Exploiting InfiniBand and GPUDirect Technology for High Performance Collectives on GPU Clusters

Ching-Hsiang Chu

chu.368@osu.edu

Department of Computer Science and Engineering
The Ohio State University
Outline

• Introduction

• Advanced Designs in MVAPICH2-GDR
  – CUDA-Aware MPI_Bcast
  – CUDA-Aware MPI_Allreduce / MPI_Reduce

• Concluding Remarks
Drivers of Modern HPC Cluster Architectures - Hardware

- Multi-core/many-core technologies
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
- Solid State Drives (SSDs), NVM, Parallel Filesystems, Object Storage Clusters
- Accelerators (NVIDIA GPGPUs and Intel Xeon Phi)

Multi-/Many-core Processors

High Performance Interconnects – InfiniBand (with SR-IOV) 
<1usec latency, 200Gbps Bandwidth>

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

SSD, NVMe-SSD, NVRAM

Sierra@LLNL

Stampede2@TACC

Comet@SDSC
Architectures for Deep Learning (DL)

- **Multi-core CPUs within a node**
  - Multi-core CPUs + Multi-GPU within a node
  - (E.g., Sierra/Summit)

- **Multi-core CPUs across nodes**
  - Multi-core CPUs + Multi-GPU across nodes

- **Multi-core CPUs + Single GPU across nodes**

**Networks**

IB Networks
Streaming-like Applications

- Streaming-like applications on HPC systems

1. Communication (MPI)
   - Broadcast
   - Allreduce/Reduce

2. Computation (CUDA)
   - Multiple GPU nodes as workers
High-performance Deep Learning

• Computation using **GPU**

• Communication using **MPI**
  – Exchanging partial gradients after each minibatch
  – All-to-all (Multi-Source) communications
    - E.g., MPI_Bcast, MPI_Allreduce

• Challenges
  – High computation-communication **overlap**
  – Good **scalability** for upcoming large-scale GPU clusters
  – No application-level modification
Outline

• Introduction

• Advanced Designs in MVAPICH2-GDR
  – CUDA-Aware MPI_Bcast
  – CUDA-Aware MPI_Allreduce / MPI_Reduce

• Concluding Remarks
Hardware Multicast-based Broadcast

- For GPU-resident data, using
  - GPUDirect RDMA (GDR)
  - InfiniBand Hardware Multicast (IB-MCAST)

- Overhead
  - IB UD limit
  - GDR limit

Hardware Multicast-based Broadcast (con’t)

- Heterogeneous Broadcast for streaming applications
  - Free-up PCIe resources

Multicast steps
IB SL step

Optimized Broadcast Send

• Preparing Intermediate buffer \((im\_buf)\)
  – Page-locked (pinned) host buffer
    ➢ Fast Device-Host data movement
  – Allocated at initialization phase
    ➢ Low overhead, one time effort

• Streaming data through host
  – Fine-tuned chunked data
  – Asynchronous copy operations
    ➢ Three-stage fine-tuned pipeline

Optimized Broadcast Receive

- Zero-copy broadcast receive
  - Pre-posted user buffer \((d_{in})\)
  - Avoids additional data movement
  - Leverages IB Scatter and GDR features
  - Low-latency
  - Free-up PCIe resources for applications

\[
\text{MPI\_Bcast}(d_{in},...)\]

Broadcast on Multi-GPU systems

- Proposed Intra-node Topology-Aware Broadcast
  - CUDA InterProcess Communication (IPC)

Benchmark Evaluation

- @ RI2 cluster, 16 GPUs, 1 GPU/node

- Provide near-constant latency over the system sizes
- Reduces up to 65% of latency for large messages

Lower is better

Streaming Workload @ RI2 (16 GPUs) & CSCS (88 GPUs)

- IB-MCAST + GDR + IPC-based MPI_Bcast schemes
  - Stable high throughput compared to existing schemes

Performance Benefits with CNTK Deep Learning Framework @ RI2 cluster, 16 GPUs

- CUDA-Aware Microsoft Cognitive Toolkit (CA-CNTK) without modification

### CA-CNTK - Image Classification

<table>
<thead>
<tr>
<th>Model</th>
<th>CA-CNTK</th>
<th>Knomial-GDR</th>
<th>Ring-GDR-Pipeling</th>
<th>Zcpy-MCAST-GDR-Pipeline</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>0.88</td>
<td>0.5</td>
<td>1.0</td>
<td>0.8</td>
</tr>
<tr>
<td>VGG</td>
<td>0.9</td>
<td>0.45</td>
<td>1.15</td>
<td>0.95</td>
</tr>
<tr>
<td>ResNet-50</td>
<td>0.88</td>
<td>0.45</td>
<td>1.15</td>
<td>0.95</td>
</tr>
</tbody>
</table>

#### Speedup vs Scale (Number of GPU nodes)

- Reduces up to 24%, 15%, 18% of latency for AlexNet, VGG, and ResNet-50 models
- Higher improvement is expected for larger system sizes

---

CUDA-Aware MPI_Allreduce

• **Existing designs**
  1. Explicit copy the data from GPU to host memory
  2. Host-to-Host communication to remote processes
  3. Perform computation on CPU
  4. Explicit copy the data from host to GPU memory

• **Proposed designs**
  1. GPU-to-GPU communication
    - **NVIDIA GPUDirect RDMA (GDR)**
    - Pipeline through host for large msg
  2. Perform computation on GPU
    - Efficient CUDA kernels

![Diagram showing network-based computing with CUDA-Aware MPI_Allreduce](network.png)

- **Expensive!**
- **Fast**
- **Good for small data**
- **Relative slow for large data**
Benchmark Evaluation @ RI2 cluster, 16 GPUs


Outline

• Introduction

• Advanced Designs in MVAPICH2-GDR
  – CUDA-Aware MPI_Bcast
  – CUDA-Aware MPI_Allreduce / MPI_Reduce

• Concluding Remarks
Concluding Remarks

• High-performance broadcast schemes to leverage GDR and IB-MCAST features for streaming and deep learning applications
  – Optimized streaming design for large messages transfers
  – High-performance reliability support for IB-MCAST

• High-performance CUDA-Aware Allreduce for deep learning
  – Efficient reduction kernel on GPUs

➤ These features are included in MVAPICHER2-GDR 2.3
  ➢ http://mvapich.cse.ohio-state.edu/
  ➢ http://mvapich.cse.ohio-state.edu/userguide/gdr/2.3/
Thank You!

- Join us for more tech talks from MVAPICH2 team
  - [http://mvapich.cse.ohio-state.edu/talks/](http://mvapich.cse.ohio-state.edu/talks/)