Optimizing & Tuning Techniques for Running MVAPICH2 over IB

Talk at 2nd Annual IBUG (InfiniBand User's Group) Workshop (2014)

by

Hari Subramoni
The Ohio State University
E-mail: subramon@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~subramon

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

Presentation Outline

- **Overview of MVAPICH2 and MVAPICH2-X**
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- Future Plans and Concluding Remarks

Drivers of Modern HPC Cluster Architectures

Multi-core Processors

High Performance Interconnects - InfiniBand
<1usec latency, >100Gbps Bandwidth

Accelerators / Coprocessors
high compute density, high performance/watt
>1 TFlop DP on a chip

- Multi-core processors are ubiquitous
- InfiniBand is very popular in HPC clusters
- Accelerators/Coprocessors are becoming common in high-end systems
- Pushing the envelope for Exascale computing

Tianhe – 2 (1)

Titan (2)

Stampede (6)

Tianhe – 1A (10)

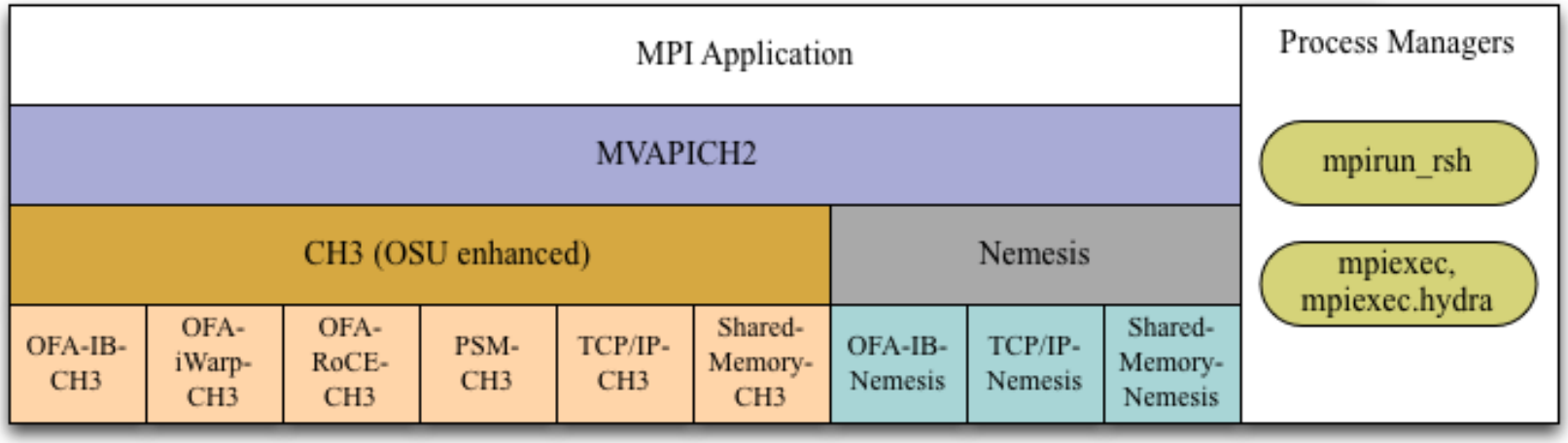
MVAPICH2/MVAPICH2-X Software

- High Performance open-source MPI Library for InfiniBand, 10Gig/iWARP, and RDMA over Converged Enhanced Ethernet (RoCE)
  - MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.0), Available since 2002
  - MVAPICH2-X (MPI + PGAS), Available since 2012
  - Support for GPGPUs and MIC
  - **Used by more than 2,150 organizations in 72 countries**
  - **More than 207,000 downloads from OSU site directly**
  - Empowering many TOP500 clusters
    - 7th ranked 462,462-core cluster (Stampede) at TACC
    - 11th ranked 74,358-core cluster (Tsubame 2.5) at Tokyo Institute of Technology
    - 16th ranked 96,192-core cluster (Pleiades) at NASA
    - 75th ranked 16,896-core cluster (Keenland) at GaTech and many others . . .
  - Available with software stacks of many IB, HSE, and server vendors including Linux Distros (RedHat and SuSE)
  - http://mvapich.cse.ohio-state.edu
- Partner in the U.S. NSF-TACC Stampede System

Major Features in MVAPICH2/MVAPICH2X for Multi-Petaflop and Exaflop Systems

- Scalability for million to billion processors
  - Support for highly-efficient inter-node and intra-node communication (both two-sided and one-sided)
  - Extremely minimum memory footprint
- Scalable Job Startup
- Support for Efficient Process Mapping and Multi-threading
- High-performance Inter-node / Intra-node Point-to-point Communication
- Support for Multiple IB Transport Protocols for Scalable Communication
- Support for Multi-rail Clusters and 3D Torus Networks
- QoS support for Communication and I/O
- Scalable Collective Communication
- Support for GPGPUs and Accelerators
- Hybrid Programming (MPI + OpenMP, MPI + UPC, MPI + OpenSHMEM, …)
- Enhanced Debugging System
- *and many more…*

MVAPICH2 Architecture (Latest Release 2.0rc1)



All Different PCI, PCI-Ex interfaces

Major Computing Platforms: IA-32, Ivybridge, Nehalem, Westmere, Sandybridge, Opteron, Magny, ..

Strong Procedure for Design, Development and Release

- Research is done for exploring new designs
- Designs are first presented through 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
- Provides detailed User guides and FAQs
  - http://mvapich.cse.ohio-state.edu

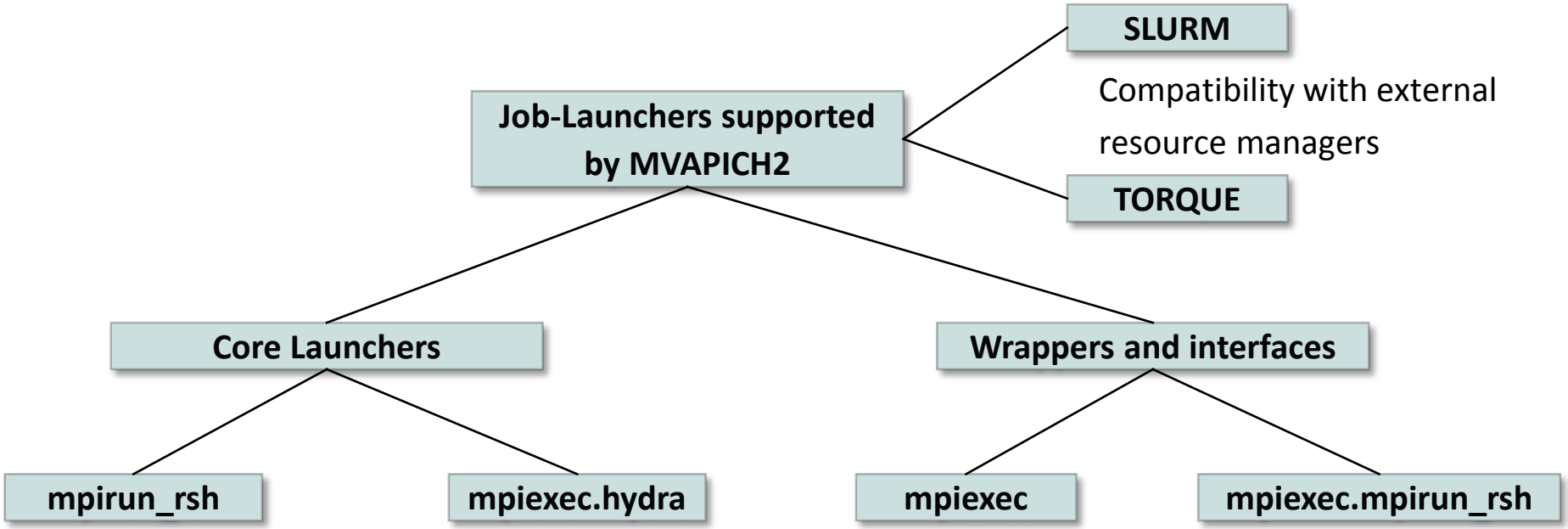
MVAPICH2 2.0RC1 and MVAPICH2-X 2.0RC1

- Released on 03/24/14
- Major Features and Enhancements
  - Based on MPICH-3.1
  - Improved performance for MPI_Put and MPI_Get operations in CH3 channel
  - Enabled MPI-3 RMA support in PSM channel
  - Enabled multi-rail support for UD-Hybrid channel
  - Optimized architecture based tuning for blocking and non-blocking collectives
  - Optimized Bcast and Reduce collectives designs
  - Improved hierarchical job startup time
  - Optimization for sub-array data-type processing for GPU-to-GPU communication
  - Updated hwloc to version 1.8
  - Enhanced build system to avoid separate builds for different networks/interfaces
    - Updated compiler wrappers (example: mpicc) to avoid adding dependencies on network and other libraries
- MVAPICH2-X 2.0RC1 supports hybrid MPI + PGAS (UPC and OpenSHMEM) programming models
  - Based on MVAPICH2 2.0RC1 including MPI-3 features; Compliant with UPC 2.18.0 and OpenSHMEM v1.0f
  - Improved intra-node performance using Shared memory and Cross Memory Attach (CMA)
  - Optimized UPC collectives

Presentation Outline

- Overview of MVAPICH2 and MVAPICH2-X
- **Optimizing and Tuning Job Startup**
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- Future Plans and Concluding Remarks

Job-Launchers supported by MVAPICH2



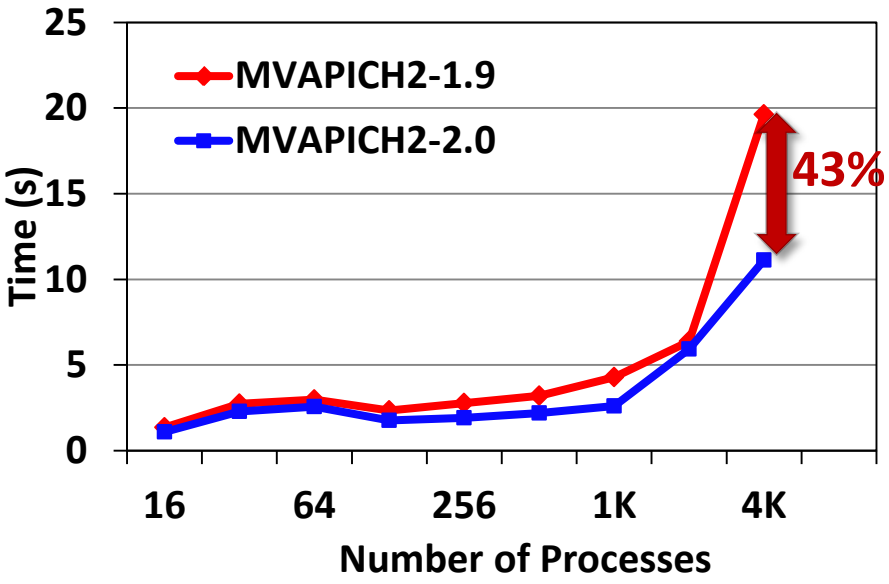
- Default and preferred launcher for all interfaces
- Supports scalable hierarchical job-startup mechanism

Developed at ANL by the MPICH team

To support interfaces prescribed by the MPI standard

Hydra-like interface to mpirun_rsh

Tuning Job-Launch with mpirun_rsh



- Job startup performance on Stampede
  - MV2_HOMOGENEOUS_CLUSTER=1
  - MV2_ON_DEMAND_UD_INFO_EXCHANGE=1
- 43% reduction in time for MPI hello world program at *4K cores*
- ***Continually being improved***

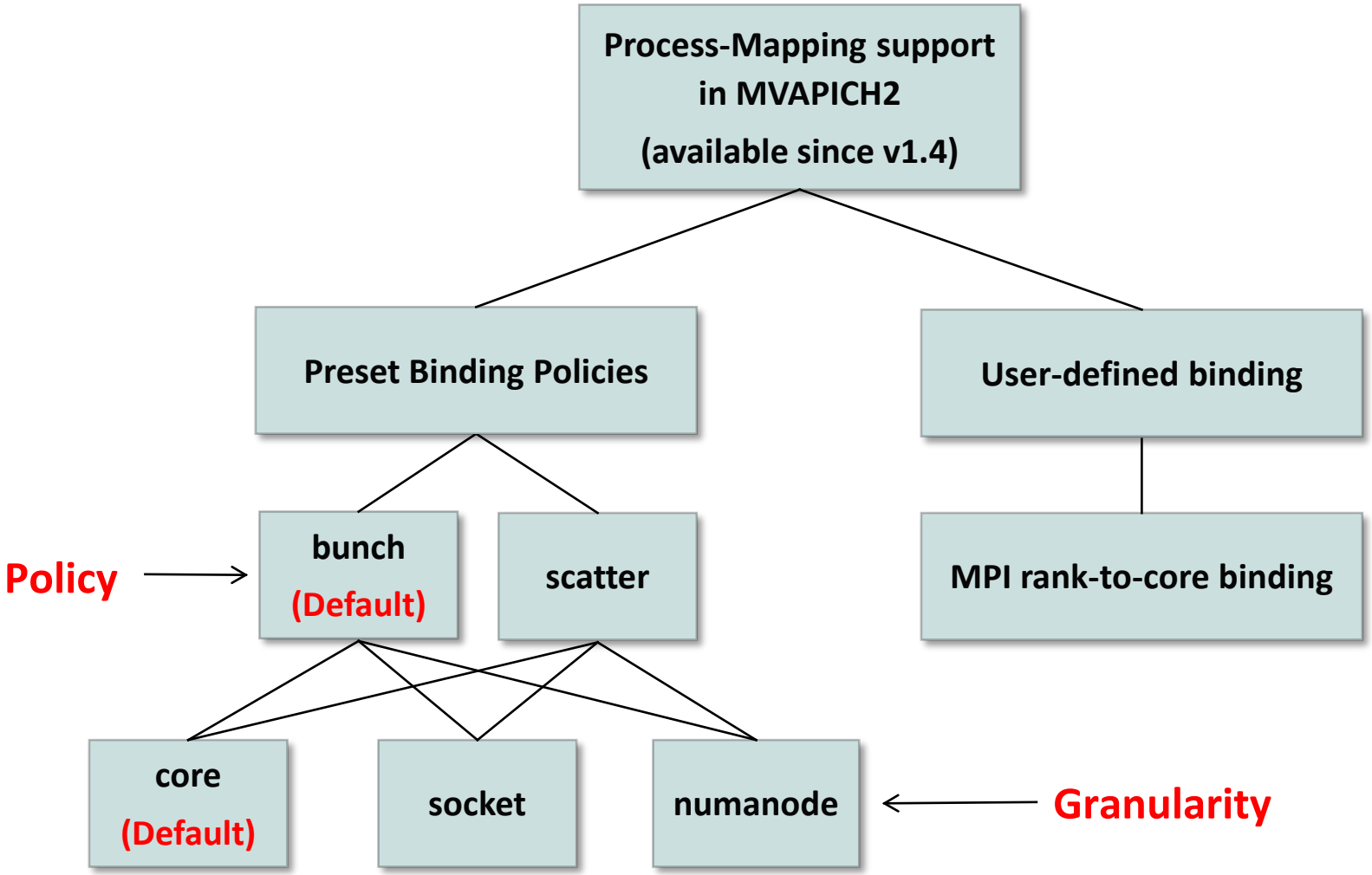
Parameter	Significance	Default
MV2_MT_DEGREE	• Degree of the hierarchical tree used by mpirun_rsh	32
MV2_FASTSSH_THRESHOLD	• #nodes beyond which hierarchical-ssh scheme is used	256
MV2_NPROCS_THRESHOLD	• #nodes beyond which file-based communication is used for hierarchical-ssh during start up	8192
MV2_HOMOGENEOUS_CLUSTER	• Optimizes startup for homogeneous clusters	Disabled
MV2_ON_DEMAND_UD_INFO_EXCHANGE	• Optimize start-up by exchanging UD connection info on-demand	Enabled

- **Refer to Job Launch Tuning section of MVAPICH2 user guide for more information**
- **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-950008.2**

Presentation Outline

- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- **Efficient Process Mapping Strategies**
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- Future Plans and Concluding Remarks

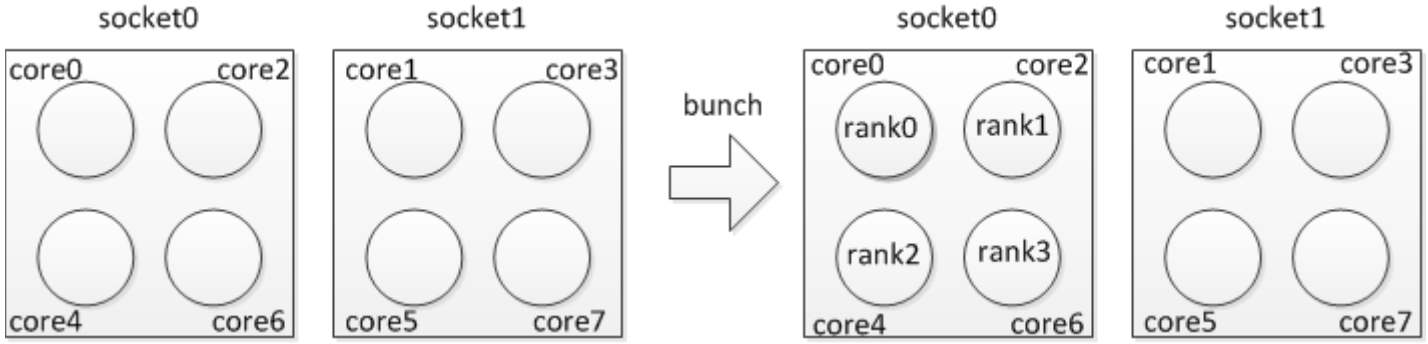
Process Mapping support in MVAPICH2



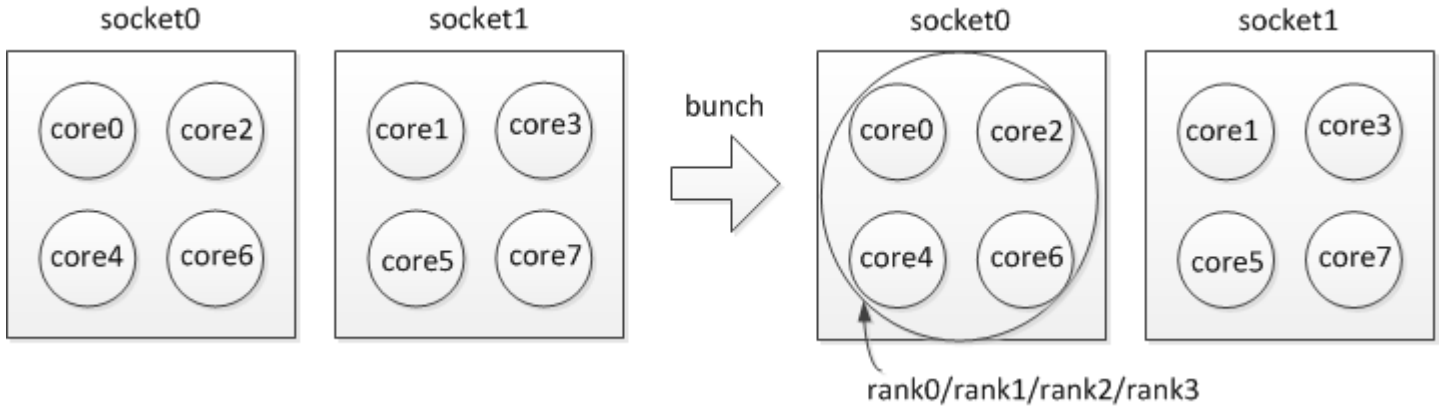
- MVAPICH2 detects processor architecture at job-launch

Preset Process-binding Policies – Bunch

- "Core" level "Bunch" mapping (Default)
  - MV2_CPU_BINDING_POLICY=bunch

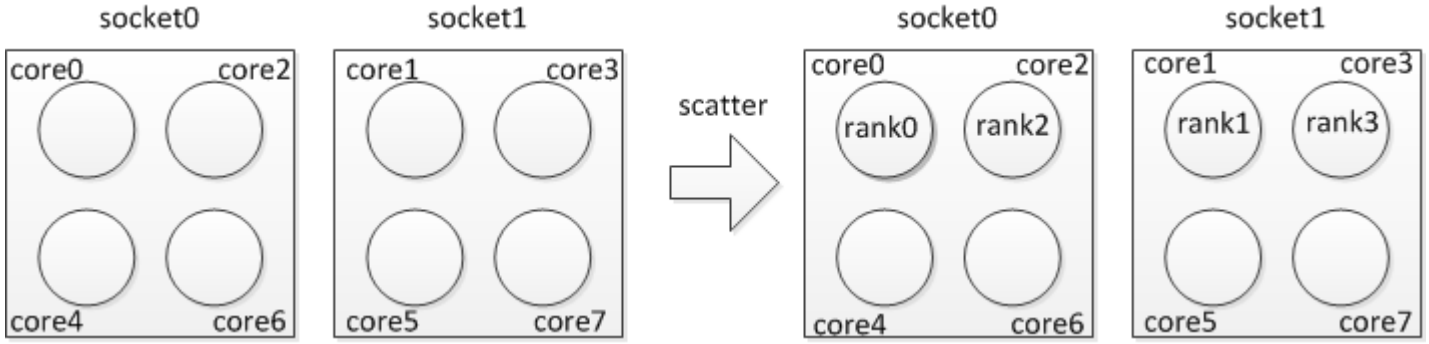


- "Socket/Numanode" level "Bunch" mapping
  - MV2_CPU_BINDING_LEVEL=socket MV2_CPU_BINDING_POLICY=bunch

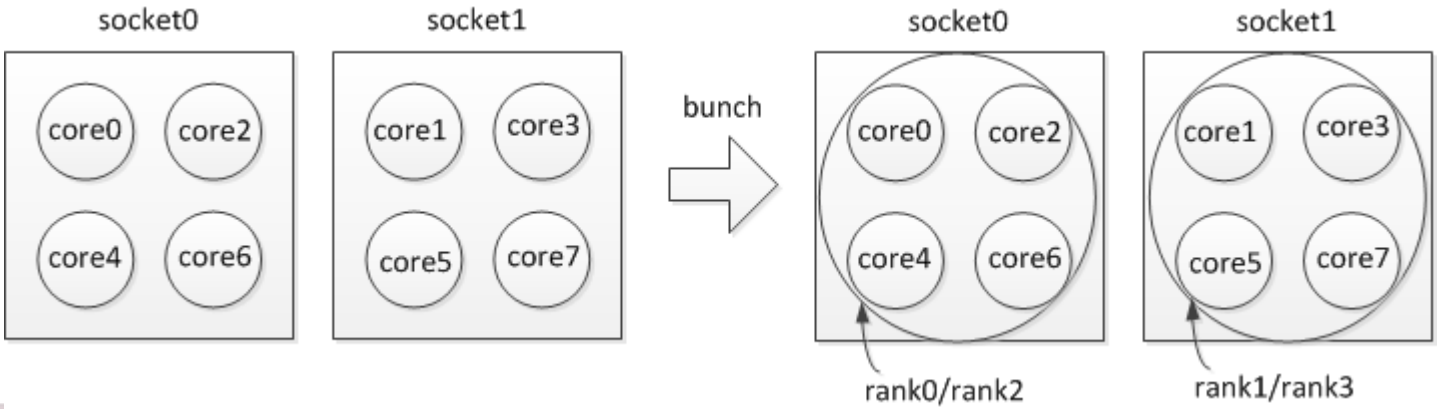


Preset Process-binding Policies – Scatter

- "Core" level "Scatter" mapping
  - MV2_CPU_BINDING_POLICY=scatter



- "Socket/Numanode" level "Scatter" mapping
  - MV2_CPU_BINDING_LEVEL=socket MV2_CPU_BINDING_POLICY=scatter



User-Defined Process Mapping

- User has complete-control over process-mapping
- To run 4 processes on cores 0, 1, 4, 5:
  - $ mpirun_rsh -np 4 -hostfile hosts MV2_CPU_MAPPING=0:1:4:5 ./a.out
- Use ',' or '-' to bind to a set of cores:
  - $mpirun_rsh -np 64 -hostfile hosts MV2_CPU_MAPPING=0,2-4:1:5:6 ./a.out
- Is process binding working as expected?
  - **MV2_SHOW_CPU_BINDING=1**
    - Display CPU binding information
    - Launcher independent
    - Example
      - MV2_SHOW_CPU_BINDING=1 MV2_CPU_BINDING_POLICY=scatter

        ----------CPU AFFINITY----------

        RANK:0 CPU_SET: 0

        RANK:1 CPU_SET: 8

- **Refer to Running with Efficient CPU (Core) Mapping section of MVAPICH2 user guide for more information**
- **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-530006.5**

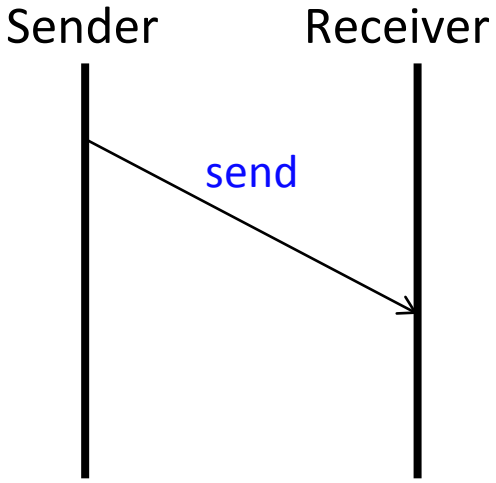
Presentation Outline

- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- **Point-to-Point Tuning and Optimizations**
  - **Inter-Node Tuning and Optimizations**
  - **Intra-Node Tuning and Optimizations**
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- Future Plans and Concluding Remarks

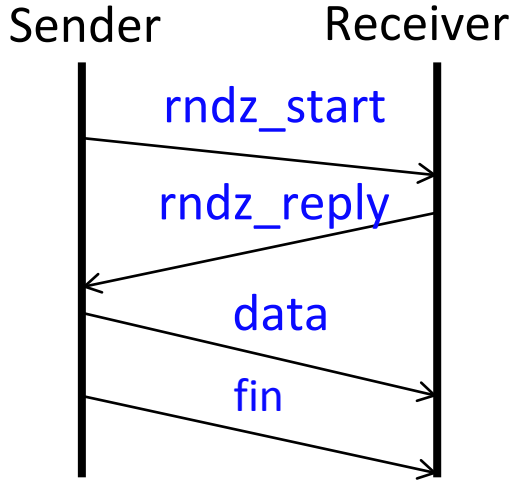
Inter-node Point-to-Point Communication

- EAGER (buffered, used for small messages)
  - RDMA Fast Path *(Memory Semantics)*
  - Send/Recv *(Channel Semantics)*

- RENDEZVOUS (un-buffered, used for large messages)
  - Reduces memory requirement by MPI library
  - Zero-Copy
  - No remote side involvement
  - **Protocols**
    - **RPUT** (RDMA Write)
    - **RGET** (RDMA Read)
    - **R3** (Send/Recv with Packetized Send)

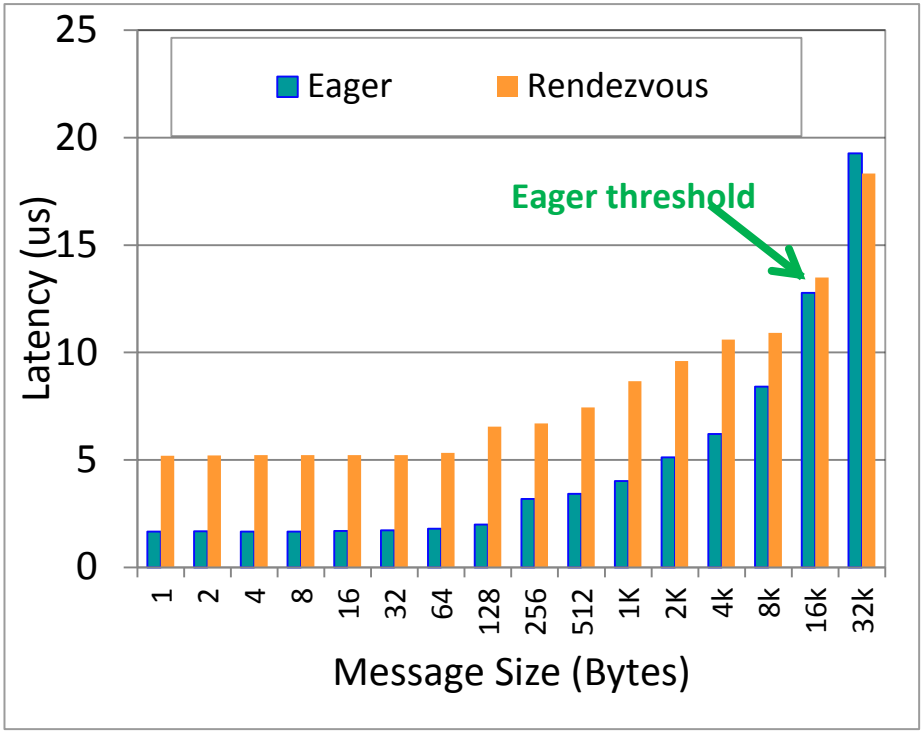


Eager Protocol

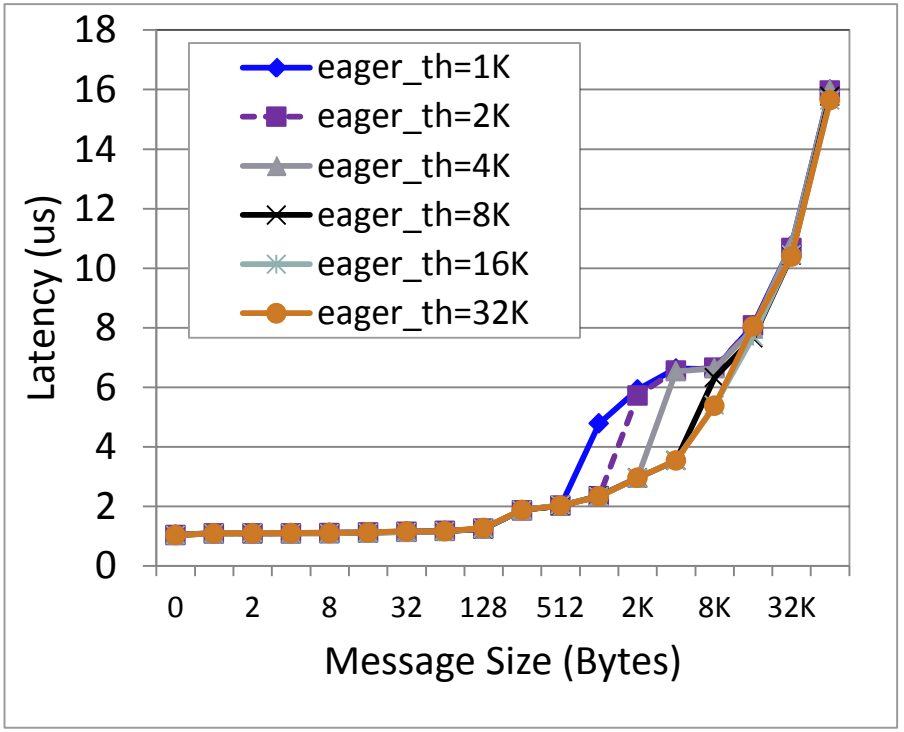


Rendezvous Protocol

Inter-node Tuning and Optimizations: Eager Thresholds



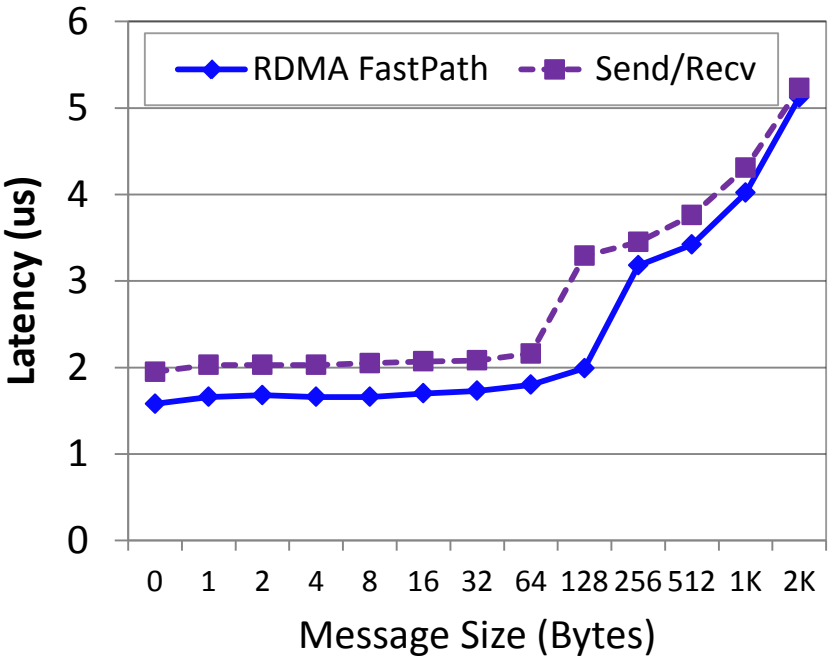
Eager vs Rendezvous



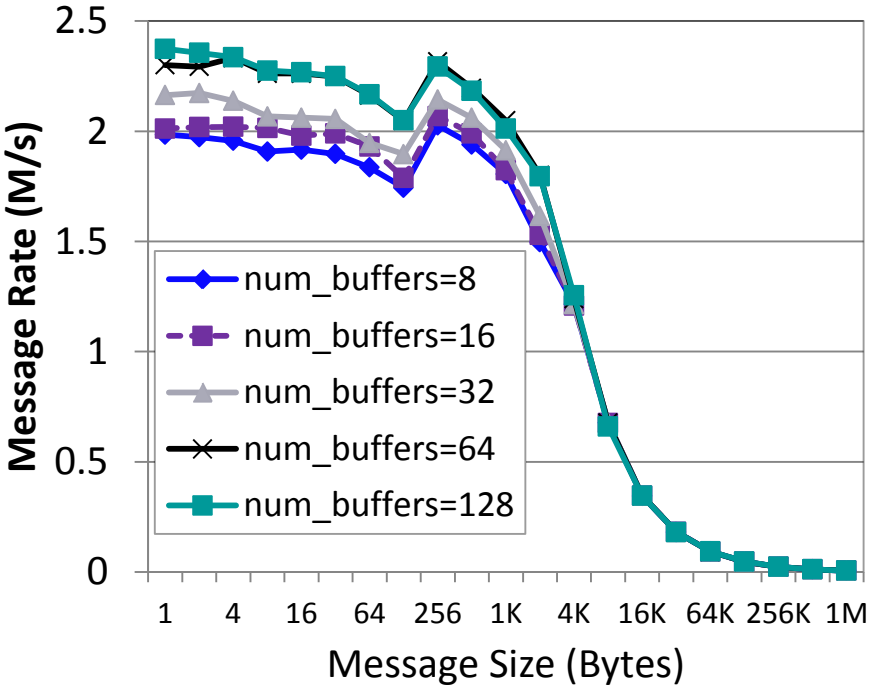
Impact of Eager Threshold

- Switching Eager to Rendezvous transfer
  - Default: Architecture dependent on common platforms, in order to achieve both best performance and memory footprint
- Threshold can be modified by users to get smooth performance across message sizes
  - mpirun_rsh –np 2 –f hostfile MV2_IBA_EAGER_THRESHOLD=32K a.out
  - Memory footprint can increase along with eager threshold

Inter-Node Tuning and Optimizations: RDMA Fast Path and RNDV Protocols



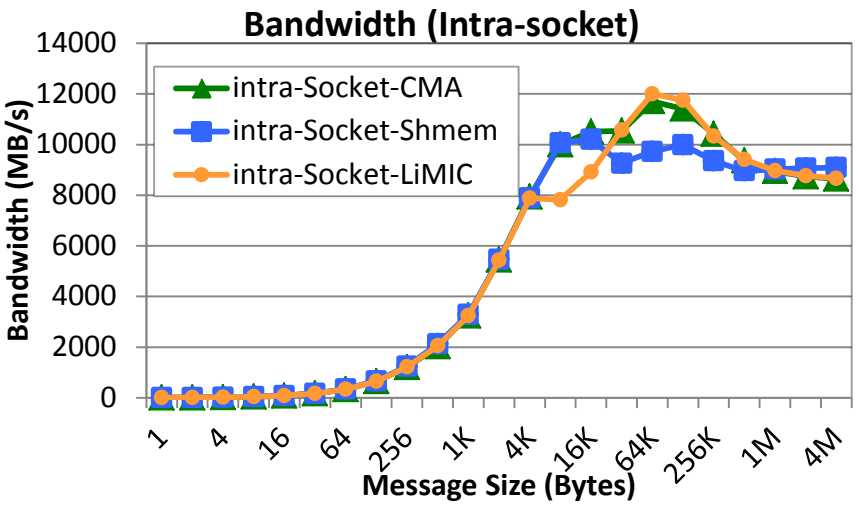
Eager: Send/Recv vs RDMA FP

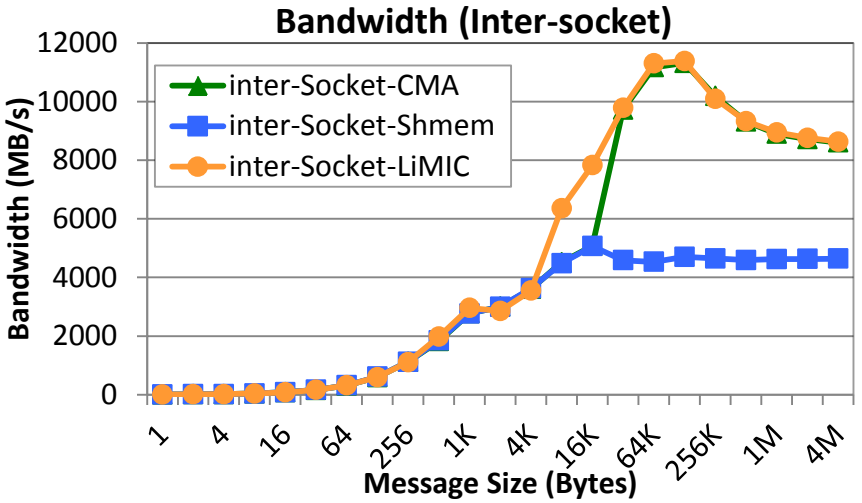


Impact of RDMA FP buffers

- **RDMA Fast Path has advantages for smaller message range (Enabled by Default)**
  - Disable: mpirun_rsh –np 2 –f hostfile MV2_USE_RDMA_FASTPATH=0 a.out
- Adjust the number of RDMA Fast Path buffers (benchmark window size = 64):
  - mpirun_rsh –np 2 –f hostfile MV2_NUM_RDMA_BUFFER=64 a.out
- Switch between Rendezvous protocols depending on applications:
  - mpirun_rsh –np 2 –f hostfile MV2_RNDV_PROTOCOL=RGET a.out (**Default: RPUT**)

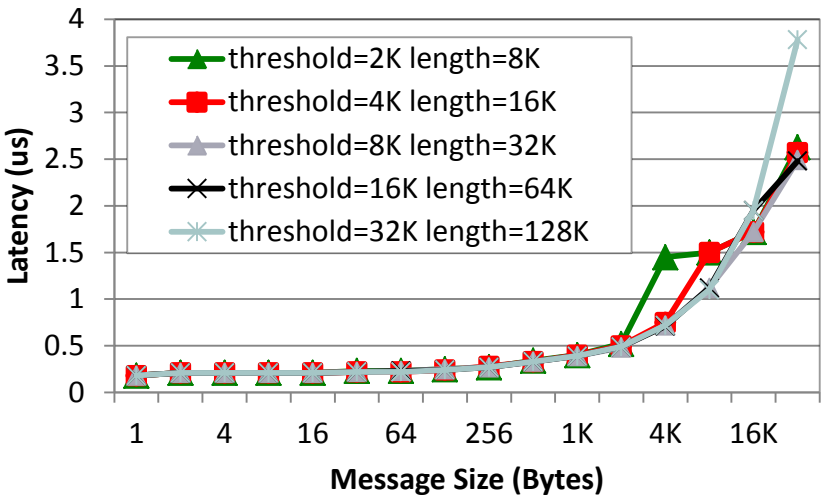
MVAPICH2 Two-Sided Intra-Node Tuning:
Shared memory and Kernel-based Zero-copy Support (LiMIC and CMA)

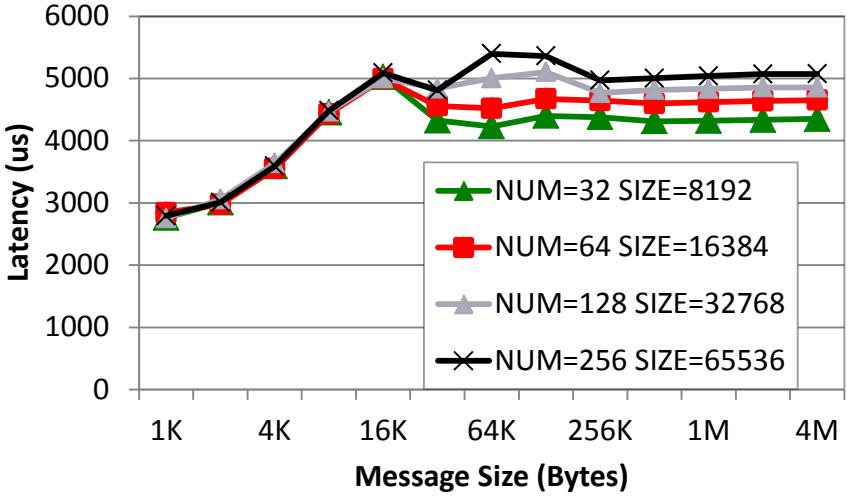




- LiMIC2:
  - Configure the library with '--with-limic2'
  - mpirun_rsh –np 2 –f hostfile a.out (To disable: MV2_SMP_USE_LIMIC2=0)
- CMA (Cross Memory Attach):
  - Configure the library with '--with-cma'
  - mpirun_rsh –np 2 –f hostfile a.out (To disable: MV2_SMP_USE_CMA=0)
- LiMIC2 is chosen by default when library is built with both LiMIC2 and CMA
- Shared-memory based design used if neither LiMIC2 and CMA is used
  - **Refer to Running with LiMIC2 section of MVAPICH2 user guide for more information**
  - **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-570006.6**

MVAPICH2 Two-Sided Intra-Node Tuning:
Shared-Memory based Runtime Parameters



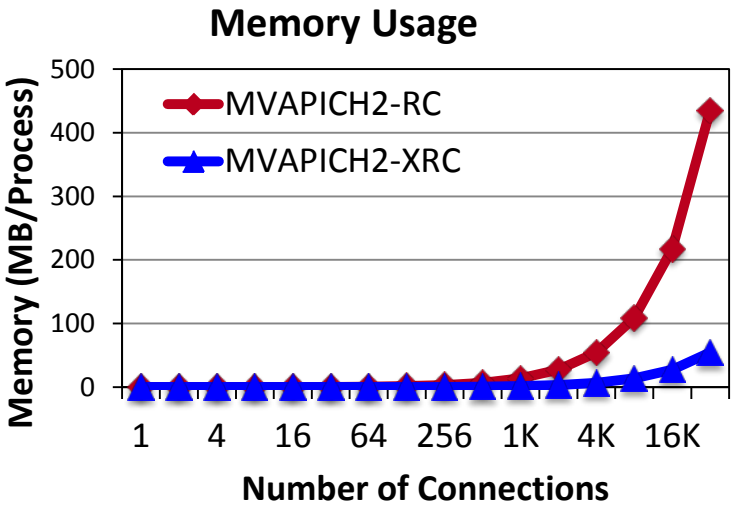


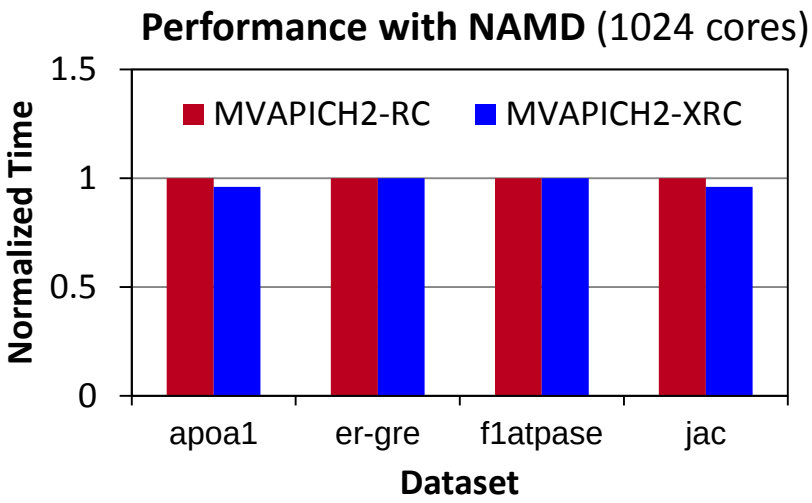
- Adjust eager threshold and eager buffer size:
  - mpirun_rsh –np 2 –f hostfile MV2_SMP_EAGERSIZE=16K MV2_SMPI_LENGTH_QUEUE=64 a.out
  - Note: Will affect the performance of small messages and memory footprint
- Adjust number of buffers /buffer size for shared-memory based Rendezvous protocol:
  - mpirun_rsh –np 2 –f hostfile MV2_SMP_SEND_BUFFER=32 MV2_SMP_SEND_BUFF_SIZE=8192 a.out
  - Note: Will affect the performance of large messages and memory footprint

Presentation Outline

- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- **InfiniBand Transport Protocol Based Tuning**
  - **eXtended Reliable Connection (XRC)**
  - **Using UD Transport**
  - **Hybrid (UD/RC/XRC) Transport in MVAPICH2**
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- Future Plans and Concluding Remarks

Using eXtended Reliable Connection (XRC) in MVAPICH2





- Memory usage for 32K processes with 8-cores per node can be 54 MB/process (for connections)
- NAMD performance improves when there is frequent communication to many peers
- Enabled by setting **MV2_USE_XRC** to **1** (Default: Disabled)
- Requires OFED version > 1.3
  - Unsupported in earlier versions (< 1.3), OFED-3.x and MLNX_OFED-2.0
  - MVAPICH2 build process will automatically disable XRC if unsupported by OFED
- Automatically enables SRQ and ON-DEMAND connection establishment

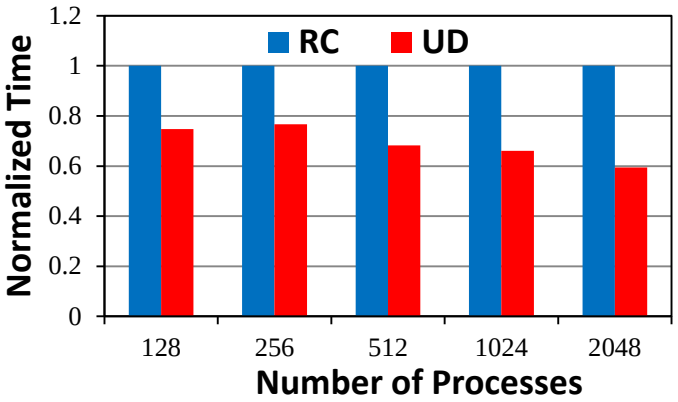
- **Refer to eXtended Reliable Connection (XRC) section of MVAPICH2 user guide for more information**
- **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-990008.6**

Using UD Transport with MVAPICH2

Memory Footprint of MVAPICH2

	RC (MVAPICH2 2.0a)				UD (MVAPICH2 2.0a)		
Number of Processes	Conn.	Buffers	Struct	Total	Buffers	Struct	Total
512	22.9	24	0.3	47.2	24	0.2	24.2
1024	29.5	24	0.6	54.1	24	0.4	24.4
2048	42.4	24	1.2	67.6	24	0.9	24.9

Performance with SMG2000

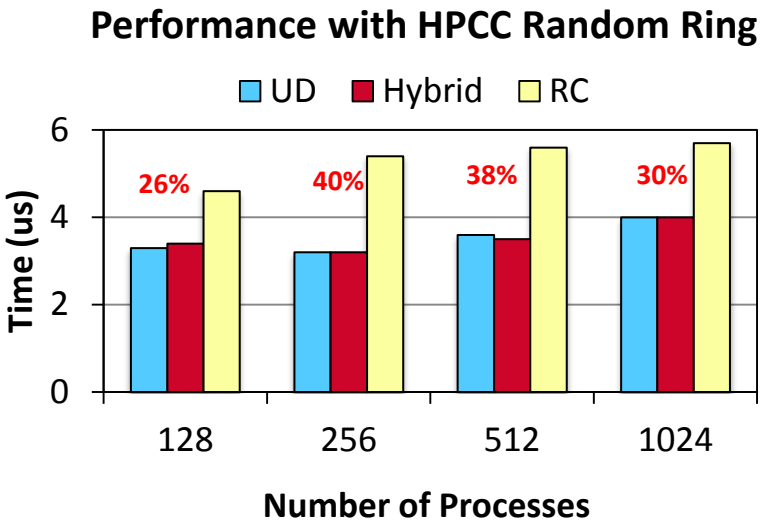


- Can use UD transport by configuring MVAPICH2 with the **–enable-hybrid**
  - Reduces QP cache trashing and memory footprint at large scale

Parameter	Significance	Default	Notes
MV2_USE_ONLY_UD	• Enable only UD transport in hybrid configuration mode	Disabled	• RC/XRC not used
MV2_USE_UD_ZCOPY	• Enables zero-copy transfers for large messages on UD	Enabled	• Always Enable when UD enabled
MV2_UD_RETRY_TIMEOUT	• Time (in usec) after which an unacknowledged message will be retried	500000	• Increase appropriately on large / congested systems
MV2_UD_RETRY_COUNT	• Number of retries before job is aborted	1000	• Increase appropriately on large / congested systems

- **Refer to Running with scalable UD transport section of MVAPICH2 user guide for more information**
- **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-610006.10**

Hybrid (UD/RC/XRC) Mode 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

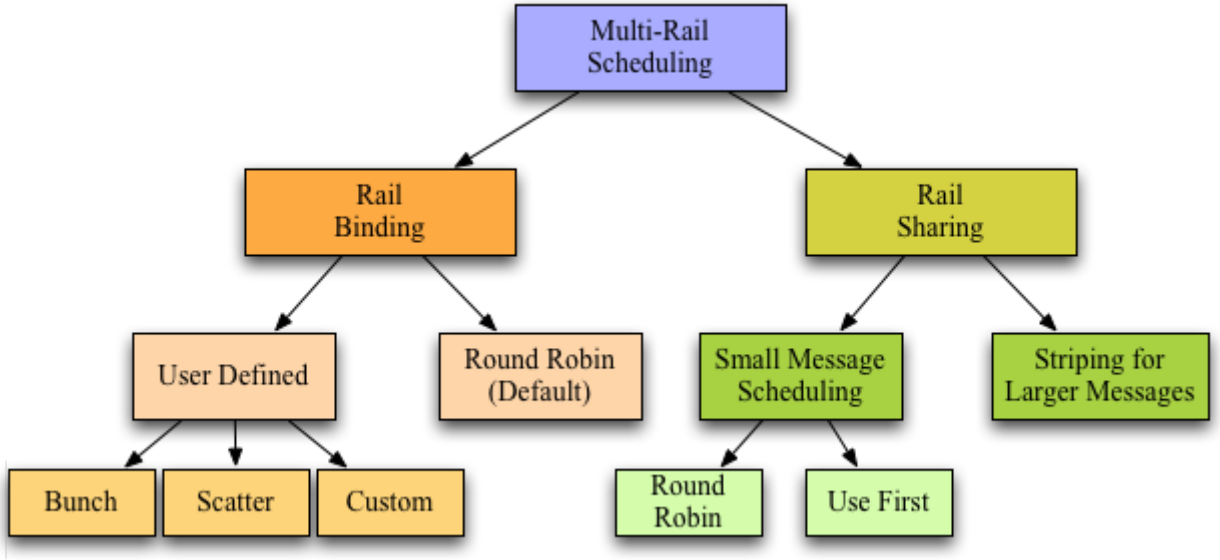
Parameter	Significance	Default	Notes
MV2_USE_UD_HYBRID	• Enable / Disable use of UD transport in Hybrid mode	Enabled	• Always Enable
MV2_HYBRID_ENABLE_THRESHOLD_SIZE	• Job size in number of processes beyond which hybrid mode will be enabled	1024	• Uses RC/XRC connection until job size < threshold
MV2_HYBRID_MAX_RC_CONN	• 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**
- **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-620006.11**

Presentation Outline

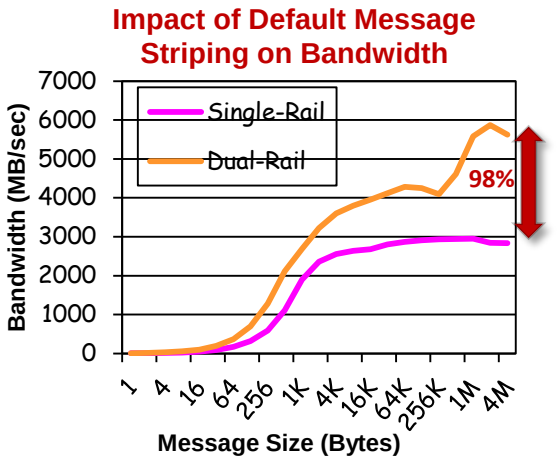
- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- **Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support**
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- Future Plans and Concluding Remarks

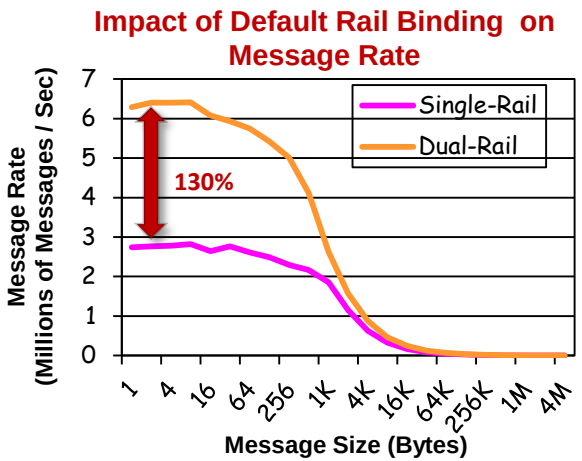
MVAPICH2 Multi-Rail Design

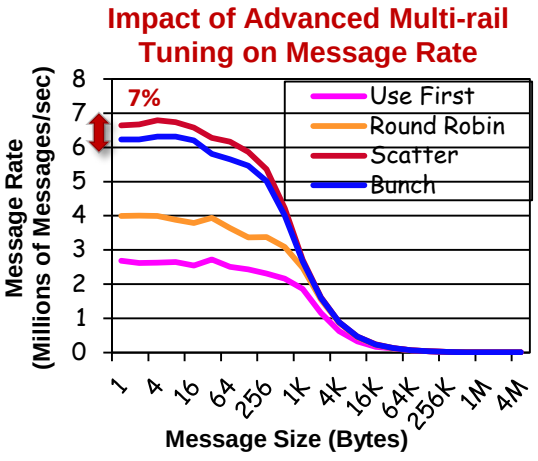


- What is a rail?
  - **HCA, Port, Queue Pair**
- Automatically detects and uses all active HCAs in a system
  - Automatically handles heterogeneity
- Supports multiple rail usage policies
  - Rail Sharing – Processes share all available rails
  - Rail Binding – Specific processes are bound to specific rails

Performance Tuning on Multi-Rail Clusters







Two 24-core Magny Cours nodes with two Mellanox ConnectX QDR adapters

Six pairs with OSU Multi-Pair bandwidth and messaging rate benchmark

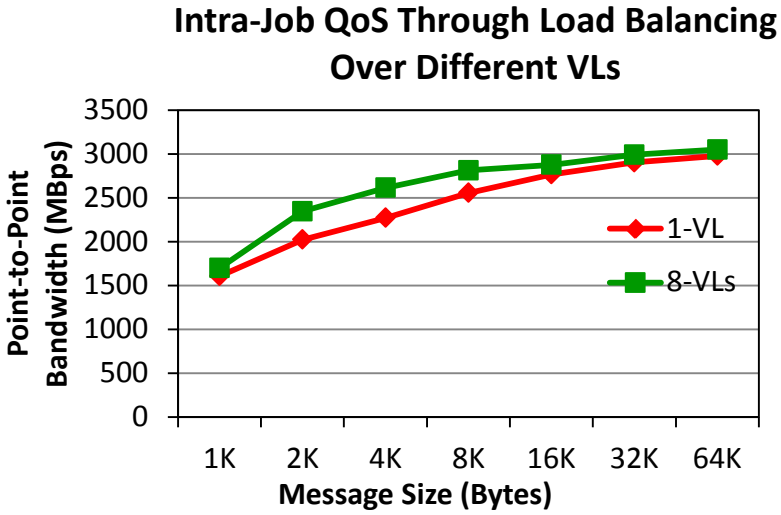
Parameter	Significance	Default	Notes
MV2_IBA_HCA	• Manually set the HCA to be used	Unset	• To get names of HCA ibstat | grep "^CA"
MV2_DEFAULT_PORT	• Select the port to use on a active multi port HCA	0	• Set to use different port
MV2_RAIL_SHARING_LARGE_MSG _THRESHOLD	• Threshold beyond which striping will take place	16 Kbyte	
MV2_RAIL_SHARING_POLICY	• Choose multi-rail rail sharing / binding policy • For Rail Sharing set to USE_FIRST or ROUND_ROBIN • Set to FIXED_MAPPING for advanced rail binding options	Rail Binding in Round Robin mode	• Advanced tuning can result in better performance
MV2_PROCESS_TO_RAIL_MAPPING	• Determines how HCAs will be mapped to the rails	BUNCH	• Options: SCATTER and custom list

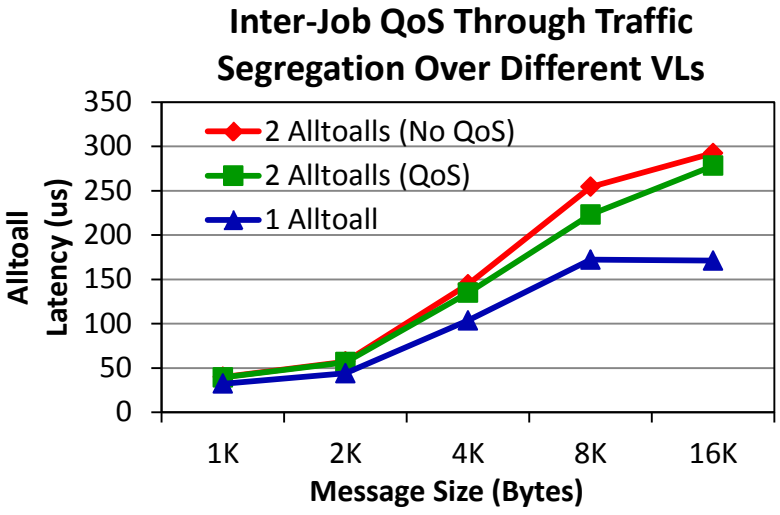
- **Refer to Enhanced design for Multiple-Rail section of MVAPICH2 user guide for more information**
- **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-640006.13**

Support for 3D Torus Networks in MVAPICH2/MVAPICH2-X

- Deadlocks possible with common routing algorithms in 3D Torus InfiniBand networks
  - Need special routing algorithm for OpenSM
- Users need to interact with OpenSM
  - Use appropriate SL to prevent deadlock
- MVAPICH2 supports 3D Torus Topology
  - Queries OpenSM at runtime to obtain appropriate SL
- Usage
  - Enabled at configure time
    - --enable-3dtorus-support
  - MV2_NUM_SA_QUERY_RETRIES
    - Control number of retries if PathRecord query fails

Exploiting QoS Support in MVAPICH2





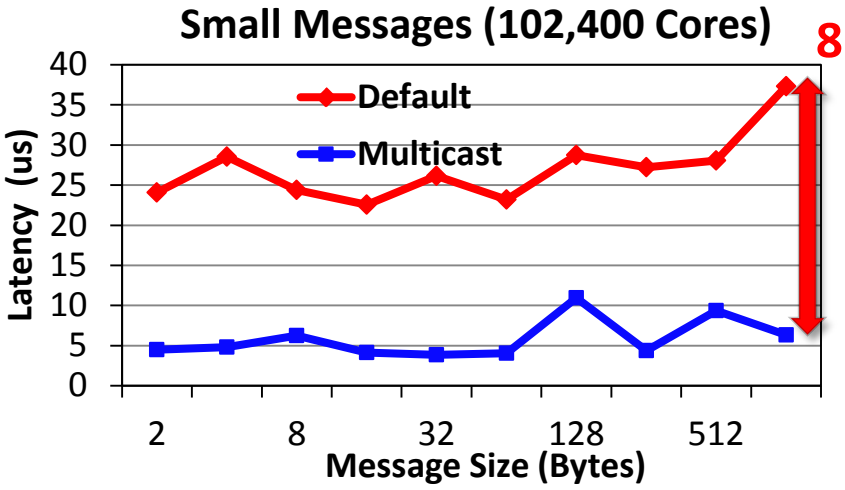
- IB is capable of providing network level differentiated service – QoS
- Uses Service Levels (SL) and Virtual Lanes (VL) to classify traffic
- Enabled at configure time using CFLAG ENABLE_QOS_SUPPORT
- Check with System administrator before enabling
  - Can affect performance of other jobs in system

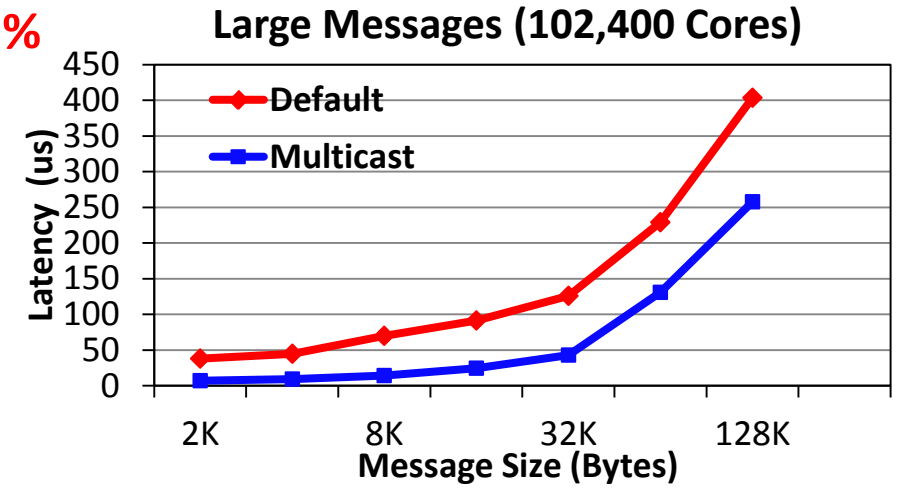
Parameter	Significance	Default	Notes
MV2_USE_QOS	• Enable / Disable use QoS	Disabled	• Check with System administrator
MV2_NUM_SLS	• Number of Service Levels user requested	8	• Use to see benefits of Intra-Job QoS
MV2_DEFAULT_SERVICE_LEVEL	• Indicates the default Service Level to be used by job	0	• Set to different values for different jobs to enable Inter-Job QoS

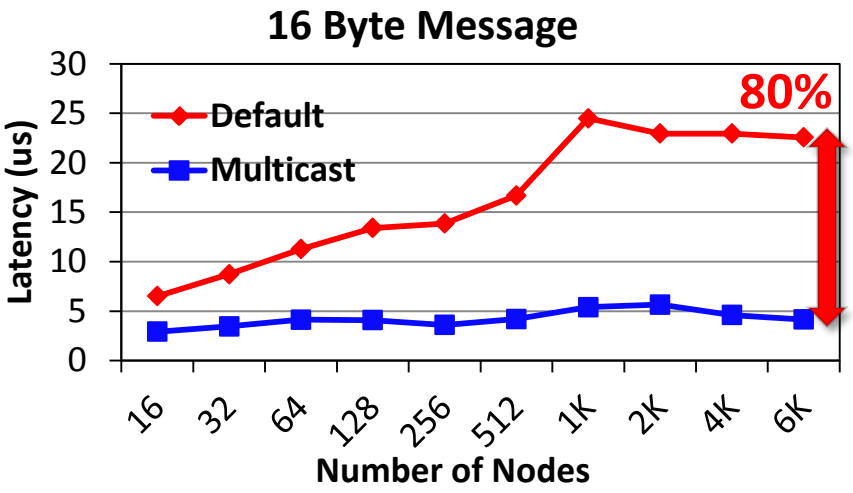
Presentation Outline

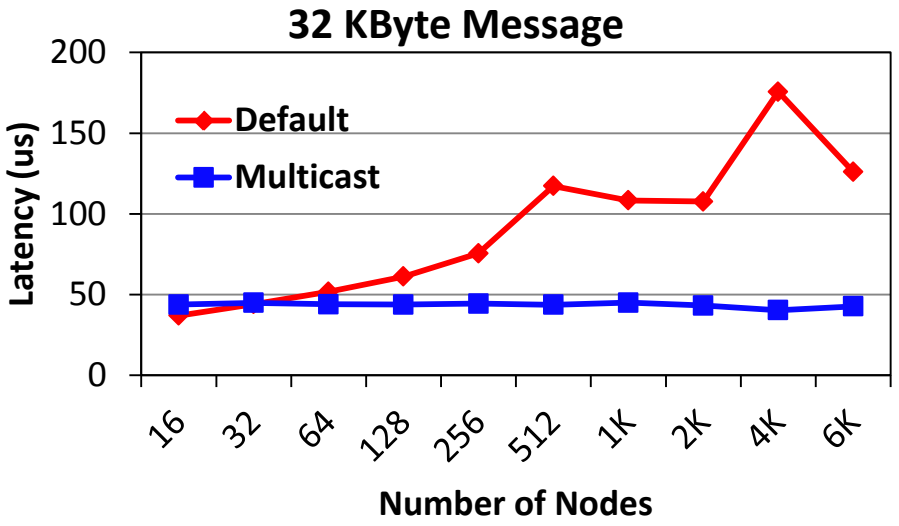
- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- **Collective Optimizations using Hardware-based Multicast**
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- Future Plans and Concluding Remarks

Hardware Multicast-aware MPI_Bcast on TACC Stampede



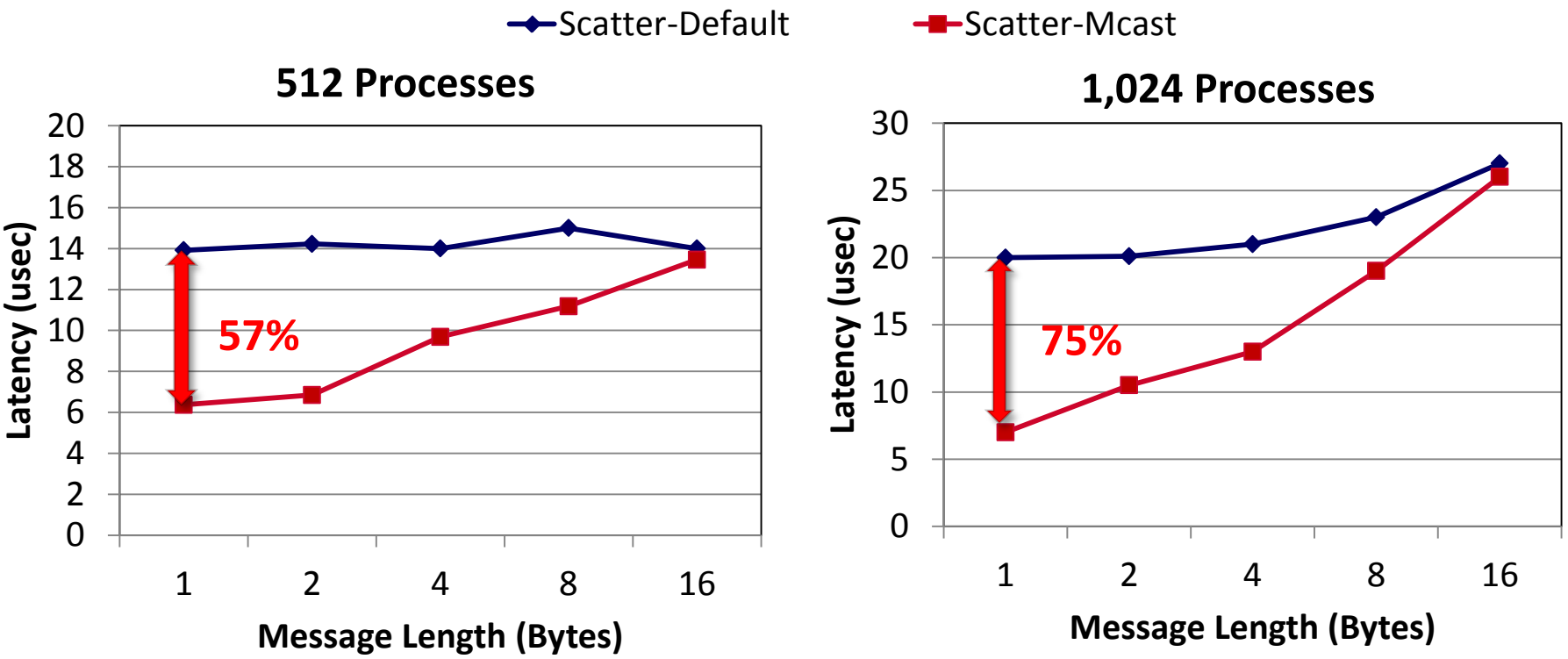






- MCAST-based designs improve latency of MPI_Bcast by up to **85%**

Hardware Multicast-aware MPI_Scatter



- Improves small message latency by up to **75%**

Parameter	Description	Default
MV2_USE_MCAST = 1	Enables hardware Multicast features	Disabled
--enable-mcast	Configure flag to enable	Enabled

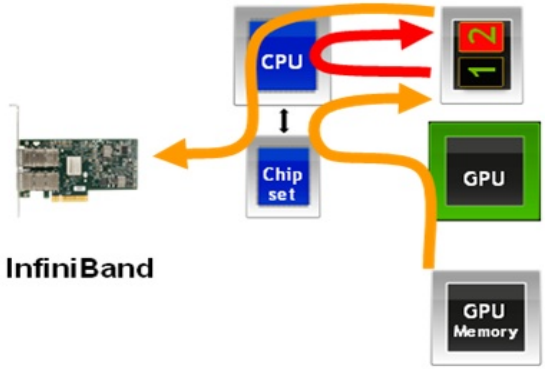
- **Refer to Running Collectives with Hardware based Multicast support section of MVAPICH2 user guide**
- **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-590006.8**

Presentation Outline

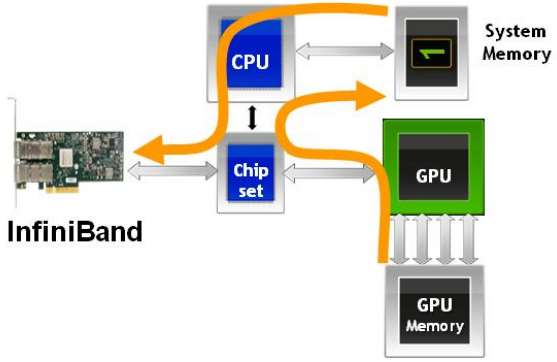
- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- **Optimizing and Tuning GPU Support in MVAPICH2**
  - **Pipelined Data Movement**
  - **GPUDirect RDMA (GDR) with CUDA**
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- Future Plans and Concluding Remarks

MVAPICH2-GPU: CUDA-Aware MPI

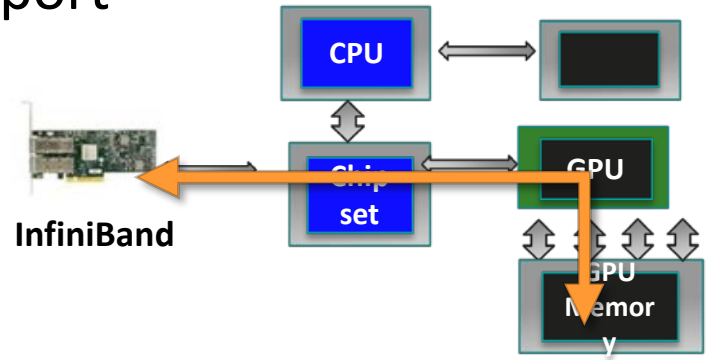
- Before CUDA 4: Additional copies
  - Low performance and low productivity



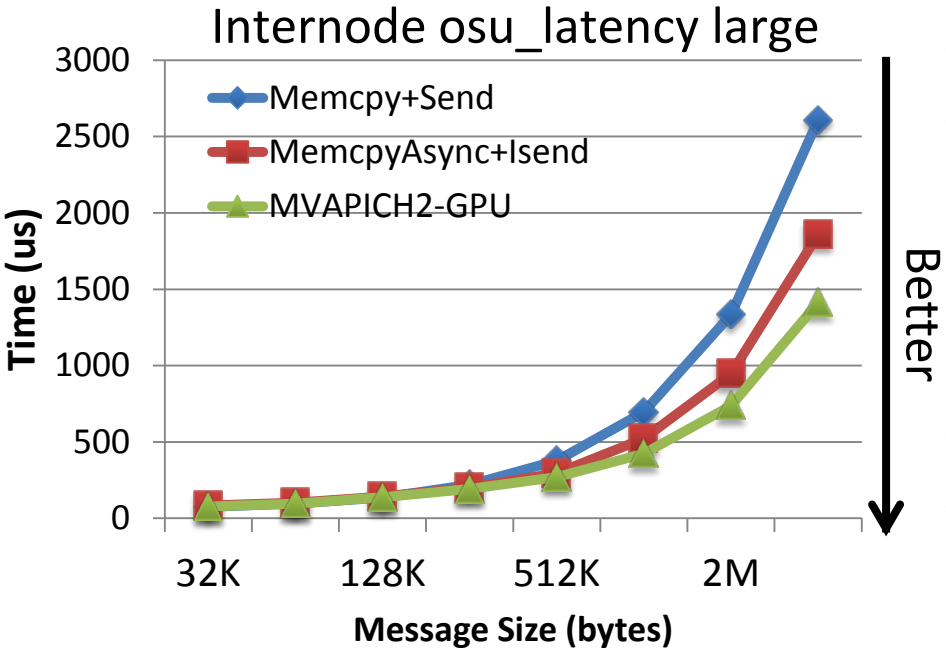
- After CUDA 4: Host-based pipeline
  - Unified Virtual Address
  - Pipeline CUDA copies with IB transfers
  - High performance and high productivity



- After CUDA 5.5: GPUDirect-RDMA support
  - GPU to GPU direct transfer
  - Bypass the host memory
  - Hybrid design to avoid PCI bottlenecks



Tuning Pipelined Data Movement in MVAPICH2

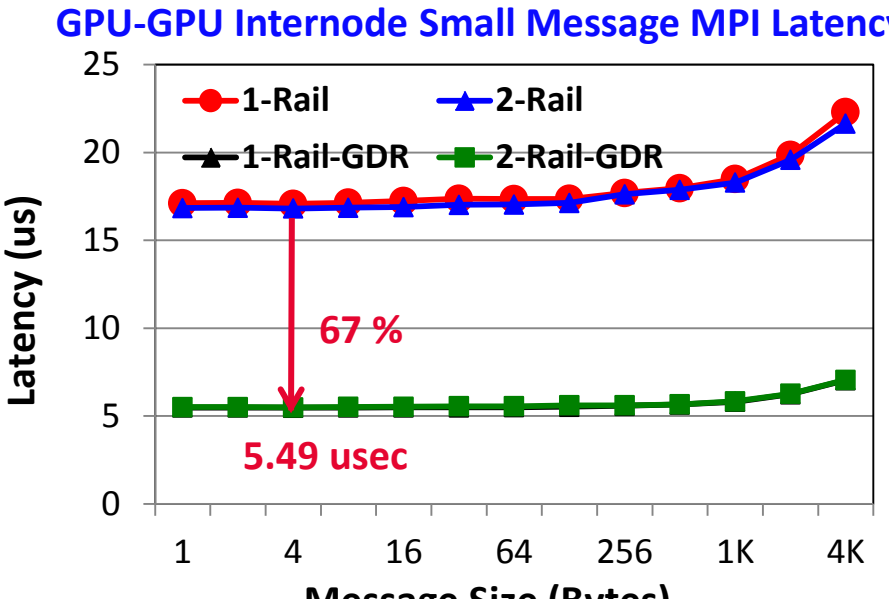


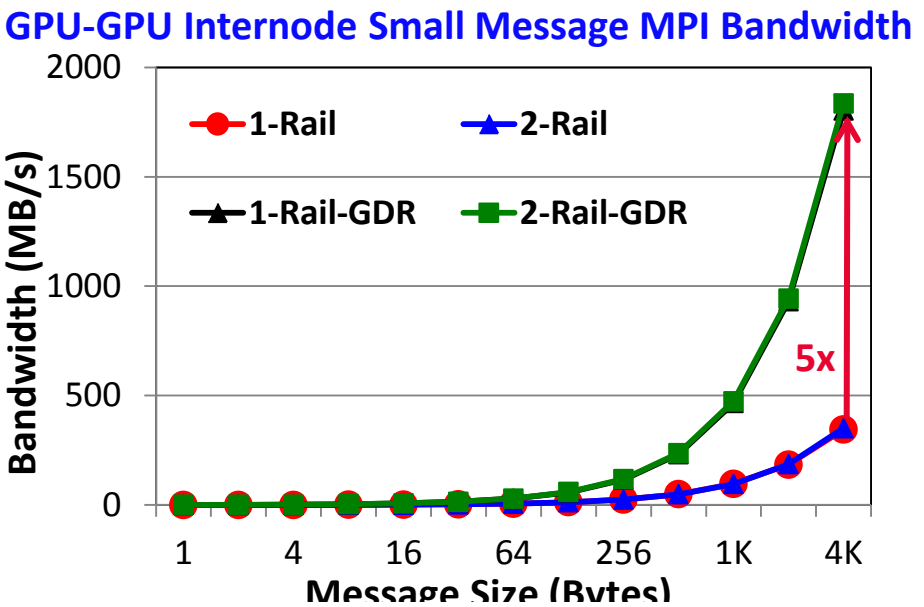
- Pipelines data movement from the GPU
- Overlaps
  - device-to-host CUDA copies
  - inter-process data movement (network transfers or shared memory copies)
  - host-to-device CUDA copies
- 45% improvement over naïve (Memcpy+Send)
- 24% improvement compared with an advanced user-level implementation (MemcpyAsync+Isend)

Parameter	Significance	Default	Notes
MV2_USE_CUDA	• Enable / Disable GPU designs	0 (Disabled)	• Disabled to avoid pointer checking overheads for host communication • Always enable to support MPI communication from GPU Memory
MV2_CUDA_BLOCK_SIZE	• Controls the pipeline blocksize	256 KByte	• Tune for your system and application • Varies based on - CPU Platform, IB HCA and GPU - CUDA driver version - Communication pattern (latency/bandwidth)

- **Refer to Running on Clusters with NVIDIA GPU Accelerators section of MVAPICH2 user guide**
- **http://mvapich.cse.ohio-state.edu/support/user_guide_mvapich2-2.0rc1.html#x1-810006.19**

Performance of MVAPICH2 with GPUDirect-RDMA





Parameter	Significance	Default	Notes
MV2_USE_GPUDIRECT	• Enable / Disable GDR-based designs	0 (Disabled)	• Always enable
MV2_GPUDIRECT_LIMIT	• Controls messages size until which GPUDirect RDMA is used	8 KByte	• Tune for your system • GPU type, host architecture and CUDA version: impact pipelining overheads and P2P bandwidth bottlenecks

Based on MVAPICH2-2.0b; Intel Ivy Bridge (E5-2680 v2) node with 20 cores; NVIDIA Tesla K40c GPU, Mellanox Connect-IB Dual-FDR HCACUDA 5.5, Mellanox OFED 2.0 with GPUDirect-RDMA Patch

How can I get Started with GDR Experimentation?

- MVAPICH2-2.0b with GDR support can be downloaded from

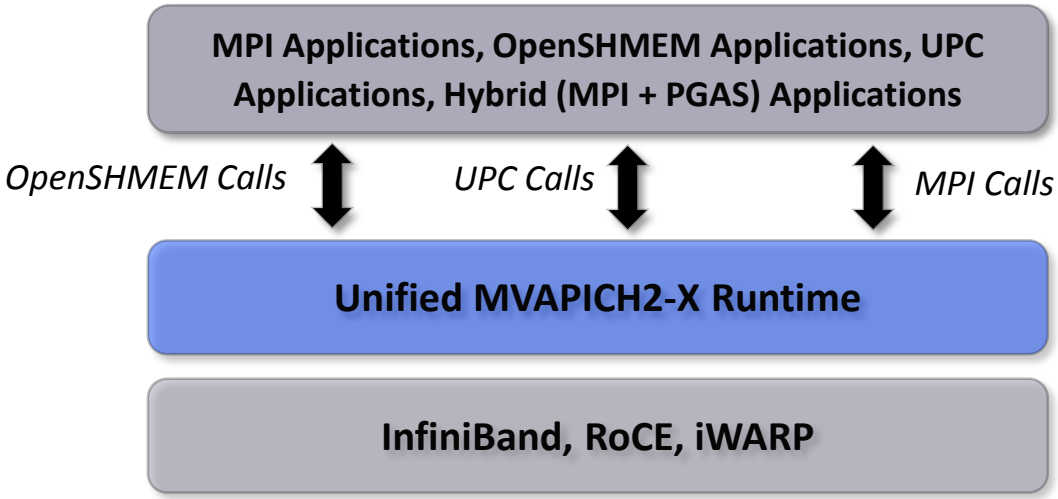
  https://mvapich.cse.ohio-state.edu/download/mvapich2gdr/
- System software requirements
  - Mellanox OFED 2.1
  - NVIDIA Driver 331.20 or later
  - NVIDIA CUDA Toolkit 5.5
  - Plugin for GPUDirect RDMA

    (http://www.mellanox.com/page/products_dyn?product_family=116)
- Has optimized designs for point-to-point communication using GDR
- Work under progress for optimizing collective and one-sided communication
- Contact MVAPICH help list with any questions related to the package
  - mvapich-help@cse.ohio-state.edu
- MVAPICH2-GDR-RC1 with additional optimizations coming soon!!

Presentation Outline

- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- **MVAPICH2-X for Hybrid MPI + PGAS**
- Enhanced Debugging System
- Future Plans and Concluding Remarks

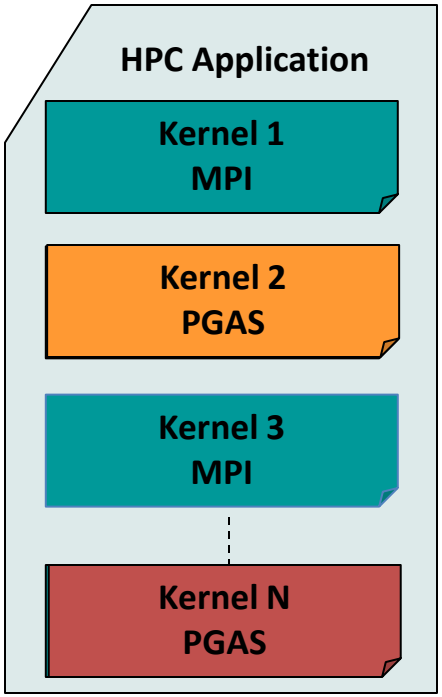
MVAPICH2-X for Hybrid MPI + PGAS Applications



- Unified communication runtime for MPI, UPC, OpenSHMEM available with MVAPICH2-X 1.9 onwards!
  - http://mvapich.cse.ohio-state.edu
- Feature Highlights
  - Supports MPI(+OpenMP), OpenSHMEM, UPC, MPI(+OpenMP) + OpenSHMEM, MPI(+OpenMP) + UPC
  - MPI-3 compliant, OpenSHMEM v1.0 standard compliant, UPC v1.2 standard compliant
  - Scalable Inter-node and Intra-node communication – point-to-point and collectives

Hybrid (MPI+PGAS) Programming

- Application sub-kernels can be re-written in MPI/PGAS based on communication characteristics
- Benefits:
  - Best of Distributed Computing Model
  - Best of Shared Memory Computing Model
- Exascale Roadmap*:
  - "Hybrid Programming is a practical way to program exascale systems"



* The International Exascale Software Roadmap, Dongarra, J., Beckman, P. et al., Volume 25, Number 1, 2011, International Journal of High Performance Computer Applications, ISSN 1094-3420

MVAPICH2-X Runtime Parameters

- **MPI Parameters**
  - MVAPICH2-X MPI based on MVAPICH2 OFA-IB-CH3 channel
  - All parameters for MVAPICH2 OFA-IB-CH3 channel are applicable
- **OpenSHMEM Parameters**
  - OOSHM_SYMMETRIC_HEAP_SIZE
    - Set OpenSHMEM Symmetric Heap Size (***Default: 512M***)
  - OOSHM_USE_SHARED_MEM
    - Use shared memory for intra-node (heap memory) communication (***Default: Enabled***)
  - OSHM_USE_CMA
    - Use Cross Memory Attach (CMA) for intra-node (heap/static memory) communication (***Default: Enabled***)
- **UPC Parameters**
  - UPC_SHARED_HEAP_SIZE
    - Set OpenSHMEM Symmetric Heap Size (***Default 512M***)
- **Hybrid Program Features**
  - Supports hybrid programming using MPI(+OpenMP), MPI(+OpenMP)+UPC and MPI(+OpenMP)+OpenSHMEM
  - Corresponding MPI/UPC/OpenSHMEM parameters can be selected

Presentation Outline

- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- **Enhanced Debugging System**
- Future Plans and Concluding Remarks

Getting Help

- Check the MVAPICH2 FAQ
- Check the Mailing List Archives
- Basic System Diagnostics
  - ibv_devinfo - at least one port should be PORT_ACTIVE
  - ulimit -l - should be “unlimited” on all compute nodes
  - host resolution: DNS or /etc/hosts
  - password-less ssh login
  - run IB perf tests for all the message sizes(-a option)
    - Ib_send_lat, ib_send_bw
  - run system program (like hostname) and MPI hello world program
- More diagnostics
  - Already fixed issue: always try with latest release
  - Regression: verifying with previous release
  - Application issue: verify with other MPI libraries
  - Launcher issue: verifying with multiple launchers (mpirun_rsh, mpiexec.hydra)
  - Debug mode (Configure with ***--enable-g=dbg, --enable-fast=none***)
  - Compiler optimization issues: try with different compiler

Submitting Bug Report

- Subscribe to mvapich-discuss and send problem report
- Include as much information as possible
- ***What information to include ???***
- Run-time issues
  - Config flags (“mpiname –a” output)
  - Exact command used to run the application
  - Run-rime parameters in the environment
    - **What parameters are being used?**
      - **MV2_SHOW_ENV_INFO=<1/2>**
      - Show values of the run time parameters
      - 1 (short list), 2 (full list)
  - Where is the segmentation fault?
    - **MV2_DEBUG_SHOW_BACKTRACE=1**
    - Shows backtrace with debug builds
      - Use following configure flags
      - ***--enable-g=dbg, --enable-fast=none***
  - Standalone reproducer program
  - Information about the IB network
    - OFED version
    - ibv_devinfo
  - Remote system access
- Build and Installation issues
  - MVAPICH2 version
  - Compiler version
  - Platform details (OS, kernel version..etc)
  - Configure flags
  - Attach Config.log file
  - Attach configure, make and make install step output
    - ./configure {–flags} 2>&1 | tee config.out
    - make 2>&1 | tee make.out
    - make install 2>&1 | tee install.out

User Resources

- MVAPIVH2 Quick Start Guide
- MVAPICH2 User Guide
  - Long and very detailed
  - FAQ
- MVAPICH2 Web-Site
  - Overview and Features
  - Reference performance
  - Publications
- Mailing List Support
  - mvapich-discuss@cse.ohio-state.edu
- Mailing List Archives
- All above resources accessible from: http://mvapich.cse.ohio-state.edu/

Presentation Outline

- Overview of MVAPICH2 and MVAPICH2-X
- Optimizing and Tuning Job Startup
- Efficient Process Mapping Strategies
- Point-to-Point Tuning and Optimizations
- InfiniBand Transport Protocol Based Tuning
- Tuning for Multi-rail Clusters, 3D Torus Networks and QoS Support
- Collective Optimizations using Hardware-based Multicast
- Optimizing and Tuning GPU Support in MVAPICH2
- MVAPICH2-X for Hybrid MPI + PGAS
- Enhanced Debugging System
- **Future Plans and Concluding Remarks**

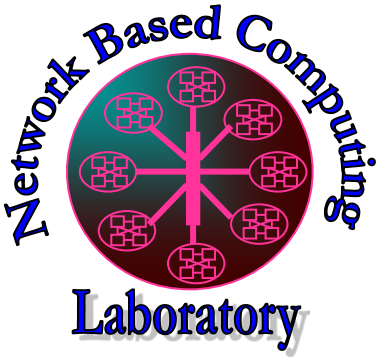
MVAPICH2/MVPICH2-X – Plans for Exascale

- Performance and Memory scalability toward 500K-1M cores
  - Dynamically Connected Transport (DCT) service with Connect-IB
- Enhanced Optimization for GPGPU and Coprocessor Support
  - Extending the GPGPU support (GPU-Direct RDMA) with CUDA 6.0 and Beyond
  - Support for Intel MIC (Knight Landing)
- Taking advantage of Collective Offload framework
  - Including support for non-blocking collectives (MPI 3.0)
- RMA support (as in MPI 3.0)
- Extended topology-aware collectives
- Power-aware collectives
- Support for MPI Tools Interface (as in MPI 3.0)
- Checkpoint-Restart and migration support with in-memory checkpointing
- Hybrid MPI+PGAS programming support with GPGPUs and Accelertors

Concluding Remarks

- Provided an overview of MVAPICH2 and MVAPICH2-X Libraries
- Presented in-depth details on configuration and runtime parameters, optimizations and their impacts
- Provided an overview of debugging support
- Demonstrated how MPI and PGAS users can use these optimization techniques to extract performance and scalability while using MVAPICH2 and MVAPICH2-X
- MVAPICH2 has many more features not covered here
  - Fault tolerance, Dynamic Process Management etc
  - Please visit http://mvapich.cse.ohio-state.edu for details
- More information about optimizing / tuning MVAPICH2 / MVAPICH2-X available at MVAPICH User Group Meeting (MUG) 2013 website
  - http://mug.mvapich.cse.ohio-state.edu

Pointers



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

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

panda@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~panda