NETWORK-BASED COMPUTING LABORATORY

A Scalable and Portable Approach to Accelerate Hybrid HPL on Heterogeneous CPU-GPU Clusters

Rong Shi* Sreeram Potluri* Khaled Hamidouche* Xiaoyi Lu*

Karen Tomko+ Dhabaleswar K. Panda*

*Network-Based Computing Laboratory
Department of Computer Science and Engineering
The Ohio State University

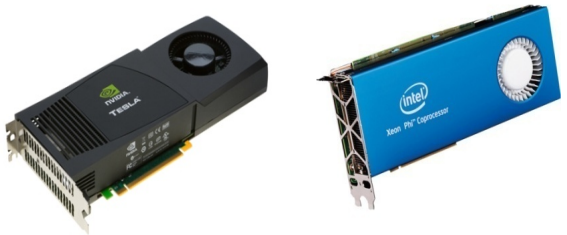
+Ohio SuperComputer Center

Outline

- Introduction
- Motivation & Problem Statement
- Proposed Design for Hybrid HPL
- Performance Evaluation
- Conclusion and Future Work

Drivers of Heterogeneous HPC Cluster

Multi-core Processors



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

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

Tianhe – 1A (10)

Stampede (6)

Oakley (OSC)

Blue Waters (NCSA)

Example of Heterogeneous Cluster

Oakley Cluster (Ohio Supercomputer Center)

https://www.osc.edu/supercomputing

- 8280 CPU cores (690 CPU nodes)
  Linpack Performance (Rmax-CPU): 79.3 Tflops
  Theoretical Peak (Rpeak-CPU): 88.1 Tflops

- One in every 10 nodes have two Nvidia Tesla GPU accelerators (64 GPU nodes)
  Linpack Performance (Rmax-GPU): 33.3 Tflops
  Theoretical Peak (Rpeak-GPU): 74.1 Tflops

Existing Work

HPL Version	Target Cluster	Open source	Multi-thread Support	GPU Programming
UTK Netlib's HPL	CPUs	Y	N	N
Intel's HPL	CPUs	N	Y	N
Frankfurt's HPL	AMD GPUs	Y	Y	OpenCL
NVIDIA's HPL	NVIDIA GPUs	Y	Y	CUDA
Endo's HPL	CPUs + GPUs	N	Y	CUDA

- Limitation of Endo's work
  - Dedicate one CPU core per MPI process for communication
  - Based on standard HPL and optimized for TSUBAME supercomputer
  - No parallelism of DTRSM

Outline

- Introduction
- Motivation & Problem Statement
- Proposed Design for Hybrid HPL
- Performance Evaluation
- Conclusion and Future Work

Motivation

- Current limitation

  Report the peak performance of only a subset of the compute resources

  In the absence of Hybrid version of CPU-GPU HPL, the full potential of a large number of GPU clusters are not being reported in the TOP500 list.

- Goal

  Design and implement a scalable and portable hybrid HPL benchmark for general heterogeneous clusters

Problem Statement

- Can we design the hybrid benchmark to measure the overall computation capacity of these heterogeneous Clusters ?

- Can our design provide these features?
  Performance (fully utilize all available CPU and GPU resources)
  Load balancing (considering different computation capacity)
  Minimize communication overhead

- Can the performance of hybrid HPL beat the performance of either pure CPU nodes or pure GPU nodes?

Outline

- Introduction
- Motivation & Problem Statement
- Proposed Design for Hybrid HPL
- Performance Evaluation
- Conclusion and Future Work

High Performance Linpack

- Benchmark

  Performance measure for ranking supercomputers in the top500 list

- Time Complexity: N is the problem size

  : LU Decomposition

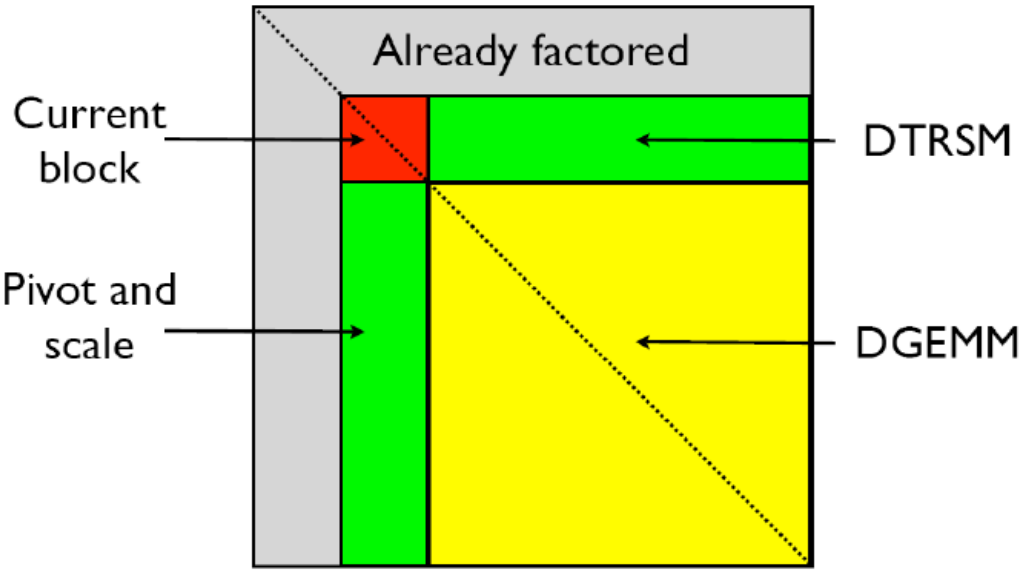
  : Backward Substitution

- Iterative Procedure of LU

  Factorize the current block

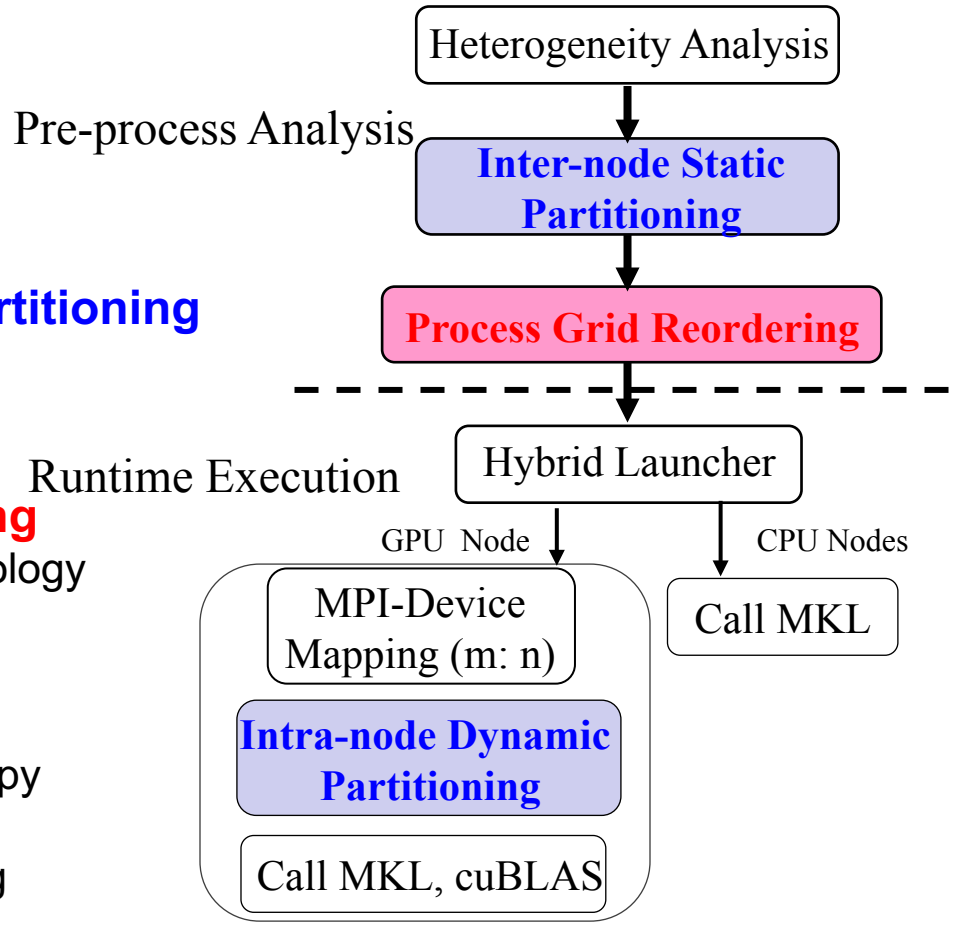
  Broadcast and update the green parts

  Update the yellow parts

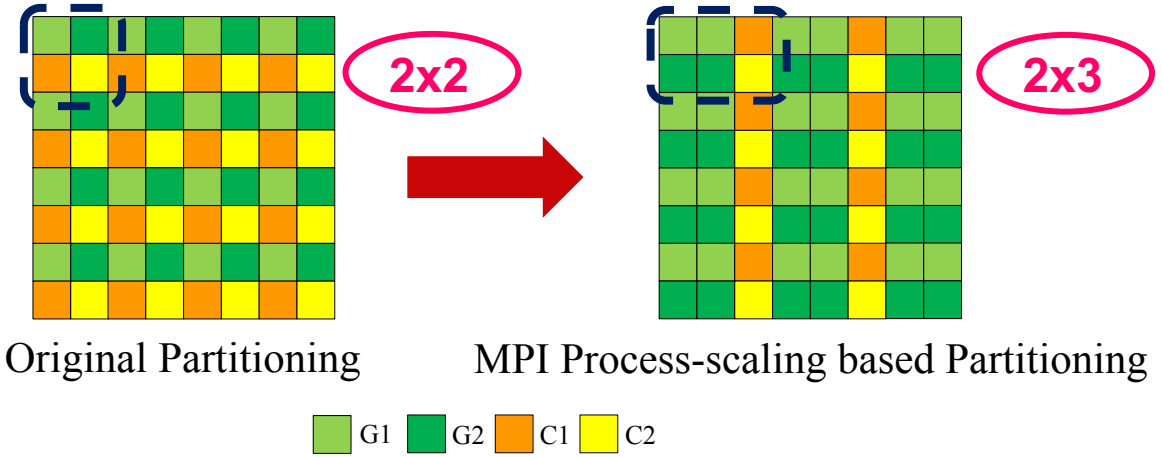


Overview of Hybrid HPL Design

- **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



Two Level Workload Partitioning



- **Inter-node Static Partitioning**

  Original design: uniform distribution, bottleneck on CPU nodes

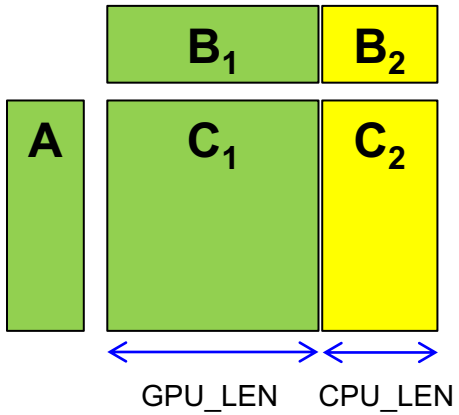
  New design: identical block size, schedules more MPI processes on GPU nodes

  MPI_GPU = ACTUAL_PEAK_GPU / ACTUAL_PEAK_CPU + β

  (NUM_CPU_CORES **mod** MPI_GPU = 0)

  Evenly split the cores

Two Level Workload Partitioning



- **Intra-node Dynamic Partitioning**

  MPI-to-Device Mapping

  Original design: 1:1

  New design: M: N (M > N)

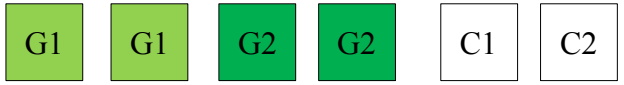
  Initial Split Ratio Tuning: alpha = GPU_LEN / (GPU_LEN + CPU_LEN)

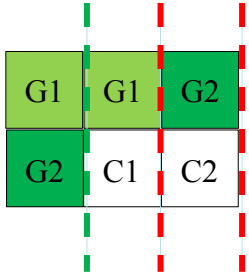
  Fewer CPU cores per MPI processes

  Overhead caused by scheduling multiple MPI processes on GPU nodes

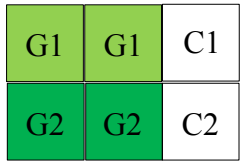
Process Grid Reordering

- Default Process Grid





2 x 3 Grid



- Synchronization overhead of Panel Broadcast

  G1 → G1 → G2 G1 → G1 → C1

  G2 → C1 → C2 G2 → G2 → C2

- Unbalanced Workload

  G1 might get more blocks than G2

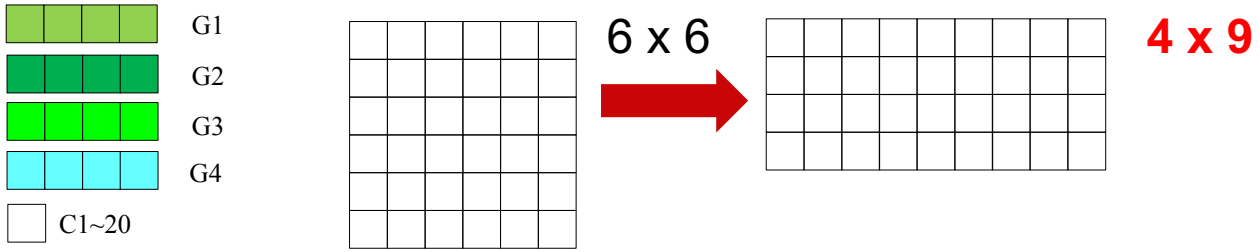
  C1 might get more blocks than C2

Process Grid Reordering

- Optimized Process Grid

  Calculate Parameters: mpi_gpu, total_num_mpi, choose initial process grid

  Strategy 1: Adjust P x Q grid (4 GPUs + 20 CPUs, mpi_g=4)



  Strategy 2: Adjust MPI_GPU (3 GPUs + 18 CPUs, **mpi_g=6→3**)

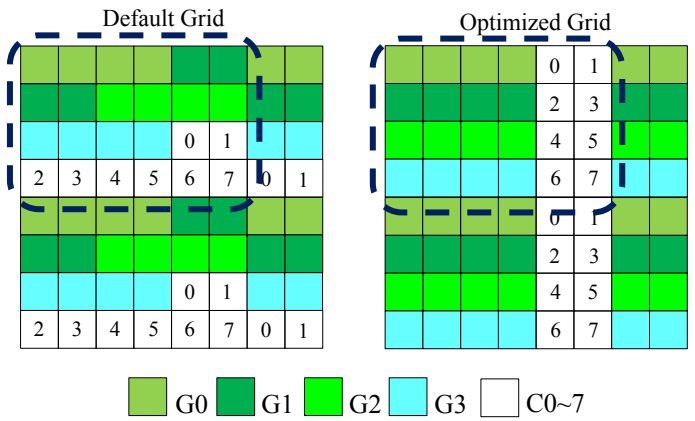
Process Grid Reordering

- Examples

  4 GPU nodes + 8 CPU nodes

  4 x 6 Process grid, mpi_gpu = 4

  N = 81,920, NB=512



Process Grid	Number of Blocks				
	g0,3	g1,2	c0,1,6,7	c2,3,4,5	Total
Default	4320	4240	1040	1080	25600
Optimized	4320		1040		25600

Better load balancing across CPU and GPU nodes

Process Grid	Total	Max rfact	Max bcast	Max update
Default	267.6	17.5	82.8	229.3
Optimized	245.6	12.4	72.1	222.8

Take advantage of shared memory for panel broadcast

Outline

- Introduction
- Motivation & Problem Statement
- Proposed Design for Hybrid HPL
- Performance Evaluation
- Conclusion and Future Work

Experimental Setup

- Experiment Environment

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

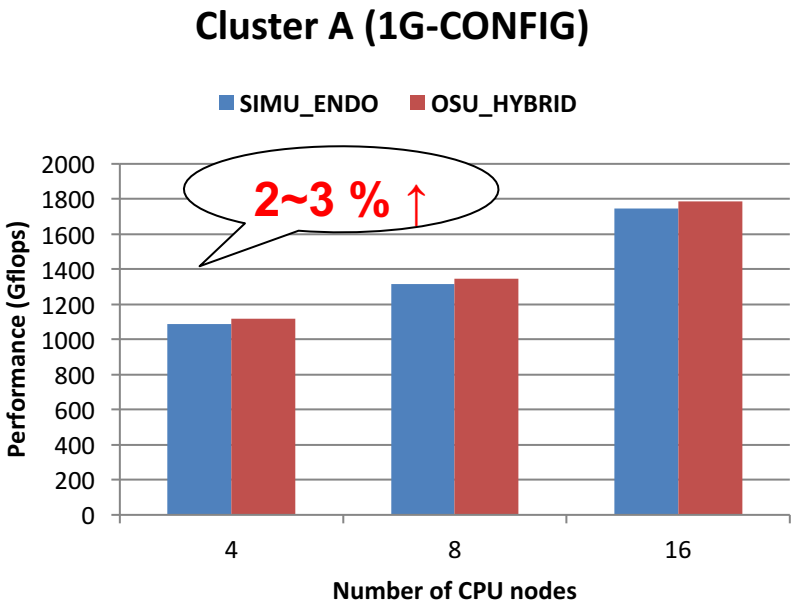
- MPI Library: MVAPICH2

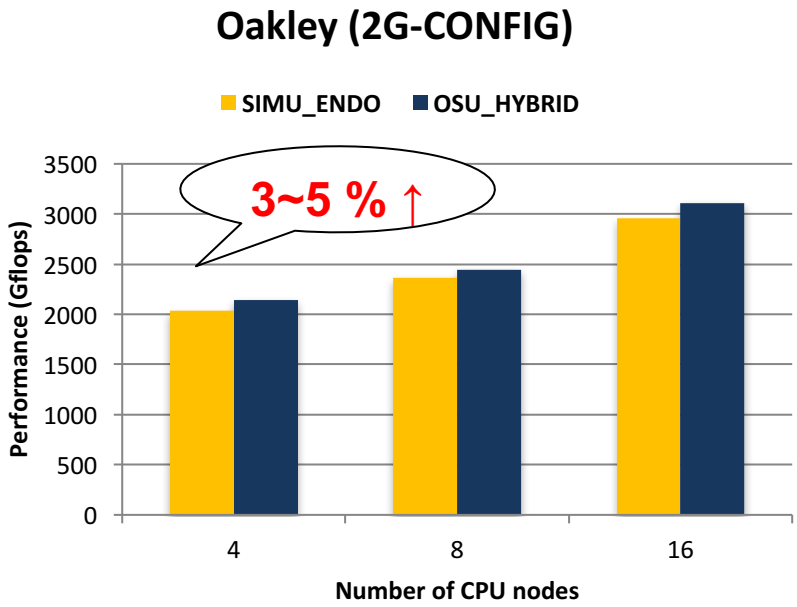
High Performance open-source MPI Library for InfiniBand, 10Gig/iWARP, and RDMA over Converged Enhanced Ethernet (RoCE)

Used by more than 2,077 organizations (HPC Centers, Industry and Universities) in 70 countries

http://mvapich.cse.ohio-state.edu/

Performance of Parallel DTRSM within GPU Nodes





4 GPU nodes with increasing number of CPU nodes

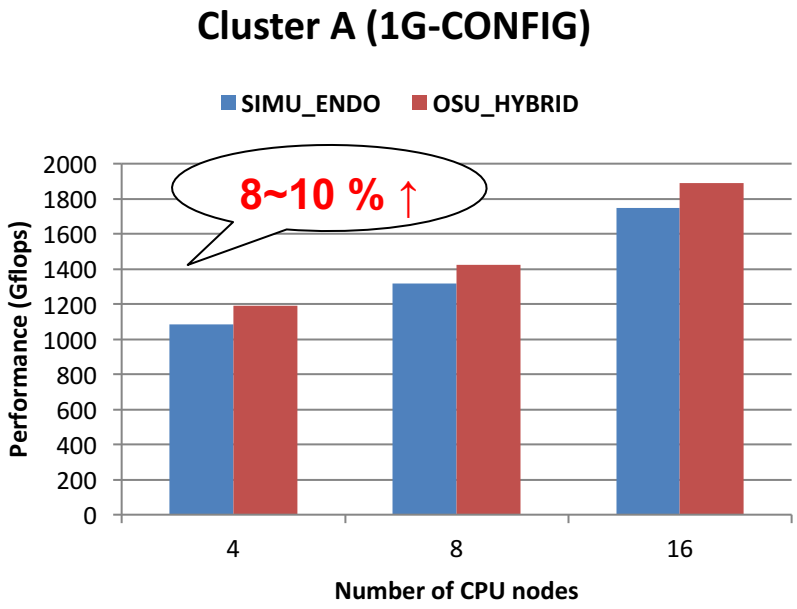
SIMU_ENDO: simulation of Endo's work

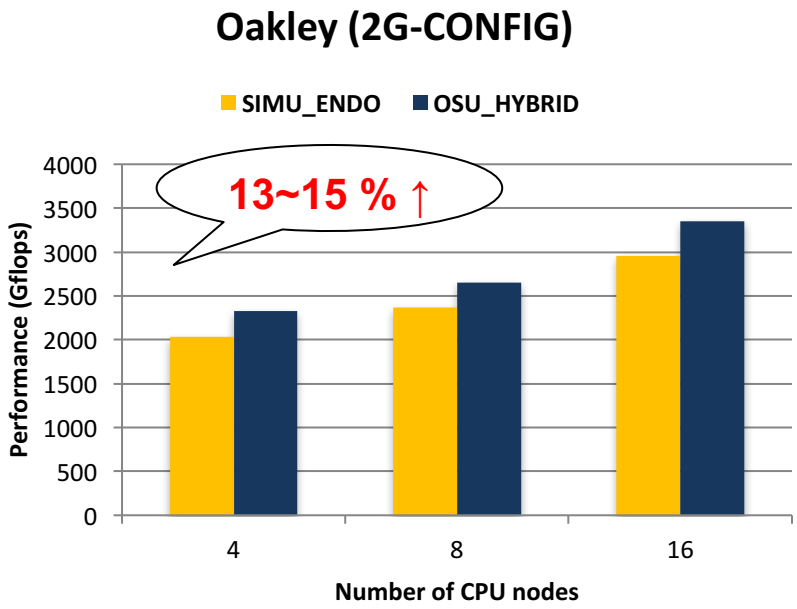
OSU_HYBRID: our design

1G-CONFIG & 2G-CONFIG: each GPU node has one or two GPU accelerators respectively

- **Parallel DTRSM brings 2~3% and 3~5% performance gain on Cluster A and Oakley Cluster**

Performance with Full Core Utilization

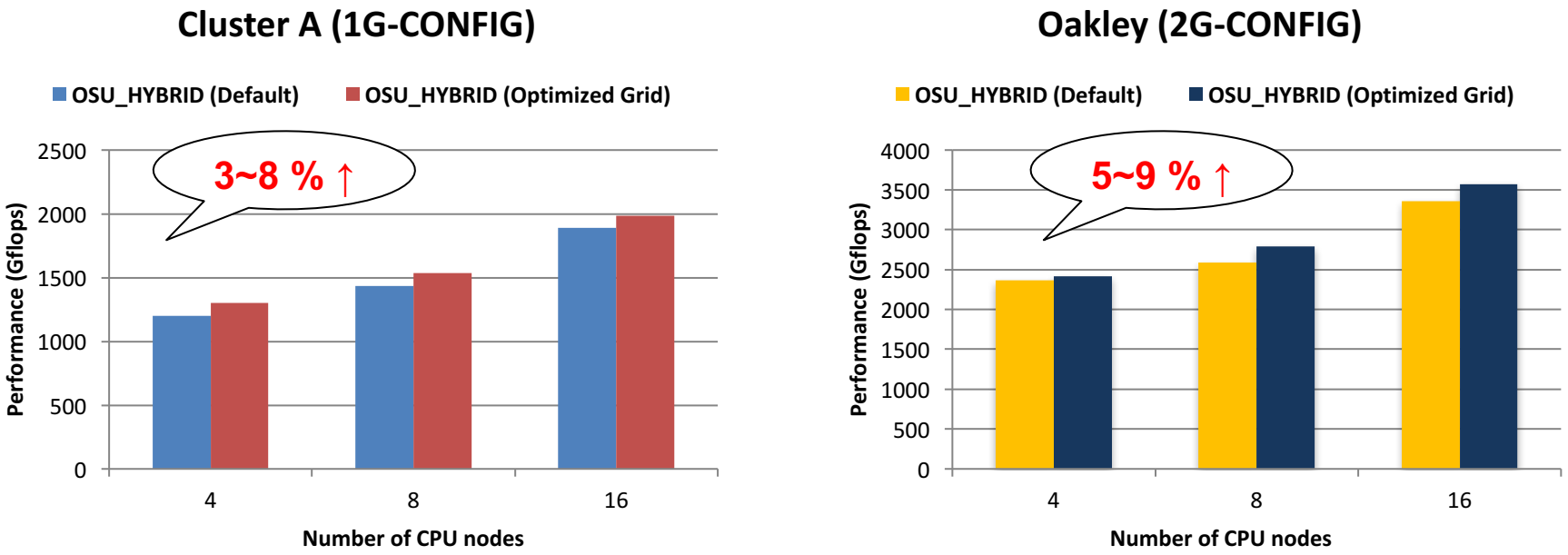




4 GPU nodes with increasing number of CPU nodes

- **One more core used for computing brings 8~10% and 13~15% performance gain on Cluster A and Oakley Cluster**

Performance with Process Grid Reordering



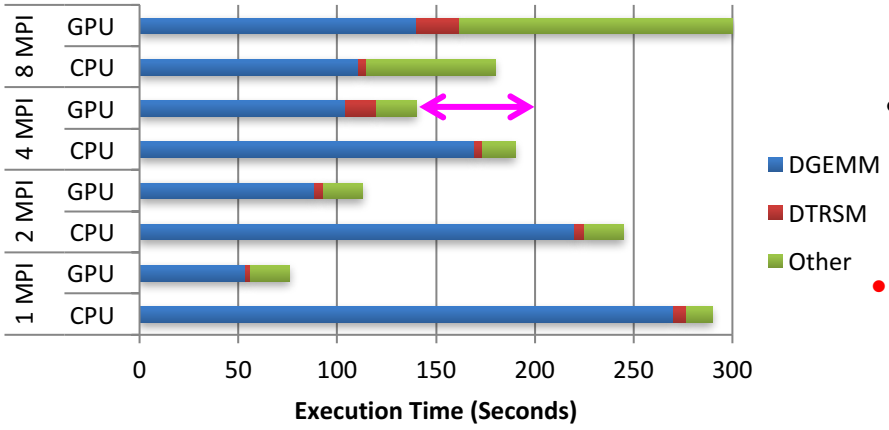
4 GPU nodes with increasing number of CPU nodes

OSU_HYBRID (Default)
OSU_HYBRID (Optimized Grid)

- **Optimized Grid brings 3~8% and 5~9% performance gain on Cluster A and Oakley Cluster**

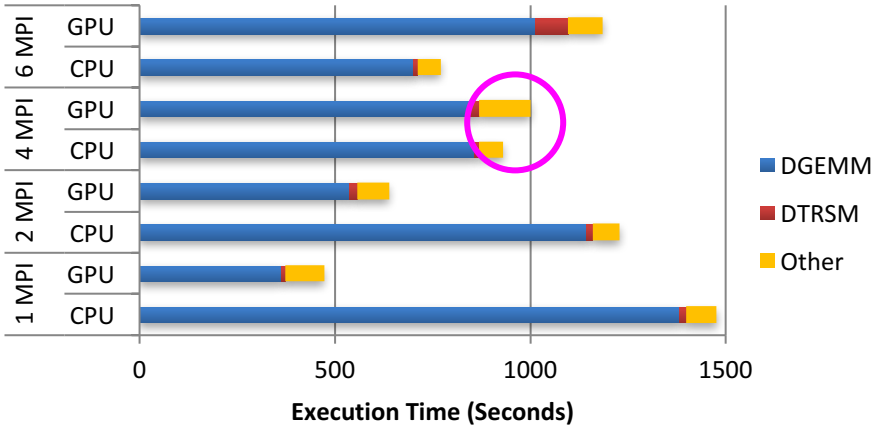
Load Balance Tuning

Cluster A (1G-CONFIG)

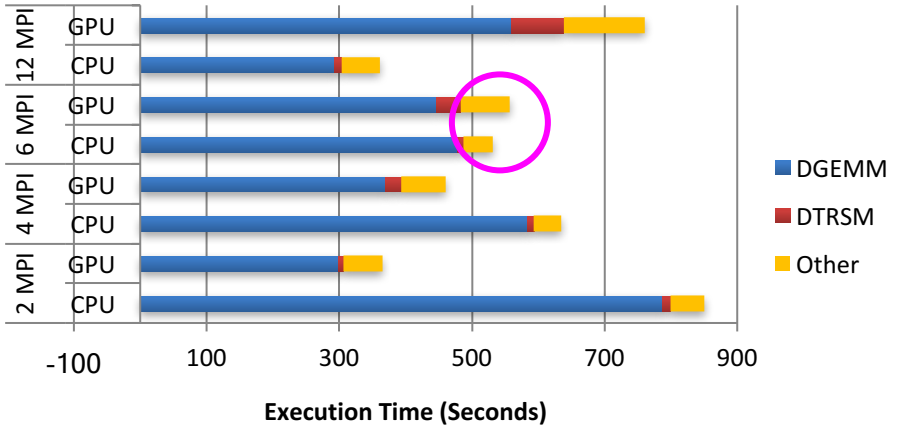


- 4 GPU nodes + 16 CPU nodes with different number of MPI processes/GPU
- MPI_GPU = ACTUAL_PEAK_GPU / ACTUAL_PEAK_CPU + β
- **The optimal number of MPI processes /GPU varies with different configurations**

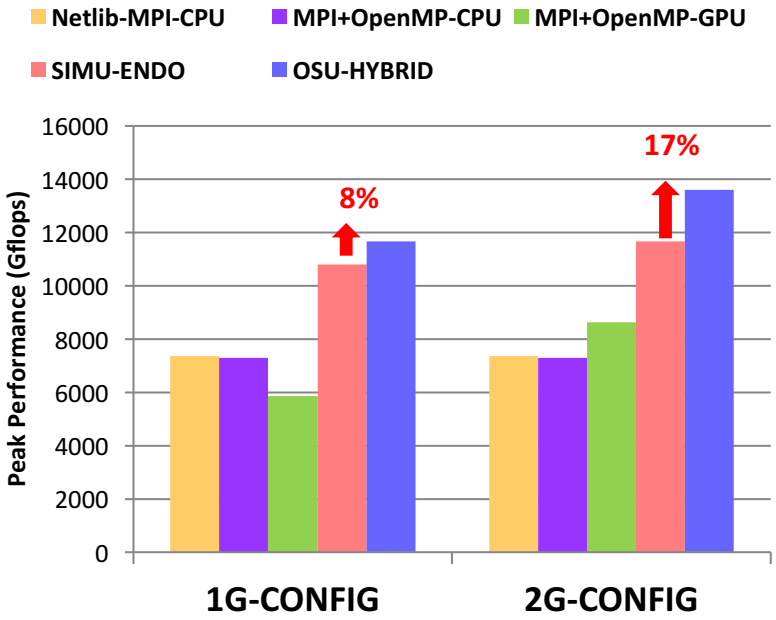
Oakley Cluster (1G-CONFIG)

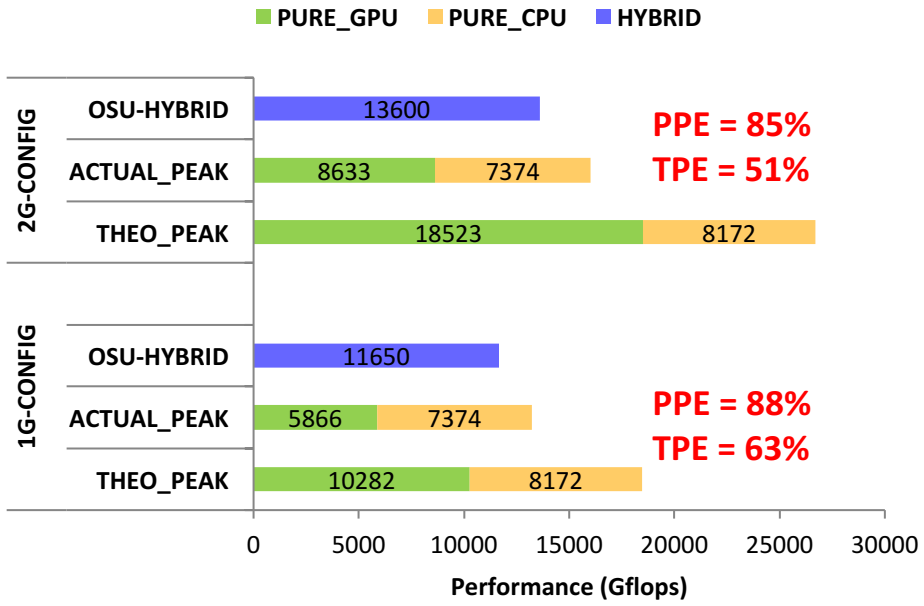


Oakley Cluster (2G-CONFIG)



Comparison of Peak Performance





- 16 GPU nodes + 64 CPU nodes on Oakley Clusters

  Netlib-MPI-CPU: Standard HPL on 64 CPU nodes

  MPI+OpenMP-CPU: NVIDIA's HPL on 64 CPU nodes

  MPI+OpenMP-GPU: NVIDIA's HPL on 16 GPU nodes

- Peak Performance Efficiency (**PPE**)
- Theoretical Performance Efficiency (**TPE**)

Peak Performance

- CPU/GPU ratio = 4

  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

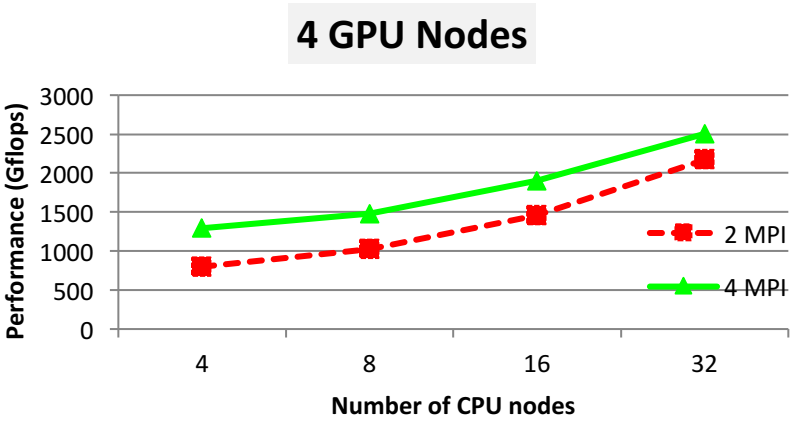
Configuration	Peak Perf (Gflops)	Problem size	Mem Use % (GPU)	PPE %	TPE %
1G-CONFIG-A	3888	140,000	78.5	80.7	52.8
1G-CONFIG-Oakley	22040	500,000	62	83.2	59.7
2G-CONFIG-Oakley	27110	512,000	77.5	86.3	50.8

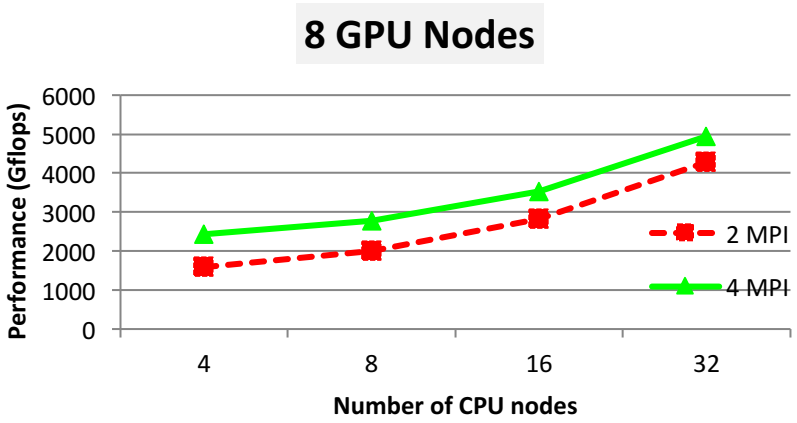
- CPU/GPU ratio = 6 (**32 GPU nodes + 192 CPU nodes)** on Oakley Cluster

Configuration	Peak Perf (Gflops)	Problem size	Mem Use % (GPU)	PPE %	TPE %
1G-CONFIG-Oakley	25,300	560,000	61.8	77	56.1
2G-CONFIG-Oakley	30,820	560,000	77.3	80.6	50.1

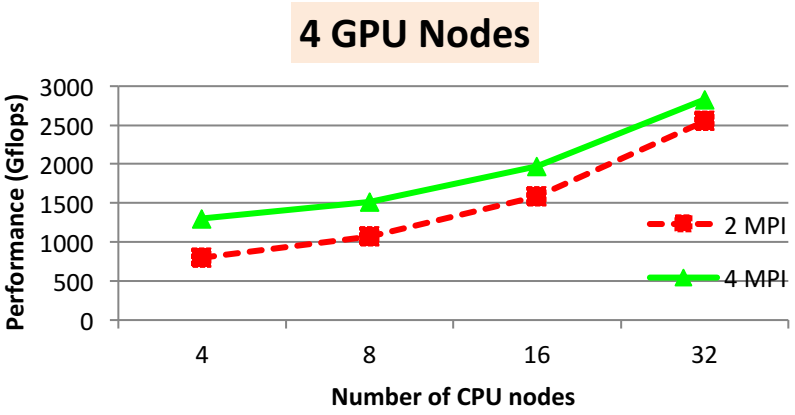
Strong & Weak Performance Scalability

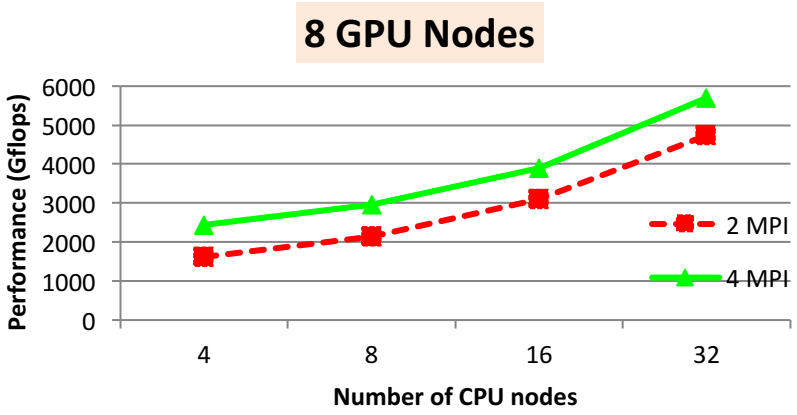
- 4/8 GPU nodes + 4/8/16/32 CPU nodes on Cluster A (1G_CONFIG)
- Strong Scalability: N = **80,000** and **110,000**



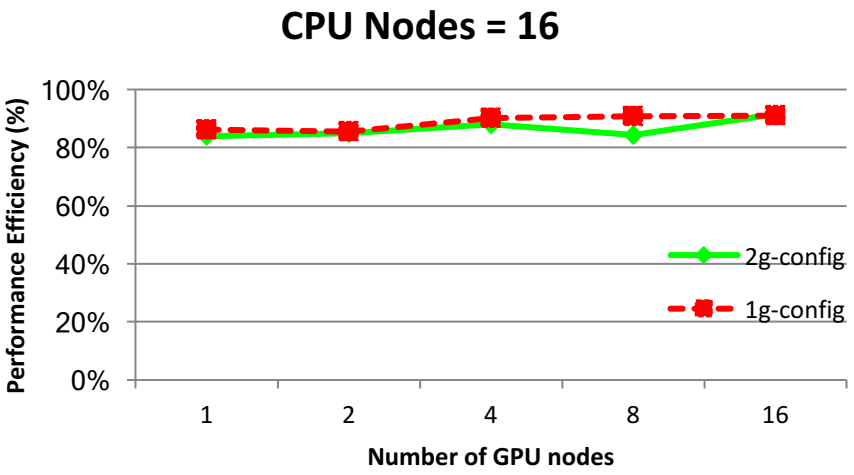


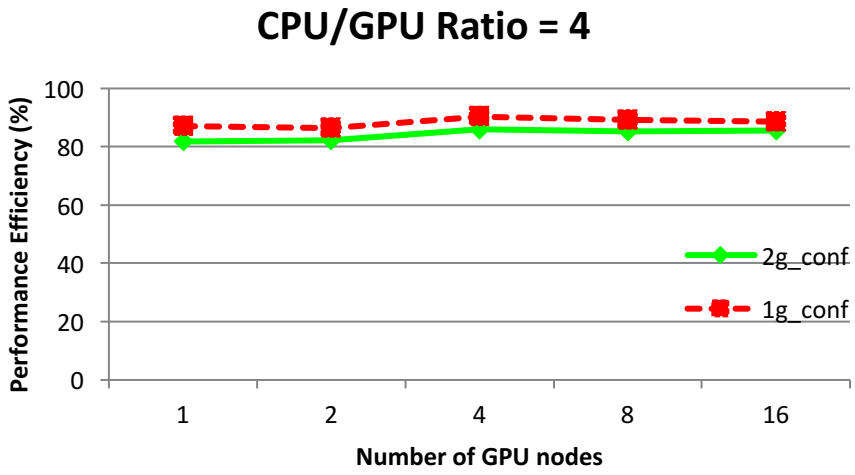
- Weak Scalability: memory usage of GPU nodes **≈ 80%**

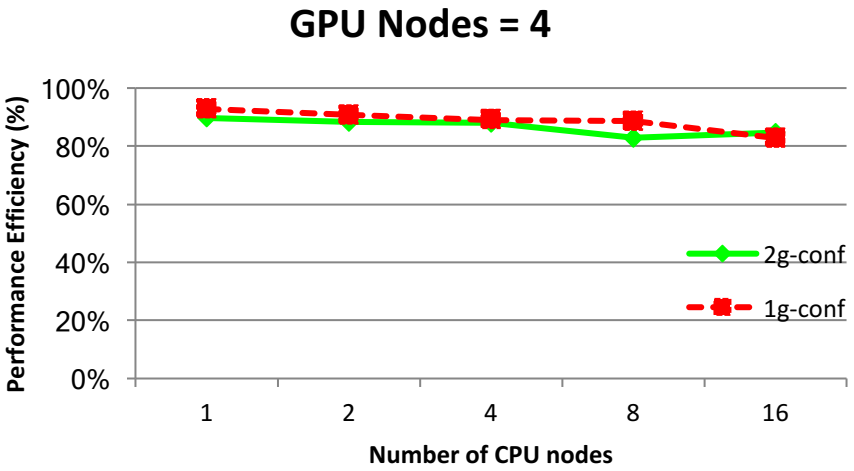




Peak Performance Efficiency Scalability







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

Outline

- Introduction
- Motivation & Problem Statement
- Proposed Design for Hybrid HPL
- Performance Evaluation
- Conclusion and Future Work

Conclusion

- Propose a novel approach to enable the **Portable** and **Scalable** HPL to efficiency utilize all computing resources on GPU-CPU clusters
- Be able to measure the peak performance of different heterogeneous clusters with varied configurations without code modification
- Exhibit sustained strong & weak performance scalability
  Exhibit sustained performance efficiency scalability
- Achieve **80%** of the combined actual peak performance of pure CPU and pure GPU nodes

Future Work

- Does the hybrid scheme apply to other applications?
- How to incorporate the design with new architecture? such as NVIDIA Kepler GPUs, Intel MICs, etc.

Acknowledgement

Mark Arnold (The Ohio State University)

Doug Johnson (Ohio Supercomputer Center)

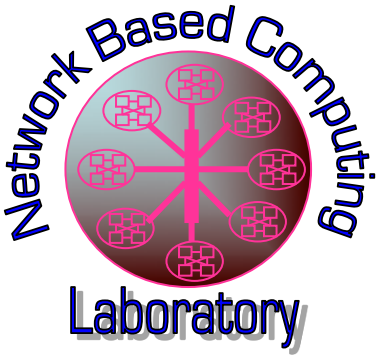
Everett Philips (NVIDIA)

Massimiliano Fatica (NVIDIA)

Thank You!

{shir, potluri, hamidouc, luxi, panda} @cse.ohio-state.edu

ktomko@osc.edu



Network-Based Computing Laboratory

http://nowlab.cse.ohio-state.edu/

MVAPICH Web Page

http://mvapich.cse.ohio-state.edu/