GPU-Aware Design, Implementation, and Evaluation of Non-blocking Collective Benchmarks

Presented By: Esthela Gallardo

Ammar Ahmad Awan, Khaled Hamidouche, Akshay Venkatesh, Jonathan Perkins, Hari Subramoni, and Dhabaleswar K. Panda

Network-Based Computing Laboratory
Department of Computer Science and Engineering
The Ohio State University, Columbus, OH, U.S.A
Outline

• Introduction
  – Non-Blocking Collectives
  – GPU-Aware MPI
• Research Challenges
• Existing Benchmark Suites
• Contributions
  – GPU-Aware Benchmark
  – Design and Implementation
• Performance Comparison
• Conclusion and Future Work
Two important trends can be observed
- A lot of emphasis on overlapping computation with communication.
- Ever increased focus on heterogeneity in HPC architectures*

Both are considered important and emerging strategies for the march towards Exa-scale

The No. 2 system, Titan, and the No. 6 system, Piz Daint, use NVIDIA GPUs to accelerate computation. -- www.top500.org
Collective Communication

- Important and widely used in MPI programs
- Primitives available in the MPI standard
  - Reduce, Broadcast, Scatter, Gather, Barrier etc.
- Collectives have been blocking
  - The context remains in the library until completion

Images from: www.mpitutorial.com
Non-Blocking Collectives (NBC)

• Have been used since 2007. Recently, made part of the MPI-3 standard

• The focus is on overlapping computation with communication

• NBC performance is good *
  – Latency is good with acceptable overhead posed by NBC operations.
  – Overlap is the new parameter – maximizing it enables independent computation to proceed in background

• 56 out of 500 top HPC systems use Nvidia GPUs
  – http://www.top500.org - June 2015 latest list

• Scientists have been writing applications with Message Passing Interface (MPI) and CUDA API

• GPU-Aware (CUDA-Aware) MPI libraries are high performance / high productivity tools for application programmers
  – MVAPICH2 – pioneered the concept of GPU-Aware MPI libraries
  – Other MPI libraries also have GPU-Aware support
    • OpenMPI, Platform MPI, Cray MPI

Source: www.top500.org

Accommodation provided by The Ohio State University
GPU-Aware MPI + NBC

- GPU-Aware MPI
  - The send and recv buffers in the collective call reside in GPU memory
  - This is transparent to the application developer
  - The MPI implementation’s runtime handles the allocation of respective buffers on CPU and GPU

- GPU-Aware NBC implementations available for some NBC operations
  - MPI_Ialltoall
  - MPI_Iallreduce
  - MPI_Iallgather
  - MPI_Ibcast

EuroMPI 2015
• Can we develop a standard benchmark that evaluates performance of different GPU-Aware NBC implementations?

• Can we identify new and meaningful parameters like
  – overlap
  – time for initiating an NBC operation
  – time for MPI_Wait and MPI_Test
  – effect of dummy GPU-CPU copies
  – effect of independent computation on CPU, GPU, and Both
    for getting a holistic performance perspective instead of latency numbers only)

• Can we provide the flexibility to the user of the benchmark to select evaluation parameters according to the needs and scale?

• Can we compare well-known and widely used MPI libraries that have a GPU-Aware NBC implementation?
• Intel MPI Benchmark (IMB) has non-blocking collective (NBC) benchmarks

• OSU Micro-Benchmark Suite (OMB) has GPU-Aware benchmarks for blocking collectives

• Natural extension is to introduce GPU-Aware NBC benchmarks
## State-of-the-art vs. Proposed

<table>
<thead>
<tr>
<th>Benchmarks/Features</th>
<th>Pt-to-Pt, One-Sided</th>
<th>Blocking Collectives</th>
<th>Non-Blocking Collectives</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Host-based</td>
<td>GPU-Aware</td>
<td>Host-based</td>
</tr>
<tr>
<td>IMB [16]</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>COMB [19]</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>SMB [20]</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>NBCBench [15]</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>OMB [8]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>OMB (w/ Proposed GPU-NBC)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

*EuroMPI 2015*
Contributions

- Present the design and implementation of the proposed GPU-Aware Non-Blocking Collective Benchmarks
- Provide useful insights on designing an effective benchmark for GPU-Aware NBC operations by discussing performance metrics like overlap and latency, communication progress mechanisms in MPI libraries, and independent CPU-GPU communication
- Discuss usage and performance effects of different runtime parameters including support for dummy compute on CPU, dummy compute on GPU, and independent CPU/GPU communication
- Illustrate the efficacy of our benchmarks by providing a comprehensive performance comparison of NBC operations in MVAPICH2 and OpenMPI on a GPU cluster
Scenario 1: NBC – CPU only

1st Call to compute/test

MPI_Ibcast (CPU/GPU buffer)

Compute_on_CPU ()

MPI_Test ()

'n'th Call to compute/test

Compute_on_CPU ()

MPI_Test ()

MPI_Wait ()

Overlapped Computation with Communication
Scenario 2: NBC – GPU only

- **1st Call to compute/test**:
  - `Compute_on_GPU()`
  - `MPI_Test()`

- **‘n’th Call to compute/test**:
  - `Compute_on_GPU()`
  - `MPI_Test()`

- Overlapped Computation with Communication

- **MPI_Igather (CPU/GPU buffer)**

- **MPI_Wait ()**
Scenario 3: NBC – CPU and GPU

1\textsuperscript{st} Call to compute/test

‘n’th Call to compute/test

MPI\_Igather (CPU/GPU buffer)

\textbf{Compute\_on\_GPU()}

\textbf{ MPI\_Wait ()}

\textbf{ Compute\_on\_CPU ()}

\textbf{ MPI\_Test ()}

\textbf{ Compute\_on\_GPU ()}

\textbf{ MPI\_Test ()}

Overlapped Computation with Communication

EuroMPI 2015
## Review: Features in Benchmarks

<table>
<thead>
<tr>
<th>Evaluation Parameters</th>
<th>IMB 4</th>
<th>NBC Bench</th>
<th>Proposed (GPU-NBC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overlap</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Latency</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>MPI_Test</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>MPI_Wait</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Coll. Init</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Dummy Compute (CPU)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Dummy Compute (GPU)</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Dummy Copy (GPU)</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
</tbody>
</table>

---

EuroMPI 2015
• To highlight the efficacy of our proposed benchmarks, we have evaluated two widely used MPI libraries; MVAPICH2 and OpenMPI
  – Both have GPU-Aware NBC implementations for some of the collectives

• We evaluate for all the parameters we have discussed so far.
• High Performance open-source MPI Library for InfiniBand, 10Gig/iWARP, and RoCE
  – MVAPICH (MPI-1), Available since 2002
  – MVAPICH2 (MPI-2.2, MPI-3.0 and MPI-3.1), Available since 2004
  – MVAPICH2-X (Advanced MPI + PGAS), Available since 2012
  – Support for GPGPUs (MVAPICH2-GDR), Available since 2014
  – Support for MIC (MVAPICH2-MIC), Available since 2014
  – Support for Virtualization (MVAPICH2-Virt), Available since 2015
  – Support for Energy-Aware MPI communications (MVAPICH2-EA), available since 2015
  – Used by more than 2,450 organizations in 76 countries
  – More than 285,000 downloads from the OSU site directly
  – Empowering many TOP500 clusters (Jun’15 ranking)
    • 8th ranked 519,640-core cluster (Stampede) at TACC
    • 11th ranked 185,344-core cluster (Pleides) at NASA
    • 22nd ranked 76,032-core cluster (Tsubame 2.5) at Tokyo Institute of Technology 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](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)
  – Stampede at TACC (8th in Jun’15, 462,462 cores, 5.168 Pflops)
Available from our website

- http://mvapich.cse.ohio-state.edu/benchmarks/
- Widely used benchmark for evaluating MPI libraries
- OMB 5.0 released recently has Host-based NBC benchmarks
• We made extensions to the OMB for evaluating NBC operations

• We then added support for evaluating the newly identified parameters for GPU-Aware NBC operations

• These benchmarks will be released publicly with our next MVAPICH-2 GDR release

• Will greatly help in obtaining a holistic view of performance for GPU-Aware NBC implementations
Wilkes cluster, deployed in Nov 2013 at Cambridge, U.K., has been used for the performance evaluation.

The cluster is partitioned with different configurations.

For our purpose we use the Tesla partition which has 128 nodes.

Each node has a 6-core dual-socket Intel IvyBridge processor.

Each node is equipped with 2 Tesla K20 NVIDIA GPUs and 2 FDR IB HCAs.
Some Terminology..

1. **Pure Comm. Latency** - Latency of an NBC when we call the collective immediately followed by MPI_Wait () call

2. **Overall Latency** - Latency of an NBC operation when we call the collective, followed by independent computation and specified number of test calls, followed by MPI_Wait () call

3. **Collective Initialization Time (Coll. Init)** – Time take by a collective init call e.g MPI_Ibcast ()

4. **Compute Time** - Time taken by the dummy compute - independent overlapped computation function (executed on CPU, GPU, and Both)

5. **Test Time** - Time taken by MPI_Test() calls

6. **NBC Overhead** - This is the difference in performance of collective when its Pure Comm. latency is compared with Overall latency
Performance Evaluation

• NBC Overhead Comparison
• Effect of Dummy Copy
• Effect of MPI_Test calls on Latency and Overlap
• Effect of Dummy Compute
  – On CPU
  – On GPU
  – On Both
• 20-30% overhead for small messages for both MVAPICH2 and OpenMPI
As expected, we do not experience overhead in the large message size range for MVAPICH2.
Effect of Dummy Copy - Ibcast

- Data shown from small message range only (little overhead for large messages)
- The dummy copies between CPU and GPU use separate streams so overhead should be minimal
- The overhead is almost constant around 15-20% for both MV2 and OpenMPI in the small message range
Effect of MPI_Test calls : Ibcast

But if used in a wrong message size range, increase in test calls can have negative effects. Latency is increasing here in the small message range.

Large Messages : Reduction - move from 2K to 10K test calls.

But if used in a wrong message size range, increase in test calls can have negative effects. Latency is increasing here in the small message range.
For small message range, latency is minimum with 1000 test calls
For Large messages, latency is best with 10,000 calls
Both outcomes are as expected
Overlap : Compute on GPU - OMPI

There is a drop in overlap - for messages larger than 32K we need 10k test calls. With 1k calls, the overlap drops dramatically.
We can see that the drop in overlap spot moves towards even larger message sizes for **MVAPICH2**

---

**Overlap : Compute on GPU – MV2**

**MVAPICH2 - Small Messages**

**MVAPICH2 - Large Messages**

**EuroMPI 2015**
• Discussed the trends in HPC and highlighted that GPU-Aware NBC operations are emerging

• Elaborated the design space for NBC benchmarks and identified the limitations in existing benchmarks

• Proposed new designs and implemented GPU-Aware NBC benchmarks

• Provided useful insights and new parameters like overlap, time of test calls, time of dummy computations, and effect of GPU dummy copies.
  – Compared MVAPICH2 and OpenMPI
  – Platform MPI and Cray MPI can also be evaluated but we did not have access

• Benchmarks will be made publicly available
Ammar Ahmad Awan, Khaled Hamidouche, Akshay Venkatesh, Jonathan Perkins, Hari Subramoni, and Dhabaleswar K. Panda

{awan.10, hamidouche.2, venkatesh.19, perkins.173, subramoni.1, panda.2} @osu.edu

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

MVAPICH Web Page
http://mvapich.cse.ohio-state.edu/
Latency: Compute on GPU – MV2

**MVAPICH2 - Small Messages**

- MV2-Latency-0
- MV2-Latency-1000
- MV2-Latency-10000

**MVAPICH2 - Large Messages**

- MV2-Latency-0
- MV2-Latency-1000
- MV2-Latency-10000

MVAPICH2 - EuroMPI 2015