MVAPICH2-GDR: Pushing the Frontier of HPC and Deep Learning

GPU Technology Conference GTC 2017

by

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

Hari Subramoni
The Ohio State University
E-mail: subramon@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~subramon
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, Omni-Path, Ethernet/iWARP, and RDMA over Converged Ethernet (RoCE)
  - MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.0), Started in 2001, First version available in 2002
  - MVAPICH2-X (MPI + PGAS), Available since 2011
  - Support for GPGPUs (MVAPICH2-GDR) and MIC (MVAPICH2-MIC), Available since 2014
  - Support for Virtualization (MVAPICH2-Virt), Available since 2015
  - Support for Energy-Awareness (MVAPICH2-EA), Available since 2015
  - Support for InfiniBand Network Analysis and Monitoring (OSU INAM) since 2015
  - Used by more than 2,750 organizations in 84 countries
  - More than 416,000 (> 0.4 million) downloads from the OSU site directly
  - Empowering many TOP500 clusters (Nov ‘16 ranking)
    - 1st, 10,649,600-core (Sunway TaihuLight) at National Supercomputing Center in Wuxi, China
    - 13th, 241,108-core (Pleiades) at NASA
    - 17th, 462,462-core (Stampede) at TACC
    - 40th, 74,520-core (Tsubame 2.5) at Tokyo Institute of Technology
  - Available with software stacks of many vendors and Linux Distros (RedHat and SuSE)
  - http://mvapich.cse.ohio-state.edu

- Empowering Top500 systems for over a decade
  - System-X from Virginia Tech (3rd in Nov 2003, 2,200 processors, 12.25 TFlops) ->
  - Sunway TaihuLight (1st in Jun’16, 10M cores, 100 PFlops)
MVAPICH2 Release Timeline and Downloads
# MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>Requirements</th>
<th>MVAPICH2 Library to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2</td>
</tr>
<tr>
<td>Advanced MPI, OSU INAM, PGAS and MPI+PGAS with IB and RoCE</td>
<td>MVAPICH2-X</td>
</tr>
<tr>
<td>MPI with IB &amp; GPU</td>
<td>MVAPICH2-GDR</td>
</tr>
<tr>
<td>MPI with IB &amp; MIC</td>
<td>MVAPICH2-MIC</td>
</tr>
<tr>
<td>HPC Cloud with MPI &amp; IB</td>
<td>MVAPICH2-Virt</td>
</tr>
<tr>
<td>Energy-aware MPI with IB, iWARP and RoCE</td>
<td>MVAPICH2-EA</td>
</tr>
</tbody>
</table>
Architecture of MVAPICH2 Software Family

High Performance Parallel Programming Models

- Message Passing Interface (MPI)
- PGAS (UPC, OpenSHMEM, CAF, UPC++)
- Hybrid --- MPI + X (MPI + PGAS + OpenMP/Cilk)

High Performance and Scalable Communication Runtime

Diverse APIs and Mechanisms

- Point-to-point Primitives
- Collectives Algorithms
- Job Startup
- Energy-Awareness
- Remote Memory Access
- I/O and File Systems
- Fault Tolerance
- Virtualization
- Active Messages
- Introspection & Analysis

Support for Modern Networking Technology
(InfiniBand, iWARP, RoCE, Omni-Path)

- Transport Protocols: RC, XRC, UD, DC
- Modern Features: UMR, ODP, SR-IOV, Multi Rail

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

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

* Upcoming

Network Based Computing Laboratory
GTC 2017
Optimizing MPI Data Movement on GPU Clusters

- Connected as PCIe devices – Flexibility but Complexity

1. Intra-GPU
2. Intra-Socket GPU-GPU
3. Inter-Socket GPU-GPU
4. Inter-Node GPU-GPU
5. Intra-Socket GPU-Host
6. Inter-Socket GPU-Host
7. Inter-Node GPU-Host

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 1.8-2.3 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
Installing MVAPICH2-GDR

• MVAPICH2-2.2 with GDR support can be downloaded from
  
  https://mvapich.cse.ohio-state.edu/download/mvapich2gdr/

• Please select the best matching package for your system
  
  • We have most of common combinations. If you do not find your match here please email OSU with the following details
    – OS versions
    – OFED version
    – CUDA Version
    – Compiler (GCC, Intel and PGI)

• Install instructions
  
  • Having root permissions
    • On default Path: rpm -Uvh --nodeps mvapich2-gdr-cuda7.0-gnu-2.2-0.3.el6.x86_64.rpm
    • On specific Path: rpm --prefix /custom/install/prefix -Uvh --nodeps mvapich2-gdr-cuda7.0-gnu-2.2-0.3.el6.x86_64.rpm
  
  • Do not have root permissions:
    • rpm2cpio mvapich2-gdr-cuda7.0-gnu-2.2-0.3.el6.x86_64.rpm | cpio –id

• More details on the installation process refer to:
  
  http://mvapich.cse.ohio-state.edu/userguide/gdr/2.2#_installing_mvapich2_gdr_library
Performance of MVAPICH2-GPU with GPU-Direct RDMA (GDR)

GPU-GPU internode latency

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

Latency (us)
Message Size (bytes)

2.18us
3X
10X

GPU-GPU Internode Bandwidth

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

Bandwidth (MB/s)
Message Size (bytes)

11X
2X

GPU-GPU Internode Bi-Bandwidth

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

Bi-Bandwidth (MB/s)
Message Size (bytes)

11X
2X

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

Small Message Latency

Latency (us) vs Message Size (bytes)

MVAPICH2-GDR-2.2
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

2.88 us

6X
Performance of MVAPICH2-GDR with GPU-Direct RDMA and Multi-Rail Support

GPU-GPU Internode MPI Uni-Directional Bandwidth

<table>
<thead>
<tr>
<th>Message Size (bytes)</th>
<th>Bandwidth (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MV2-GDR 2.1</td>
</tr>
<tr>
<td></td>
<td>MV2-GDR 2.1 RC2</td>
</tr>
</tbody>
</table>

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

GPU-GPU Internode Bi-directional Bandwidth

<table>
<thead>
<tr>
<th>Message Size (bytes)</th>
<th>Bi-Bandwidth (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MV2-GDR 2.1</td>
</tr>
<tr>
<td></td>
<td>MV2-GDR 2.1 RC2</td>
</tr>
</tbody>
</table>

MV2-GDR 2.1: 8,759 MB/s at 40%
MV2-GDR 2.1 RC2: 15,111 MB/s at 20%
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


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

- 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 support in MVAPICH2

- Datatypes support in MPI
  - Operate on customized datatypes to improve productivity
  - Enable MPI library to optimize non-contiguous data

At Sender:

```
MPI_Type_vector (n_blocks, n_elements, stride, old_type, &new_type);
MPI_Type_commit(&new_type);
...
MPI_Send(s_buf, size, new_type, dest, tag, MPI_COMM_WORLD);
```

- Inside MVAPICH2
  - Use datatype specific CUDA Kernels to pack data in chunks
  - Efficiently move data between nodes using RDMA
  - In progress - currently optimizes vector and hindexed datatypes
  - Transparent to the user

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

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


- 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/
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 since MVAPICH2-GDR 2.2b)
- OSU Micro-benchmarks extended to evaluate the performance of point-to-point and collective communications using managed buffers

D. S. Banerjee, K Hamidouche, and D. K Panda, Designing High Performance Communication Runtime for GPUManaged Memory: Early Experiences, GPGPU-9 Workshop, held in conjunction with PPoPP ’16
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
- Available since MVAPICH2-GDR 2.2RC1
Enhanced Support for Inter-node Managed Memory

- Enhance inter-node managed memory to use GDR
- SGL-based scheme:
  - Eager Pre-allocated GPU VBUFs
  - Register the GPU vbufs with the HCA
  - Scatter-Gather List to combine control and data messages
  - Brings GDR performance to Managed memory
  - High performance and high productivity
  - Up to 25% improvement for small and medium messages (micro-benchmark)
  - Up to 1.92X improvement for GPU-LBM application
- Will be available in a future MVAPICH2-GDR 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
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
- Available since 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
Latency oriented: Send+kernel and Recv+kernel

- 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
Point-to-point: Hide the kernel launch overhead

- 30% improvement at 16K Bytes compared to traditional MPI+CUDA programs

Collective: Overlap computations with GPU communications (GDS)

- 36% improvement at 64K Bytes compared to traditional MPI+CUDA programs

Intel Broadwell CPU, NVIDIA K80 and Mellanox EDR 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
Deep Learning Frameworks

- Many Deep Learning (DL) frameworks have emerged
  - Berkeley Caffe
  - Google TensorFlow
  - Microsoft CNTK
  - Facebook Torch
  - Facebook Caffe2

- Broadly, DL Frameworks are being developed along two directions
  1. The HPC Eco-system: MPI-based Deep Learning
  2. Enterprise Eco-system: BigData-based Deep Learning

Parallel Training: Scale-up and Scale-out

- **Scale-up**: Intra-node Performance
  - Many improvements like:
    - NVIDIA cuDNN, cuBLAS, NCCL, etc.

- **Scale-out**: Inter-node Performance
  - DL frameworks - single-node only
  - Distributed (Parallel) Training – an emerging trend
    - S-Caffe or OSU-Caffe – MPI-based
    - Microsoft CNTK – MPI-based
    - Google TensorFlow – gRPC-based
    - Facebook Caffe2 – Hybrid
How to efficiently scale-out a Deep Learning (DL) framework and take advantage of heterogeneous High Performance Computing (HPC) resources?
New Challenges for MPI Runtimes

- Deep Learning frameworks are a different game altogether
  - Unusually large message sizes (order of megabytes)
  - Most communication based on GPU buffers
- How to address these newer requirements?
  - GPU-specific Communication Libraries (NCCL)
    - NVidia's NCCL library provides inter-GPU communication
  - CUDA-Aware MPI (MVAPICH2-GDR)
    - Provides support for GPU-based communication
- Can we exploit CUDA-Aware MPI and NCCL to support Deep Learning applications?
Efficient Broadcast: MVAPICH2-GDR and NCCL

- NCCL has some limitations
  - Only works for a single node, thus, no scale-out on multiple nodes
  - Degradation across IOH (socket) for scale-up (within a node)
- We propose optimized MPI_Bcast
  - Communication of very large GPU buffers (order of megabytes)
  - Scale-out on large number of dense multi-GPU nodes
- Hierarchical Communication that efficiently exploits:
  - CUDA-Aware MPI_Bcast in MV2-GDR
  - NCCL Broadcast primitive

Large Message Optimized Collectives for Deep Learning

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

- Benefits and Weaknesses
  - Multi-GPU Training within a single node
  - Performance degradation for GPUs across different sockets
  - Limited Scale-out
- OSU-Caffe: MPI-based Parallel Training
  - Enable Scale-up (within a node) and Scale-out (across multi-GPU nodes)
  - network on ImageNet dataset

GoogLeNet (ImageNet) on 128 GPUs

OSU-Caffe is publicly available from: http://hidl.cse.ohio-state.edu

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

```c
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);
```

```c
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 and RoCE 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/RoCE 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

- **S7324 - Bringing NVIDIA GPUs to the PGAS/OpenSHMEM World: Challenges and Solutions**
  - **Day:** Today, 05/11
  - **Time:** 15:00 - 15:25
  - **Location:** Room 211B
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
subramon@cse.ohio-state.edu

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

MVAPICH
MPI, PGAS and Hybrid MPI+PGAS Library

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

HiDL
High-Performance Deep Learning

The High-Performance Deep Learning Project
http://hidl.cse.ohio-state.edu/