Accelerating HPL on Heterogeneous GPU Clusters

Presentation at GTC 2014

by

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

• Introduction and Motivation
• Proposed Design for Hybrid HPL
• Performance Evaluation
• Conclusion and Future Work
High-End Computing (HEC): PetaFlop to ExaFlop

Projected Performance Development

100 PFlops in 2015

223.54 PF/s

33.86 PF/s

1 EFlops in 2018

96.62 TF/s

Expected to have an ExaFlop system in 2019 -2022!
Drivers of Heterogeneous HPC Cluster

- Multi-core processors are ubiquitous
- High Performance Linpack (HPL) is used to measure the peak performance
- NVIDIA GPUs are becoming common in high-end systems
- Pushing the envelope for heterogeneous computing

Multi-core Processors

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

Tianhe – 1A  Stampede  Oakley (OSC)  Blue Waters (NCSA)
• Homogeneous CPU Clusters
  – No GPU accelerators (ex: BlueGene)

• Homogeneous GPU Clusters
  – All nodes have the same configuration
  – Titan, Keeneland, Wilkes

• Heterogeneous GPU Clusters
  – CPU nodes + GPU nodes
  – Ratio CPU/GPU > 1
  – Oakley@OSC: 634 CPU + 64 GPU
  – BlueWaters@NCSA: 22,500 XE + 4200 XK
HPL - High Performance Linpack

- Benchmark
  Performance measure for ranking supercomputers in the top500 list

- Time Complexity: $\frac{2}{3} N^3 + 2N^2 + O(N)$  $N$ is the problem size

  $O(N^3)$: LU Decomposition
  $O(N^2)$: Backward Substitution

- Iterative Procedure of LU
  Factorize the current block
  Broadcast and update the green parts
  Update the yellow parts
Current Execution of HPL on Heterogeneous GPU Clusters

• Current HPL support for GPU Clusters
  – Heterogeneity inside a node CPU+GPU
  – Homogeneity across nodes

• Current HPL execution on heterogeneous GPU Clusters
  – Only CPU nodes (using all the CPU cores)
  – Only GPU nodes (using CPU+GPU on only GPU nodes)
  – As the ratio CPU/GPU is higher => report the “Only CPU” runs

• Need for HPL which supports heterogeneous systems
  – Heterogeneity inside a node (CPU+GPU)
  – Heterogeneity across nodes (nodes w/o GPUs)
Outline

• Introduction and Motivation

• Proposed Design for Hybrid HPL
  – Two-level Workload Partitioning (Inter-node and Intra-node)
  – Process-grid Reordering

• Performance Evaluation

• Conclusion and Future Work
Two Level Workload Partitioning: Inter-node

Inter-node Static Partitioning

Original design: uniform distribution, bottleneck on CPU nodes

New design: identical block size, schedules more MPI processes on GPU nodes

\[
\text{MPI}_\text{GPU} = \frac{\text{ACTUAL\_PEAK\_GPU}}{\text{ACTUAL\_PEAK\_CPU}} + \beta \\
\text{(NUM\_CPU\_CORES mod MPI\_GPU} = 0 \) \\
\text{Evenly split the cores}
\]
Two Level Workload Partitioning: Intra-node

• Intra-node Dynamic Partitioning
  • MPI-to-Device Mapping
    Original design: 1:1
    New design: M: N (M > N), N= number of GPUs/Node, M= number of MPI processes

• Initial Split Ratio Tuning: alpha = GPU_LEN / (GPU_LEN + CPU_LEN)
  Fewer CPU cores per MPI processes
  Overhead caused by scheduling multiple MPI processes on GPU nodes
Process Grid Reordering

- Default (blocking) Process Grid with Multiple MPI processes/GPU nodes

- Synchronization overhead of Panel Broadcast
  - Default Design

- Unbalanced Workload
  G1 might get more blocks than G2
  C1 might get more blocks than C2

- Balanced Workload
  All the parties have the adequate workload
Overview of Hybrid HPL Design

- **Heterogeneity Analysis**
  - Pure CPU nodes
  - Pure GPU nodes
  - Hybrid CPU+GPU nodes

- **Two-level Workload Partitioning**
  - Inter-node Static
  - Intra-node Dynamic

- **Process Grid Reordering**
  - Generate efficient node topology

- **Hybrid Launcher**
  - GPU nodes
    - Asynchronous Memory Copy
    - MPI-Device Mapping
    - Adaptive Split Ratio Tuning
  - CPU nodes

- **Pre-process Analysis**
  - Inter-node Static Partitioning

- **Runtime Execution**
  - Hybrid Launcher
    - Call MKL
    - Call MKL, cuBLAS
    - Intra-node Dynamic Partitioning
    - MPI-Device Mapping (m: n)
Outline

• Introduction and Motivation

• Proposed Design for Hybrid HPL

• Performance Evaluation

• Conclusion and Future Work
Experimental Setup

- **Experiment Environment**

<table>
<thead>
<tr>
<th>Specifications</th>
<th>Cluster A</th>
<th>Oakley Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Processor Type</td>
<td>Intel Xeon E5630 2.53GHz</td>
<td>Intel Xeon X5650 2.66GHz</td>
</tr>
<tr>
<td>CPU Clock</td>
<td>two quad-core sockets 11.6 GB</td>
<td>two 6-core sockets 46 GB</td>
</tr>
<tr>
<td>Node Type</td>
<td>80.96 Gflops</td>
<td>127.68 Gflops</td>
</tr>
<tr>
<td>CPU Memory</td>
<td>NVIDIA Tesla C2050 515 Gflops/GPU</td>
<td>NVIDIA Tesla M2070 515 Gflops/GPU</td>
</tr>
<tr>
<td>CPU Theo.peak (double)</td>
<td>MKL 10.3/cuBLAS</td>
<td>MKL 10.3/cuBLAS</td>
</tr>
<tr>
<td>GPU Processor Type</td>
<td>Intel Compilers 11.1 MVAPICH2 1.9 RHEL 6.1</td>
<td>Intel Compilers 11.1 MVAPICH2 1.9 RHEL 6.3</td>
</tr>
<tr>
<td>GPU Theo.peak (double)</td>
<td>Mellanox IB QDR</td>
<td>Mellanox IB QDR</td>
</tr>
<tr>
<td>BLAS Lib</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compilers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPI Lib</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interconnect</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **MPI Library: MVAPICH2**
  - High Performance open-source MPI Library for InfiniBand, 10Gig/iWARP, and RDMA over Converged Enhanced Ethernet (RoCE)
  - Support for heterogeneous GPU Clusters
  - Used by more than 2,150 organizations (HPC Centers, Industry and Universities) in 72 countries

http://mvapich.cse.ohio-state.edu/
Actual Peak Performance

1G-CONFIG-A: 8 GPU nodes (1 GPU accelerators) + 32 CPU nodes
1G-CONFIG-Oakley: 32 GPU nodes (1 GPU accelerators) + 128 CPU nodes
2G-CONFIG-Oakley: 32 GPU nodes (2 GPU accelerators) + 128 CPU nodes

CPU/GPU ratio = 4

- Hybrid designs outperforms the PURE_CPU by 32%

R. Shi, S. Potluri, K. Hamidouche, X. Lu, K. Tomko and D. K. Panda, A Scalable and Portable Approach to Accelerate Hybrid HPL on Heterogeneous CPU-GPU Clusters, IEEE Cluster (Cluster '13), Best Student Paper Award
Efficiency

- **PPE**: Peak Performance Efficiency
- **TPE**: Theoretical Performance Efficiency

Performance (Gflops)

<table>
<thead>
<tr>
<th>Configuration</th>
<th>THEO_PEAK</th>
<th>ACTUAL_PEAK</th>
<th>OSU-HYBRID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1G-CONFIG</td>
<td>10282</td>
<td>5866</td>
<td>11650</td>
</tr>
<tr>
<td>2G-CONFIG</td>
<td>8172</td>
<td>8633</td>
<td>13600</td>
</tr>
</tbody>
</table>

**PPE = 88%**
**TPE = 63%**

**PPE = 85%**
**TPE = 51%**

THEO_PEAK VS ACTUAL_PEAK VS OSU-HYBRID
Peak Performance Efficiency (PPE) Scalability

- Constant PPE for fixed CPUs, fixed GPUs and fixed ratio
Outline

• Introduction and Motivation
• Proposed Design for Hybrid HPL
• Performance Evaluation
• Conclusion and Future Work
Conclusion

• Propose a novel approach of HPL on GPU clusters with heterogeneity on intra- and inter-nodes configuration.

• Achieve 80% of the combined actual peak performance of pure CPU and pure GPU nodes

• Studying the impact of such designs for other benchmarks and applications.
Thank You!

panda@cse.ohio-state.edu

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

MVAPICH Web Page
http://mvapich.cse.ohio-state.edu/