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)
Streaming-like Applications

- Streaming-like applications on HPC systems

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

2. Computation (CUDA)
   - Multiple GPU nodes as workers

Real-time streaming

HPC resources for real-time analytics

Data streaming-like broadcast operations

Worker
- CPU
- GPU
- GPU

Worker
- CPU
- GPU
- GPU

Worker
- CPU
- GPU
- GPU

Worker
- CPU
- GPU
- GPU

Worker
- CPU
- GPU
- GPU
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

• Heterogeneous Broadcast for streaming applications

➢ Free-up PCIe resources

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

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

Efficient Reliability Support for IB-MCAST

- When a receiver experiences timeout (lost MCAST packet)
  - Performs the RMA Get operation to the sender’s backup buffer to retrieve lost MCAST packets
  - **Sender is not interrupted**
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

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

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

Deep Learning Frameworks @ RI2 cluster, 16 GPUs

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

**CA-CNTK - Image Classification**

<table>
<thead>
<tr>
<th></th>
<th>Knomial-GDR</th>
<th>Ring-GDR-Pipelining</th>
<th>Zcpy-MCAST-GDR-Pipeline</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 AlexNet</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 AlexNet</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 VGG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 VGG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 ResNet-50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 ResNet-50</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 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
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 MVAPICH2-GDR 2.3
  - [http://mvapich.cse.ohio-state.edu/](http://mvapich.cse.ohio-state.edu/)
  - [http://mvapich.cse.ohio-state.edu/userguide/gdr/2.3/](http://mvapich.cse.ohio-state.edu/userguide/gdr/2.3/)
Thank You!

Ching-Hsiang Chu

chu.368@osu.edu

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

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


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/)