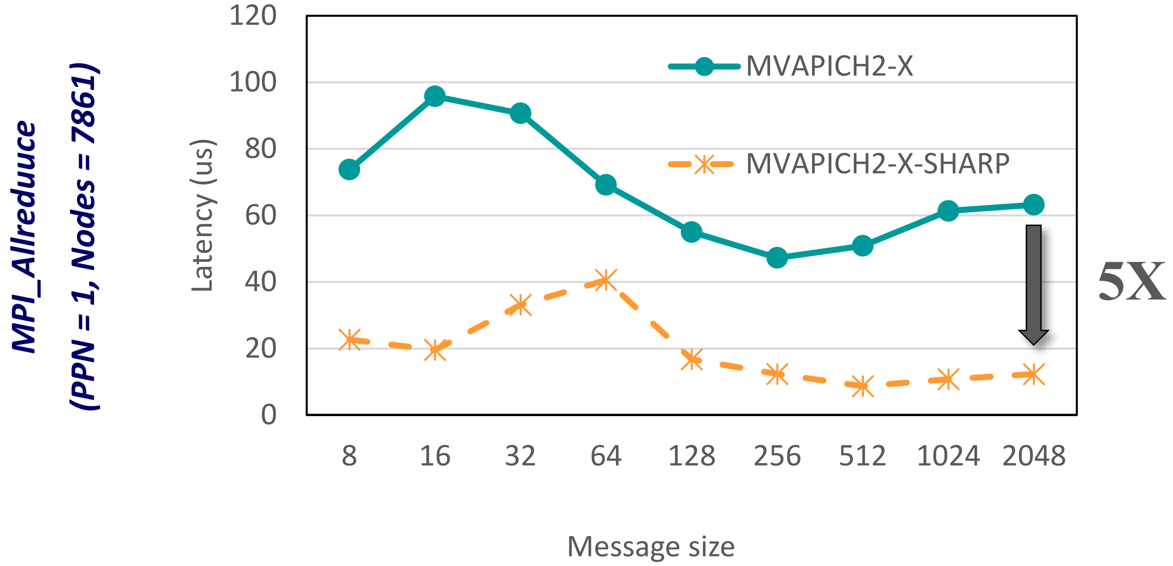
Physical Network Topology*

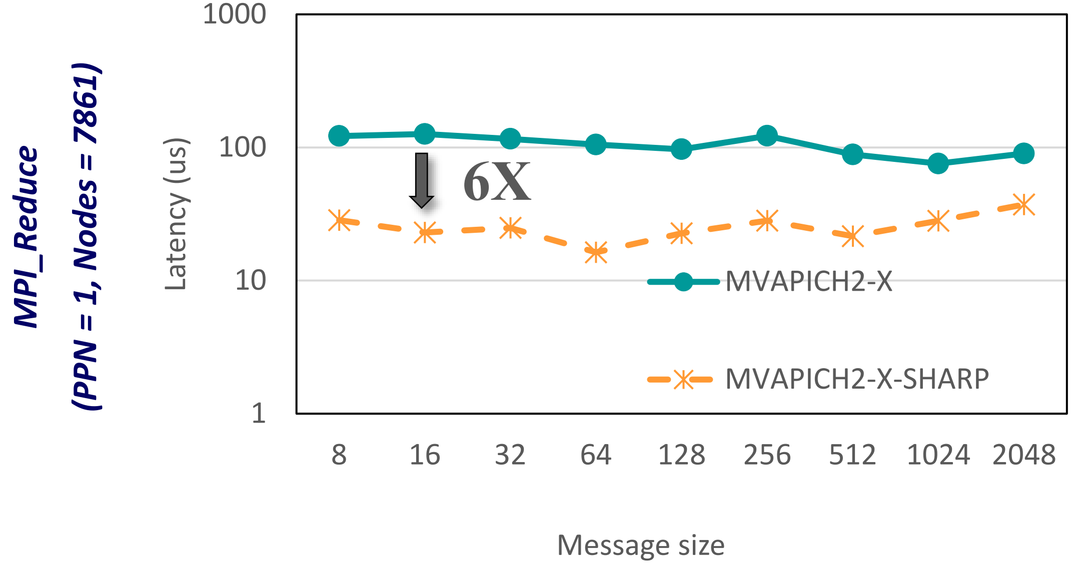


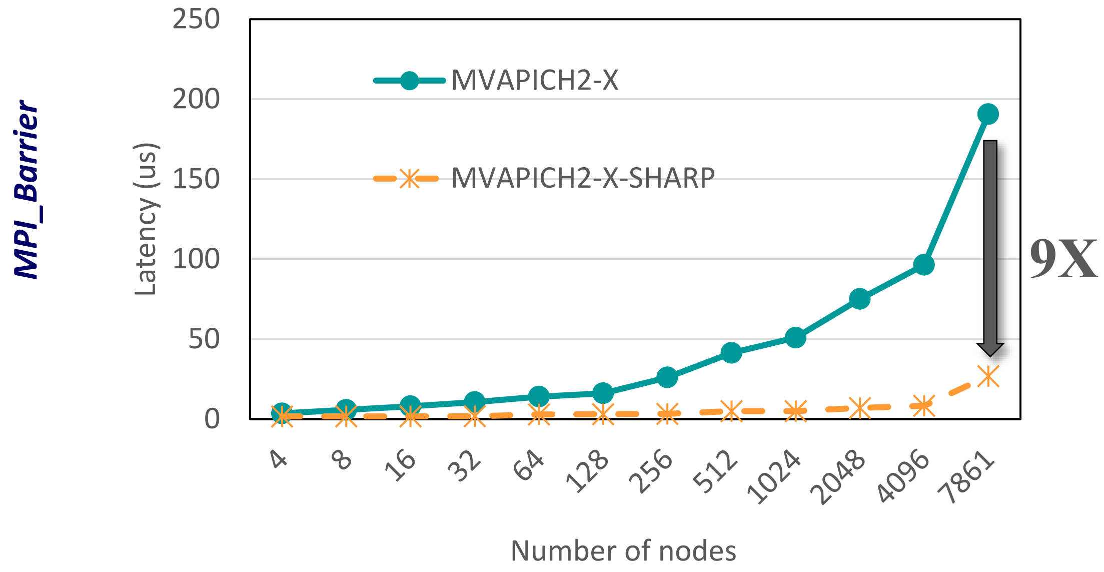
Logical SHArP Tree*

* Bloch et al. Scalable Hierarchical Aggregation Protocol (SHArP): A Hardware Architecture for Efficient Data Reduction

Performance of Collectives with SHARP on TACC Frontera







Optimized SHARP designs in MVAPICH2-X

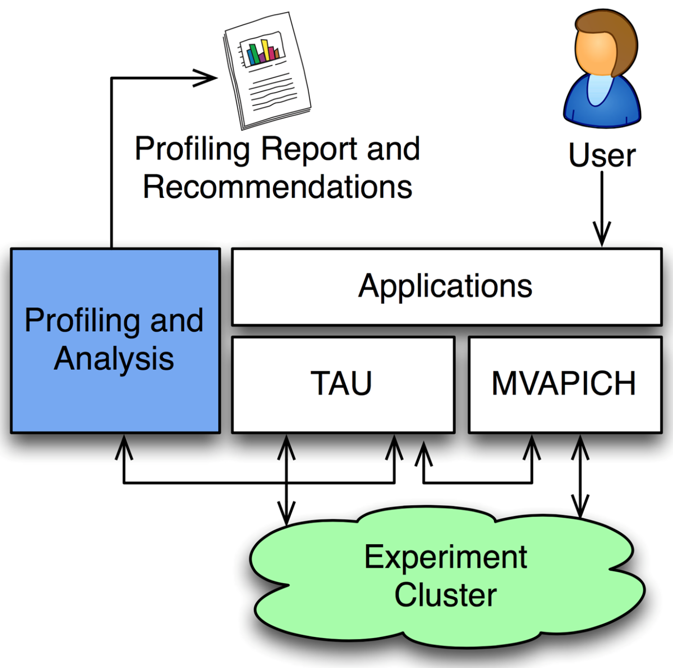
Up to 9X performance improvement with SHARP over MVAPICH2-X default for 1ppn MPI_Barrier, ***6X*** for 1ppn MPI_Reduce and ***5X*** for 1ppn MPI_Allreduce

B. Ramesh , K. Suresh , N. Sarkauskas , M. Bayatpour , J. Hashmi , H. Subramoni , and D. K. Panda, Scalable MPI Collectives using SHARP: Large Scale Performance Evaluation on the TACC Frontera System, ExaMPI2020 - Workshop on Exascale MPI 2020, Nov 2020.

Optimized Runtime Parameters: MV2_ENABLE_SHARP = 1

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*



Available in MVAPICH2

VBUF usage without CVAR based tuning as displayed by ParaProf

TAU: ParaProf: Context Events for: node 0 - mpit_withoutcvar_bt.C.1k.ppk

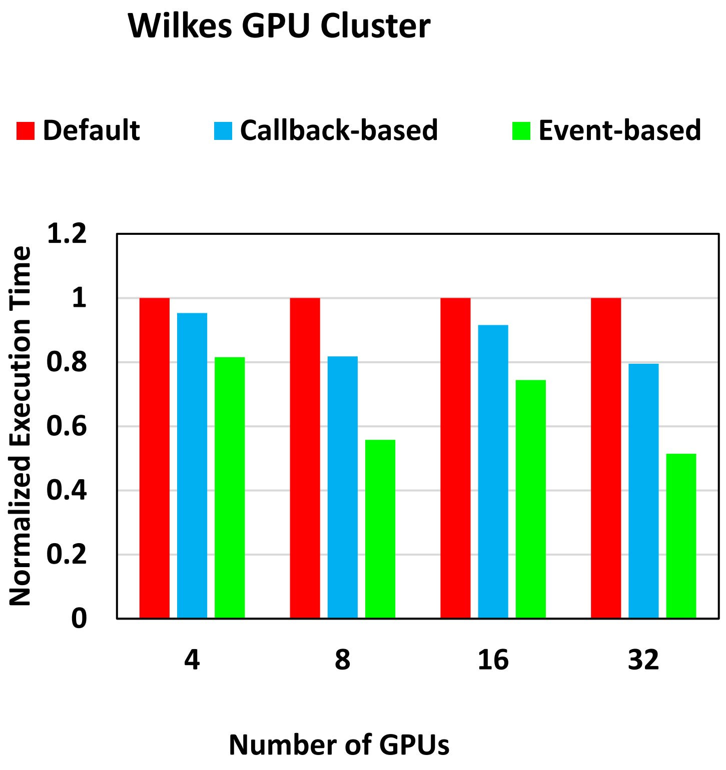
Name	MaxValue	MinValue	MeanValue	Std. Dev.	NumSamples	Total
mv2_total_vbuf_memory (Total amount of memory in bytes used for VBUFs)	3,313,056	3,313,056	3,313,056	0	1	3,313,056
mv2_ud_vbuf_allocated (Number of UD VBUFs allocated)	0	0	0	0	0	0
mv2_ud_vbuf_available (Number of UD VBUFs available)	0	0	0	0	0	0
mv2_ud_vbuf_freed (Number of UD VBUFs freed)	0	0	0	0	0	0
mv2_ud_vbuf_inuse (Number of UD VBUFs inuse)	0	0	0	0	0	0
mv2_ud_vbuf_max_use (Maximum number of UD VBUFs used)	0	0	0	0	0	0
mv2_vbuf_allocated (Number of VBUFs allocated)	320	320	320	0	1	320
mv2_vbuf_available (Number of VBUFs available)	255	255	255	0	1	255
mv2_vbuf_freed (Number of VBUFs freed)	25,545	25,545	25,545	0	1	25,545
mv2_vbuf_inuse (Number of VBUFs inuse)	65	65	65	0	1	65
mv2_vbuf_max_use (Maximum number of VBUFs used)	65	65	65	0	1	65
num_calloc_calls (Number of MPIT_calloc calls)	89	89	89	0	1	89

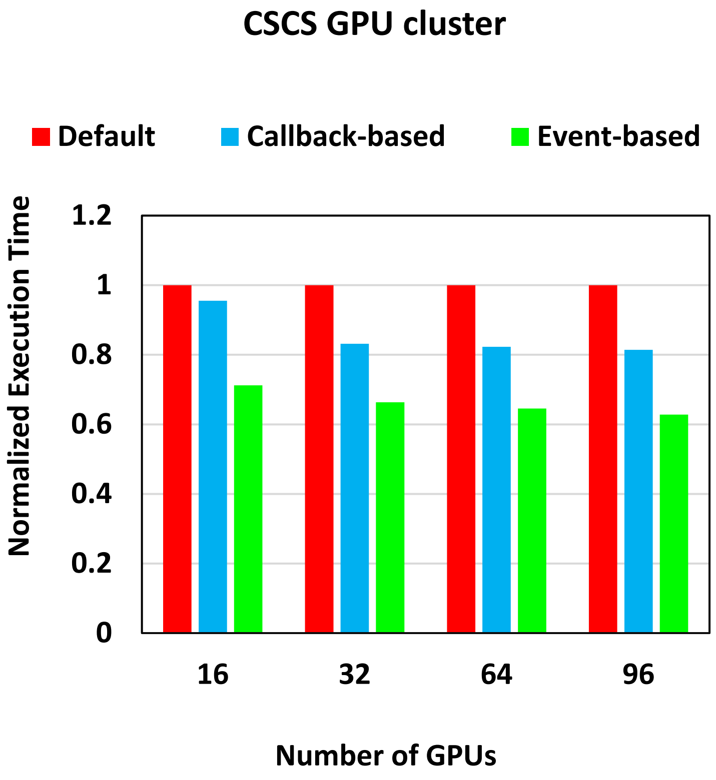
VBUF usage with CVAR based tuning as displayed by ParaProf

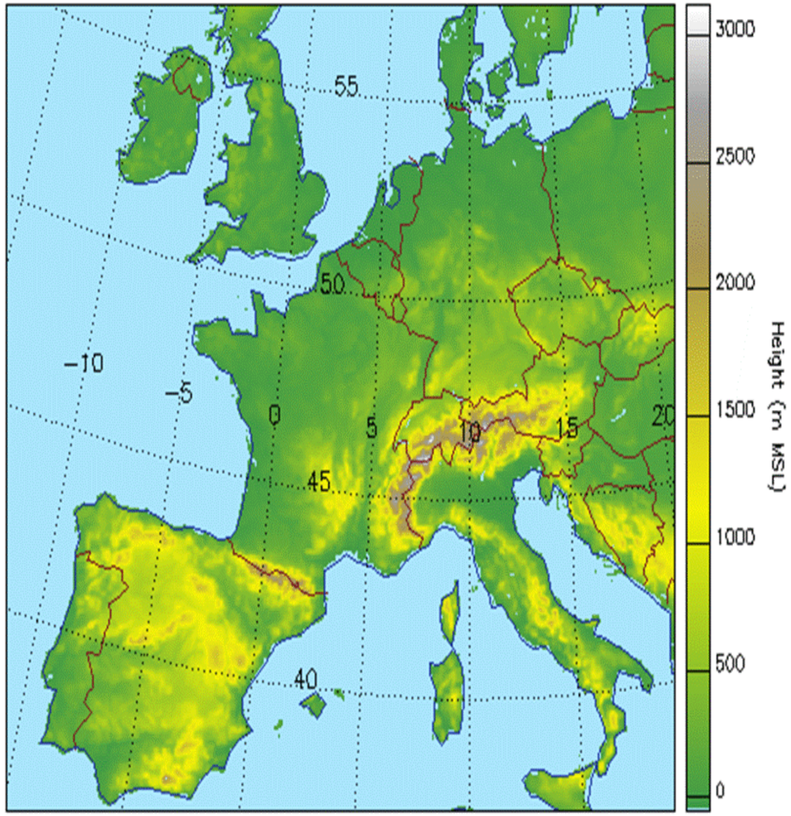
TAU: ParaProf: Context Events for: node 0 - bt-mz.E.vbuf_pool_16.1k.ppk

Name	MaxValue	MinValue	MeanValue	Std. Dev.	NumSamp...	Total
mv2_total_vbuf_memory (Total amount of memory in bytes used for VBUFs)	1,815,056	1,815,056	1,815,056	0	1	1,815,056
mv2_ud_vbuf_allocated (Number of UD VBUFs allocated)	0	0	0	0	0	0
mv2_ud_vbuf_available (Number of UD VBUFs available)	0	0	0	0	0	0
mv2_ud_vbuf_freed (Number of UD VBUFs freed)	0	0	0	0	0	0
mv2_ud_vbuf_inuse (Number of UD VBUFs inuse)	0	0	0	0	0	0
mv2_ud_vbuf_max_use (Maximum number of UD VBUFs used)	0	0	0	0	0	0
mv2_vbuf_allocated (Number of VBUFs allocated)	160	160	160	0	1	160
mv2_vbuf_available (Number of VBUFs available)	94	94	94	0	1	94
mv2_vbuf_freed (Number of VBUFs freed)	5,479	5,479	5,479	0	1	5,479
mv2_vbuf_inuse (Number of VBUFs inuse)	66	66	66	0	1	66

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







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

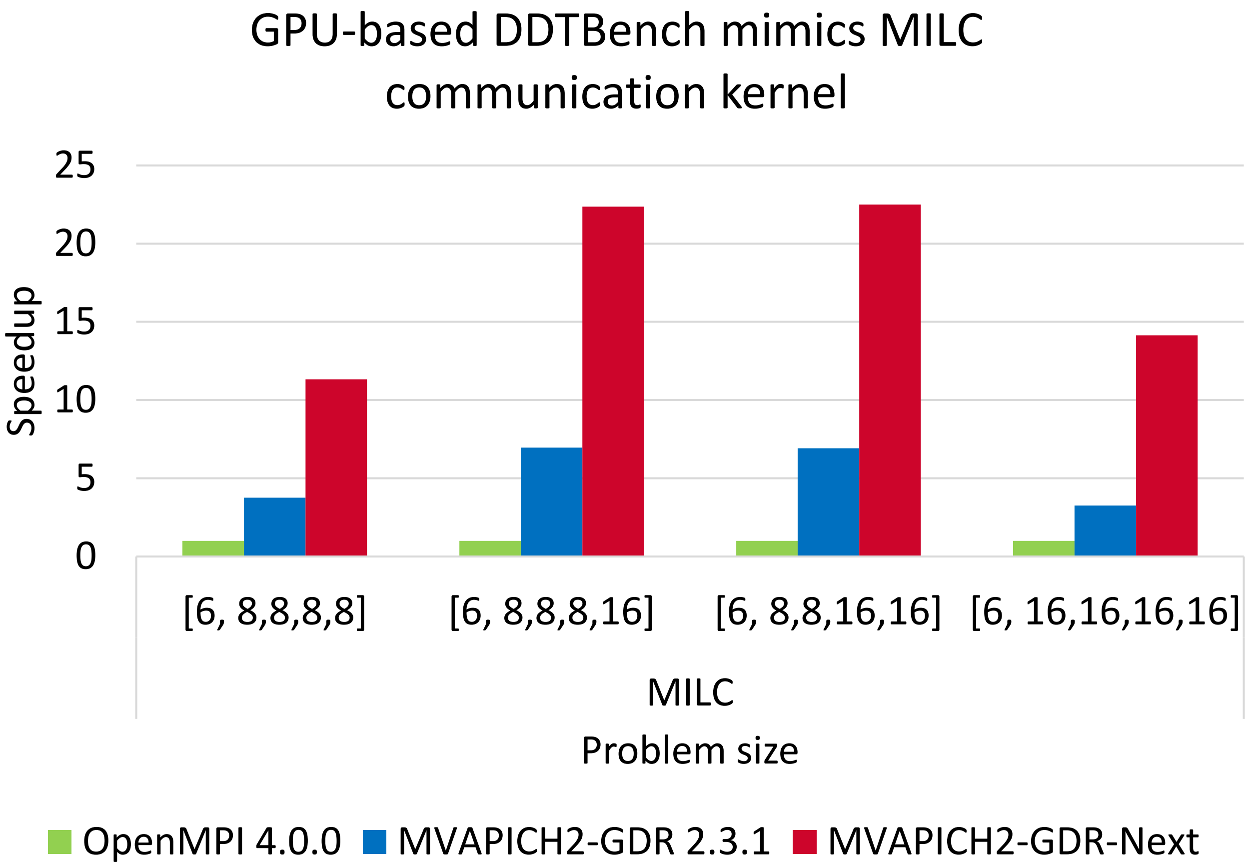
Cosmo model: http://www2.cosmo-model.org/content/tasks/operational/meteoSwiss/

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

C. Chu, K. Hamidouche, A. Venkatesh, D. Banerjee , H. Subramoni, and D. K. Panda, Exploiting Maximal Overlap for Non-Contiguous Data Movement Processing on Modern GPU-enabled Systems, IPDPS'16

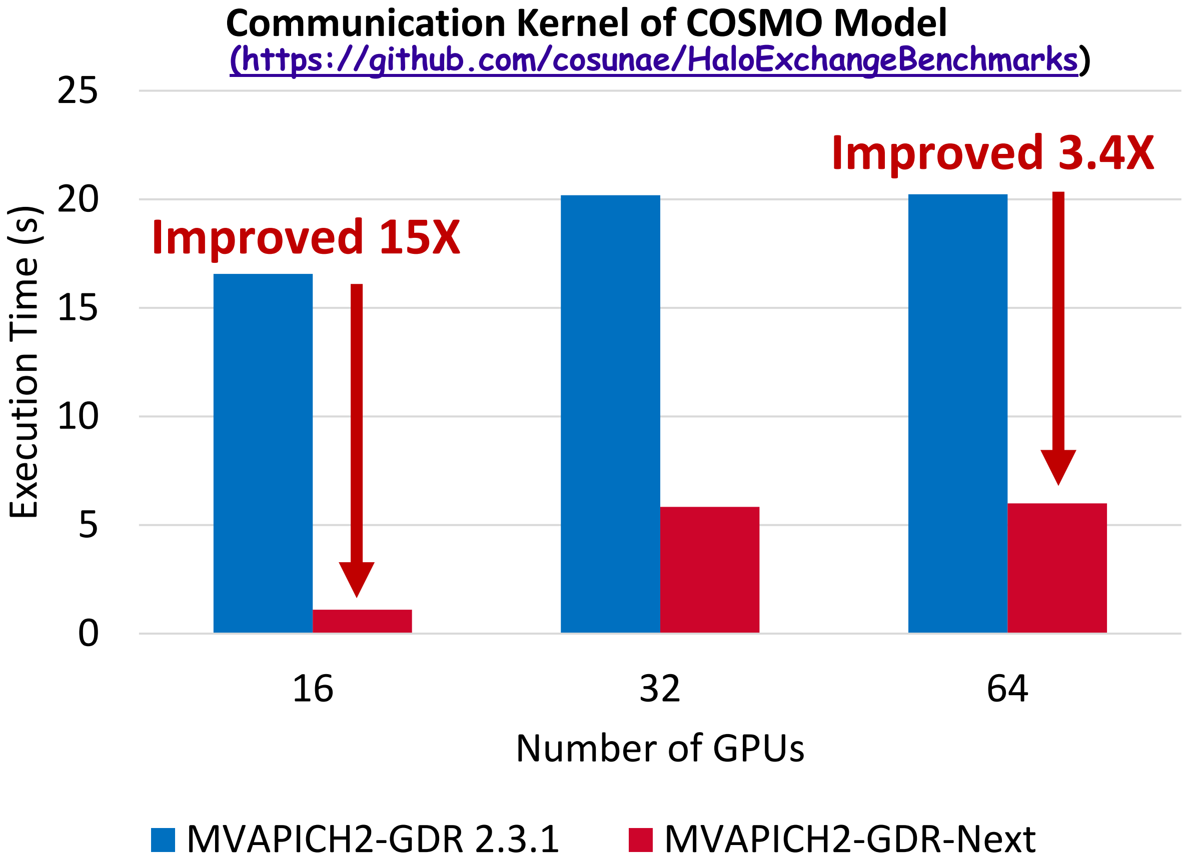
MVAPICH2-GDR: Enhanced Derived Datatype

- Kernel-based and GDRCOPY-based one-shot packing for inter-socket and inter-node communication
- Zero-copy (packing-free) for GPUs with peer-to-peer direct access over PCIe/NVLink



Platform: Nvidia DGX-2 system

(NVIDIA Volta GPUs connected with NVSwitch), CUDA 9.2

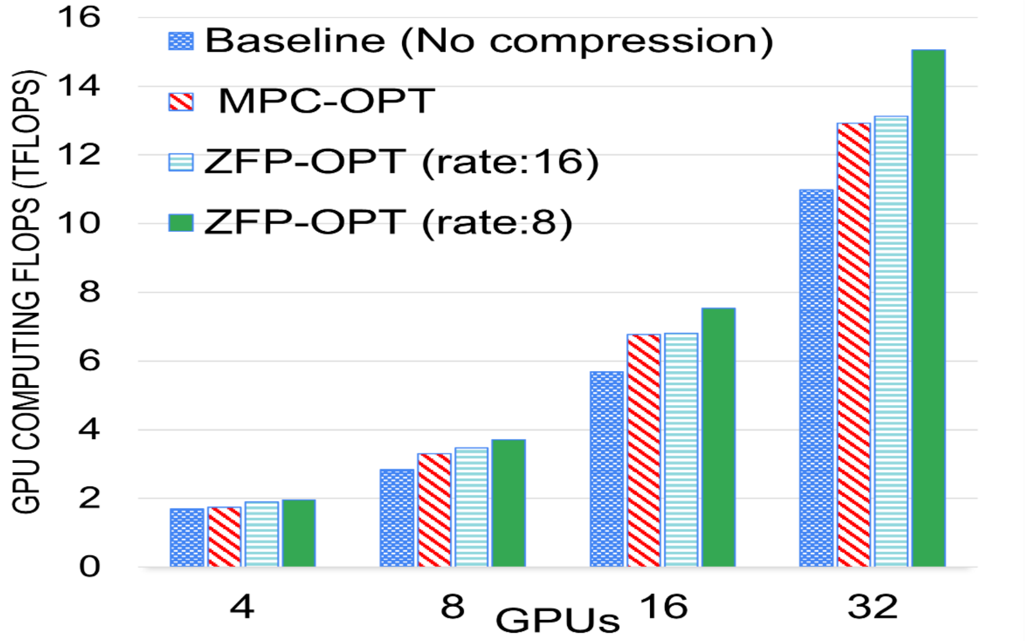


Platform: Cray CS-Storm

(16 NVIDIA Tesla K80 GPUs per node), CUDA 8.0

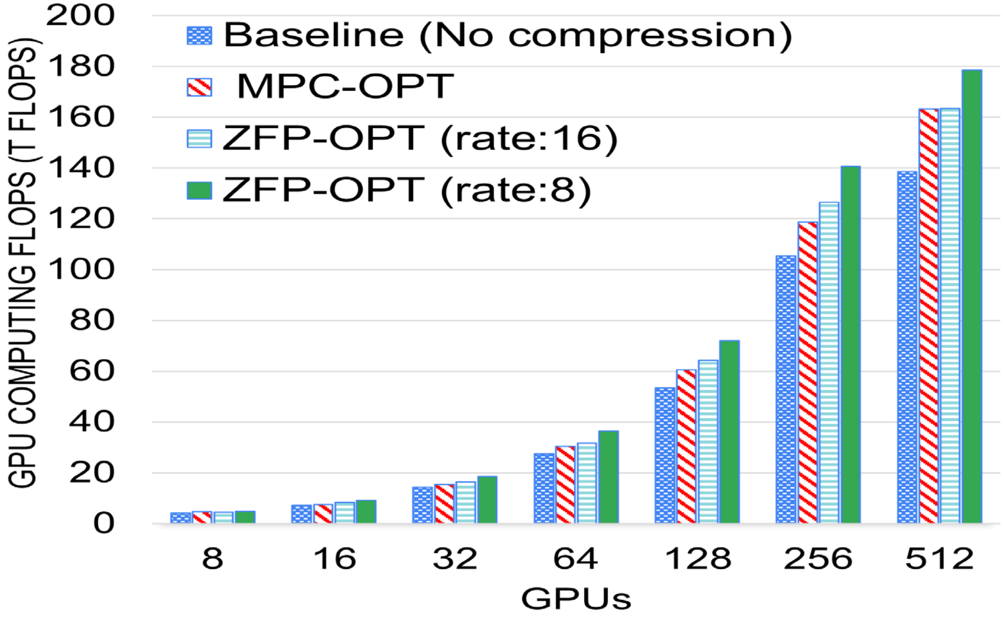
MVAPICH2-GDR: Support for Real-Time Compression

- Designs GPU-assisted on-the-fly message compression show 37% higher GFLOPs for the AWP-ODC on Frontera-Liquid and Frontera-Longhorn
- *Will be available in future MVAPICH2-GDR releases*



Weak scaling of AWP-ODC on Frontera Liquid

4 GPUs/node (higher is better)



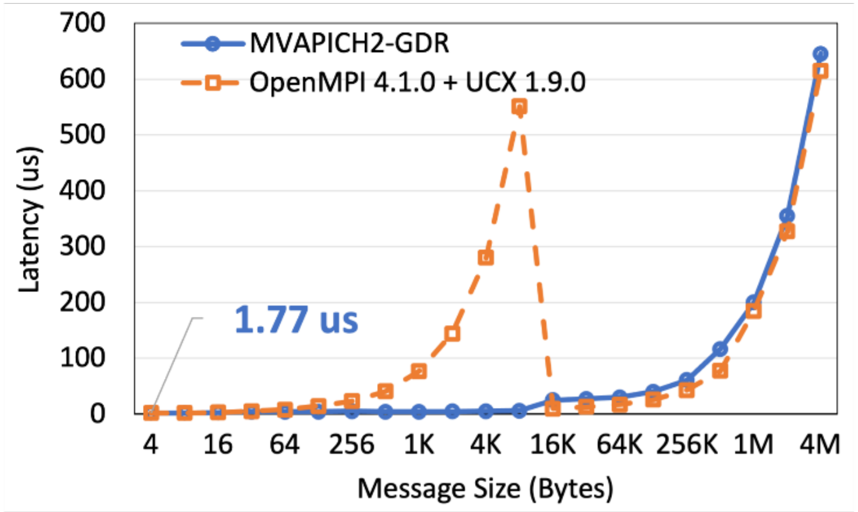
Weak scaling of AWP-ODC on Lassen

4 GPUs/node (higher is better)

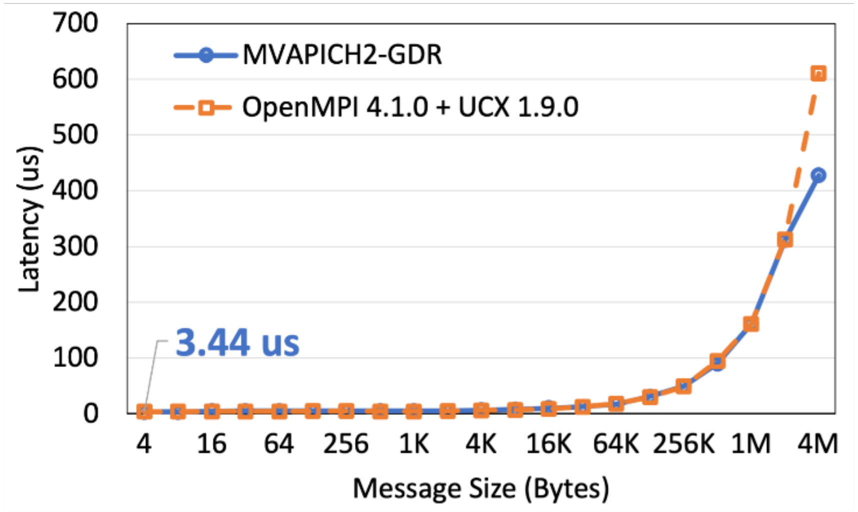
Q. Zhou, C. Chu, N. S. Kumar, P. Kousha, S. M. Ghazimirsaeed, H. Subramoni and D. K. Panda, "Designing High-Performance MPI Libraries with On-the-fly Compression for Modern GPU Clusters", IPDPS'20 ***(Accepted to be presented)***

MVAPICH2-GDR ROCm Support for AMD GPUs

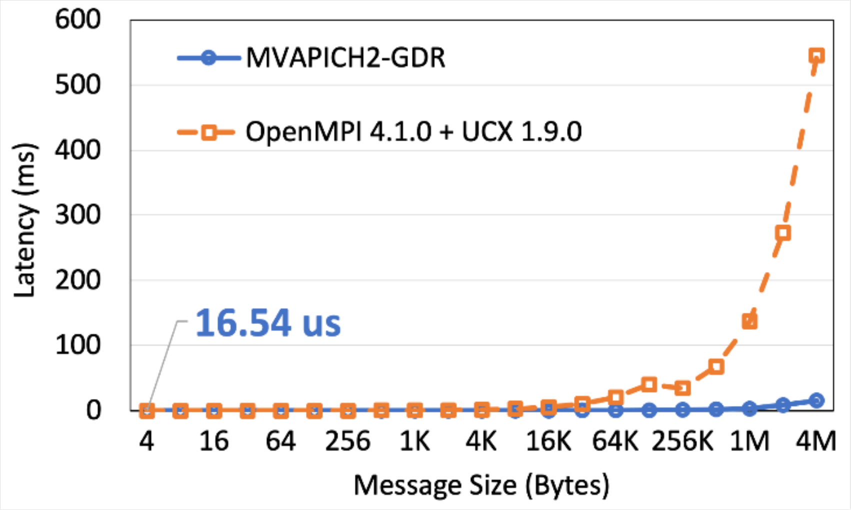
Intra-Node Point-to-Point Latency



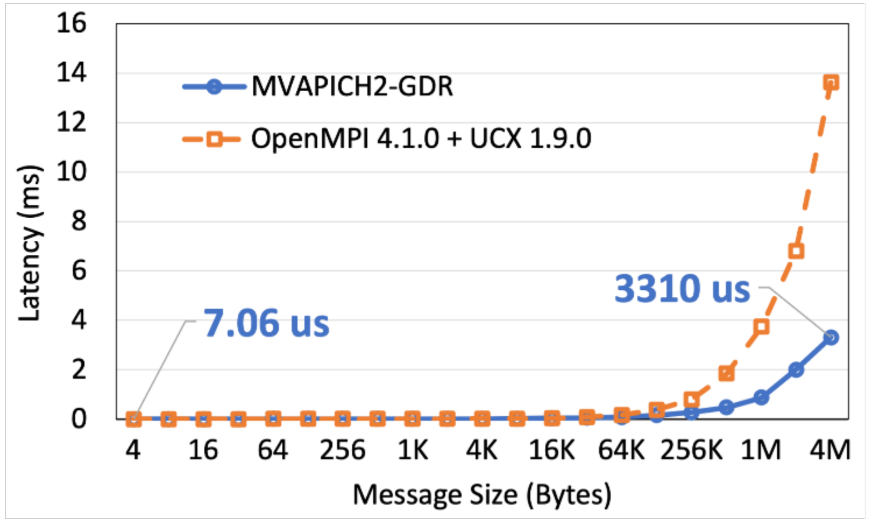
Inter-Node Point-to-Point Latency



Allreduce – 64 GPUs (8 nodes, 8 GPUs Per Node)



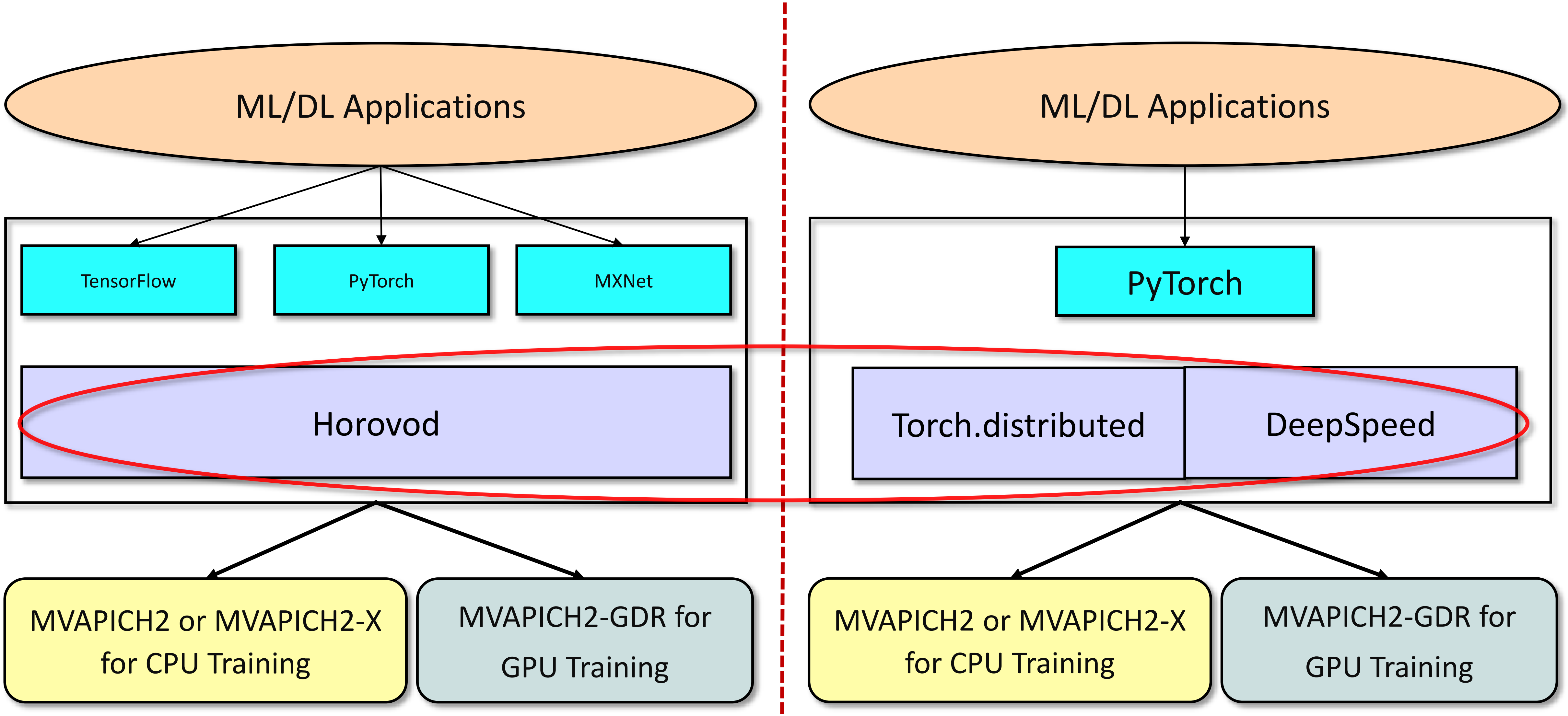
Bcast – 64 GPUs (8 nodes, 8 GPUs Per Node)



Corona Cluster - ROCm-3.9.0 (mi50 AMD GPUs)

Available with MVAPICH2-GDR 2.3.5

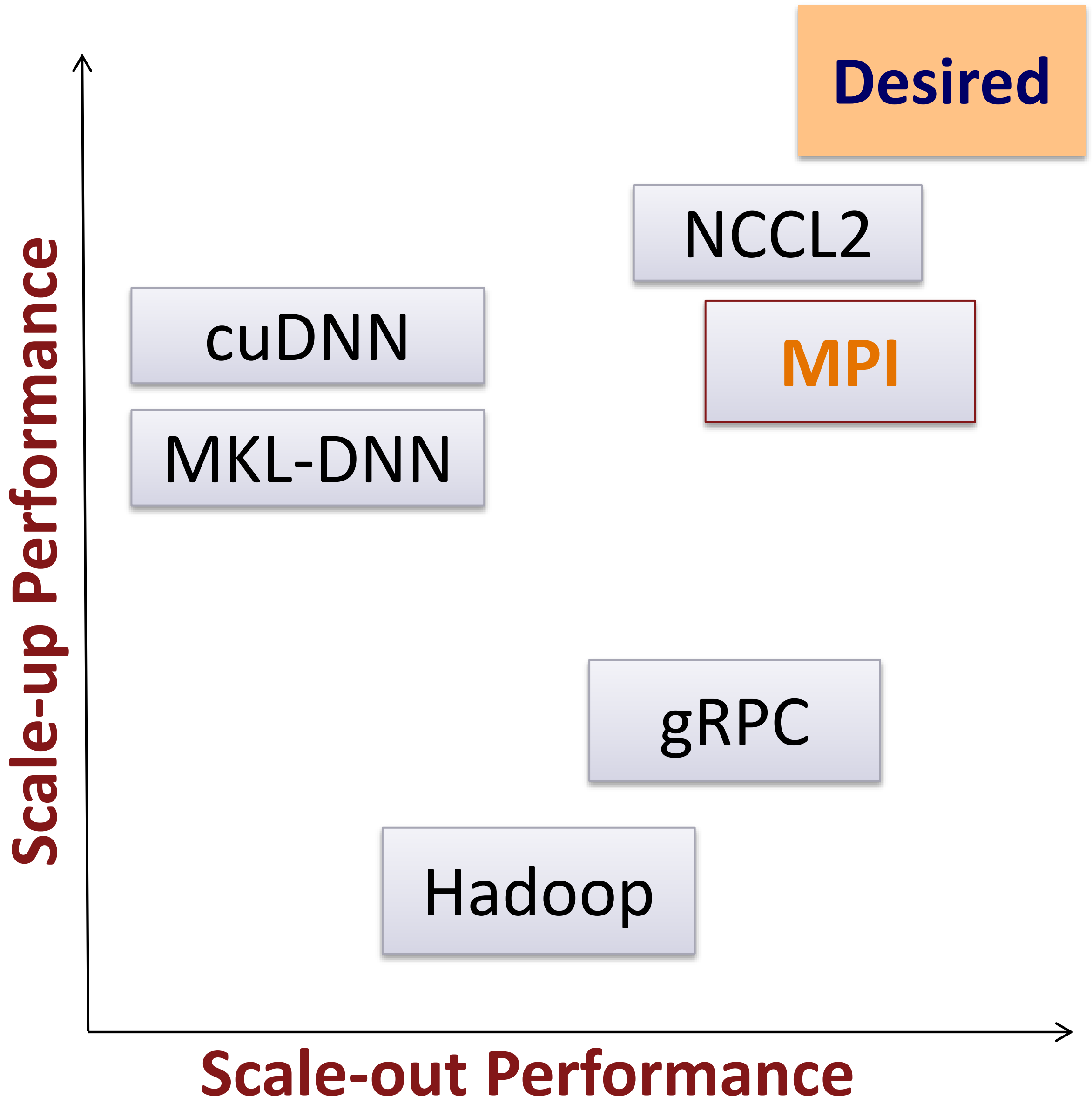
MVAPICH2 (MPI)-driven Infrastructure for ML/DL Training



More details available from: http://hidl.cse.ohio-state.edu

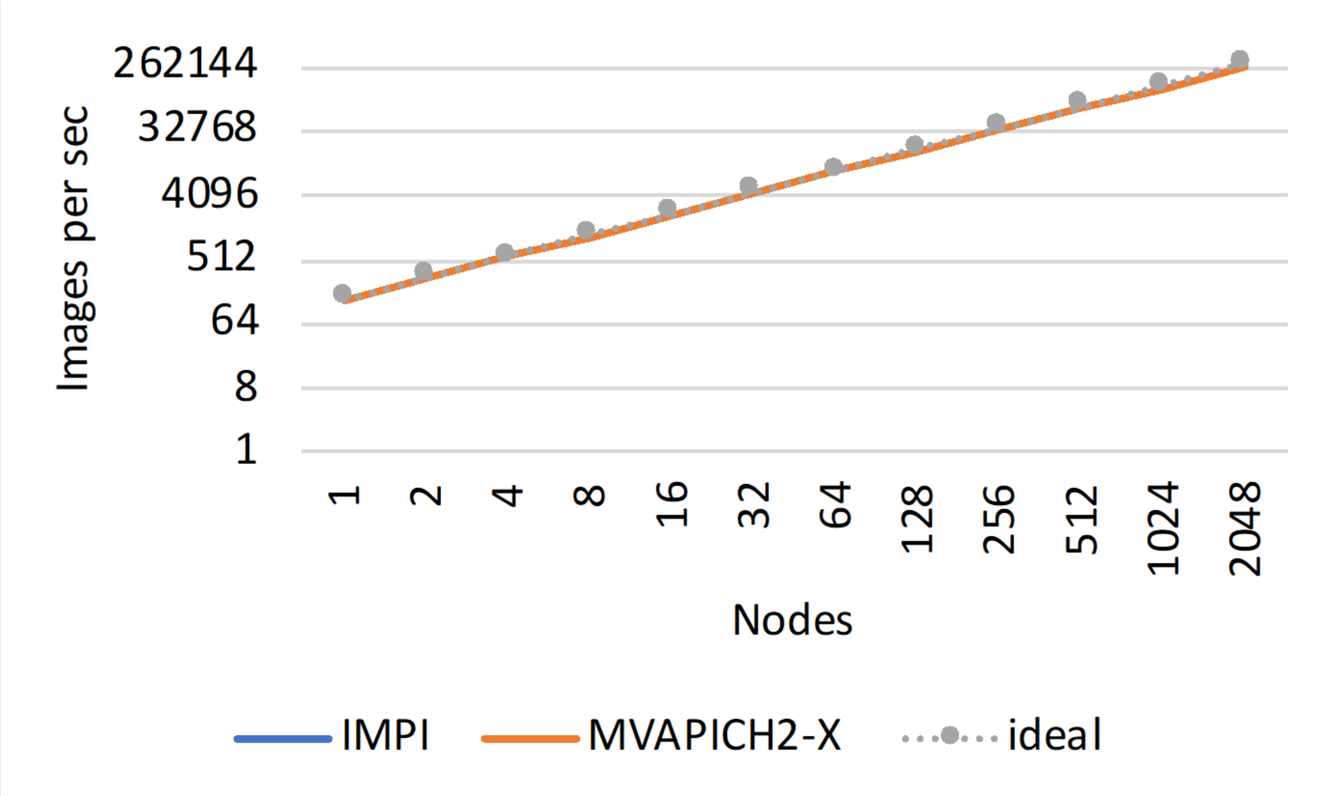
Deep Learning: New Challenges for Runtimes

- **Scale-up**: Intra-node Communication
  - Many improvements like:
    - NVIDIA cuDNN, cuBLAS, NCCL, etc.
    - CUDA 9 Co-operative Groups
- **Scale-out**: Inter-node Communication
  - DL Frameworks – most are optimized for single-node only
  - Distributed (Parallel) Training is an emerging trend
    - **OSU-Caffe – MPI-based**
    - Microsoft CNTK – MPI/NCCL2
    - Google TensorFlow – gRPC-based/MPI/NCCL2
    - Facebook Caffe2 – Hybrid (NCCL2/Gloo/MPI)
    - PyTorch



Distributed TensorFlow on TACC Frontera (2,048 CPU nodes with 114,688 cores)

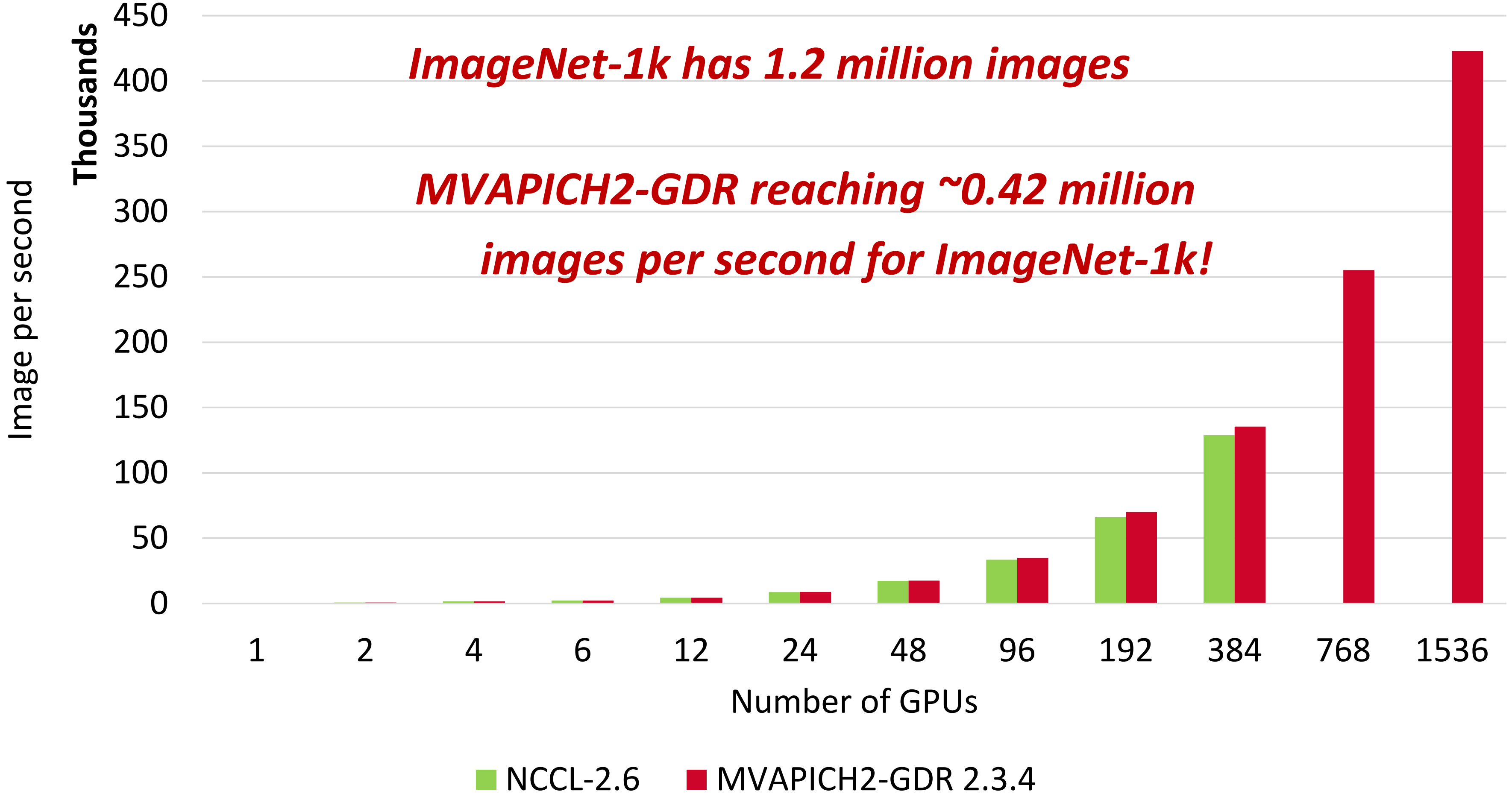
- Scaled TensorFlow to 2048 nodes on Frontera using MVAPICH2
- MVAPICH2 and IntelMPI give similar performance for DNN training
- Report a peak of 260,000 images/sec on 2,048 nodes
- On 2048 nodes, ResNet-50 can be trained in 7 minutes!



A. Jain, A. A. Awan, H. Subramoni, DK Panda, "Scaling TensorFlow, PyTorch, and MXNet using MVAPICH2 for High-Performance Deep Learning on Frontera", DLS '19 (SC '19 Workshop).

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!**

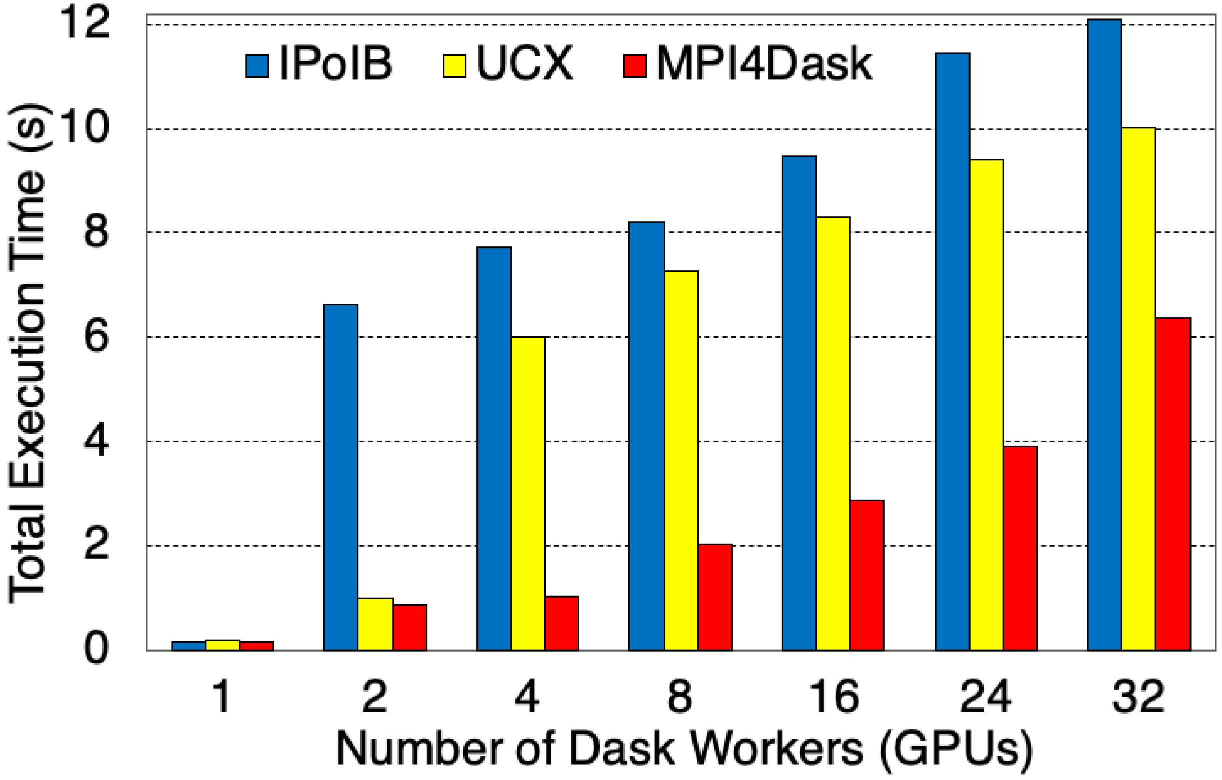


*We observed issues for NCCL2 beyond 384 GPUs

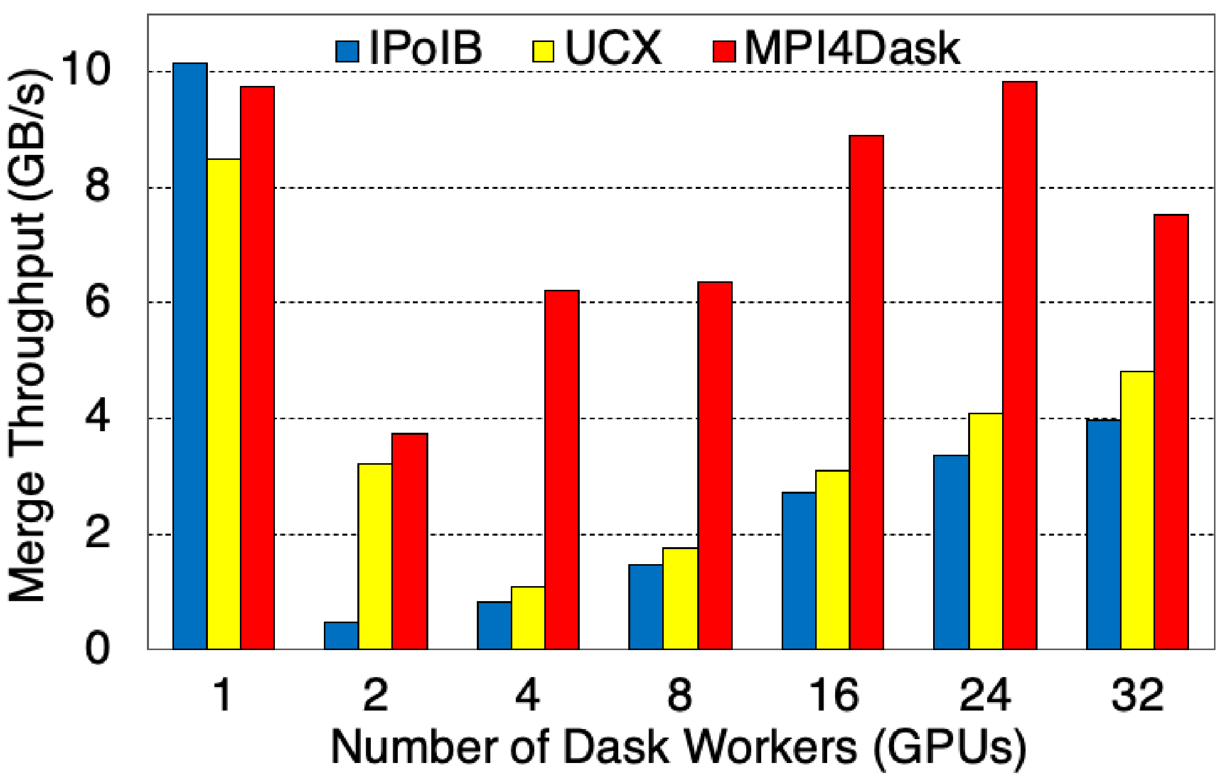
Platform: The Summit Supercomputer (#2 on Top500.org) – 6 NVIDIA Volta GPUs per node connected with NVLink, CUDA 10.1

Accelerating cuDF Merge – Longhorn (TACC Frontera GPU Subsystem)

2.91x better on average



2.90x better on average

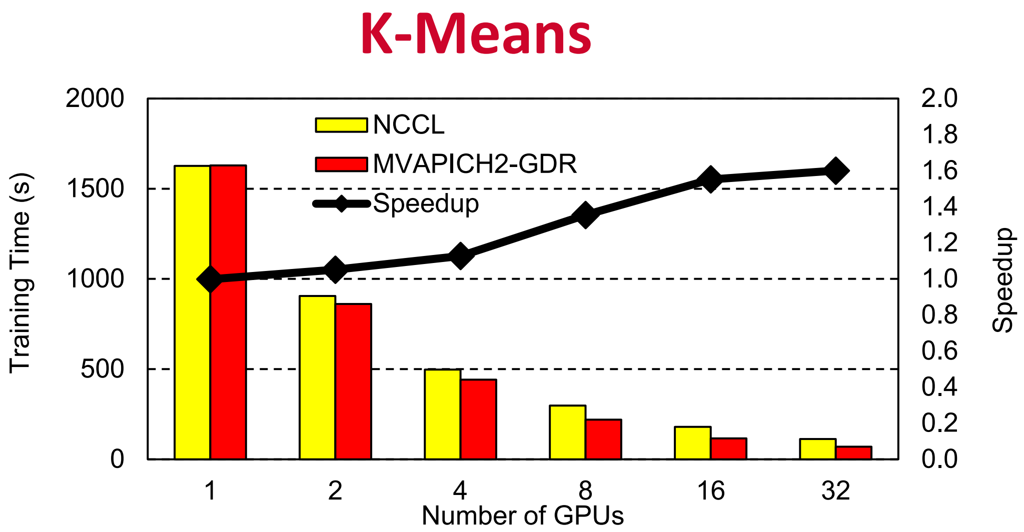


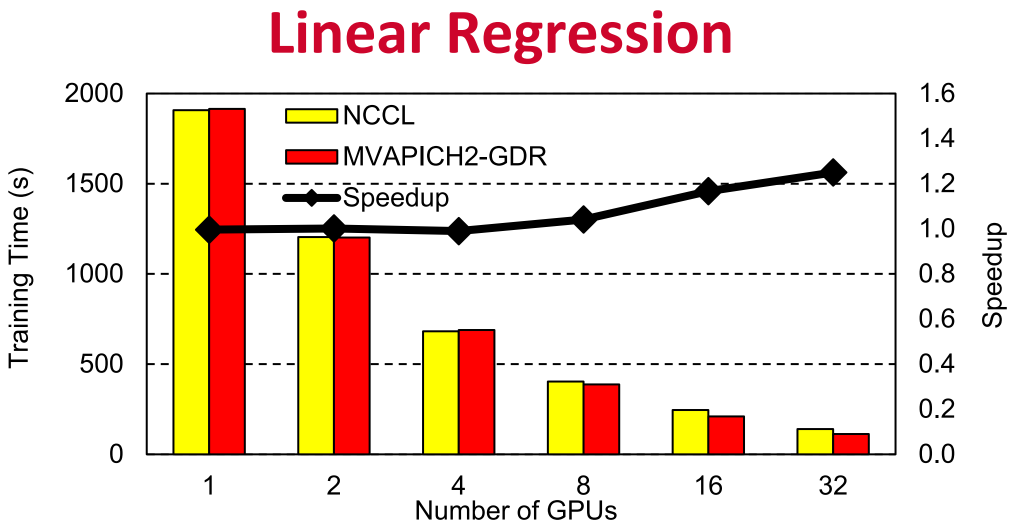
A. Shafi , J. Hashmi , H. Subramoni , and D. K. Panda, Efficient MPI-based Communication for GPU-Accelerated Dask Applications, https://arxiv.org/abs/2101.08878

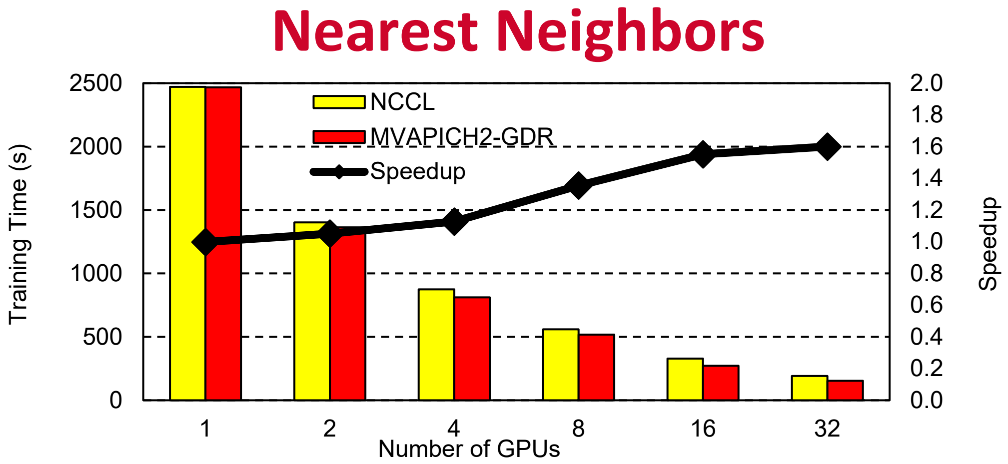
MPI4Dask 0.1 release

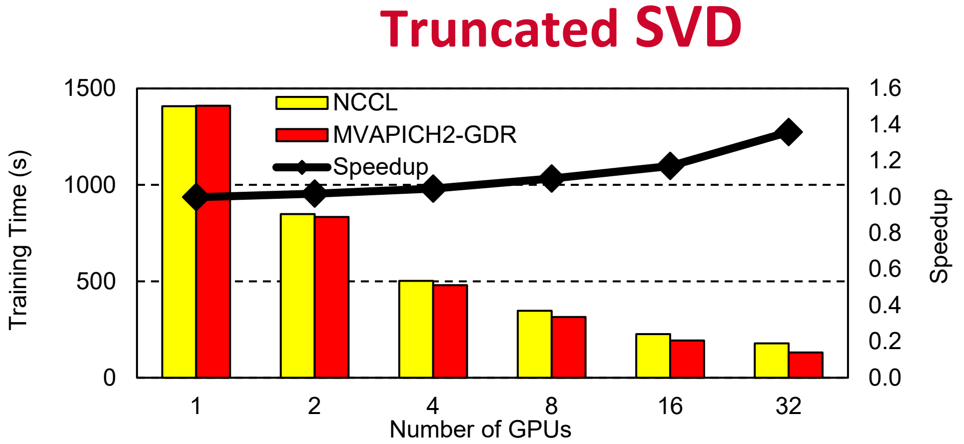
(http://hibd.cse.ohio-state.edu)

Accelerating cuML with MVAPICH2-GDR on Longhorn







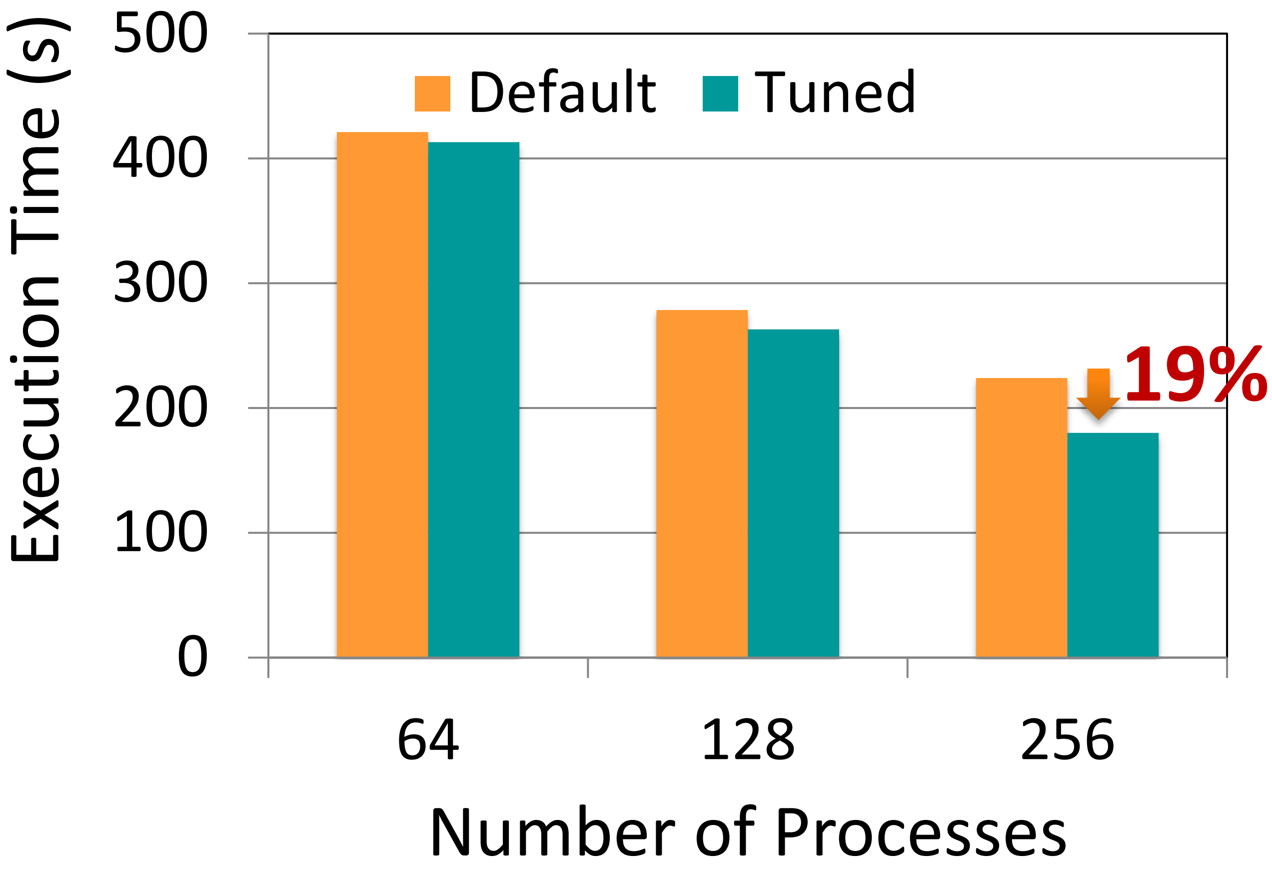


M. Ghazimirsaeed , Q. Anthony , A. Shafi , H. Subramoni , and D. K. Panda, Accelerating GPU-based Machine Learning in Python using MPI Library: A Case Study with MVAPICH2-GDR, MLHPC Workshop, Nov 2020

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/
- Initial list of applications
  - Amber
  - HoomDBlue
  - HPCG
  - Lulesh
  - MILC
  - Neuron
  - SMG2000
  - Cloverleaf
  - SPEC (LAMMPS, POP2, TERA_TF, WRF2)
- Soliciting additional contributions, send your results to mvapich-help at cse.ohio-state.edu.
- We will link these results with credits to you.

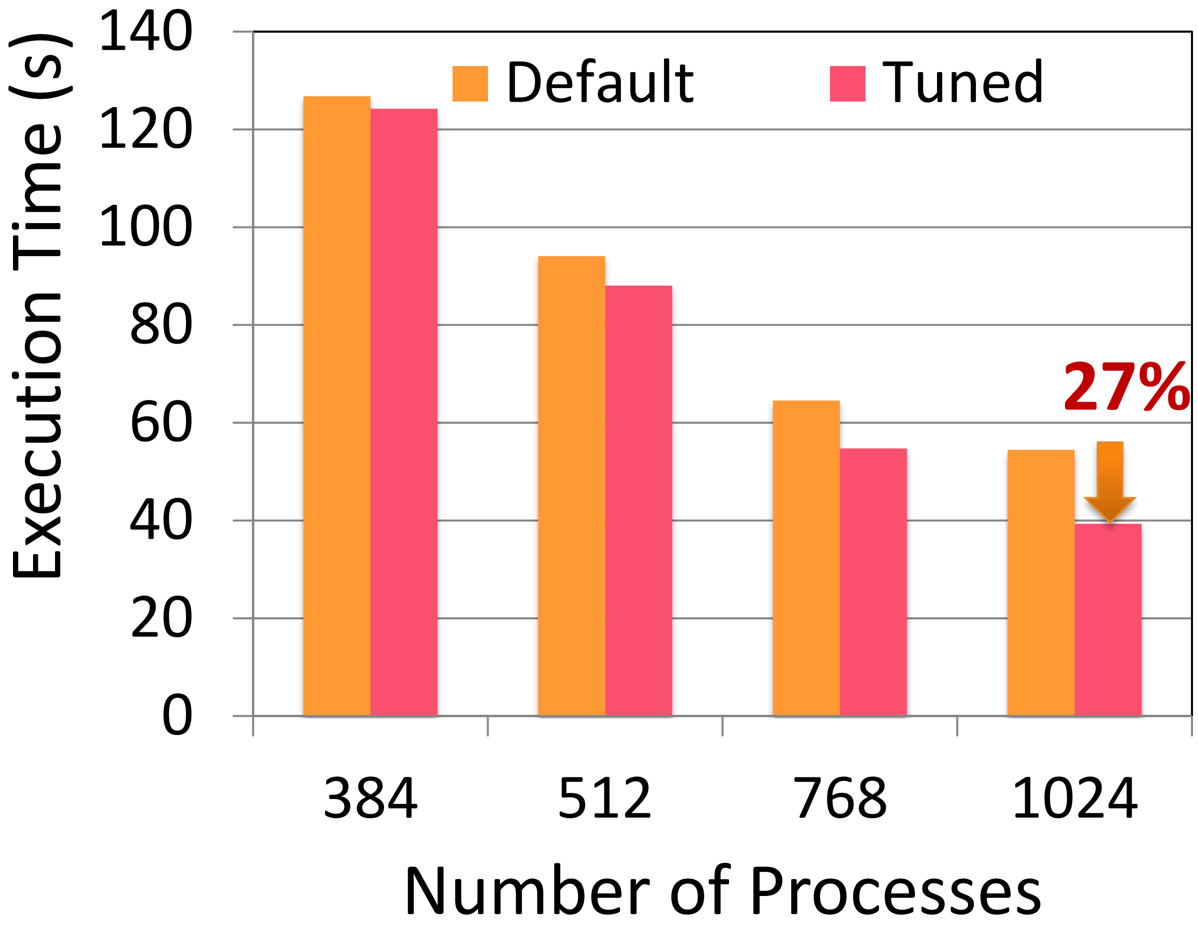
Amber: Impact of Tuning Eager Threshold



Data Submitted by: Dong Ju Choi @ UCSD

- Tuning the Eager threshold has a significant impact on application performance by avoiding the synchronization of rendezvous protocol and thus yielding better communication computation overlap
- 19% improvement in overall execution time at 256 processes
- Library Version: MVAPICH2 2.2
- MVAPICH Flags used
  - MV2_IBA_EAGER_THRESHOLD=131072
  - MV2_VBUF_TOTAL_SIZE=131072
- Input files used
  - Small: MDIN
  - Large: PMTOP

Neuron: Impact of Tuning Transport Protocol



Data Submitted by Mahidhar Tatineni @ SDSC

- UD-based transport protocol selection benefits the SMG2000 application
- 15% and 27% improvement is seen for 768 and 1,024 processes respectively
- Library Version: MVAPICH2 2.2
- MVAPICH Flags used
  - MV2_USE_ONLY_UD=1
- Input File
  - YuEtAl2012
- System Details
  - Comet@SDSC
  - Haswell nodes with dual 12-cores socket per node and Mellanox FDR (56 Gbps) network.

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*
  - Bluefield based offload*
- Enhanced communication schemes for upcoming architectures
  - Intel Optane*
  - BlueField*
  - 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

Funding Acknowledgments

Funding Support by

Equipment Support by

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. S. Kumar (M.S.)
- B. Ramesh (Ph.D.)
- K. K. Suresh (Ph.D.)
- N. Sarkauskas (Ph.D.)
- S. Srivastava (M.S.)
- A. H. Tu (Ph.D.)
- S. Xu (Ph.D.)
- Q. Zhou (Ph.D.)

Current Research Scientists

- A. Shafi
- H. Subramoni

Current Senior Research Associate

- J. Hashmi

Current Software Engineers

- A. Reifsteck
- 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. Chakraborthy (Ph.D.)
- N. Dandapanthula (M.S.)
- V. Dhanraj (M.S.)
- C.-H. Chu (Ph.D.)
- T. Gangadharappa (M.S.)
- K. Gopalakrishnan (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. Santhanaraman (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. Ghazimeersaeed
- H.-W. Jin
- J. Lin
- M. Luo
- E. Mancini
- K. Manian
- S. Marcarelli
- A. Ruhela
- J. Vienne
- H. Wang

Thank You!

panda@cse.ohio-state.edu

Network-Based Computing Laboratory

http://nowlab.cse.ohio-state.edu/

The High-Performance MPI/PGAS Project

http://mvapich.cse.ohio-state.edu/

The High-Performance Big Data Project

http://hibd.cse.ohio-state.edu/

The High-Performance Deep Learning Project

http://hidl.cse.ohio-state.edu/