MVAPICH

MPI, PGAS and Hybrid MPI+PGAS Library

MVAPICH2-GDR: Pushing the Frontier of Designing MPI Libraries Enabling GPUDirect Technologies

GPU Technology Conference GTC 2016

by

Dhabaleswar K. (DK) Panda

The Ohio State University

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

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

Khaled Hamidouche

The Ohio State University

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

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

Outline

- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What’s new with MVAPICH2-GDR
  - Efficient MPI-3 Non-Blocking Collective support
  - Maximal overlap in MPI Datatype Processing
  - Efficient Support for Managed Memory
  - RoCE and Optimized Collective
  - Initial support for GPUDirect Async feature
  - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

Overview of the MVAPICH2 Project

- High Performance open-source MPI Library for InfiniBand, 10-40Gig/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 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
  - **Used by more than 2,550 organizations in 79 countries**
  - **More than 360,000 (> 0.36 million) downloads from the OSU site directly**
  - Empowering many TOP500 clusters (Nov '15 ranking)
    - 10^{th} ranked 519,640-core cluster (Stampede) at TACC
    - 13^{th} ranked 185,344-core cluster (Pleiades) at NASA
    - 25^{th} ranked 76,032-core cluster (Tsubame 2.5) at Tokyo Institute of Technology and many others
  - 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 (3^{rd} in Nov 2003, 2,200 processors, 12.25 TFlops) ->
  - Stampede at TACC (10^{th} in Nov'15, 519,640 cores, 5.168 Plops)

MVAPICH2 Architecture

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, OmniPath)

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)

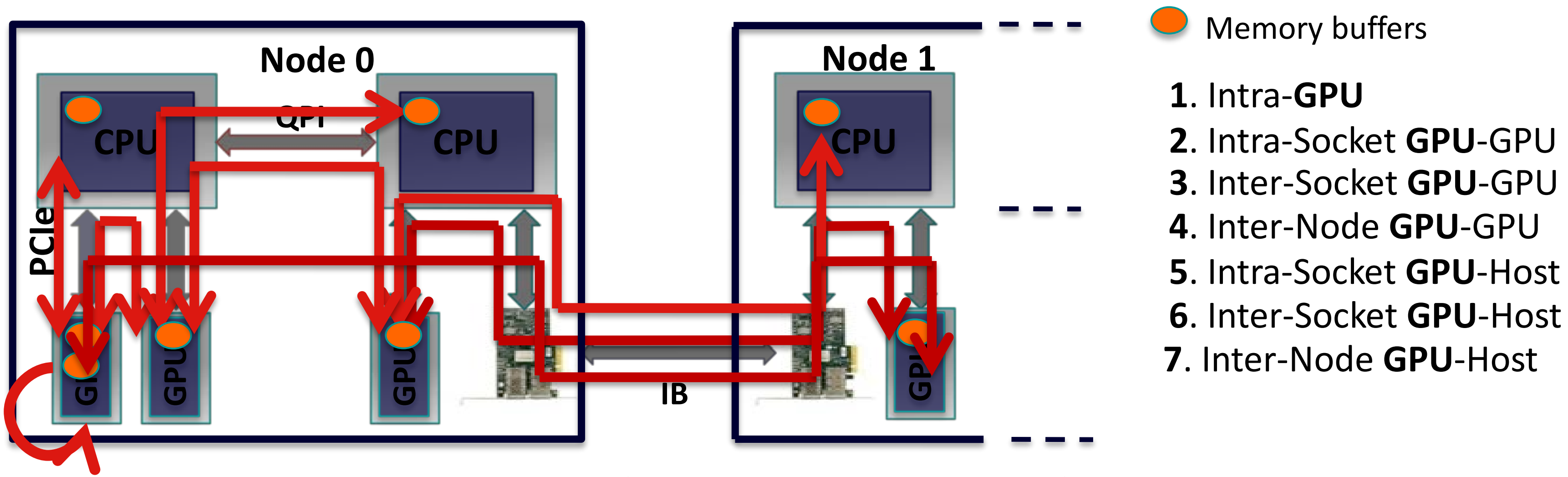
Transport Mechanisms: Shared Memory, CMA, IVSHMEM

Modern Features: MCDRAM*, NVLink*, CAPI*

* Upcoming

Optimizing MPI Data Movement on GPU Clusters

- Connected as PCIe devices – Flexibility but Complexity



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

and more . . .

- For each path different schemes: Shared_mem, IPC, GPUDirect RDMA, pipeline …
- Critical for runtimes to optimize data movement while hiding the complexity

GPU-Aware (CUDA-Aware) MPI Library: MVAPICH2-GPU

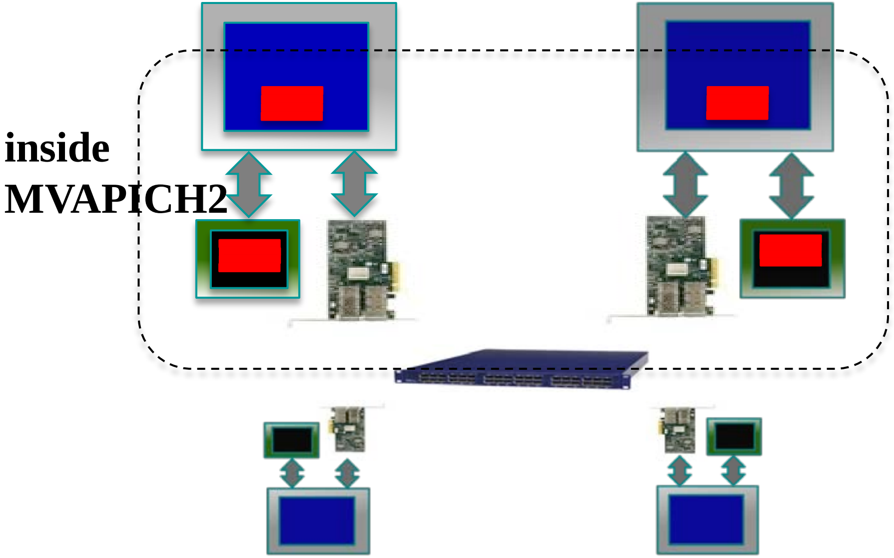
- Standard MPI interfaces used for unified data movement
- Takes advantage of Unified Virtual Addressing (>= CUDA 4.0)
- Overlaps data movement from GPU with RDMA transfers

At Sender:

MPI_Send(s_devbuf, size, …);

At Receiver:

MPI_Recv(r_devbuf, size, …);



High Performance and High Productivity

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

Using MVAPICH2-GPUDirect Version

- MVAPICH2-2.2b with GDR support can be downloaded from

  https://mvapich.cse.ohio-state.edu/download/mvapich2gdr/

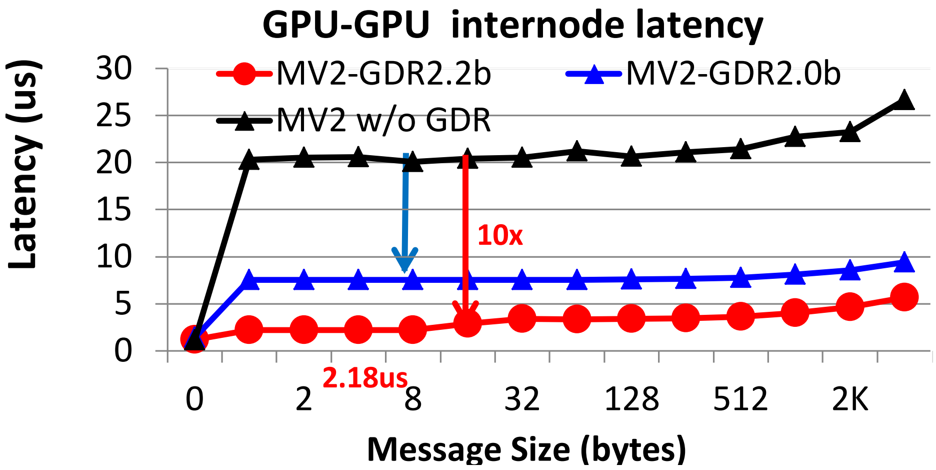
- System software requirements
  - Mellanox OFED 2.1 or later
  - NVIDIA Driver 331.20 or later
  - NVIDIA CUDA Toolkit 7.0 or later
  - Plugin for GPUDirect RDMA

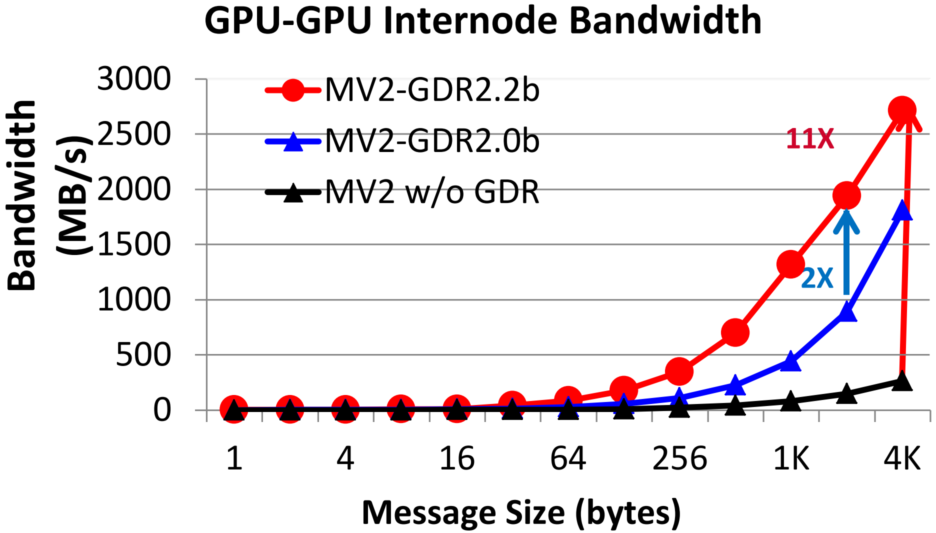
    http://www.mellanox.com/page/products_dyn?product_family=116
  - Strongly recommended
  - GDRCOPY module from NVIDIA

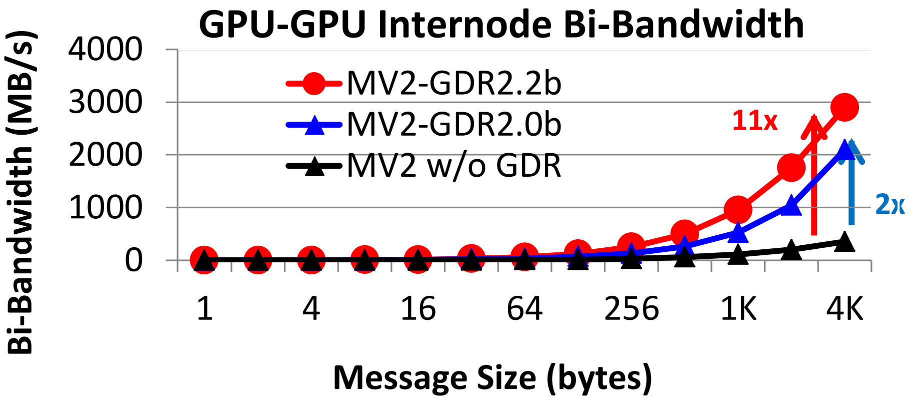
    https://github.com/NVIDIA/gdrcopy
- Contact MVAPICH help list with any questions related to the package

  mvapich-help@cse.ohio-state.edu

Performance of MVAPICH2-GPU with GPU-Direct RDMA (GDR)

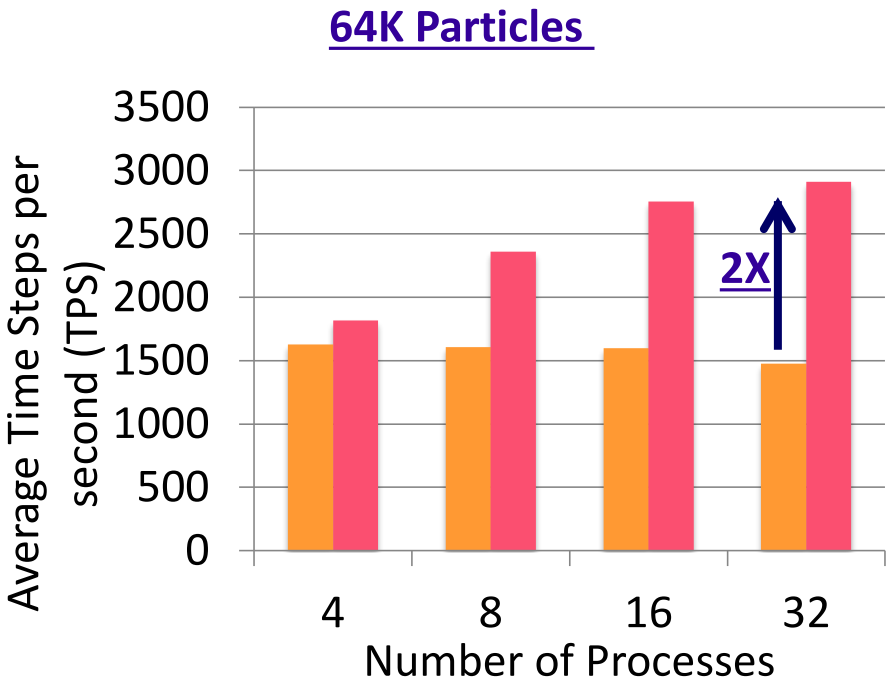


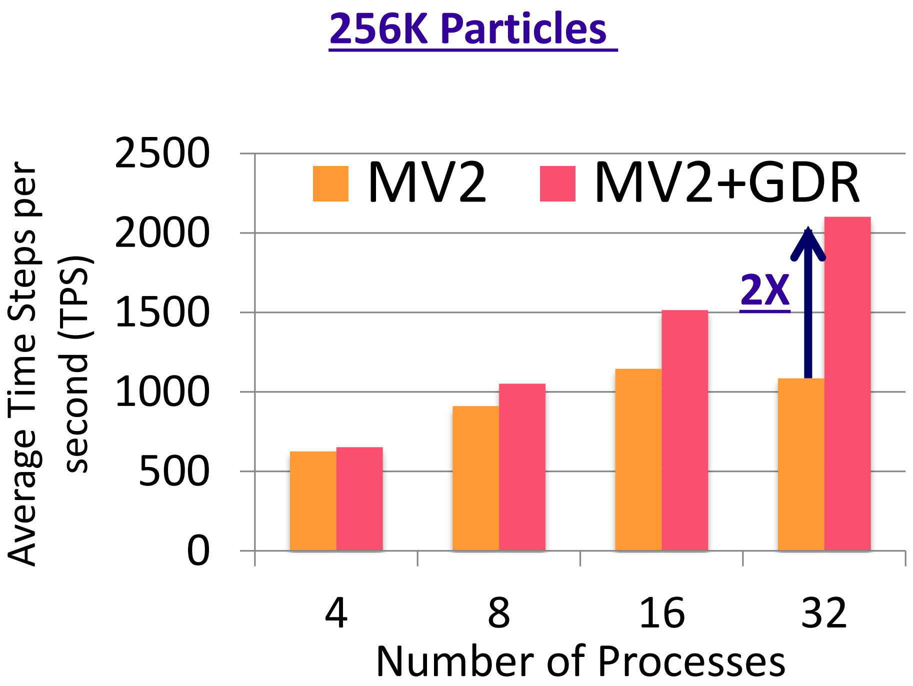




MVAPICH2-GDR-2.2b
Intel Ivy Bridge (E5-2680 v2) node - 20 cores
NVIDIA Tesla K40c GPU
Mellanox Connect-IB Dual-FDR HCA
CUDA 7
Mellanox OFED 2.4 with GPU-Direct-RDMA

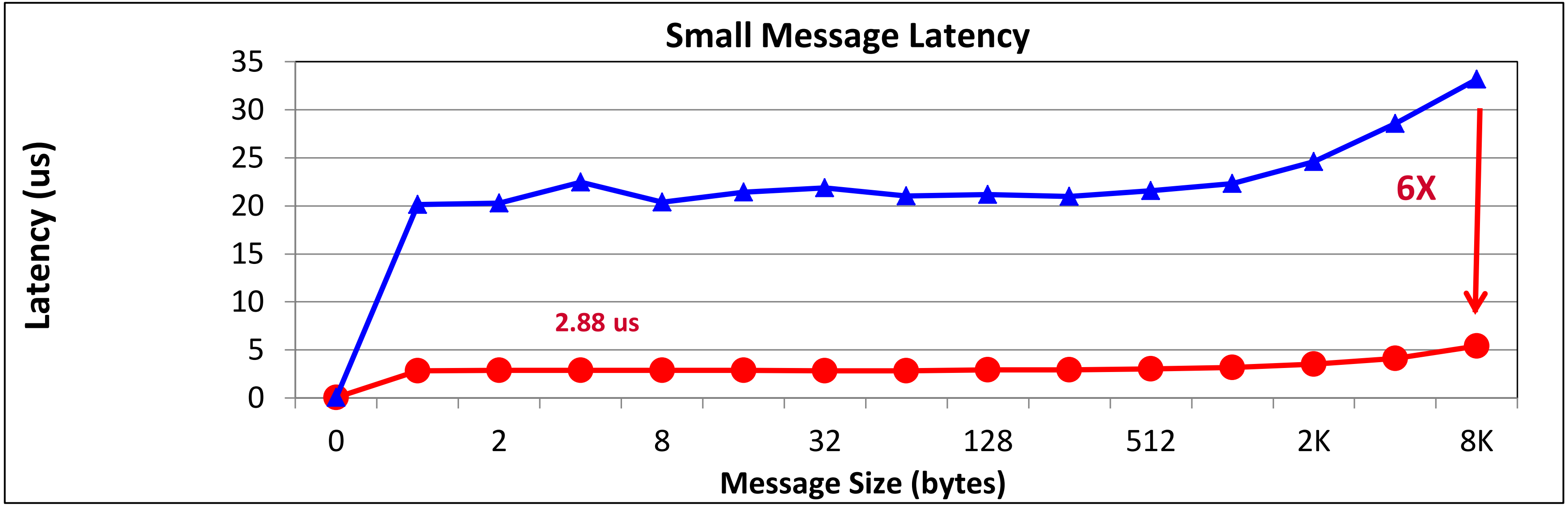
Application-Level Evaluation (HOOMD-blue)





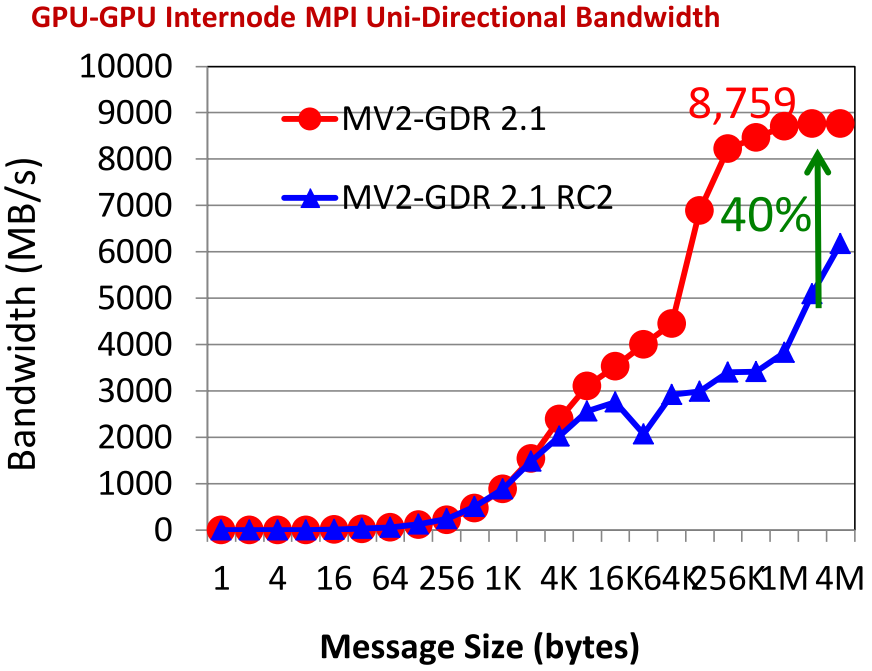
- Platform: Wilkes (Intel Ivy Bridge + NVIDIA Tesla K20c + Mellanox Connect-IB)
- HoomdBlue Version 1.0.5
  - GDRCOPY enabled: MV2_USE_CUDA=1 MV2_IBA_HCA=mlx5_0 MV2_IBA_EAGER_THRESHOLD=32768 MV2_VBUF_TOTAL_SIZE=32768 MV2_USE_GPUDIRECT_LOOPBACK_LIMIT=32768 MV2_USE_GPUDIRECT_GDRCOPY=1 MV2_USE_GPUDIRECT_GDRCOPY_LIMIT=16384

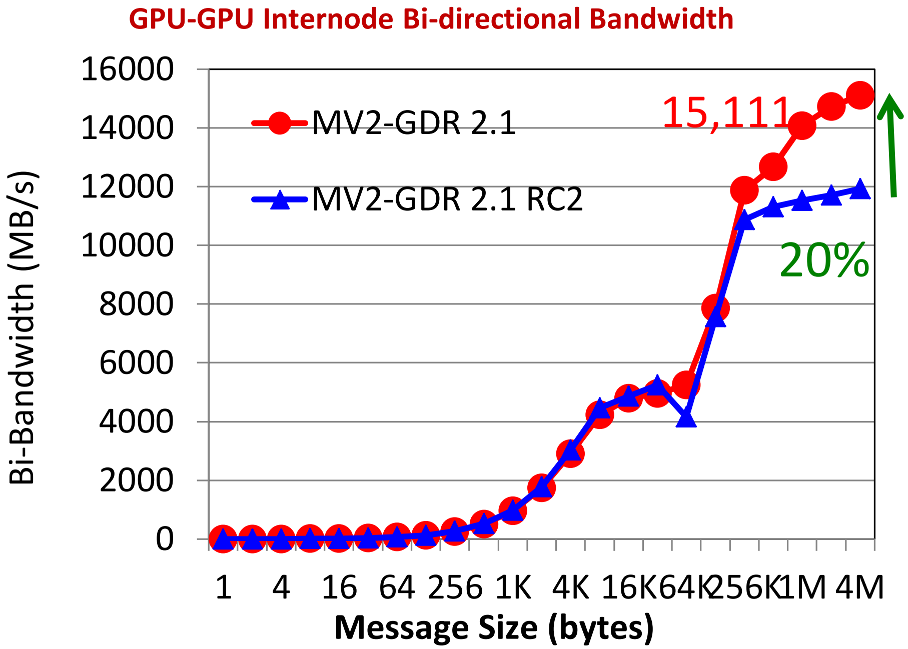
Full and Efficient MPI-3 RMA Support



MVAPICH2-GDR-2.2b
Intel Ivy Bridge (E5-2680 v2) node - 20 cores, NVIDIA Tesla K40c GPU
Mellanox Connect-IB Dual-FDR HCA, CUDA 7
Mellanox OFED 2.4 with GPU-Direct-RDMA

Performance of MVAPICH2-GDR with GPU-Direct RDMA and Multi-Rail Support





MVAPICH2-GDR-2.2.b
Intel Ivy Bridge (E5-2680 v2) node - 20 cores, NVIDIA Tesla K40c GPU
Mellanox Connect-IB Dual-FDR HCA CUDA 7
Mellanox OFED 2.4 with GPU-Direct-RDMA

Outline

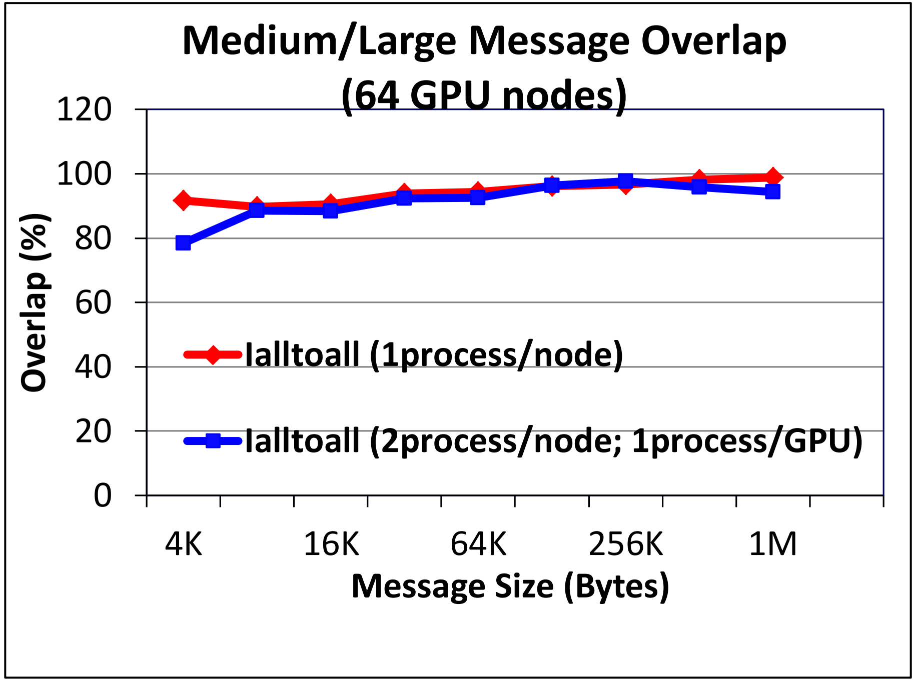
- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What's new with MVAPICH2-GDR
  - Efficient MPI-3 Non-Blocking Collective support
  - Maximal overlap in MPI Datatype Processing
  - Efficient Support for Managed Memory
  - RoCE and Optimized Collective
  - Initial support for GPUDirect Async feature
  - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

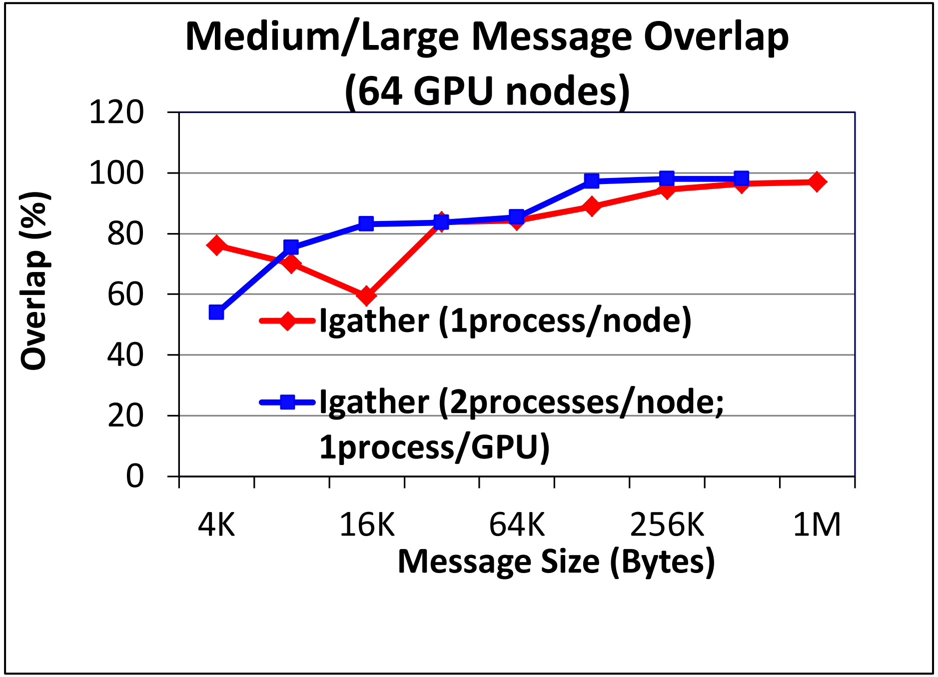
Non-Blocking Collectives (NBC) using Core-Direct Offload

- MPI NBC decouples initiation (Ialltoall) and completion (Wait) phases and provide overlap potential (Ialltoall + compute + Wait) but CPU drives progress largely in Wait (=> 0 overlap)
- CORE-Direct feature provides true overlap capabilities by providing a priori specification of a list of network-tasks which is progressed by the NIC instead of the CPU (hence freeing it)
- We propose a design that **combines GPUDirect RDMA and Core-Direct features** to provide efficient support of CUDA-Aware NBC collectives on GPU buffers
  - Overlap communication with CPU computation
  - Overlap communication with GPU computation
- Extend OMB with CUDA-Aware NBC benchmarks to evaluate
  - Latency
  - Overlap on both CPU and GPU

A. Venkatesh, K. Hamidouche, H. Subramoni, and D. K. Panda, Offloaded GPU Collectives using CORE-Direct and CUDA Capabilities on IB Clusters, HIPC, 2015

CUDA-Aware Non-Blocking Collectives





A. Venkatesh, K. Hamidouche, H. Subramoni, and D. K. Panda, Offloaded GPU Collectives using CORE-Direct and CUDA Capabilities on IB Clusters, HIPC, 2015

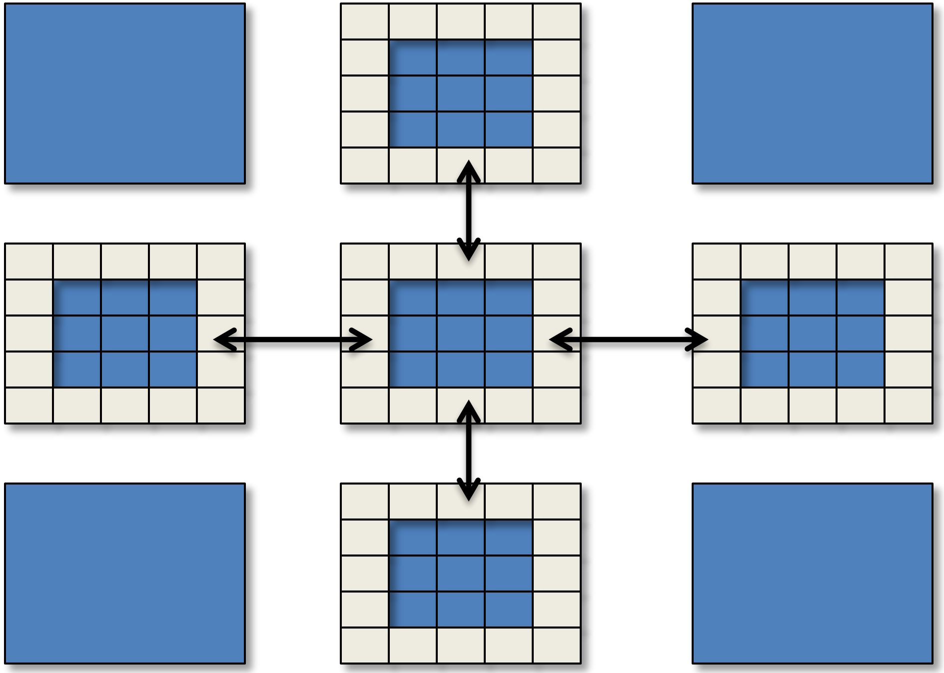
Platform: Wilkes: Intel Ivy Bridge
NVIDIA Tesla K20c + Mellanox Connect-IB
Available since MVAPICH2-GDR 2.2b

Outline

- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What's new with MVAPICH2-GDR
    - Efficient MPI-3 Non-Blocking Collective support
    - Maximal overlap in MPI Datatype Processing
    - Efficient Support for Managed Memory
    - RoCE and Optimized Collective
    - Initial support for GPUDirect Async feature
    - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

Non-contiguous Data Exchange

Halo data exchange



- Multi-dimensional data
  - Row based organization
  - Contiguous on one dimension
  - Non-contiguous on other dimensions
- Halo data exchange
  - Duplicate the boundary
  - Exchange the boundary in each iteration

MPI Datatype Processing (Computation Optimization)

- Comprehensive support
  - Targeted kernels for regular datatypes - vector, subarray, indexed_block
  - Generic kernels for all other irregular datatypes
- Separate non-blocking stream for kernels launched by MPI library
  - Avoids stream conflicts with application kernels
- Flexible set of parameters for users to tune kernels
  - Vector
    - MV2_CUDA_KERNEL_VECTOR_TIDBLK_SIZE
    - MV2_CUDA_KERNEL_VECTOR_YSIZE
  - Subarray
    - MV2_CUDA_KERNEL_SUBARR_TIDBLK_SIZE
    - MV2_CUDA_KERNEL_SUBARR_XDIM
    - MV2_CUDA_KERNEL_SUBARR_YDIM
    - MV2_CUDA_KERNEL_SUBARR_ZDIM
  - Indexed_block
    - MV2_CUDA_KERNEL_IDXBLK_XDIM

MPI Datatype Processing (Communication Optimization)

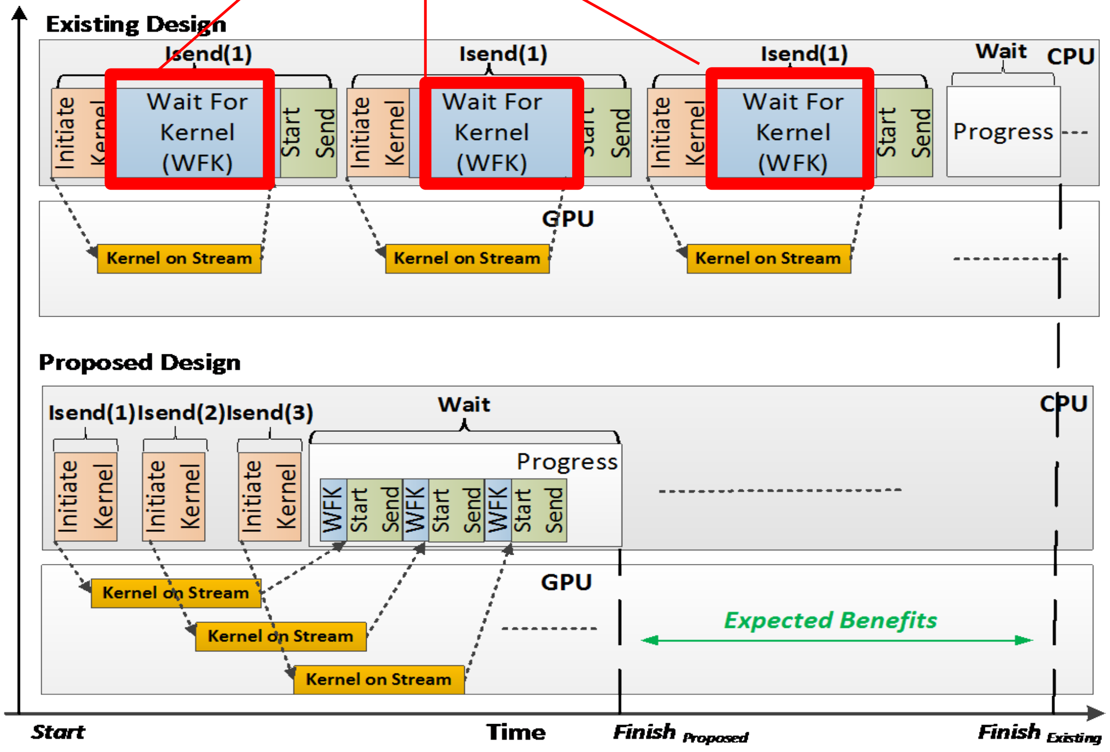
Common Scenario

MPI_Isend (A,.. Datatype,…)
MPI_Isend (B,.. Datatype,…)
MPI_Isend (C,.. Datatype,…)
MPI_Isend (D,.. Datatype,…)
…

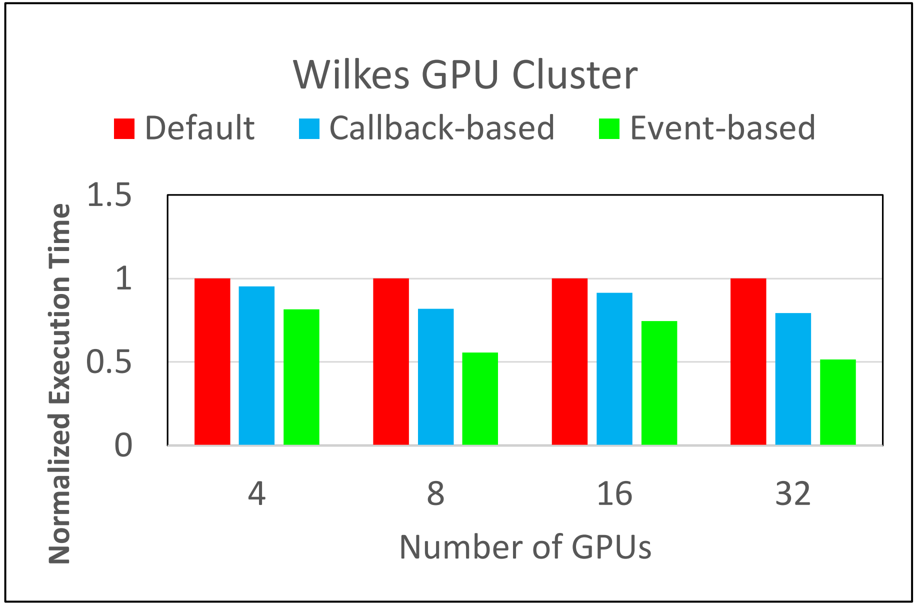
MPI_Waitall (…);

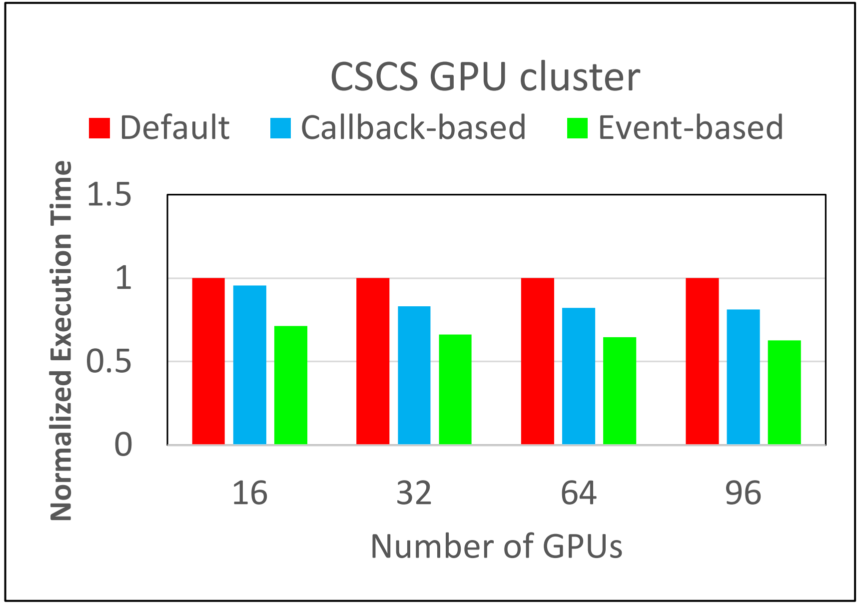
*A, B…contain non-contiguous MPI Datatype

Waste of computing resources on CPU and GPU



Application-Level Evaluation (HaloExchange - Cosmo)





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

On-going Collaboration with CSCS and Meteo Swiss

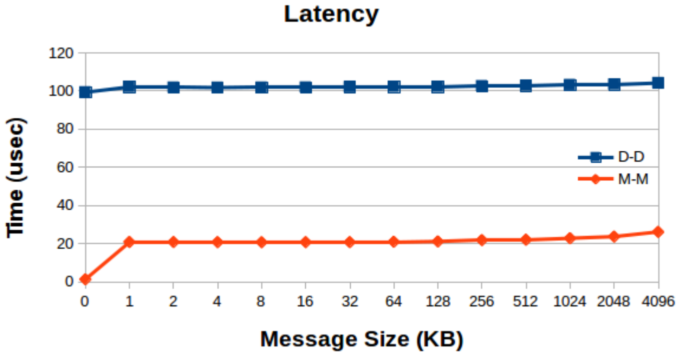
C. Chu, K. Hamidouche, A. Venkatesh, D. Banerjee , H. Subramoni, and D. K. Panda, Exploiting Maximal Overlap for Non-Contiguous Data Movement Processing on Modern GPU-enabled Systems, IPDPS'16

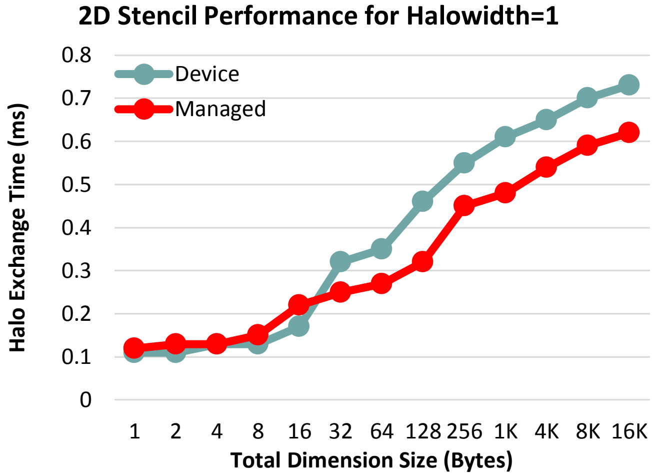
Outline

- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What's new with MVAPICH2-GDR
  - Efficient MPI-3 Non-Blocking Collective support
  - Maximal overlap in MPI Datatype Processing
  - Efficient Support for Managed Memory
  - RoCE and Optimized Collective
  - Initial support for GPUDirect Async feature
  - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

Initial (Basic) Support for GPU Managed Memory

- CUDA 6.0 NVIDIA introduced CUDA Managed (or Unified) memory allowing a common memory allocation for GPU or CPU through *cudaMallocManaged()* call
- Significant productivity benefits due to abstraction of explicit allocation and *cudaMemcpy()*
- Extended MVAPICH2 to perform communications directly from managed buffers (Available in MVAPICH2-GDR 2.2b)
- OSU Micro-benchmarks extended to evaluate the performance of point-to-point and collective communications using managed buffers
  - Available since OMB 5.2

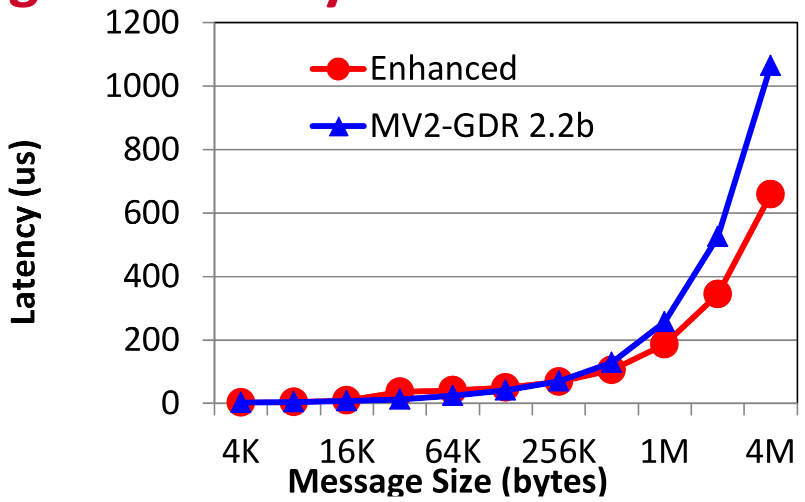


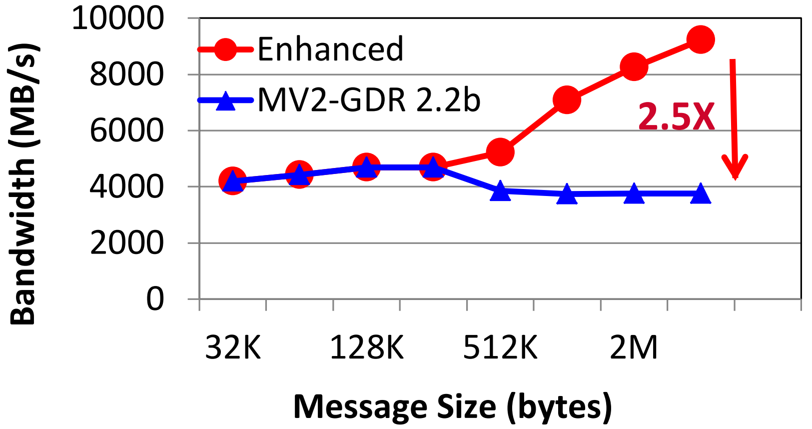


D. S. Banerjee, K Hamidouche, and D. K Panda, Designing High Performance Communication Runtime for GPUManaged Memory: Early Experiences, GPGPU-9 Workshop, to be held in conjunction with PPoPP ‘16

Enhanced Support for Intra-node Managed Memory

- CUDA Managed => no memory pin down
  - No IPC support for intra-node communication
  - No GDR support for Inter-node communication
- Initial and basic support in MVAPICH2-GDR
  - For both intra- and inter-nodes use “pipeline through” host memory
- Enhance intra-node 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
- Will be available in MVAPICH2-GDR 2.2RC1





Outline

- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What's new with MVAPICH2-GDR
  - Efficient MPI-3 Non-Blocking Collective support
  - Maximal overlap in MPI Datatype Processing
  - Efficient Support for Managed Memory
  - RoCE and Optimized Collective
  - Initial support for GPUDirect Async feature
  - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

ROCE and Optimized Collectives Support

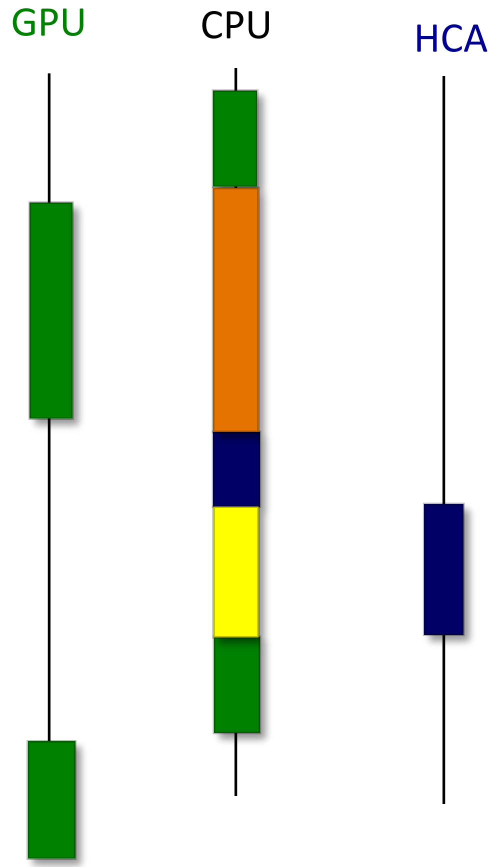
- RoCE V1 and V2 support
- RDMA_CM connection support
- CUDA-Aware Collective Tuning
  - Point-point Tuning (available since MVAPICH2-GDR 2.0)
    - Tuned thresholds for the different communication patterns and features
    - Depending on the system configuration (CPU, HCA and GPU models)
  - Tuning Framework for GPU based collectives
    - Select the best algorithm depending on message size, system size and system configuration
    - Support for Bcast and Gather operations for different GDR-enabled systems
- Will be available with the upcoming MVAPICH2-GDR 2.2RC1 release

Outline

- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What's new with MVAPICH2-GDR
  - Efficient MPI-3 Non-Blocking Collective support
  - Maximal overlap in MPI Datatype Processing
  - Efficient Support for Managed Memory
  - RoCE and Optimized Collective
  - Initial support for GPUDirect Async feature
  - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

Overview of GPUDirect aSync (GDS) Feature: Current MPI+CUDA interaction

```
CUDA_Kernel_a<<<>>>(A…., stream1)
cudaStreamSynchronize(stream1)
MPI_ISend (A,…., req1)
MPI_Wait (req1)
CUDA_Kernel_b<<<>>>(B…., stream1)
```

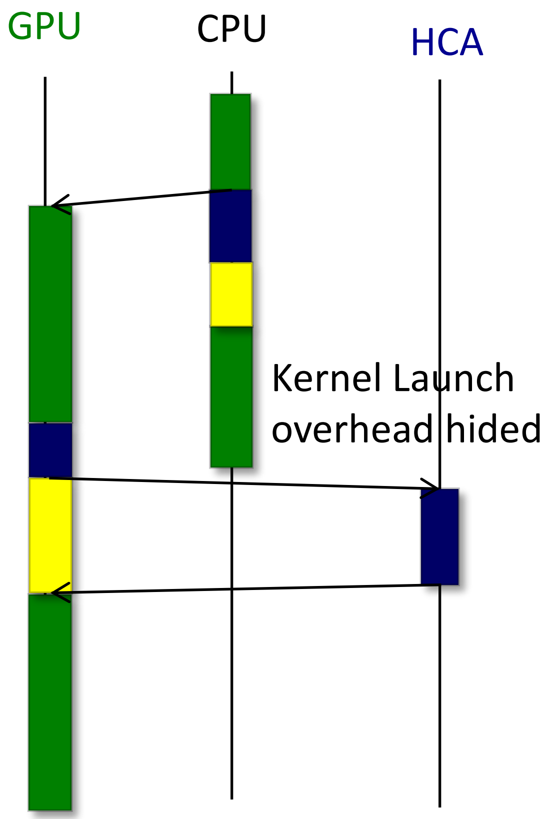


100% CPU control

- Limit the throughput of a GPU
- Limit the asynchronous progress
- Waste CPU cycles

MVAPICH2-GDS: Decouple GPU Control Flow from CPU

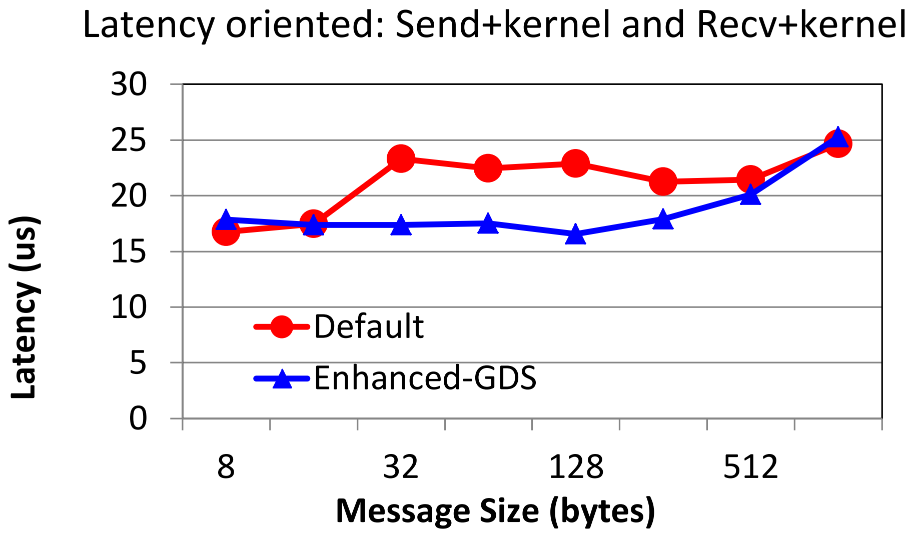
CUDA_Kernel_a<<<>>>(A…., stream1)
MPI_ISend (A,…., req1, stream1)
MPI_Wait (req1, stream1) (non-blocking from CPU)
CUDA_Kernel_b<<<>>>(B…., stream1)

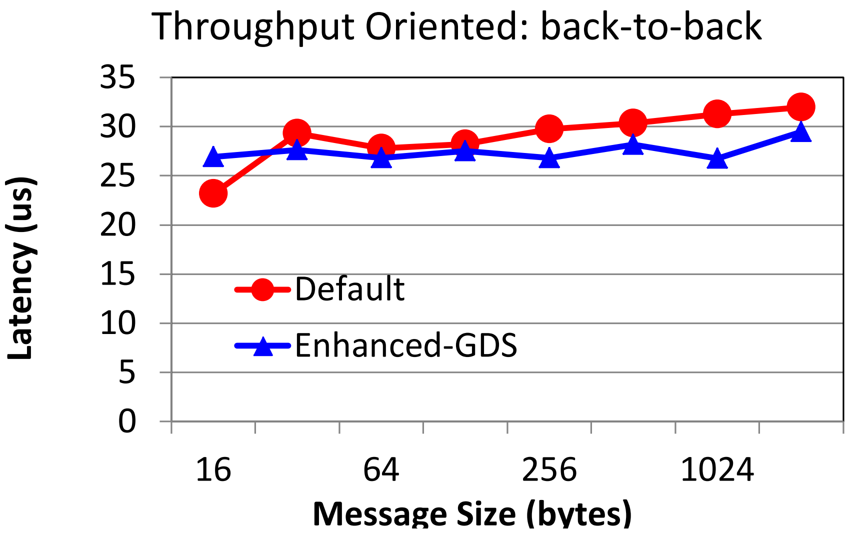


CPU offloads the compute, communication and synchronization tasks to GPU

- CPU is out of the critical path
- Tight interaction between GPU and HCA
- Hide the overhead of kernel launch
- Requires MPI semantics extensions
    - All operations are asynchronous from CPU
    - Extend MPI semantics with Stream-based semantics

MVAPICH2-GDS: Preliminary Results





- 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
  - Requires some tuning and expect better performance for Application with different Kernels

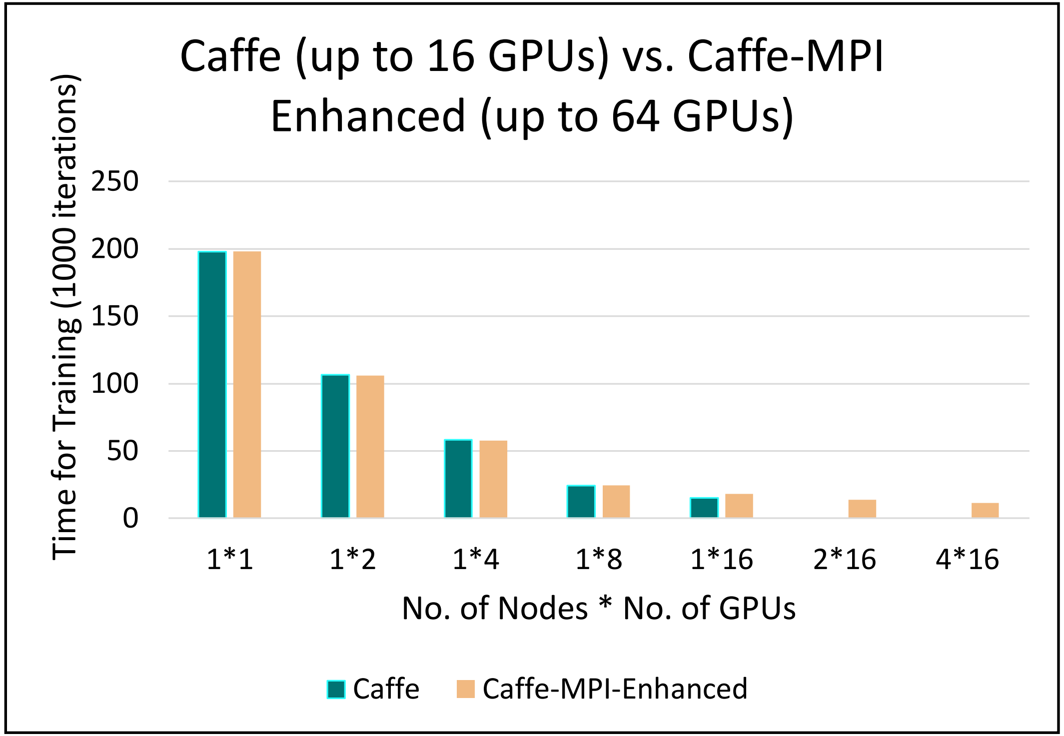
Intel SandyBridge, NVIDIA K20 and Mellanox FDR HCA

Outline

- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What's new with MVAPICH2-GDR
  - Efficient MPI-3 Non-Blocking Collective support
  - Maximal overlap in MPI Datatype Processing
  - Efficient Support for Managed Memory
  - RoCE and Optimized Collective
  - Initial support for GPUDirect Async feature
  - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

Efficient Deep Learning with MVAPICH2-GDR

- Caffe : A flexible and layered Deep Learning framework.
- Benefits and Weaknesses
  - Multi-GPU Training within a single node
  - Performance degradation for GPUs across different sockets
- Can we enhance Caffe with MVAPICH2-GDR?
  - Caffe-MPI Enhanced: A CUDA-Aware MPI version
  - Enable Scale-up (within a node) and Scale-out (across multi-GPU nodes)
  - Initial Evaluation suggests that we can scale up to 64 GPUs for training the CIFAR-10 model



Outline

- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What's new with MVAPICH2-GDR
  - Efficient MPI-3 Non-Blocking Collective support
  - Maximal overlap in MPI Datatype Processing
  - Efficient Support for Managed Memory
  - RoCE and Optimized Collective
  - Initial support for GPUDirect Async feature
  - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

OpenACC-Aware MPI

- acc_malloc to allocate device memory
  - No changes to MPI calls
  - MVAPICH2 detects the device pointer and optimizes data movement
- acc_deviceptr to get device pointer (in OpenACC 2.0)
  - Enables MPI communication from memory allocated by compiler when it is available in OpenACC 2.0 implementations
  - MVAPICH2 will detect the device pointer and optimize communication
- Delivers the same performance as with CUDA

```
A = acc_malloc(sizeof(int) * N);
......
#pragma acc parallel loop deviceptr(A) . . .
//compute for loop

MPI_Send (A, N, MPI_INT, 0, 1, MPI_COMM_WORLD);

......
acc_free(A);
```

```
A = malloc(sizeof(int) * N);
......
#pragma acc data copyin(A) . . .
{
#pragma acc parallel loop . . .
//compute for loop
MPI_Send(acc_deviceptr(A), N, MPI_INT, 0, 1,
MPI_COMM_WORLD);
}
......
free(A);
```

Outline

- Overview of the MVAPICH2 Project
- MVAPICH2-GPU with GPUDirect-RDMA (GDR)
- What's new with MVAPICH2-GDR
  - Efficient MPI-3 Non-Blocking Collective support
  - Maximal overlap in MPI Datatype Processing
  - Efficient Support for Managed Memory
  - RoCE and Optimized Collective
  - Initial support for GPUDirect Async feature
  - Efficient Deep Learning with MVAPICH2-GDR
- OpenACC-Aware support
- Conclusions

Conclusions

- MVAPICH2 optimizes MPI communication on InfiniBand clusters with GPUs
- Provides optimized designs for point-to-point two-sided and one-sided communication, datatype processing and collective operations
- Efficient and maximal overlap for MPI-3 NBC collectives
- Delivers high performance and high productivity with support for the latest NVIDIA GPUs and InfiniBand Adapters
- Looking forward to next-generation designs with GPUDirect Async (GDS) and applications domain like Deep Learning
- Users are strongly encouraged to use the latest MVAPICH2-GDR release to avail all features and performance benefits

A Follow-up Talk on PGAS/OpenSHMEM

- **S6418 - Bringing NVIDIA GPUs to the PGAS/OpenSHMEM World: Challenges and Solutions**
  - **Day:** Wednesday, 04/06
  - **Time:** 16:30 - 16:55
  - **Location:** Room 211A

Acknowledgments

Dr. Davide Rossetti
Dr. Sreeram Potluri

Filippo Spiga and Stuart Rankin,
HPCS, University of Cambridge

(Wilkes Cluster)

Thank You!

panda@cse.ohio-state.edu, hamidouche@cse.ohio-state.edu

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

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