Building Efficient HPC Clouds with MVAPICH2 and RDMA-Hadoop over SR-IOV InfiniBand Clusters

Talk at OpenFabrics Alliance Workshop (OFAW ‘17)

by

Xiaoyi Lu
The Ohio State University
E-mail: luxi@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~luxi

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

- Cloud Computing focuses on maximizing the effectiveness of the shared resources
- Virtualization is the key technology for resource sharing in the Cloud
- Widely adopted in industry computing environment
Drivers of Modern HPC Cluster and Cloud Architecture

- Multi-core/many-core technologies, Accelerators
- Large memory nodes
- Solid State Drives (SSDs), NVM, Parallel Filesystems, Object Storage Clusters
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
- Single Root I/O Virtualization (SR-IOV)

SSDs, Object Storage Clusters

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

Large memory nodes (Upto 2 TB)
Single Root I/O Virtualization (SR-IOV)

- Single Root I/O Virtualization (SR-IOV) is providing new opportunities to design HPC cloud with very little low overhead.
- Allows a single physical device, or a Physical Function (PF), to present itself as multiple virtual devices, or Virtual Functions (VFs).
- VFs are designed based on the existing non-virtualized PFs, no need for driver change.
- Each VF can be dedicated to a single VM through PCI pass-through.
- Work with 10/40 GigE and InfiniBand.
Building HPC Cloud with SR-IOV and InfiniBand

- High-Performance Computing (HPC) has adopted advanced interconnects and protocols
  - InfiniBand
  - 10/40/100 Gigabit Ethernet/iWARP
  - RDMA over Converged Enhanced Ethernet (RoCE)
- Very Good Performance
  - Low latency (few micro seconds)
  - High Bandwidth (100 Gb/s with EDR InfiniBand)
  - Low CPU overhead (5-10%)
- OpenFabrics software stack with IB, iWARP and RoCE interfaces are driving HPC systems
- How to Build HPC Clouds with SR-IOV and InfiniBand for delivering optimal performance?
Network Based Computing Laboratory

Broad Challenges in Designing Communication and I/O Middleware for HPC on Clouds

- Virtualization Support with Virtual Machines and Containers
  - KVM, Docker, Singularity, etc.
- Communication coordination among optimized communication channels on Clouds
  - SR-IOV, IVShmem, IPC-Shm, CMA, etc.
- Locality-aware communication
- Scalability for million to billion processors
  - Support for highly-efficient inter-node and intra-node communication (both two-sided and one-sided)
- Scalable Collective communication
  - Offload; Non-blocking; Topology-aware
- Balancing intra-node and inter-node communication for next generation nodes (128-1024 cores)
  - Multiple end-points per node
- NUMA-aware communication for nested virtualization
- Integrated Support for GPGPUs and Accelerators
- Fault-tolerance/resiliency
  - Migration support with virtual machines
- QoS support for communication and I/O
- Support for Hybrid MPI+PGAS programming (MPI + OpenMP, MPI + UPC, MPI + OpenSHMEM, MPI+UPC++, CAF, ...)
- Energy-Awareness
- Co-design with resource management and scheduling systems on Clouds
  - OpenStack, Slurm, etc.
Additional Challenges in Designing Communication and I/O Middleware for Big Data on Clouds

- High-Performance designs for Big Data middleware
  - RDMA-based designs to accelerate Big Data middleware on high-performance Interconnects
  - NVM-aware communication and I/O schemes for Big Data
  - SATA-/PCIe-/NVMe-SSD support
  - Parallel Filesystem support
  - Optimized overlapping among Computation, Communication, and I/O
  - Threaded Models and Synchronization

- Fault-tolerance/resiliency
  - Migration support with virtual machines
  - Data replication

- Efficient data access and placement policies

- Efficient task scheduling

- Fast deployment and automatic configurations on Clouds
Approaches to Build HPC Clouds

- MVAPICH2-Virt with SR-IOV and IVSHMEM
  - Standalone, OpenStack
- SR-IOV-enabled VM Migration Support in MVAPICH2
- MVAPICH2 with Containers (Docker and Singularity)
- MVAPICH2 with Nested Virtualization (Container over VM)
- MVAPICH2-Virt on SLURM
  - SLURM alone, SLURM + OpenStack
- Big Data Libraries on Cloud
  - RDMA-Hadoop, OpenStack Swift
# MVAPICH2 Software Family

<table>
<thead>
<tr>
<th>High-Performance Parallel Programming Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MVAPICH2</strong></td>
</tr>
<tr>
<td><strong>MVAPICH2-X</strong></td>
</tr>
<tr>
<td><strong>MVAPICH2-GDR</strong></td>
</tr>
<tr>
<td><strong>MVAPICH2-Virt</strong></td>
</tr>
<tr>
<td><strong>MVAPICH2-EA</strong></td>
</tr>
<tr>
<td><strong>MVAPICH2-MIC</strong></td>
</tr>
</tbody>
</table>

## Microbenchmarks

<table>
<thead>
<tr>
<th>Microbenchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OMB</strong></td>
</tr>
</tbody>
</table>

## Tools

<table>
<thead>
<tr>
<th>Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OSU INAM</strong></td>
</tr>
<tr>
<td><strong>OEMT</strong></td>
</tr>
</tbody>
</table>
HPC on Cloud Computing Systems: Challenges Addressed by OSU So Far

Applications

HPC and Big Data Middleware

HPC (MPI, PGAS, MPI+PGAS, MPI+OpenMP, etc.)

Resource Management and Scheduling Systems for Cloud Computing
(OpenStack Nova, Heat; Slurm)

Communication and I/O Library

Communication Channels
(SR-IOV, IVShmem, IPC-Shm, CMA)

Fault-Tolerance & Consolidation
(Migration)

Locality- and NUMA-aware Communication

QoS-aware

Virtualization
(Hypervisor and Container)

Future Studies

Networking Technologies
(InfiniBand, Omni-Path, 1/10/40/100 GigE and Intelligent NICs)

Commodity Computing System Architectures
(Multi- and Many-core architectures and accelerators)

Storage Technologies
(HDD, SSD, NVRAM, and NVMe-SSD)
Overview of MVAPICH2-Virt with SR-IOV and IVSHMEM

- Redesign MVAPICH2 to make it virtual machine aware
  - SR-IOV shows near to native performance for inter-node point to point communication
  - IVSHMEM offers shared memory based data access across co-resident VMs
  - Locality Detector: maintains the locality information of co-resident virtual machines
  - Communication Coordinator: selects the communication channel (SR-IOV, IVSHMEM) adaptively


MVAPICH2-Virt with SR-IOV and IVSHMEM over OpenStack

- OpenStack is one of the most popular open-source solutions to build clouds and manage virtual machines
- Deployment with OpenStack
  - Supporting SR-IOV configuration
  - Supporting IVSHMEM configuration
  - Virtual Machine aware design of MVAPICH2 with SR-IOV
- An efficient approach to build HPC Clouds with MVAPICH2-Virt and OpenStack

Application-Level Performance on Chameleon

- 32 VMs, 6 Core/VM
- Compared to Native, 2-5% overhead for Graph500 with 128 Procs
- Compared to Native, 1-9.5% overhead for SPEC MPI2007 with 128 Procs
Approaches to Build HPC Clouds

- MVAPICH2-Virt with SR-IOV and IVSHMEM
  - Standalone, OpenStack
- SR-IOV-enabled VM Migration Support in MVAPICH2
- MVAPICH2 with Containers (Docker and Singularity)
- MVAPICH2 with Nested Virtualization (Container over VM)
- MVAPICH2-Virt on SLURM
  - SLURM alone, SLURM + OpenStack
- Big Data Libraries on Cloud
  - RDMA-Hadoop, OpenStack Swift
Execute Live Migration with SR-IOV Device

```
[root@sandy1:Migration]$
[root@sandy1:Migration]# ssh sandy3-vm1 lspci
```

**Example Output:**

```
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMCH [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.2 USB controller: Intel Corporation 82371SB PIIX3 USB [Natoma/Triton II] (rev 01)
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 03)
00:02.0 VGA compatible controller: Cirrus Logic GD 5446
00:03.0 Ethernet controller: Red Hat, Inc Virtio network device
00:04.0 Infiniband controller: Mellanox Technologies MT27700 Family [ConnectX-4 Virtual Function]
00:05.0 Unclassified device [00ff]: Red Hat, Inc Virtio memory balloon
```

```
[root@sandy1:Migration]$ virsh migrate --live --rdma-pin-all --migrateuri rdma://sandy3-ib sandy1-vm1 qemu:///sandy3-tb/system
error: Requested operation is not valid: domain has assigned non-USB host devices
[root@sandy1:Migration]$`
```
High Performance SR-IOV enabled VM Migration Support in MVAPICH2

- Migration with SR-IOV device has to handle the challenges of detachment/re-attachment of virtualized IB device and IB connection
- Consist of SR-IOV enabled IB Cluster and External Migration Controller
- Multiple parallel libraries to notify MPI applications during migration (detach/reattach SR-IOV/IVShmem, migrate VMs, migration status)
- Handle the IB connection suspending and reactivating
- Propose Progress engine (PE) and migration thread based (MT) design to optimize VM migration and MPI application performance

Performance Evaluation of VM Migration Framework

- Compared with the TCP, the RDMA scheme reduces the total migration time by 20%.
- Total time is dominated by ‘Migration’ time; Times on other steps are similar across different schemes.
- Proposed migration framework could reduce up to 51% migration time.
Performance Evaluation of VM Migration Framework

- Migrate a VM from one machine to another while benchmark is running inside
- Proposed MT-based designs perform slightly worse than PE-based designs because of lock/unlock
- No benefit from MT because of NO computation involved
Performance Evaluation of VM Migration Framework

- 8 VMs in total and 1 VM carries out migration during application running
- Compared with NM, MT- worst and PE incur some overhead compared with NM
- MT-typical allows migration to be completely overlapped with computation
Approaches to Build HPC Clouds

- MVAPICH2-Virt with SR-IOV and IVSHMEM
  - Standalone, OpenStack
- SR-IOV-enabled VM Migration Support in MVAPICH2
- MVAPICH2 with Containers (Docker and Singularity)
- MVAPICH2 with Nested Virtualization (Container over VM)
- MVAPICH2-Virt on SLURM
  - SLURM alone, SLURM + OpenStack
- Big Data Libraries on Cloud
  - RDMA-Hadoop, OpenStack Swift
Container-based technologies (e.g., Docker) provide lightweight virtualization solutions.

Container-based virtualization – share host kernel by containers.
Benefits of Containers-based Virtualization for HPC on Cloud

- Experiment on NFS Chameleon Cloud
- Container has less overhead than VM
- BFS time in Graph 500 significantly increases as the number of container increases on one host. Why?

Containers-based Design: Issues, Challenges, and Approaches

- What are the performance bottlenecks when running MPI applications on multiple containers per host in HPC cloud?
- Can we propose a new design to overcome the bottleneck on such container-based HPC cloud?
- Can optimized design deliver near-native performance for different container deployment scenarios?
- Locality-aware based design to enable CMA and Shared memory channels for MPI communication across co-resident containers

Application-Level Performance on Docker with MVAPICH2

- 64 Containers across 16 nodes, pining 4 Cores per Container
- Compared to Container-Def, up to **11%** and **73%** of execution time reduction for NAS and Graph 500
- Compared to Native, less than **9 %** and **5%** overhead for NAS and Graph 500
• Less than 18% overhead on latency
• Less than 13% overhead on BW
- 512 Processes across 32 nodes
- Less than 15% and 14% overhead for Bcast and Allreduce, respectively
Application-Level Performance on Singularity with MVAPICH2

- 512 Processes across 32 nodes
- Less than 16% and 11% overhead for NPB and Graph500, respectively
Approaches to Build HPC Clouds

- MVAPICH2-Virt with SR-IOV and IVSHMEM
  - Standalone, OpenStack
- SR-IOV-enabled VM Migration Support in MVAPICH2
- MVAPICH2 with Containers (Docker and Singularity)
- MVAPICH2 with Nested Virtualization (Container over VM)
- MVAPICH2-Virt on SLURM
  - SLURM alone, SLURM + OpenStack
- Big Data Libraries on Cloud
  - RDMA-Hadoop, OpenStack Swift
Nested Virtualization: Containers over Virtual Machines

- Useful for live migration, sandbox application, legacy system integration, software deployment, etc.
- Performance issues because of the redundant call stacks (two-layer virtualization) and isolated physical resources
Multiple Communication Paths in Nested Virtualization

- Different VM placements introduce multiple communication paths on container level
  1. Intra-VM Intra-Container (across core 4 and core 5)
  2. Intra-VM Inter-Container (across core 13 and core 14)
  3. Inter-VM Inter-Container (across core 6 and core 12)
  4. Inter-Node Inter-Container (across core 15 and the core on remote node)
Two VMs are deployed on the same socket and different sockets, respectively.

*-Def and Inter-VM Inter-Container-1Layer have similar performance.

Large gap compared to native performance.

Challenges of Nested Virtualization

- How to further reduce the performance overhead of running applications on the nested virtualization environment?

- What are the impacts of the different VM/container placement schemes for the communication on the container level?

- Can we propose a design which can adapt these different VM/container placement schemes and deliver near-native performance for nested virtualization environments?
Overview of Proposed Design in MVAPICH2

Two-Layer Locality Detector: Dynamically detecting MPI processes in the co-resident containers inside one VM as well as the ones in the co-resident VMs.

Two-Layer NUMA Aware Communication Coordinator: Leverage nested locality info, NUMA architecture info and message to select appropriate communication channel.

Inter-VM Inter-Container Pt2Pt (Intra-Socket)

- 1Layer has similar performance to the Default
- Compared with 1Layer, 2Layer delivers up to 84% and 184% improvement for latency and BW
Inter-VM Inter-Container Pt2Pt (Inter-Socket)

- 1-Layer has similar performance to the Default
- 2-Layer has near-native performance for small msg, but clear overhead on large msg
- Compared to 2-Layer, Hybrid design brings up to 42% and 25% improvement for latency and BW, respectively
Application-level Evaluations

- 256 processes across 64 containers on 16 nodes
- Compared with Default, enhanced-hybrid design reduces up to 16% (28,16) and 10% (LU) of execution time for Graph 500 and NAS, respectively
- Compared with the 1Layer case, enhanced-hybrid design also brings up to 12% (28,16) and 6% (LU) performance benefit
Approaches to Build HPC Clouds

• MVAPICH2-Virt with SR-IOV and IVSHMEM
  – Standalone, OpenStack

• SR-IOV-enabled VM Migration Support in MVAPICH2

• MVAPICH2 with Containers (Docker and Singularity)

• MVAPICH2 with Nested Virtualization (Container over VM)

• MVAPICH2-Virt on SLURM
  – SLURM alone, SLURM + OpenStack

• Big Data Libraries on Cloud
  – RDMA-Hadoop, OpenStack Swift
Need for Supporting SR-IOV and IVSHMEM in SLURM

- Requirement of managing and isolating virtualized resources of SR-IOV and IVSHMEM
- Such kind of management and isolation is hard to be achieved by MPI library alone, but much easier with SLURM
- Efficient running MPI applications on HPC Clouds needs SLURM to support managing SR-IOV and IVSHMEM
  - Can critical HPC resources be efficiently shared among users by extending SLURM with support for SR-IOV and IVSHMEM based virtualization?
  - Can SR-IOV and IVSHMEM enabled SLURM and MPI library provide bare-metal performance for end applications on HPC Clouds?
SLURM SPANK Plugin based Design

- VM Configuration Reader – Register all VM configuration options, set in the job control environment so that they are visible to all allocated nodes.
- VM Launcher – Setup VMs on each allocated node:
  - File based lock to detect occupied VF and exclusively allocate free VF
  - Assign a unique ID to each IVSHMEM and dynamically attach to each VM
- VM Reclaimer – Tear down VMs and reclaim resources
SLURM SPANK Plugin with OpenStack based Design

- VM Configuration Reader – VM options register
- VM Launcher, VM Reclaimer – Offload to underlying OpenStack infrastructure
  - PCI Whitelist to passthrough free VF to VM
  - Extend Nova to enable IVSHMEM when launching VM

32 VMs across 8 nodes, 6 Core/VM

EASJ - Compared to Native, less than 4% overhead with 128 Procs

SACJ, EACJ – Also minor overhead, when running NAS as concurrent job with 64 Procs
Approaches to Build HPC Clouds

• MVAPICH2-Virt with SR-IOV and IVSHMEM
  – Standalone, OpenStack
• SR-IOV-enabled VM Migration Support in MVAPICH2
• MVAPICH2 with Containers (Docker and Singularity)
• MVAPICH2 with Nested Virtualization (Container over VM)
• MVAPICH2-Virt on SLURM
  – SLURM alone, SLURM + OpenStack

• Big Data Libraries on Cloud
  – RDMA-Hadoop, OpenStack Swift
The High-Performance Big Data (HiBD) Project

- RDMA for Apache Spark
- RDMA for Apache Hadoop 2.x (RDMA-Hadoop-2.x)
  - Plugins for Apache, Hortonworks (HDP) and Cloudera (CDH) Hadoop distributions
- RDMA for Apache HBase
- RDMA for Memcached (RDMA-Memcached)
- RDMA for Apache Hadoop 1.x (RDMA-Hadoop)
- OSU HiBD-Benchmarks (OHB)
  - HDFS, Memcached, HBase, and Spark Micro-benchmarks
- [http://hibd.cse.ohio-state.edu](http://hibd.cse.ohio-state.edu)
- Users Base: 215 organizations from 29 countries
- More than 21,000 downloads from the project site

Available for InfiniBand and RoCE
High-Performance Apache Hadoop over Clouds: Challenges

• How about performance characteristics of native IB-based designs for Apache Hadoop over SR-IOV enabled cloud environment?

• To achieve locality-aware communication, how can the cluster topology be automatically detected in a scalable and efficient manner and be exposed to the Hadoop framework?

• How can we design virtualization-aware policies in Hadoop for efficiently taking advantage of the detected topology?

• Can the proposed policies improve the performance and fault tolerance of Hadoop on virtualized platforms?

“How can we design a high-performance Hadoop library for Cloud-based systems?”
## Overview of RDMA-Hadoop-Virt Architecture

- **Virtualization-aware modules in all the four main Hadoop components:**
  - **HDFS:** Virtualization-aware Block Management to improve fault-tolerance
  - **YARN:** Extensions to Container Allocation Policy to reduce network traffic
  - **MapReduce:** Extensions to Map Task Scheduling Policy to reduce network traffic
  - **Hadoop Common:** Topology Detection Module for automatic topology detection

- **Communications in HDFS, MapReduce, and RPC go through RDMA-based designs over SR-IOV enabled InfiniBand**

---

Evaluation with Applications

- 14% and 24% improvement with Default Mode for CloudBurst and Self-Join
- 30% and 55% improvement with Distributed Mode for CloudBurst and Self-Join
OpenStack Swift Overview

- Distributed Cloud-based Object Storage Service
- Deployed as part of OpenStack installation
- Can be deployed as standalone storage solution as well
- Worldwide data access via Internet
  - HTTP-based
- Architecture
  - Multiple Object Servers: To store data
  - Few Proxy Servers: Act as a proxy for all requests
  - Ring: Handles metadata
- Usage
  - Input/output source for Big Data applications (most common use case)
  - Software/Data backup
  - Storage of VM/Docker images
- Based on traditional TCP sockets communication
Swift-X: Accelerating OpenStack Swift with RDMA for Building Efficient HPC Clouds

• Challenges
  – Proxy server is a bottleneck for large scale deployments
  – Object upload/download operations network intensive
  – Can an RDMA-based approach benefit?

• Design
  – Re-designed Swift architecture for improved scalability and performance; Two proposed designs:
    • Client-Oblivious Design: No changes required on the client side
    • Metadata Server-based Design: Direct communication between client and object servers; bypass proxy server
  – RDMA-based communication framework for accelerating networking performance
  – High-performance I/O framework to provide maximum overlap between communication and I/O

S. Gugnani, X. Lu, and D. K. Panda, Swift-X: Accelerating OpenStack Swift with RDMA for Building an Efficient HPC Cloud, accepted at CCGrid’17, May 2017
Swift-X: Accelerating OpenStack Swift with RDMA for Building Efficient HPC Clouds

- Communication time reduced by up to 3.8x for PUT and up to 2.8x for GET
- Up to 66% reduction in GET latency
Available Appliances on Chameleon Cloud*

<table>
<thead>
<tr>
<th>Appliance</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CentOS 7 KVM SR-IOV</td>
<td>Chameleon bare-metal image customized with the KVM hypervisor and a recompiled kernel to enable SR-IOV over InfiniBand. <a href="https://www.chameleoncloud.org/appliances/3/">https://www.chameleoncloud.org/appliances/3/</a></td>
</tr>
<tr>
<td>MPI bare-metal cluster complex appliance (Based on Heat)</td>
<td>This appliance deploys an MPI cluster composed of bare metal instances using the MVAPICH2 library over InfiniBand. <a href="https://www.chameleoncloud.org/appliances/29/">https://www.chameleoncloud.org/appliances/29/</a></td>
</tr>
<tr>
<td>MPI + SR-IOV KVM cluster (Based on Heat)</td>
<td>This appliance deploys an MPI cluster of KVM virtual machines using the MVAPICH2-Virt implementation and configured with SR-IOV for high-performance communication over InfiniBand. <a href="https://www.chameleoncloud.org/appliances/28/">https://www.chameleoncloud.org/appliances/28/</a></td>
</tr>
<tr>
<td>CentOS 7 SR-IOV RDMA-Hadoop</td>
<td>The CentOS 7 SR-IOV RDMA-Hadoop appliance is built from the CentOS 7 appliance and additionally contains RDMA-Hadoop library with SR-IOV. <a href="https://www.chameleoncloud.org/appliances/17/">https://www.chameleoncloud.org/appliances/17/</a></td>
</tr>
</tbody>
</table>

- Through these available appliances, users and researchers can easily deploy HPC clouds to perform experiments and run jobs with
  - High-Performance SR-IOV + InfiniBand
  - High-Performance MVAPICH2 Library over bare-metal InfiniBand clusters
  - High-Performance MVAPICH2 Library with Virtualization Support over SR-IOV enabled KVM clusters
  - High-Performance Hadoop with RDMA-based Enhancements Support

[*] Only include appliances contributed by OSU NowLab
Conclusions

- MVAPICH2-Virt over SR-IOV-enabled InfiniBand is an efficient approach to build HPC Clouds
  - Standalone, OpenStack, Slurm, and Slurm + OpenStack
  - Support Virtual Machine Migration with SR-IOV InfiniBand devices
  - Support Virtual Machine, Container (Docker and Singularity), and Nested Virtualization
- Very little overhead with virtualization, near native performance at application level
- Much better performance than Amazon EC2
- **MVAPICH2-Virt** is available for building HPC Clouds
  - SR-IOV, IVSHMEM, Docker support, OpenStack
- Big Data analytics stacks such as RDMA-Hadoop can benefit from cloud-aware designs
- Appliances for MVAPICH2-Virt and RDMA-Hadoop are available for building HPC Clouds
- Future releases for supporting running MPI jobs in VMs/Containers with SLURM, etc.
- SR-IOV/container support and appliances for other MVAPICH2 libraries (MVAPICH2-X, MVAPICH2-GDR, ...) and RDMA-Spark/Memcached
NVM-aware RDMA-Based Communication and I/O Schemes for High-Perf Big Data Analytics
Funding Acknowledgments

Funding Support by

Equipment Support by
# Personnel Acknowledgments

## Current Students
- A. Awan (Ph.D.)
- R. Biswas (M.S.)
- M. Bayatpour (Ph.D.)
- S. Chakraborty (Ph.D.)
- C.-H. Chu (Ph.D.)
- M. Li (Ph.D.)

## Past Students
- A. Augustine (M.S.)
- P. Balaji (Ph.D.)
- S. Bhagvat (M.S.)
- A. Bhat (M.S.)
- D. Buntinas (Ph.D.)
- L. Chai (Ph.D.)
- B. Chandrasekharan (M.S.)
- N. Dandapanthula (M.S.)
- V. Dhanraj (M.S.)
- T. Gangadharappa (M.S.)
- K. Gopalakrishnan (M.S.)
- W. Huang (Ph.D.)
- J. Jose (Ph.D.)
- S. Kini (M.S.)
- M. Koop (Ph.D.)
- K. Kulkarni (M.S.)
- R. Kumar (M.S.)
- S. Krishnamoorthy (M.S.)
- P. Lai (M.S.)
- J. Liu (Ph.D.)

## Past Post-Docs
- D. Banerjee
- X. Besseron
- H.-W. Jin
- J. Lin
- M. Luo
- E. Mancini

## Current Research Scientists
- X. Lu
- H. Subramoni

## Past Research Scientist
- K. Hamidouche
- S. Sur

## Past Programmers
- D. Bureddy
- M. Arnold
- J. Perkins

## Current Research Specialist
- J. Smith

## Current Research Scientists
- X. Lu
- H. Subramoni

## Past Research Scientist
- K. Hamidouche
- S. Sur

## Current Research Specialist
- J. Smith

## Current Research Scientists
- X. Lu
- H. Subramoni

## Past Research Scientist
- K. Hamidouche
- S. Sur

## Current Research Specialist
- J. Smith

## Current Research Scientists
- X. Lu
- H. Subramoni

## Past Research Scientist
- K. Hamidouche
- S. Sur

## Current Research Specialist
- J. Smith
Thank You!

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

http://www.cse.ohio-state.edu/~panda
http://www.cse.ohio-state.edu/~luxi

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

The High-Performance Big Data Project
http://hibd.cse.ohio-state.edu/