HiBD
High-Performance
Big Data

Overview of High-Performance Big Data Stacks

OSU Booth Talk at SC '17

by

Xiaoyi Lu

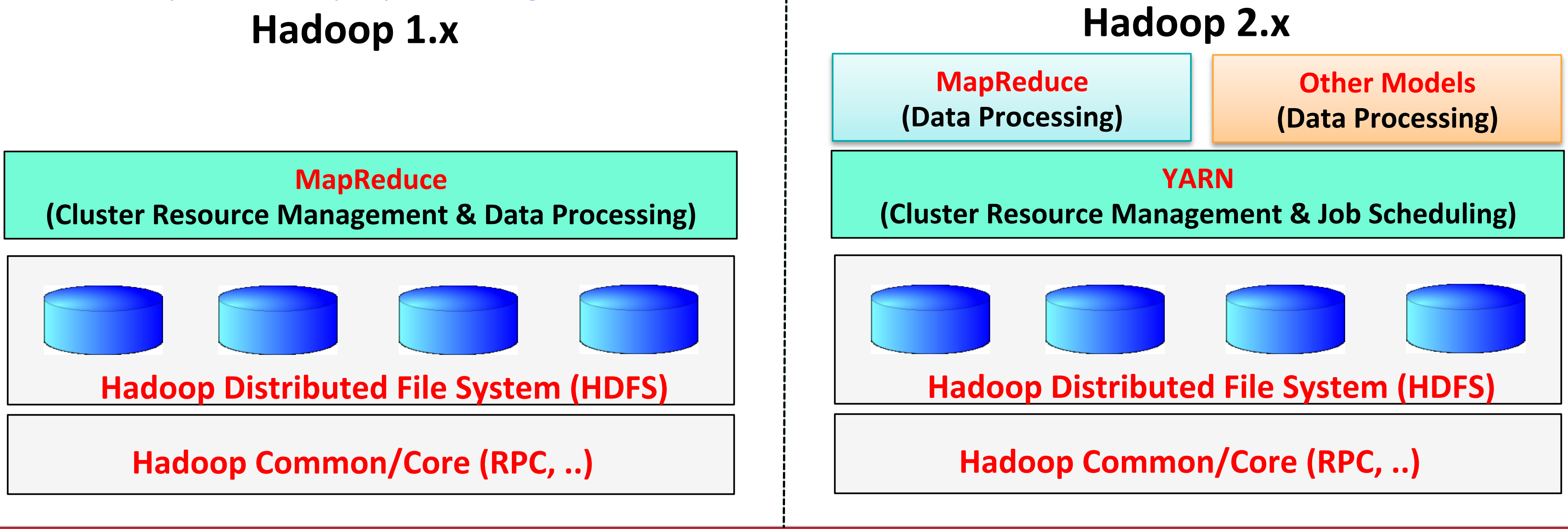
The Ohio State University

E-mail: luxi@cse.ohio-state.edu

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

Overview of Apache Hadoop Architecture

- Open-source implementation of Google MapReduce, GFS, and BigTable for Big Data Analytics
  - Hadoop Common Utilities (RPC, etc.), HDFS, MapReduce, YARN
- http://hadoop.apache.org

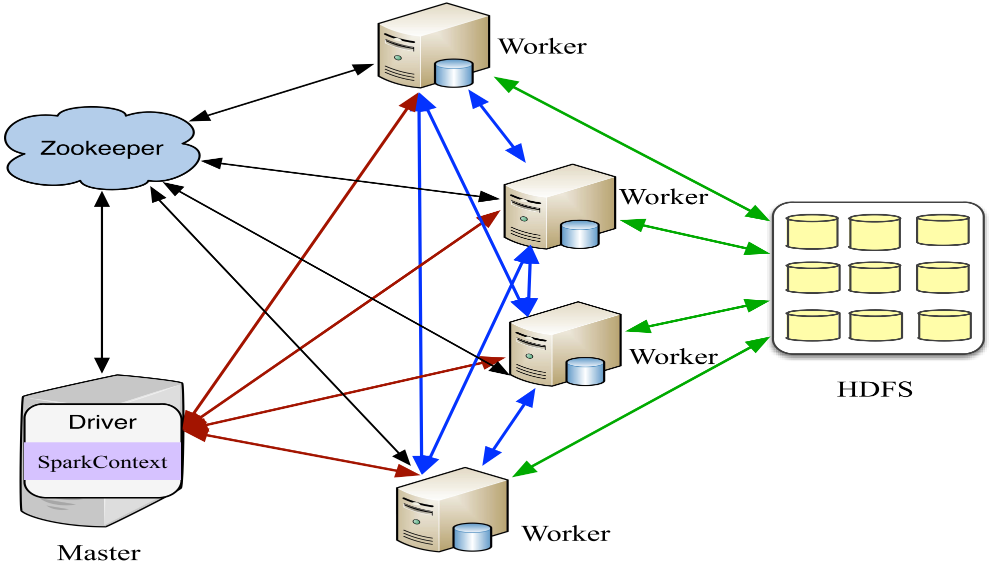

Hadoop 1.x
MapReduce
(Cluster Resource Management & Data Processing)
Hadoop Distributed File System (HDFS)
Hadoop Common/Core (RPC, ..)
Hadoop 2.x
MapReduce
(Data Processing)
Other Models
(Data Processing)
YARN
(Cluster Resource Management & Job Scheduling)
Hadoop Distributed File System (HDFS)
Hadoop Common/Core (RPC, ..)

HDFS and MapReduce in Apache Hadoop

- HDFS: Primary storage of Hadoop; highly reliable and fault-tolerant
  - NameNode stores the file system namespace
  - DataNodes store data blocks
- MapReduce: Computing framework of Hadoop; highly scalable
  - Map tasks read data from HDFS, operate on it, and write the intermediate data to local disk
  - Reduce tasks get data by shuffle, operate on it and write output to HDFS
- Adopted by many reputed organizations
  - eg: Facebook, Yahoo!
- Developed in Java for platform-independence and portability
- Uses Sockets/HTTP for communication!

Spark Architecture Overview

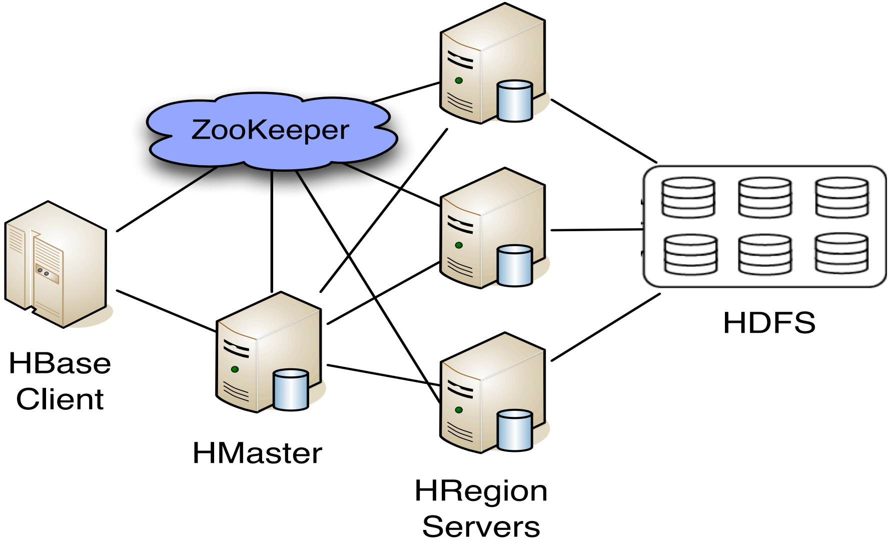
- An in-memory data-processing framework
  - Iterative machine learning jobs
  - Interactive data analytics
  - Scala based Implementation
  - Standalone, YARN, Mesos
- Scalable and communication intensive
  - Wide dependencies between Resilient Distributed Datasets (RDDs)
  - MapReduce-like shuffle operations to repartition RDDs
  - Sockets based communication



http://spark.apache.org

HBase Architecture Overview

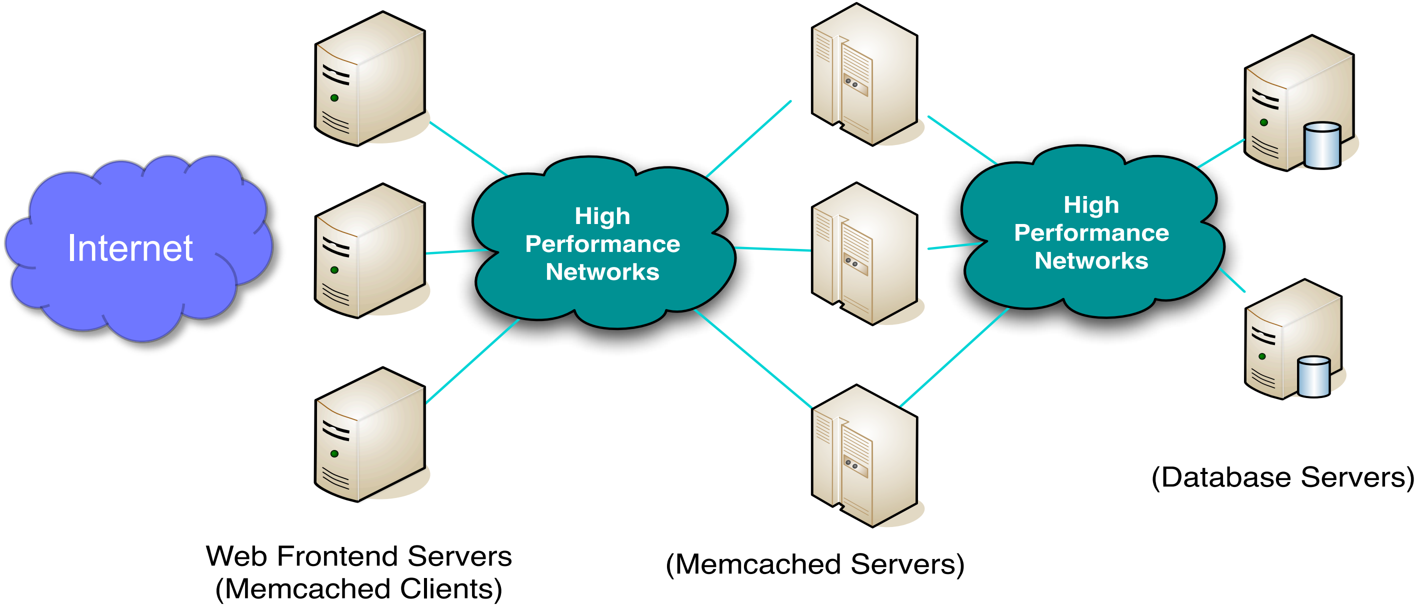
- Apache Hadoop Database (http://hbase.apache.org/)
- Semi-structured database, which is highly scalable
- Integral part of many datacenter applications
  - eg: Facebook Social Inbox
- Developed in Java for platform-independence and portability
- Uses sockets for communication!



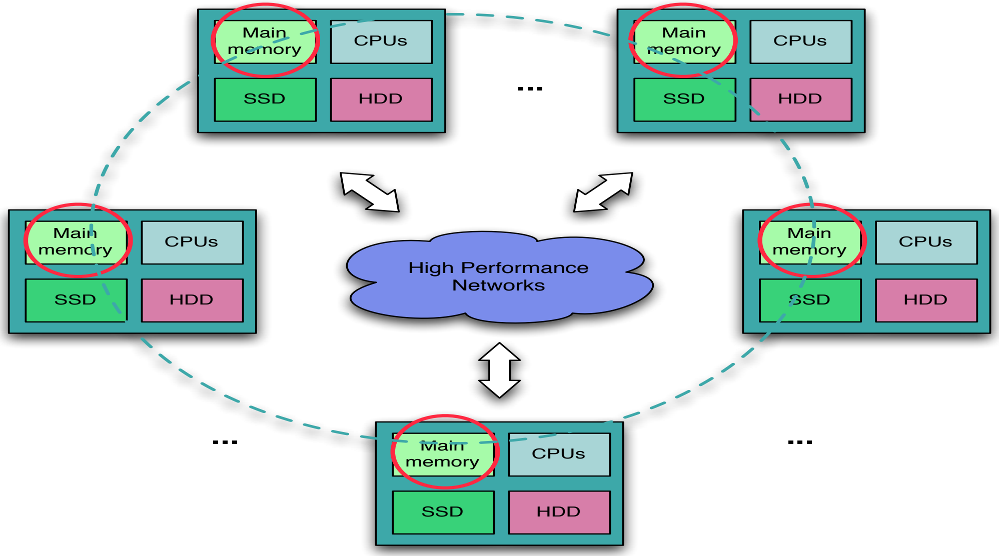
(HBase Architecture)

Overview of Web 2.0 Architecture and Memcached

- Three-layer architecture of Web 2.0
  - Web Servers, Memcached Servers, Database Servers
- Memcached is a core component of Web 2.0 architecture


Internet
High Performance Networks
High Performance Networks
Web Frontend Servers (Memcached Clients)
(Memcached Servers)
(Database Servers)

Memcached Architecture



- Distributed Caching Layer
  - Allows to aggregate spare memory from multiple nodes
  - General purpose
- Typically used to cache database queries, results of API calls
- Scalable model, but typical usage very network intensive

Presentation Outline

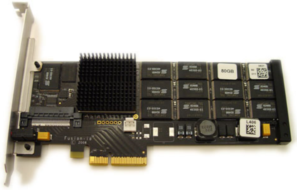
- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

Drivers of Modern HPC Cluster Architectures

Multi-core Processors

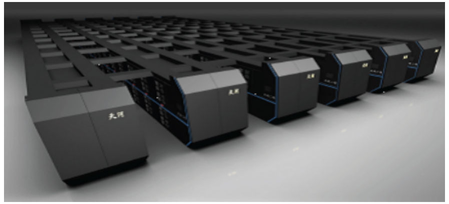
High Performance Interconnects - InfiniBand
<1usec latency, 100Gbps Bandwidth>

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



SSD, NVMe-SSD, NVRAM

- Multi-core/many-core technologies
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
- Solid State Drives (SSDs), Non-Volatile Random-Access Memory (NVRAM), NVMe-SSD
- Accelerators (NVIDIA GPGPUs and Intel Xeon Phi)



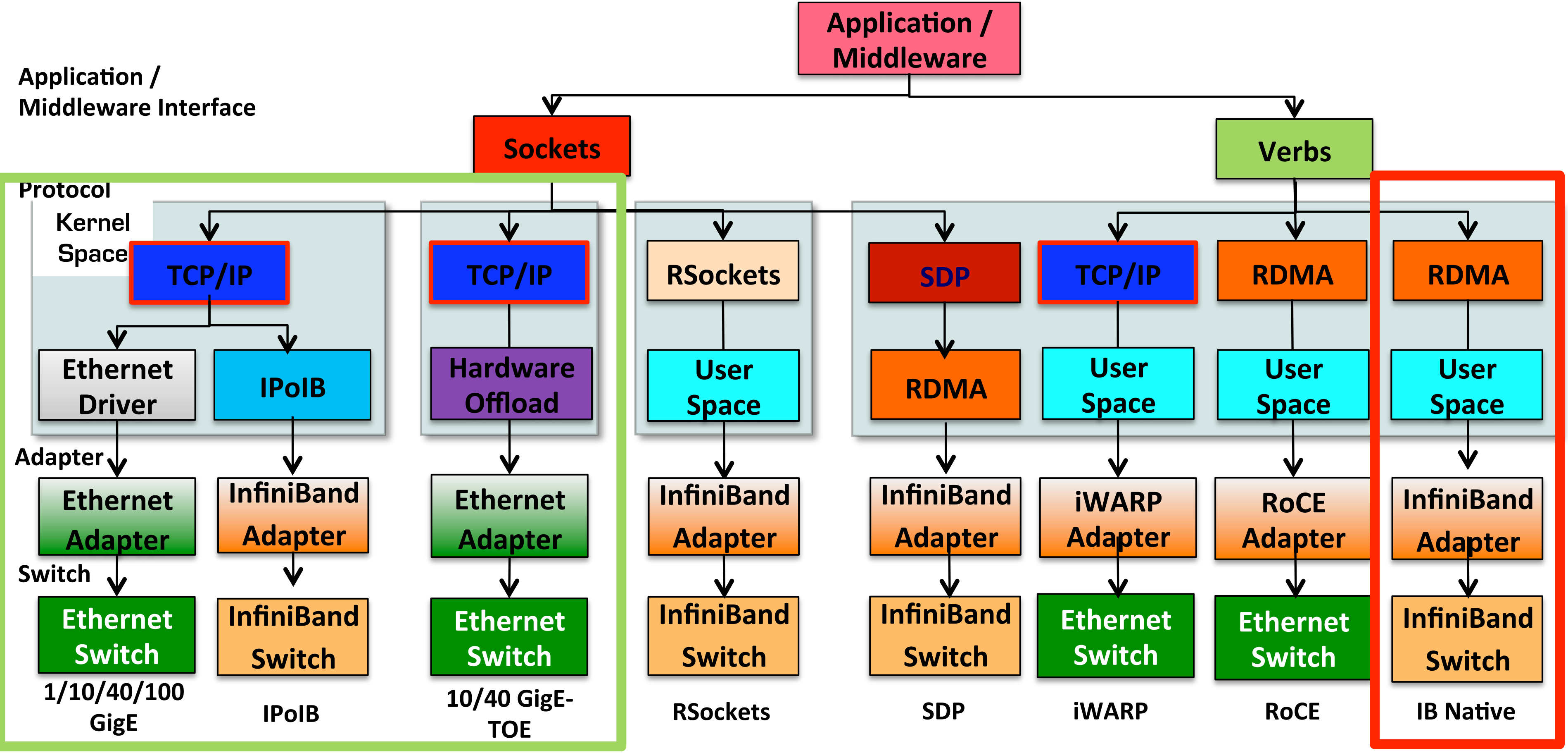
Tianhe – 2

Titan

Stampede

Tianhe – 1A

Interconnects and Protocols in OpenFabrics Stack



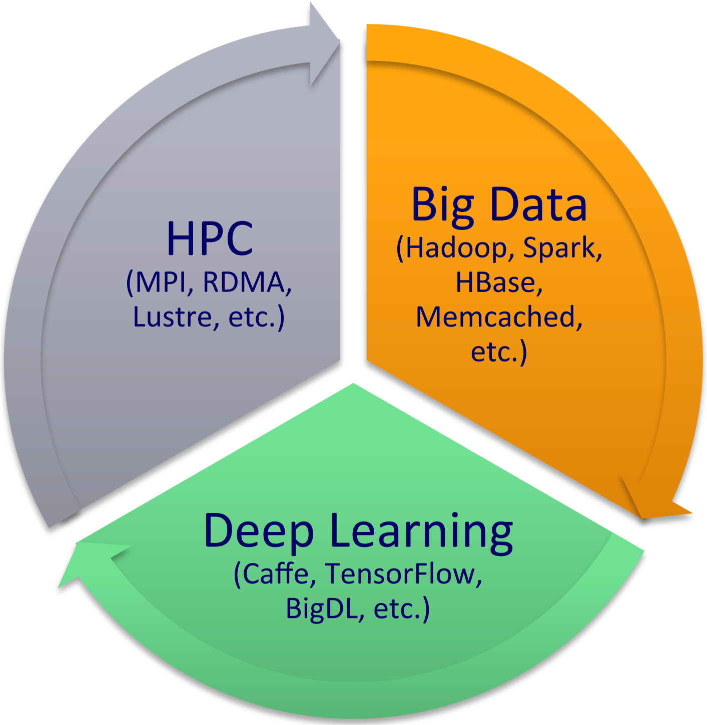
Open Standard InfiniBand Networking Technology

- Introduced in Oct 2000
- High Performance Data Transfer
  - Interprocessor communication and I/O
  - Low latency (<1.0 microsec), High bandwidth (up to 12.5 GigaBytes/sec -> 100Gbps), and low CPU utilization (5-10%)
- Flexibility for LAN and WAN communication
- Multiple Transport Services
  - Reliable Connection (RC), Unreliable Connection (UC), Reliable Datagram (RD), Unreliable Datagram (UD), and Raw Datagram
  - Provides flexibility to develop upper layers
- Multiple Operations
  - Send/Recv
  - RDMA Read/Write
  - Atomic Operations (very unique)
    - high performance and scalable implementations of distributed locks, semaphores, collective communication operations
- Leading to big changes in designing HPC clusters, file systems, cloud computing systems, grid computing systems, ….

Presentation Outline

- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

Increasing Usage of HPC, Big Data and Deep Learning


HPC
(MPI, RDMA, Lustre, etc.)
Big Data
(Hadoop, Spark, HBase, Memcached, etc.)
Deep Learning
(Caffe, TensorFlow, BigDL, etc.)

Convergence of HPC, Big Data, and Deep Learning!!!

How Can HPC Clusters with High-Performance Interconnect and Storage Architectures Benefit Big Data and Deep Learning Applications?

What are the major bottlenecks in current Big Data processing and Deep Learning middleware (e.g. Hadoop, Spark)?

Can the bottlenecks be alleviated with new designs by taking advantage of HPC technologies?

Can RDMA-enabled high-performance interconnects benefit Big Data processing and Deep Learning?

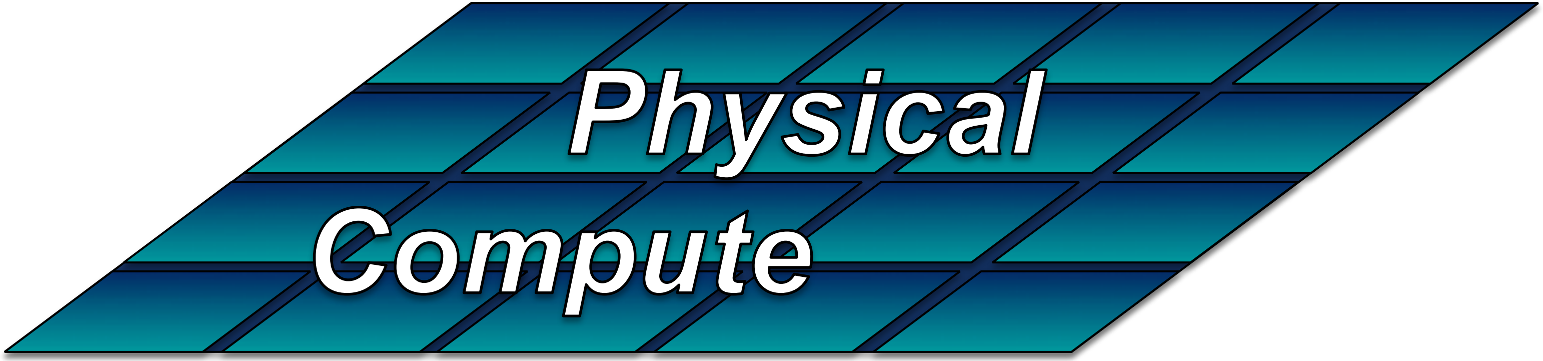
Can HPC Clusters with high-performance storage systems (e.g. SSD, parallel file systems) benefit Big Data and Deep Learning applications?

How much performance benefits can be achieved through enhanced designs?

How to design benchmarks for evaluating the performance of Big Data and Deep Learning middleware on HPC clusters?

Bring HPC, Big Data processing, and Deep Learning into a “convergent trajectory”!

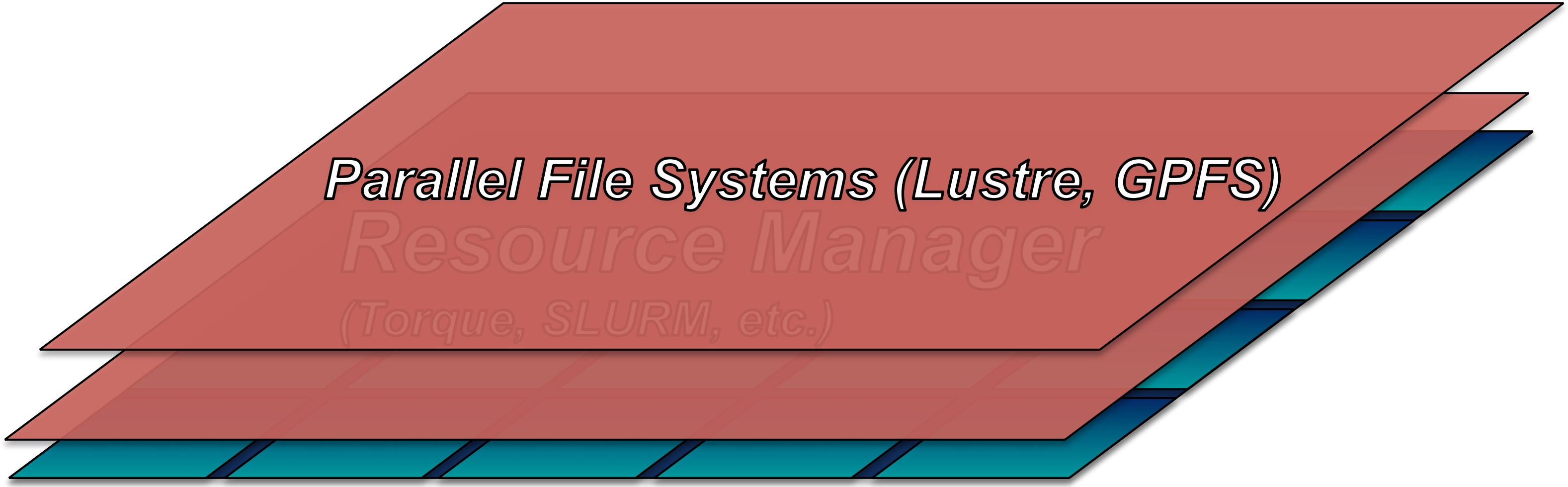
Can We Run Big Data and Deep Learning Jobs on Existing HPC Infrastructure?



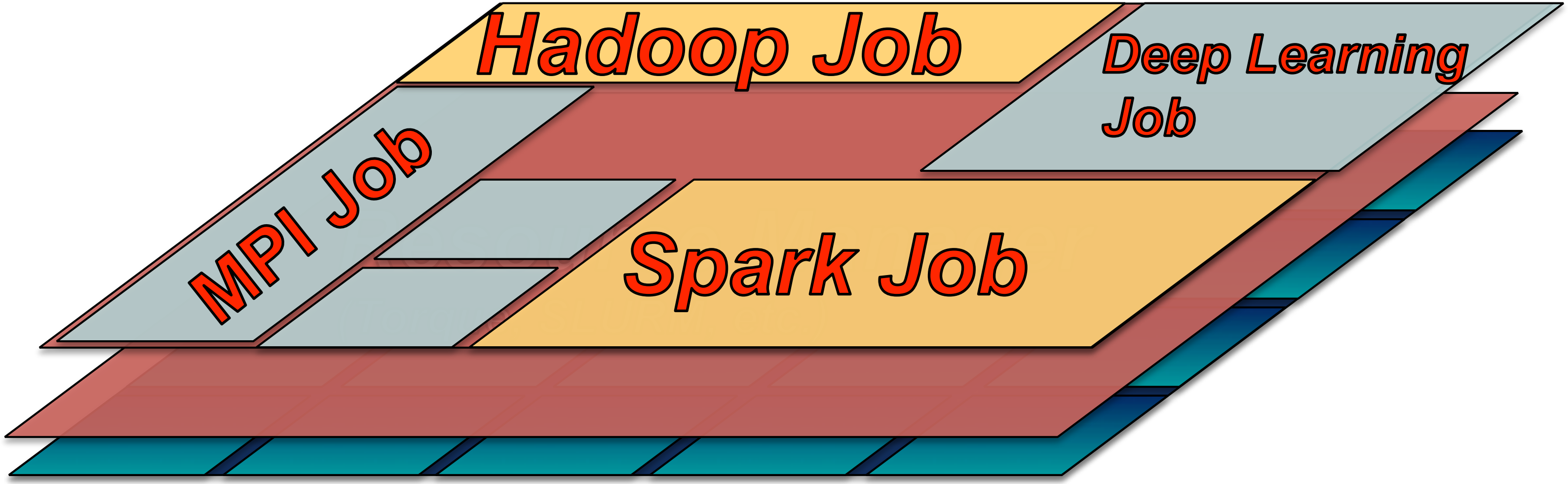
Can We Run Big Data and Deep Learning Jobs on Existing HPC Infrastructure?

Resource Manager
(Torque, SLURM, etc.)

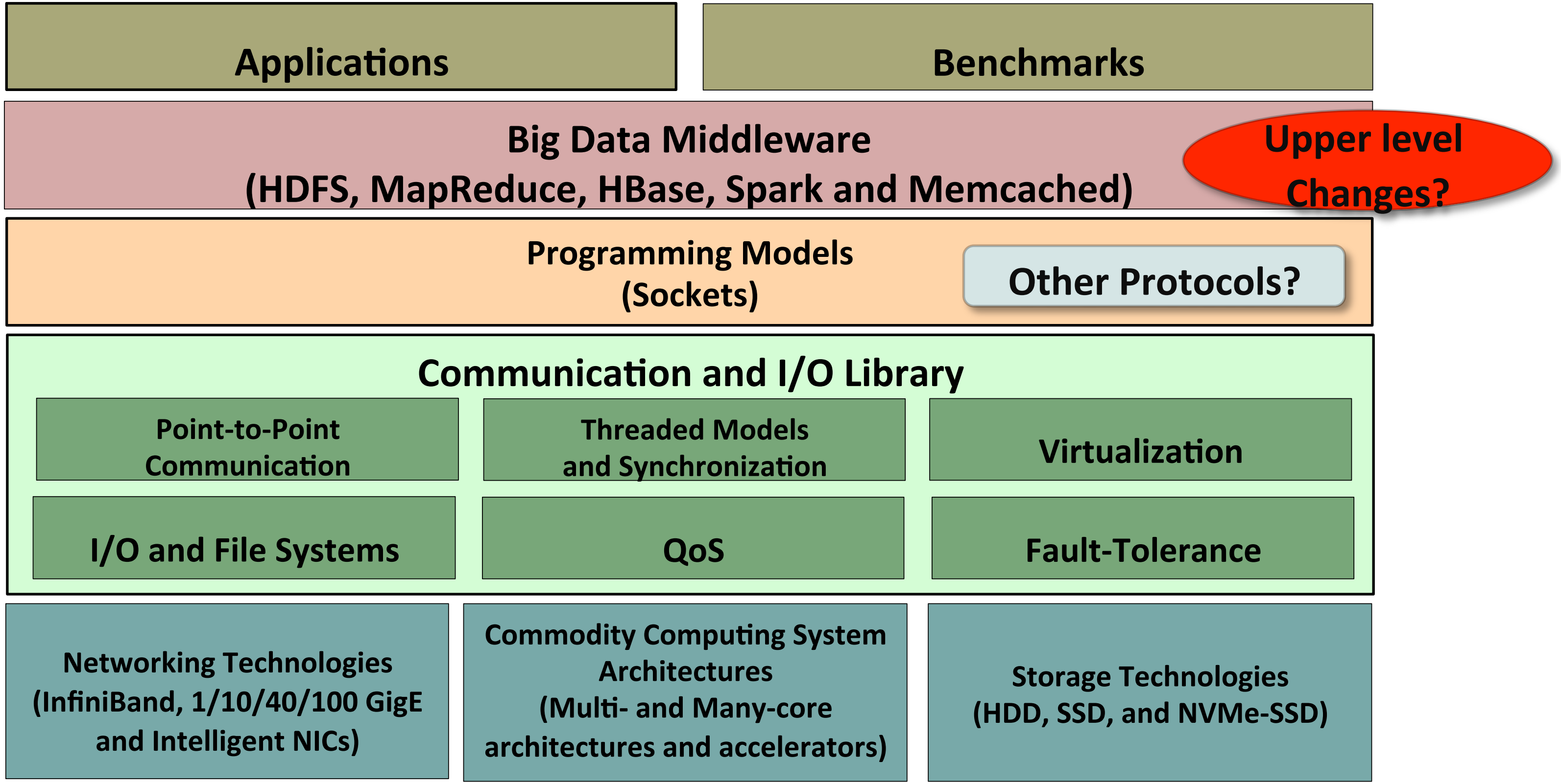
Can We Run Big Data and Deep Learning Jobs on Existing HPC Infrastructure?



Can We Run Big Data and Deep Learning Jobs on Existing HPC Infrastructure?



Designing Communication and I/O Libraries for Big Data Systems: Challenges



Applications

Benchmarks

Big Data Middleware
(HDFS, MapReduce, HBase, Spark and Memcached)

Upper level Changes?

Programming Models
(Sockets)

Other Protocols?

Communication and I/O Library

Point-to-Point Communication

Threaded Models and Synchronization

Virtualization

I/O and File Systems

QoS

Fault-Tolerance

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

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

Storage Technologies
(HDD, SSD, and NVMe-SSD)

Presentation Outline

- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

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
- Users Base: 260 organizations from 31 countries
- More than 23,900 downloads from the project site

Available for InfiniBand and RoCE

Also run on Ethernet

HiBD
High-Performance
Big Data

Network Based Computing
Laboratory

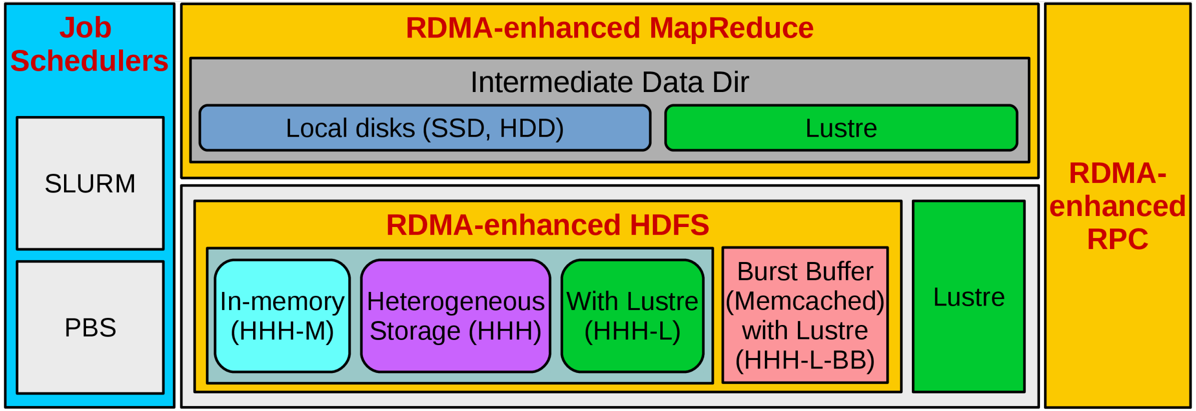
THE OHIO STATE
UNIVERSITY

RDMA for Apache Hadoop 2.x Distribution

- High-Performance Design of Hadoop over RDMA-enabled Interconnects
  - High performance RDMA-enhanced design with native InfiniBand and RoCE support at the verbs-level for HDFS, MapReduce, and RPC components
  - Enhanced HDFS with in-memory and heterogeneous storage
  - High performance design of MapReduce over Lustre
  - Memcached-based burst buffer for MapReduce over Lustre-integrated HDFS (HHH-L-BB mode)
  - Plugin-based architecture supporting RDMA-based designs for Apache Hadoop, CDH and HDP
  - Easily configurable for different running modes (HHH, HHH-M, HHH-L, HHH-L-BB, and MapReduce over Lustre) and different protocols (native InfiniBand, RoCE, and IPoIB)
- Current release: 1.2.0
  - Based on Apache Hadoop 2.8.0
  - Compliant with Apache Hadoop 2.8.0, HDP 2.5.0.3 and CDH 5.8.2 APIs and applications
  - Tested with
    - Mellanox InfiniBand adapters (DDR, QDR, FDR, and EDR)
    - RoCE support with Mellanox adapters
    - Various multi-core platforms
    - Different file systems with disks and SSDs and Lustre

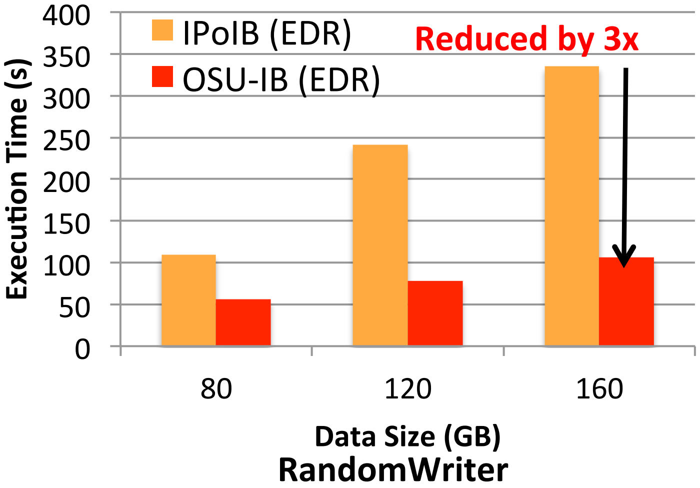
http://hibd.cse.ohio-state.edu

Different Modes of RDMA for Apache Hadoop 2.x

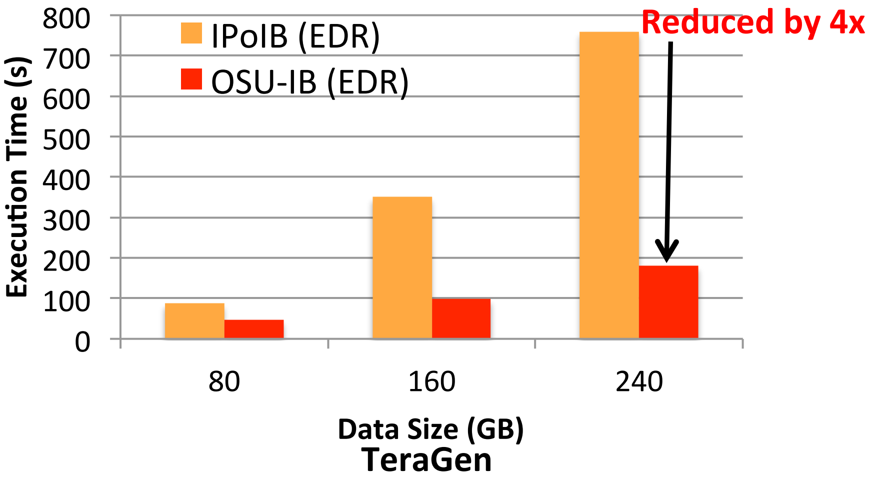


- **HHH**: Heterogeneous storage devices with hybrid replication schemes are supported in this mode of operation to have better fault-tolerance as well as performance. This mode is enabled by **default** in the package.
- **HHH-M**: A high-performance in-memory based setup has been introduced in this package that can be utilized to perform all I/O operations in-memory and obtain as much performance benefit as possible.
- **HHH-L**: With parallel file systems integrated, HHH-L mode can take advantage of the Lustre available in the cluster.
- **HHH-L-BB**: This mode deploys a Memcached-based burst buffer system to reduce the bandwidth bottleneck of shared file system access. The burst buffer design is hosted by Memcached servers, each of which has a local SSD.
- **MapReduce over Lustre, with/without local disks**: Besides, HDFS based solutions, this package also provides support to run MapReduce jobs on top of Lustre alone. Here, two different modes are introduced: with local disks and without local disks.
- **Running with Slurm and PBS**: Supports deploying RDMA for Apache Hadoop 2.x with Slurm and PBS in different running modes (HHH, HHH-M, HHH-L, and MapReduce over Lustre).

Performance Numbers of RDMA for Apache Hadoop 2.x – RandomWriter & TeraGen in OSU-RI2 (EDR)



RandomWriter

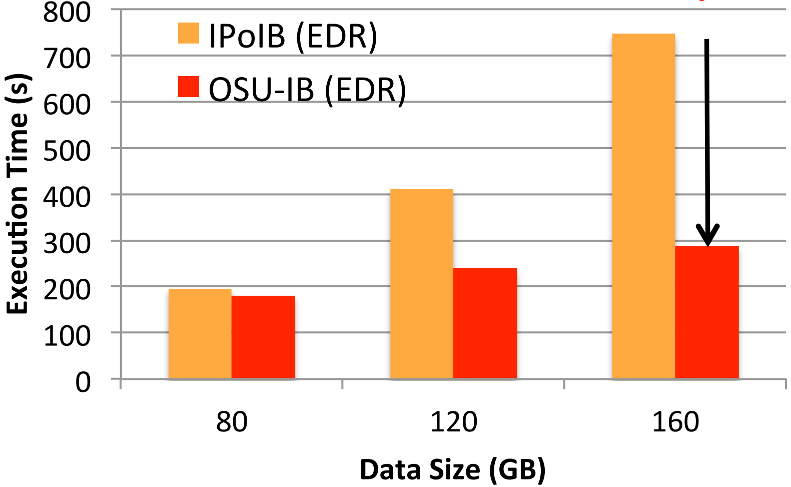


TeraGen

Cluster with 8 Nodes with a total of 64 maps

- RandomWriter
  - 3x improvement over IPoIB for 80-160 GB file size
- TeraGen
  - 4x improvement over IPoIB for 80-240 GB file size

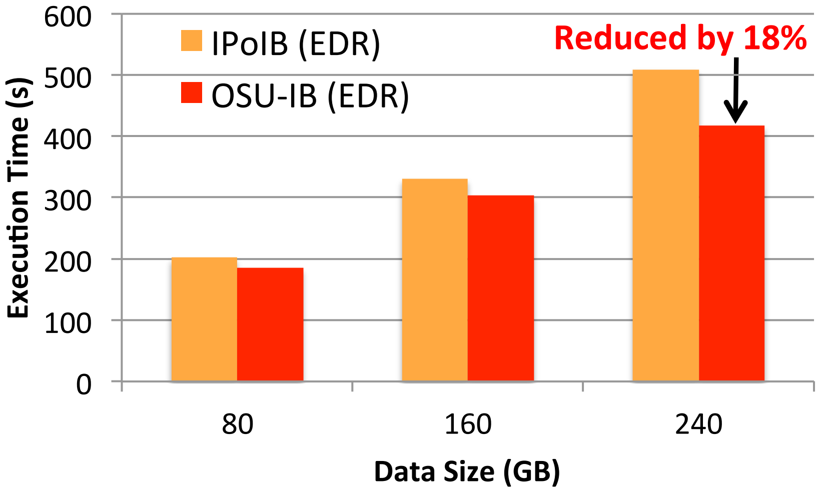
Performance Numbers of RDMA for Apache Hadoop 2.x – Sort & TeraSort in OSU-RI2 (EDR)



Reduced by 61%

Sort

Cluster with 8 Nodes with a total of 64 maps and 14 reduces



Reduced by 18%

TeraSort

Cluster with 8 Nodes with a total of 64 maps and 32 reduces

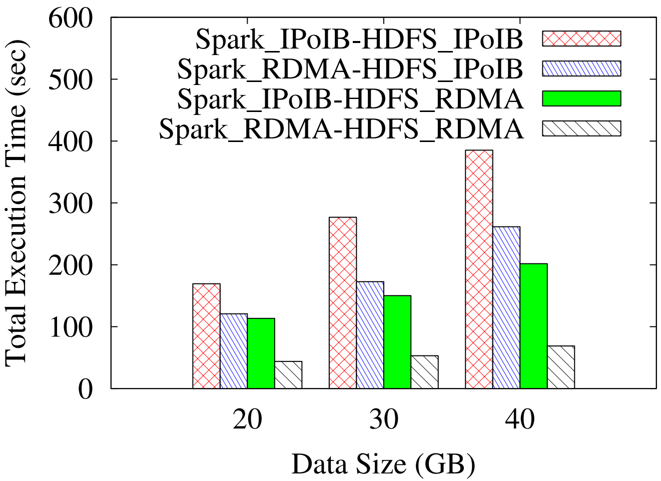
- Sort
  - 61% improvement over IPoIB for 80-160 GB data

- TeraSort
  - 18% improvement over IPoIB for 80-240 GB data

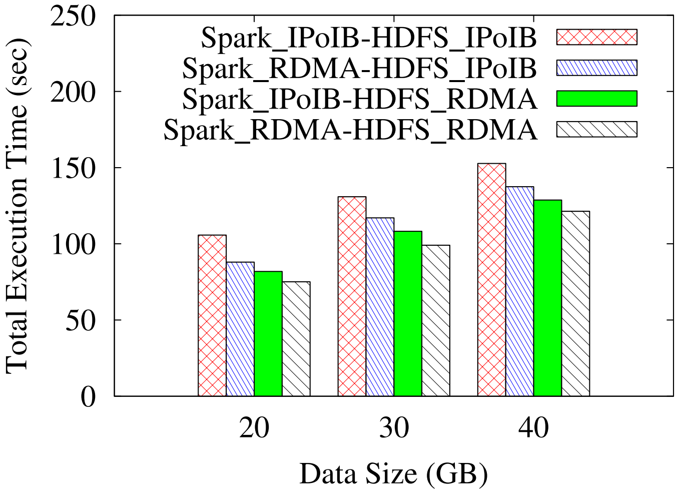
RDMA for Apache Spark Distribution

- High-Performance Design of Spark over RDMA-enabled Interconnects
  - RDMA-enhanced design with native InfiniBand and RoCE support at the verbs-level for Spark
  - RDMA-based data shuffle and SEDA-based shuffle architecture
  - Non-blocking and chunk-based data transfer
  - RDMA support for Spark SQL
  - Integration with HHH in RDMA for Apache Hadoop
  - Easily configurable for different protocols (native InfiniBand, RoCE, and IPoIB)
- Current release: 0.9.4
  - Based on Apache Spark 2.1.0
  - Tested with
    - Mellanox InfiniBand adapters (DDR, QDR and FDR)
    - RoCE support with Mellanox adapters
    - Various multi-core platforms
    - RAM disks, SSDs, and HDD
  - http://hibd.cse.ohio-state.edu

Performance Evaluation on SDSC-Comet-IB-FDR – with RDMA-HDFS



8 Worker Nodes, HiBench Sort Total Time



8 Worker Nodes, HiBench TeraSort Total Time

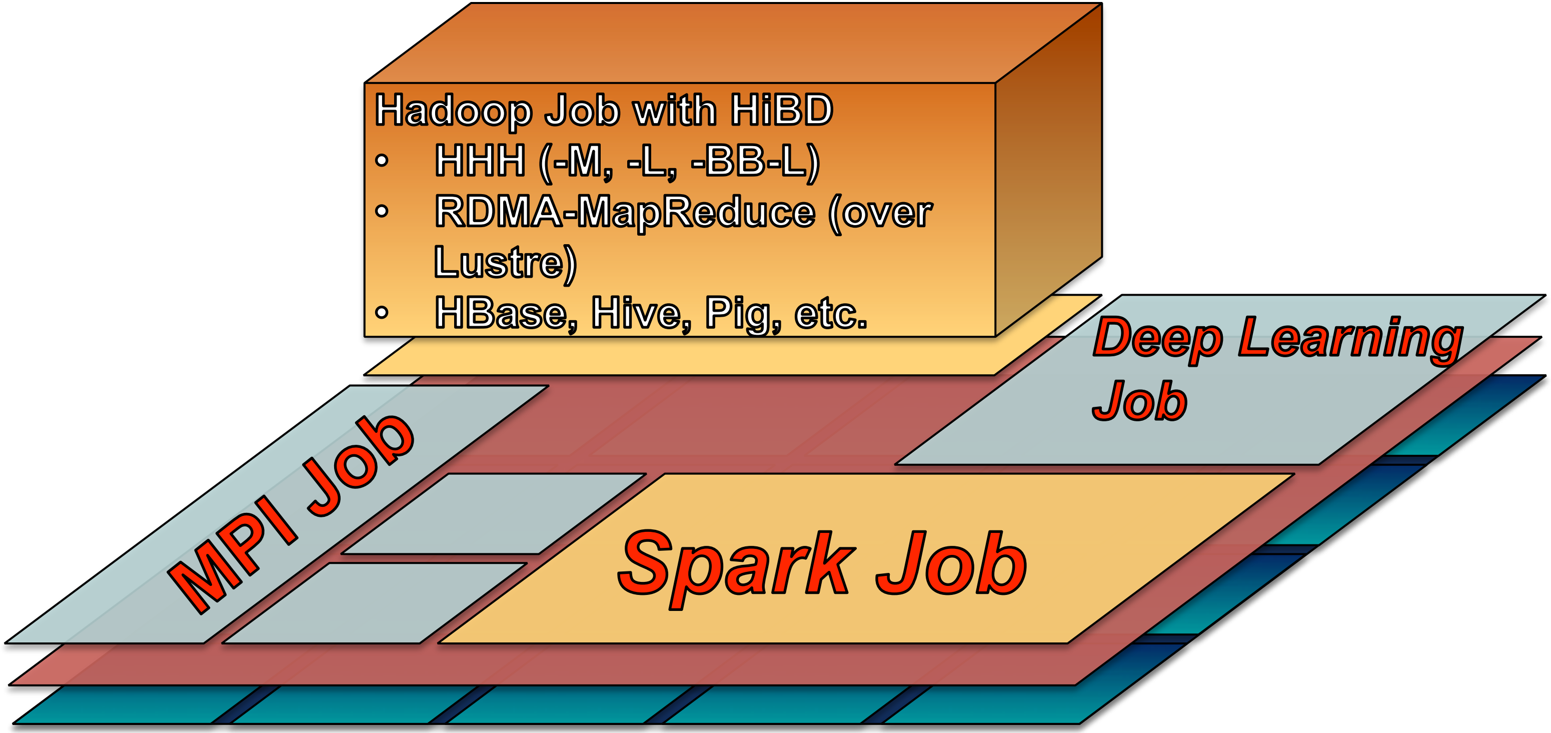
- InfiniBand FDR, SSD, YARN-Client Mode, 192 Cores
- Benefit of RDMA-Spark and the capability of combining with other advanced technologies, such as RDMA-HDFS
- A combined version of ‘RDMA-Spark+RDMA-HDFS’ can achieve the best performance
- RDMA vs. IPoIB
  - HiBench Sort: Total time reduced by up to 38% over IPoIB through RDMA-Spark, up to 82% through RDMA-Spark+RDMA-HDFS
  - HiBench TeraSort: Total time reduced by up to 17% over IPoIB through RDMA-Spark, up to 29% through RDMA-Spark+RDMA-HDFS

HiBD Packages on SDSC Comet and Chameleon Cloud

- RDMA for Apache Hadoop 2.x and RDMA for Apache Spark are installed and available on SDSC Comet.
  - Examples for various modes of usage are available in:
    - RDMA for Apache Hadoop 2.x: /share/apps/examples/HADOOP
    - RDMA for Apache Spark: /share/apps/examples/SPARK/
  - Please email help@xsede.org (reference Comet as the machine, and SDSC as the site) if you have any further questions about usage and configuration.
- RDMA for Apache Hadoop is also available on Chameleon Cloud as an appliance
  - https://www.chameleoncloud.org/appliances/17/

M. Tatineni, X. Lu, D. J. Choi, A. Majumdar, and D. K. Panda, Experiences and Benefits of Running RDMA Hadoop and Spark on SDSC Comet, XSEDE'16, July 2016

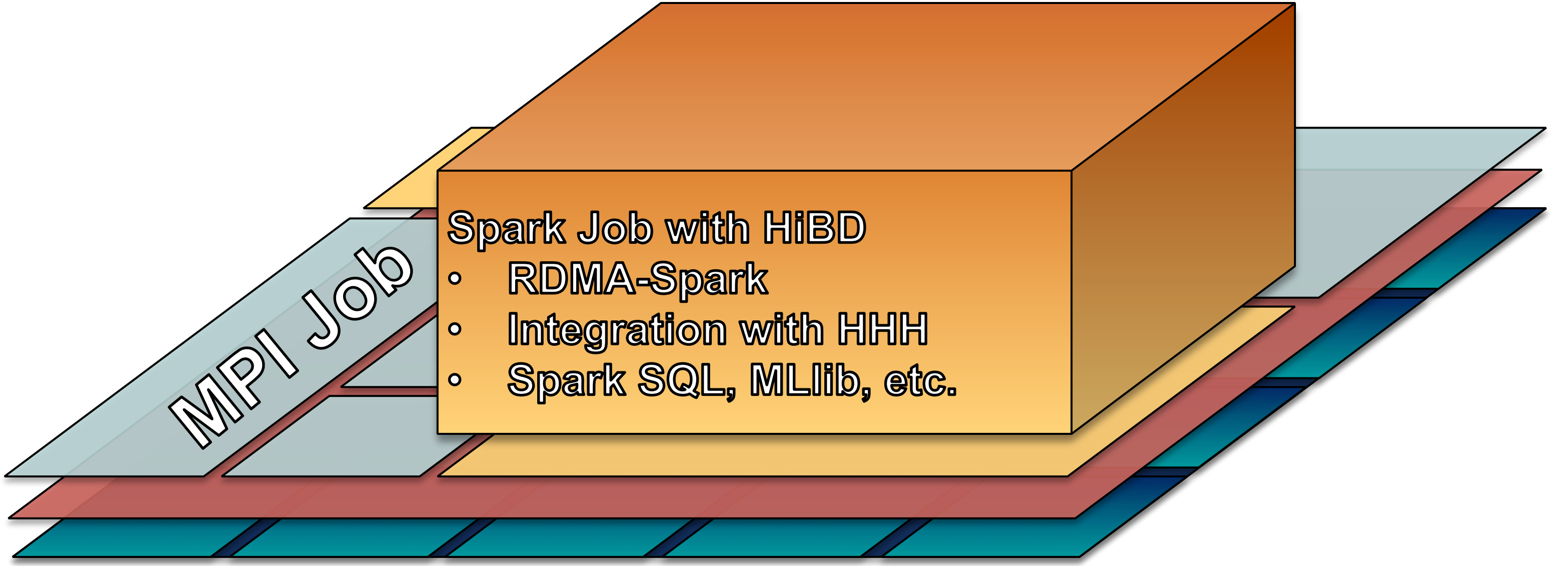
Using HiBD Packages for Big Data Processing on Existing HPC Infrastructure



Hadoop Job with HiBD

- HHH (-M, -L, -BB-L)
- RDMA-MapReduce (over Lustre)
- HBase, Hive, Pig, etc.

Using HiBD Packages for Big Data Processing on Existing HPC Infrastructure



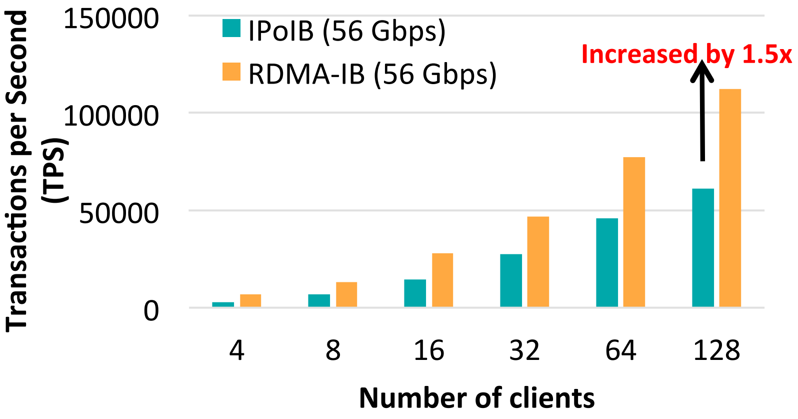
Spark Job with HiBD

- RDMA-Spark
- Integration with HHH
- Spark SQL, MLlib, etc.

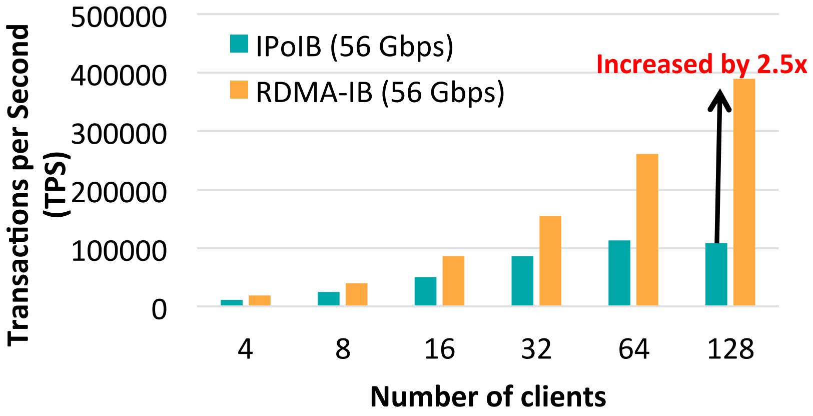
RDMA for Apache HBase Distribution

- High-Performance Design of HBase over RDMA-enabled Interconnects
  - High performance RDMA-enhanced design with native InfiniBand and RoCE support at the verbs-level for HBase
  - Compliant with Apache HBase 1.1.2 APIs and applications
  - On-demand connection setup
  - Easily configurable for different protocols (native InfiniBand, RoCE, and IPoIB)
- Current release: 0.9.1
  - Based on Apache HBase 1.1.2
  - Tested with
    - Mellanox InfiniBand adapters (DDR, QDR, FDR, and EDR)
    - RoCE support with Mellanox adapters
    - Various multi-core platforms
  - http://hibd.cse.ohio-state.edu

Performance Numbers of RDMA for Apache HBase – YCSB in SDSC-Comet



YCSB Workload A



YCSB Workload C

Cluster with 4 Region servers

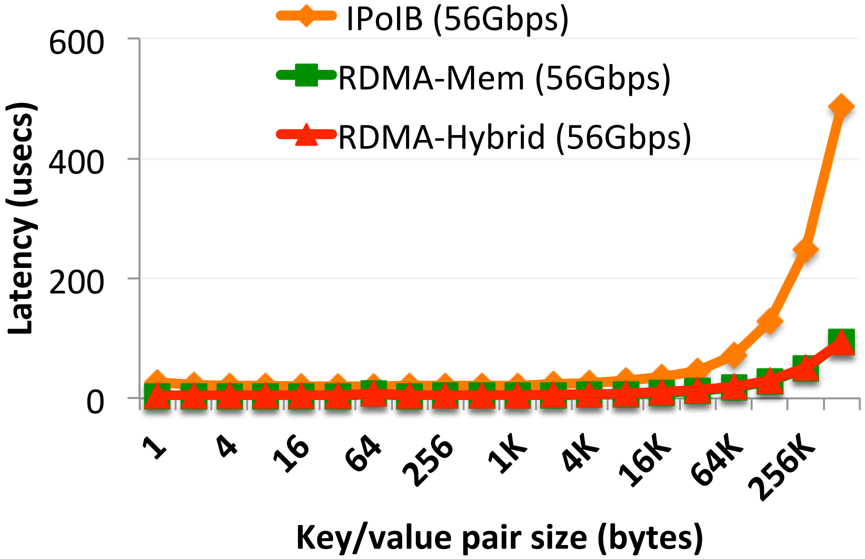
- YCSB Workload A
  - Up to 1.5x improvement over IPoIB

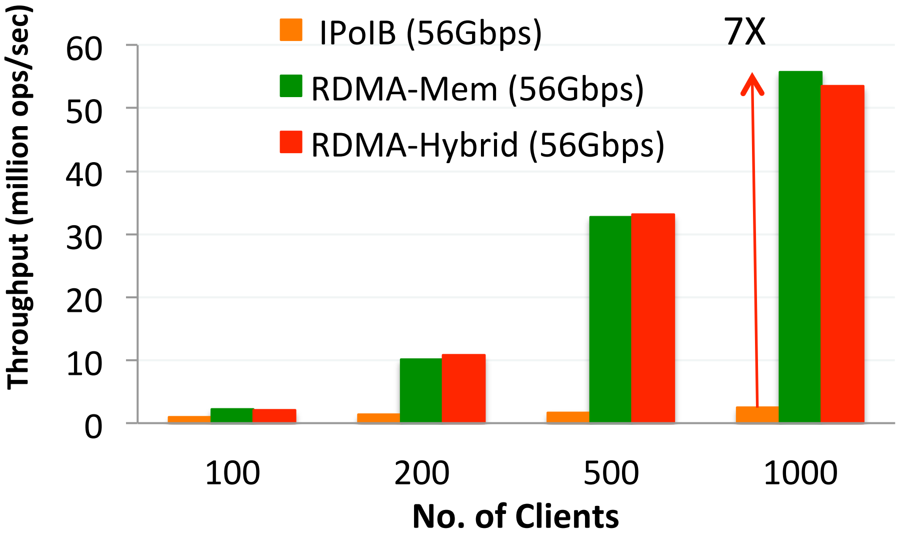
- YCSB Workload C
  - Up to 2.5x improvement over IPoIB

RDMA for Memcached Distribution

- High-Performance Design of Memcached over RDMA-enabled Interconnects
  - High performance RDMA-enhanced design with native InfiniBand and RoCE support at the verbs-level for Memcached and libMemcached components
  - High performance design of SSD-Assisted Hybrid Memory
  - Non-Blocking Libmemcached Set/Get API extensions
  - Support for burst-buffer mode in Lustre-integrated design of HDFS in RDMA for Apache Hadoop-2.x
  - Easily configurable for native InfiniBand, RoCE and the traditional sockets-based support (Ethernet and InfiniBand with IPoIB)
- Current release: 0.9.5
  - Based on Memcached 1.4.24 and libMemcached 1.0.18
  - Compliant with libMemcached APIs and applications
  - Tested with
    - Mellanox InfiniBand adapters (DDR, QDR, FDR, and EDR)
    - RoCE support with Mellanox adapters
    - Various multi-core platforms
    - SSD
  - http://hibd.cse.ohio-state.edu

Performance Numbers of RDMA for Memcached on SDSC-Comet – OHB





- ohb_memlat & ohb_memthr latency & throughput micro-benchmarks
- Memcached-RDMA can
  - improve query latency by up to 71% over IPoIB (FDR 56Gbps)
  - improve throughput by up to 7X over IPoIB (FDR 56Gbps)
  - No overhead in using hybrid mode when all data can fit in memory

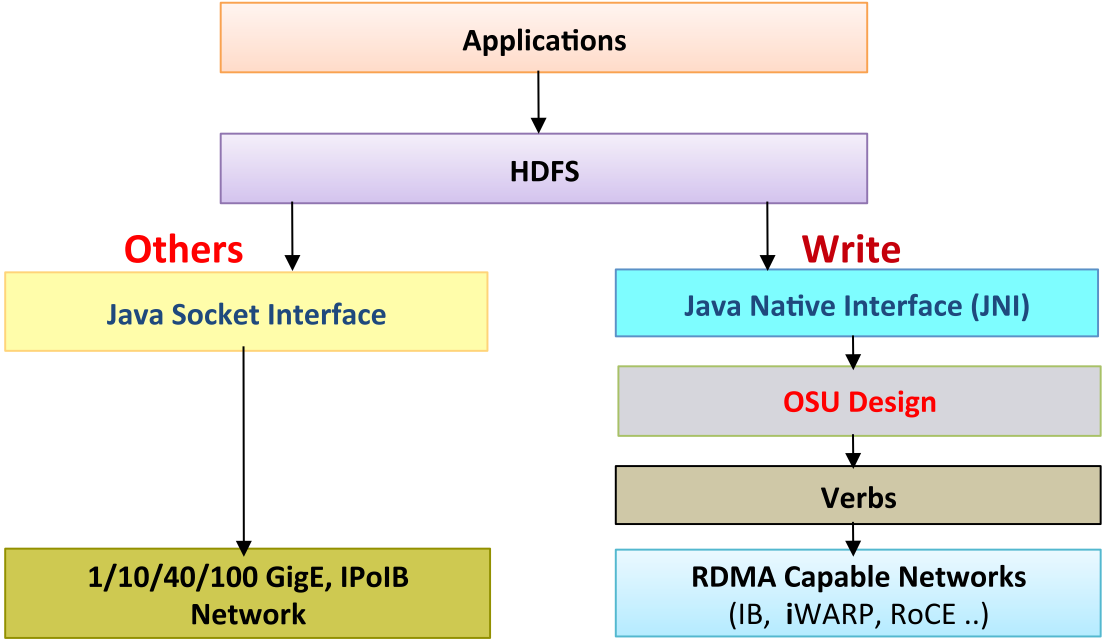
OSU HiBD Micro-Benchmark (OHB) Suite – HDFS, Memcached, HBase, and Spark

- Micro-benchmarks for Hadoop Distributed File System (HDFS)
  - Sequential Write Latency (**SWL**) Benchmark, Sequential Read Latency (**SRL**) Benchmark, Random Read Latency (**RRL**) Benchmark, Sequential Write Throughput (**SWT**) Benchmark, Sequential Read Throughput (**SRT**) Benchmark
  - Support benchmarking of
    - Apache Hadoop 1.x and 2.x HDFS, Hortonworks Data Platform (HDP) HDFS, Cloudera Distribution of Hadoop (CDH) HDFS
- Micro-benchmarks for Memcached
  - **Get** Benchmark, **Set** Benchmark, and **Mixed** Get/Set Benchmark, **Non-Blocking API** Latency Benchmark**, Hybrid Memory** Latency Benchmark
- Micro-benchmarks for HBase
  - **Get** Latency Benchmark, **Put** Latency Benchmark
- Micro-benchmarks for Spark
  - GroupBy, SortBy
- Current release: 0.9.2
- **http://hibd.cse.ohio-state.edu**

Presentation Outline

- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

Design Overview of HDFS with RDMA

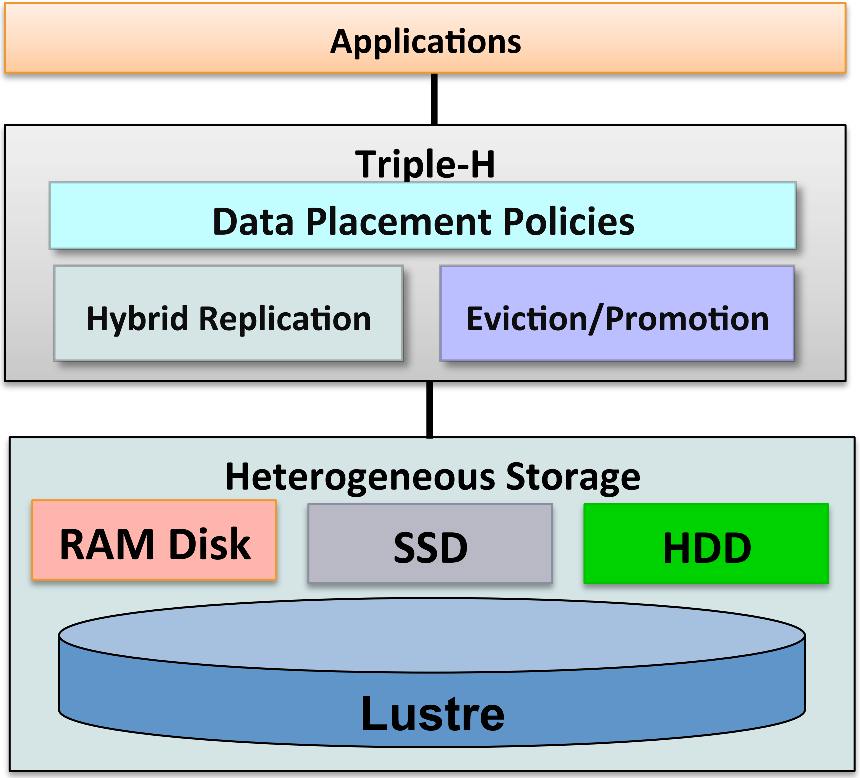


- Design Features
  - RDMA-based HDFS write
  - RDMA-based HDFS replication
  - Parallel replication support
  - On-demand connection setup
  - InfiniBand/RoCE support

N. S. Islam, M. W. Rahman, J. Jose, R. Rajachandrasekar, H. Wang, H. Subramoni, C. Murthy and D. K. Panda , High Performance RDMA-Based Design of HDFS over InfiniBand , Supercomputing (SC), Nov 2012

N. Islam, X. Lu, W. Rahman, and D. K. Panda, SOR-HDFS: A SEDA-based Approach to Maximize Overlapping in RDMA-Enhanced HDFS, HPDC '14, June 2014

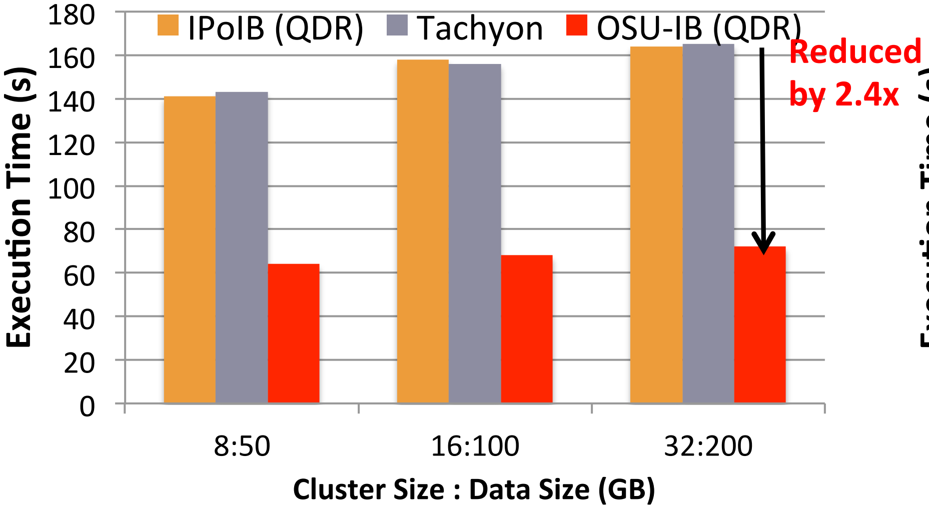
Enhanced HDFS with In-Memory and Heterogeneous Storage



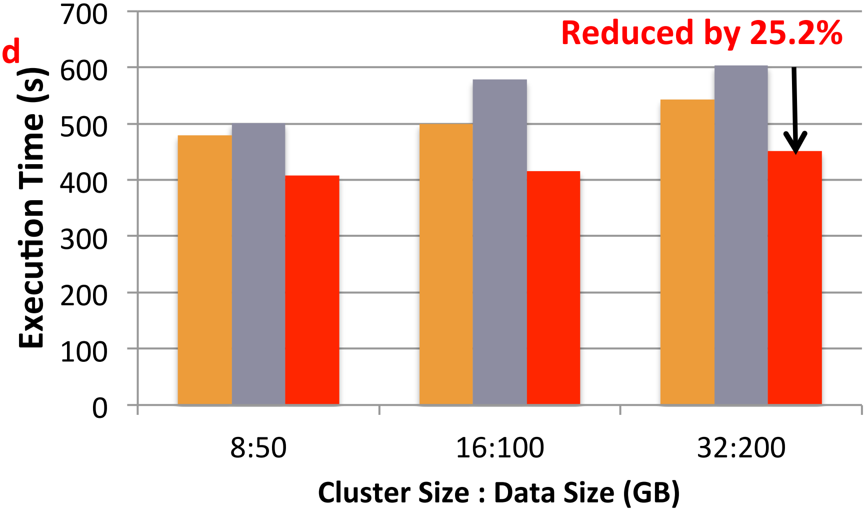
- Design Features
  - Three modes
    - Default (HHH)
    - In-Memory (HHH-M)
    - Lustre-Integrated (HHH-L)
  - Policies to efficiently utilize the heterogeneous storage devices
    - RAM, SSD, HDD, Lustre
  - Eviction/Promotion based on data usage pattern
  - Hybrid Replication
  - Lustre-Integrated mode:
    - Lustre-based fault-tolerance

N. Islam, X. Lu, M. W. Rahman, D. Shankar, and D. K. Panda, Triple-H: A Hybrid Approach to Accelerate HDFS on HPC Clusters with Heterogeneous Storage Architecture, CCGrid '15, May 2015

Evaluation with Spark on SDSC Gordon (HHH vs. Alluxio)



TeraGen

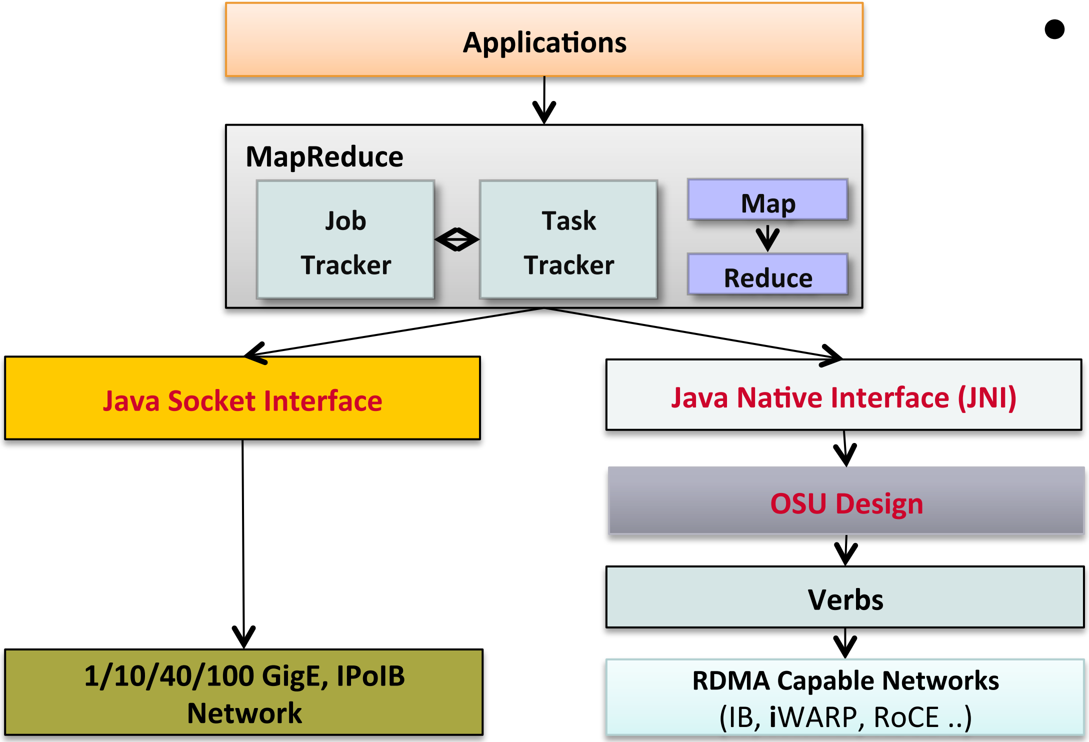


TeraSort

- For 200GB TeraGen on 32 nodes
  - Spark-TeraGen: HHH has 2.4x improvement over Tachyon; 2.3x over HDFS-IPoIB (QDR)
  - Spark-TeraSort: HHH has 25.2% improvement over Tachyon; 17% over HDFS-IPoIB (QDR)

N. Islam, M. W. Rahman, X. Lu, D. Shankar, and D. K. Panda, Performance Characterization and Acceleration of In-Memory File Systems for Hadoop and Spark Applications on HPC Clusters, IEEE BigData ’15, October 2015

Design Overview of MapReduce with RDMA

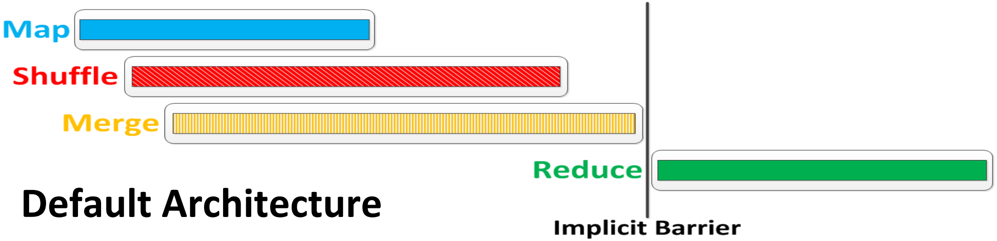


Applications
MapReduce
Job Tracker
Task Tracker
Map
Reduce
Java Socket Interface
Java Native Interface (JNI)
OSU Design
Verbs
1/10/40/100 GigE, IPoIB Network
RDMA Capable Networks (IB, iWARP, RoCE ..)

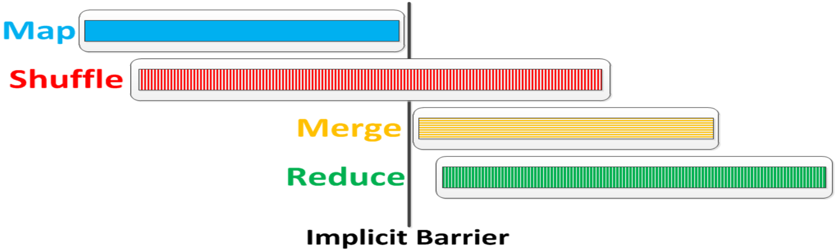
- Design Features
  - RDMA-based shuffle
  - Prefetching and caching map output
  - Efficient Shuffle Algorithms
  - In-memory merge
  - On-demand Shuffle Adjustment
  - Advanced overlapping
    - map, shuffle, and merge
    - shuffle, merge, and reduce
  - On-demand connection setup
  - InfiniBand/RoCE support

M. W. Rahman, N. S. Islam, X. Lu, J. Jose, H. Subramon, H. Wang, and D. K. Panda, High-Performance RDMA-based Design of Hadoop MapReduce over InfiniBand, HPDIC Workshop, held in conjunction with IPDPS, May 2013

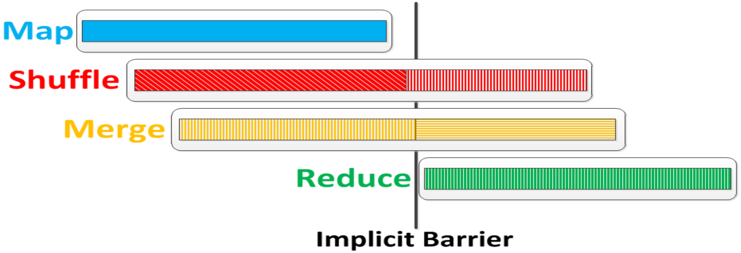
Advanced Overlapping among Different Phases



Default Architecture



Enhanced Overlapping with In-Memory Merge



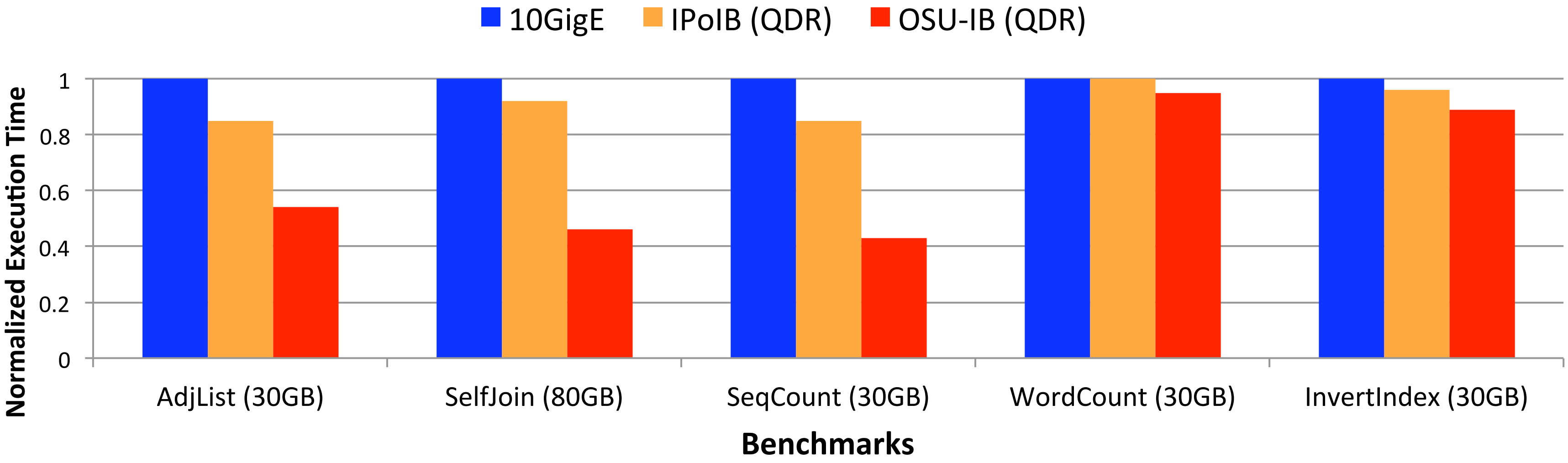
Advanced Hybrid Overlapping

A hybrid approach to achieve maximum possible overlapping in MapReduce across all phases compared to other approaches

- Efficient Shuffle Algorithms
- Dynamic and Efficient Switching
- On-demand Shuffle Adjustment

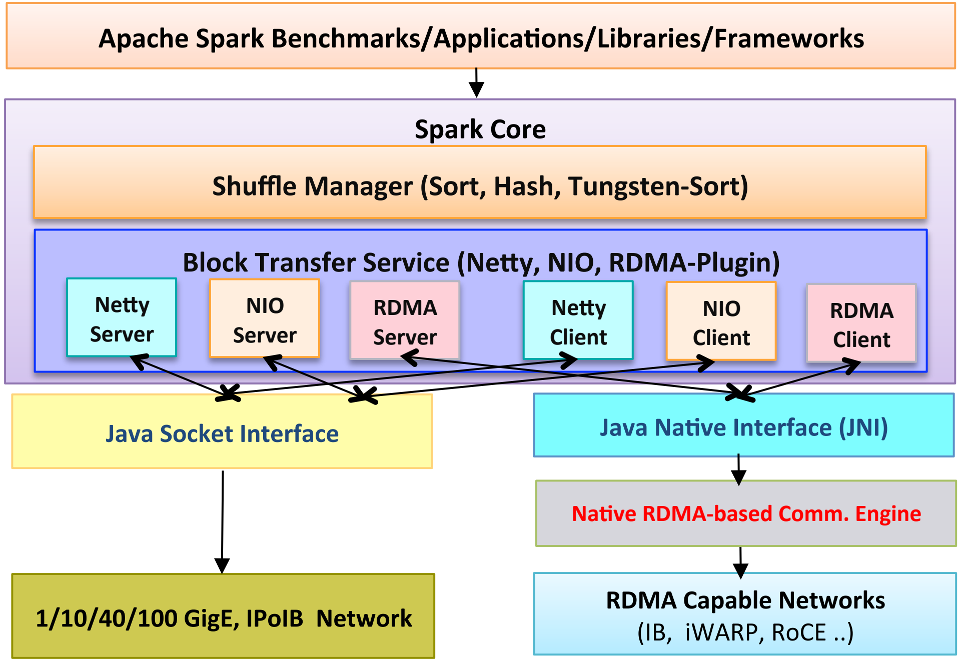
M. W. Rahman, X. Lu, N. S. Islam, and D. K. Panda, HOMR: A Hybrid Approach to Exploit Maximum Overlapping in MapReduce over High Performance Interconnects, ICS, June 2014

Evaluations using PUMA Workload



- 50% improvement in Self Join over IPoIB (QDR) for 80 GB data size
- 49% improvement in Sequence Count over IPoIB (QDR) for 30 GB data size

Design Overview of Spark with RDMA


Apache Spark Benchmarks/Applications/Libraries/Frameworks
Spark Core
Shuffle Manager (Sort, Hash, Tungsten-Sort)
Block Transfer Service (Netty, NIO, RDMA-Plugin)
Netty Server
NIO Server
RDMA Server
Netty Client
NIO Client
RDMA Client
Java Socket Interface
Java Native Interface (JNI)
Native RDMA-based Comm. Engine
1/10/40/100 GigE, IPoIB Network
RDMA Capable Networks
(IB, iWARP, RoCE ..)

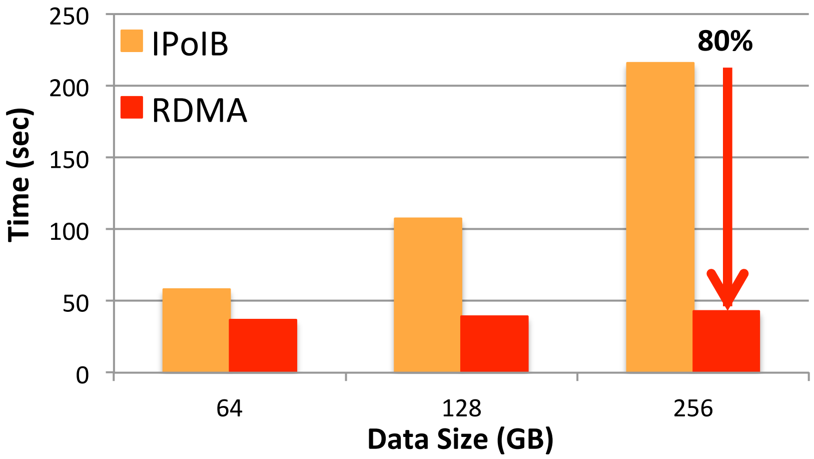
- Design Features
  - RDMA based shuffle plugin
  - SEDA-based architecture
  - Dynamic connection management and sharing
  - Non-blocking data transfer
  - Off-JVM-heap buffer management
  - InfiniBand/RoCE support

- Enables high performance RDMA communication, while supporting traditional socket interface
- JNI Layer bridges Scala based Spark with communication library written in native code

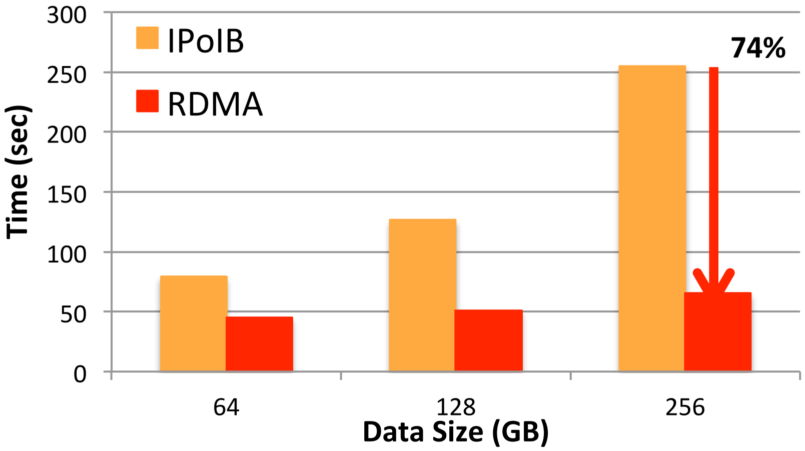
X. Lu, M. W. Rahman, N. Islam, D. Shankar, and D. K. Panda, Accelerating Spark with RDMA for Big Data Processing: Early Experiences, Int'l Symposium on High Performance Interconnects (HotI'14), August 2014

X. Lu, D. Shankar, S. Gugnani, and D. K. Panda, High-Performance Design of Apache Spark with RDMA and Its Benefits on Various Workloads, IEEE BigData '16, Dec. 2016.

Performance Evaluation on SDSC Comet – SortBy/GroupBy



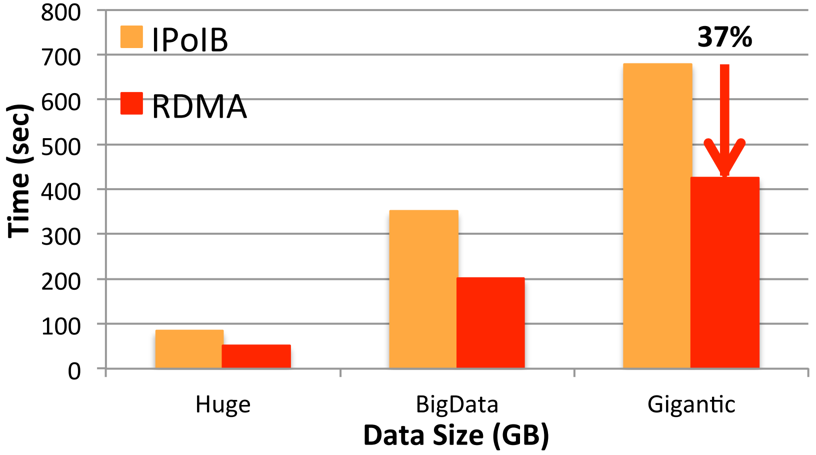
64 Worker Nodes, 1536 cores, SortByTest Total Time



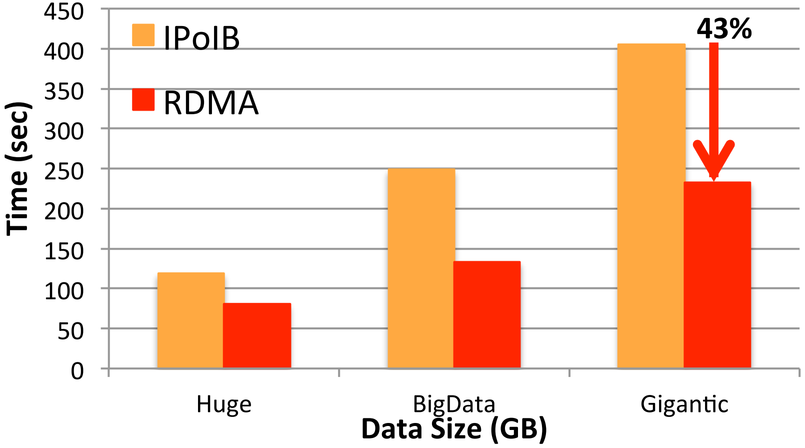
64 Worker Nodes, 1536 cores, GroupByTest Total Time

- InfiniBand FDR, SSD, 64 Worker Nodes, 1536 Cores, (1536M 1536R)
- RDMA-based design for Spark 1.5.1
- RDMA vs. IPoIB with 1536 concurrent tasks, single SSD per node.
  - SortBy: Total time reduced by up to 80% over IPoIB (56Gbps)
  - GroupBy: Total time reduced by up to 74% over IPoIB (56Gbps)

Performance Evaluation on SDSC Comet – HiBench PageRank



32 Worker Nodes, 768 cores, PageRank Total Time



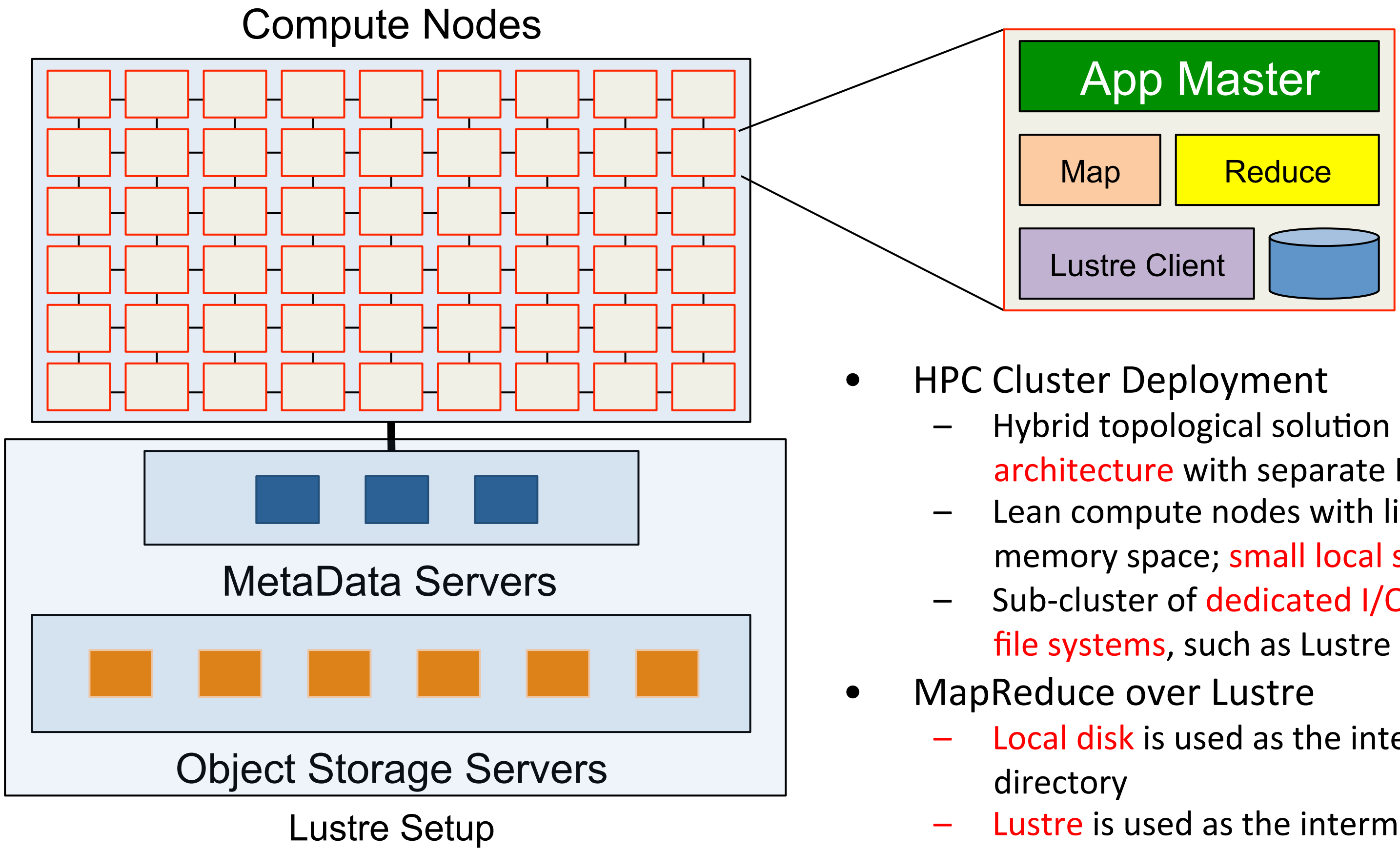
64 Worker Nodes, 1536 cores, PageRank Total Time

- InfiniBand FDR, SSD, 32/64 Worker Nodes, 768/1536 Cores, (768/1536M 768/1536R)
- RDMA-based design for Spark 1.5.1
- RDMA vs. IPoIB with 768/1536 concurrent tasks, single SSD per node.
  - 32 nodes/768 cores: Total time reduced by 37% over IPoIB (56Gbps)
  - 64 nodes/1536 cores: Total time reduced by 43% over IPoIB (56Gbps)

Presentation Outline

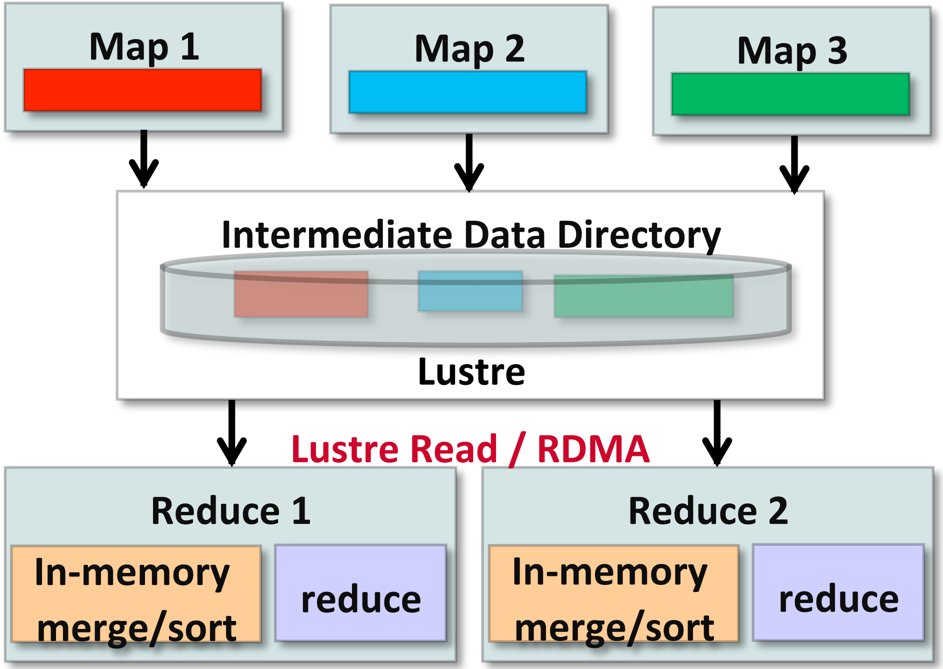
- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

Optimize Hadoop YARN MapReduce over Parallel File Systems



- HPC Cluster Deployment
  - Hybrid topological solution of Beowulf architecture with separate I/O nodes
  - Lean compute nodes with light OS; more memory space; small local storage
  - Sub-cluster of dedicated I/O nodes with parallel file systems, such as Lustre
- MapReduce over Lustre
  - Local disk is used as the intermediate data directory
  - Lustre is used as the intermediate data directory

Design Overview of Shuffle Strategies for MapReduce over Lustre

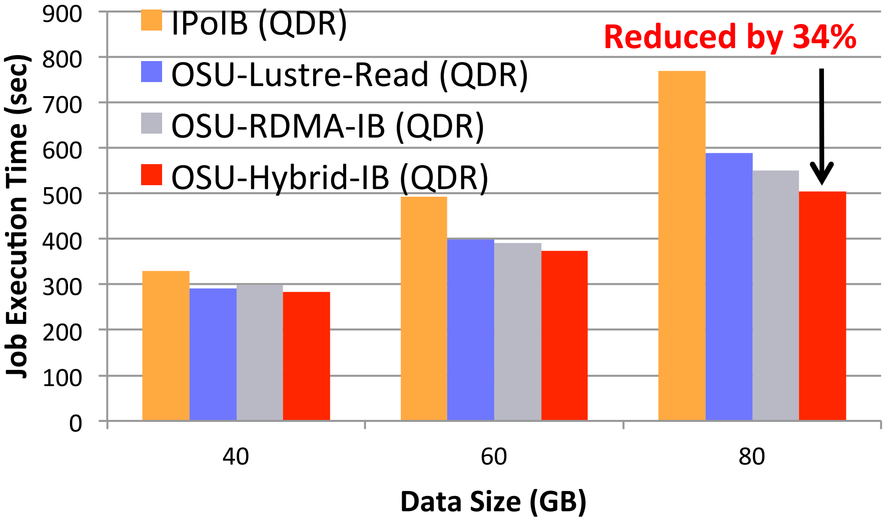


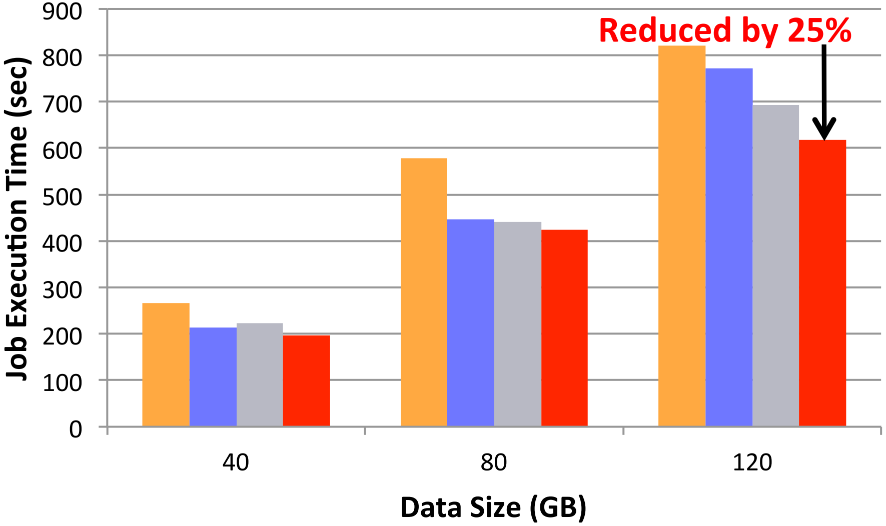
- Design Features
  - Two shuffle approaches
    - Lustre read based shuffle
    - RDMA based shuffle
  - Hybrid shuffle algorithm to take benefit from both shuffle approaches
  - Dynamically adapts to the better shuffle approach for each shuffle request based on profiling values for each Lustre read operation
  - In-memory merge and overlapping of different phases are kept similar to RDMA-enhanced MapReduce design

M. W. Rahman, X. Lu, N. S. Islam, R. Rajachandrasekar, and D. K. Panda, High Performance Design of YARN MapReduce on Modern HPC Clusters with Lustre and RDMA, IPDPS, May 2015

Case Study - Performance Improvement of MapReduce over Lustre on SDSC-Gordon

- Lustre is used as the intermediate data directory





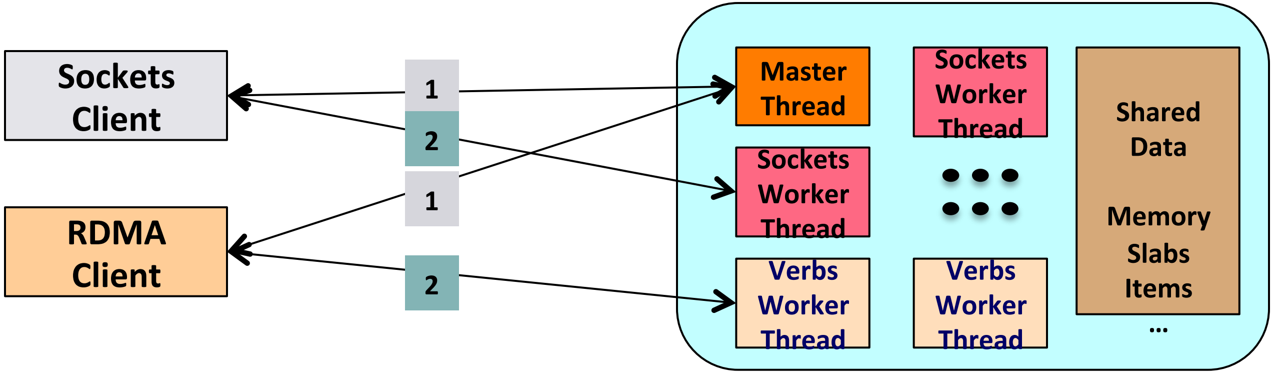
- For 80GB Sort in 8 nodes
  - 34% improvement over IPoIB (QDR)

- For 120GB TeraSort in 16 nodes
  - 25% improvement over IPoIB (QDR)

Presentation Outline

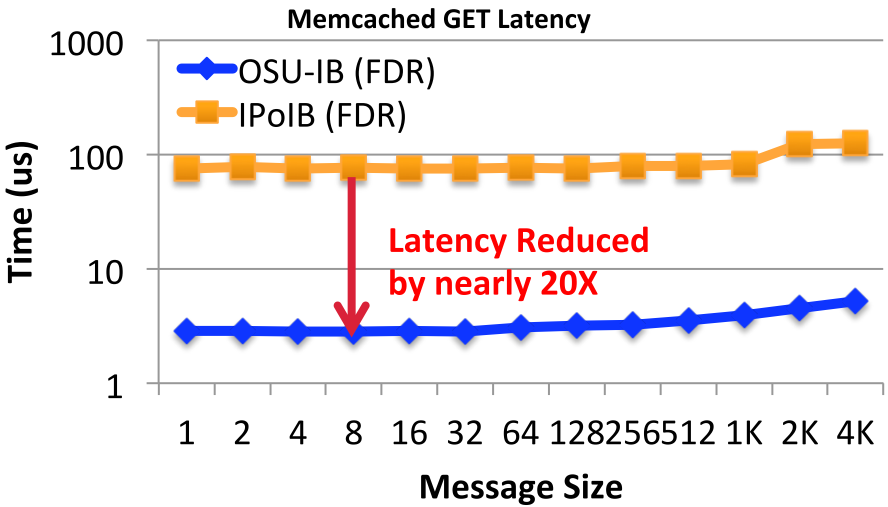
- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

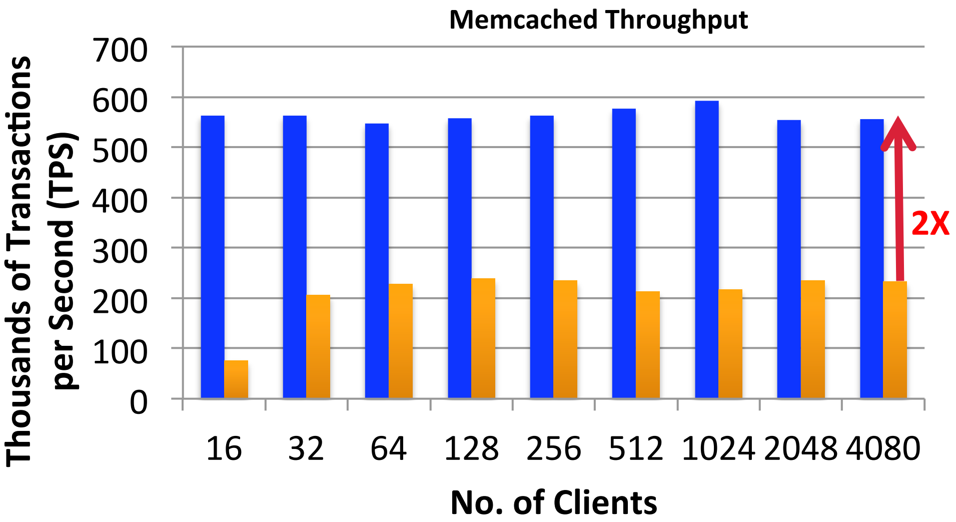
Memcached-RDMA Design



- Server and client perform a negotiation protocol
  - Master thread assigns clients to appropriate worker thread
- Once a client is assigned a verbs worker thread, it can communicate directly and is “bound” to that thread
- All other Memcached data structures are shared among RDMA and Sockets worker threads
- Memcached Server can serve both socket and verbs clients simultaneously
- Memcached applications need not be modified; uses verbs interface if available

Memcached Performance (FDR Interconnect)

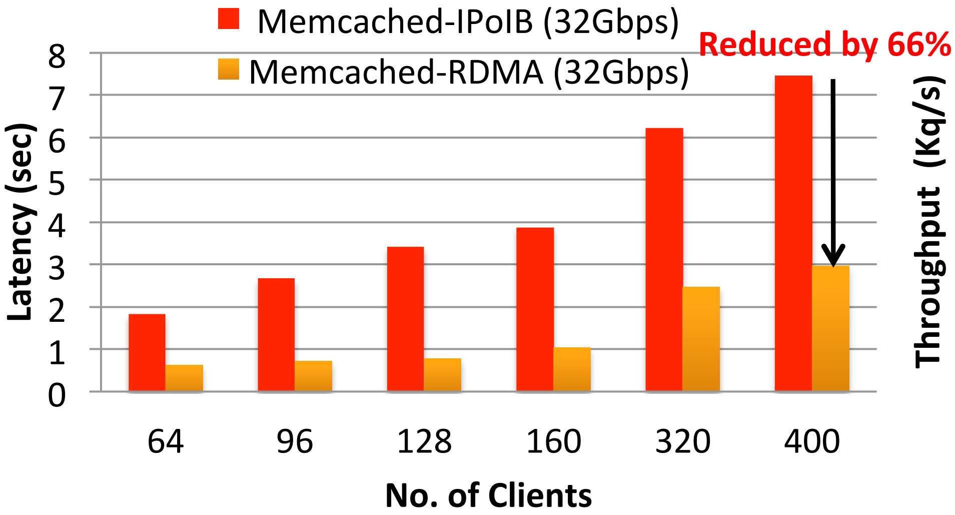


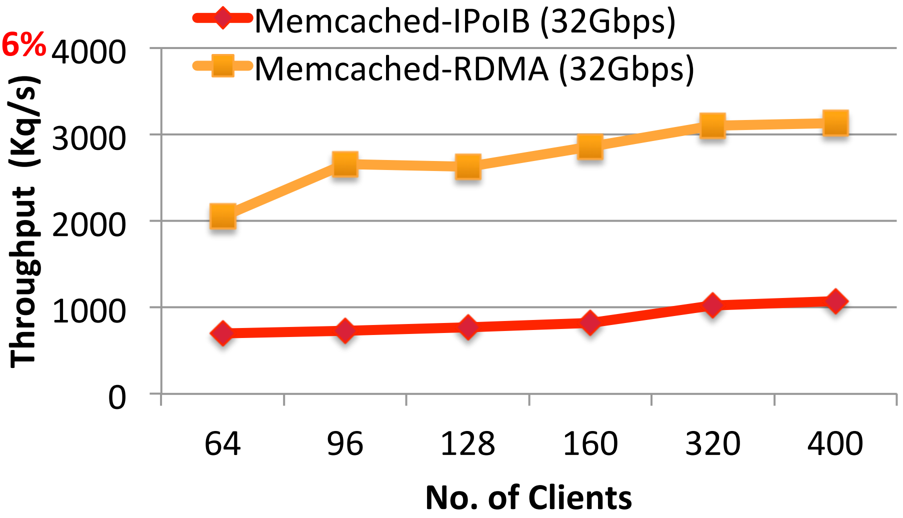


Experiments on TACC Stampede (Intel SandyBridge Cluster, IB: FDR)

- Memcached Get latency
  - 4 bytes OSU-IB: 2.84 us; IPoIB: 75.53 us
  - 2K bytes OSU-IB: 4.49 us; IPoIB: 123.42 us
- Memcached Throughput (4bytes)
  - 4080 clients OSU-IB: 556 Kops/sec, IPoIB: 233 Kops/s
  - Nearly 2X improvement in throughput

Micro-benchmark Evaluation for OLDP workloads

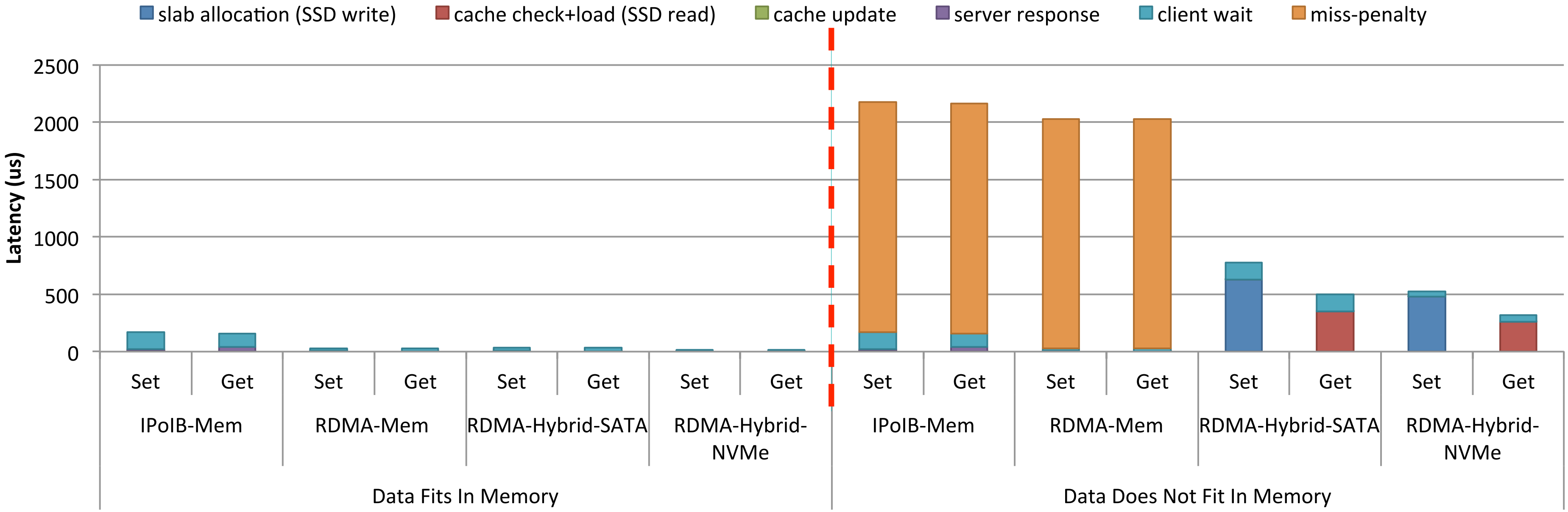




- Illustration with Read-Cache-Read access pattern using modified mysqlslap load testing tool
- Memcached-RDMA can
  - improve query latency by up to 66% over IPoIB (32Gbps)
  - throughput by up to 69% over IPoIB (32Gbps)

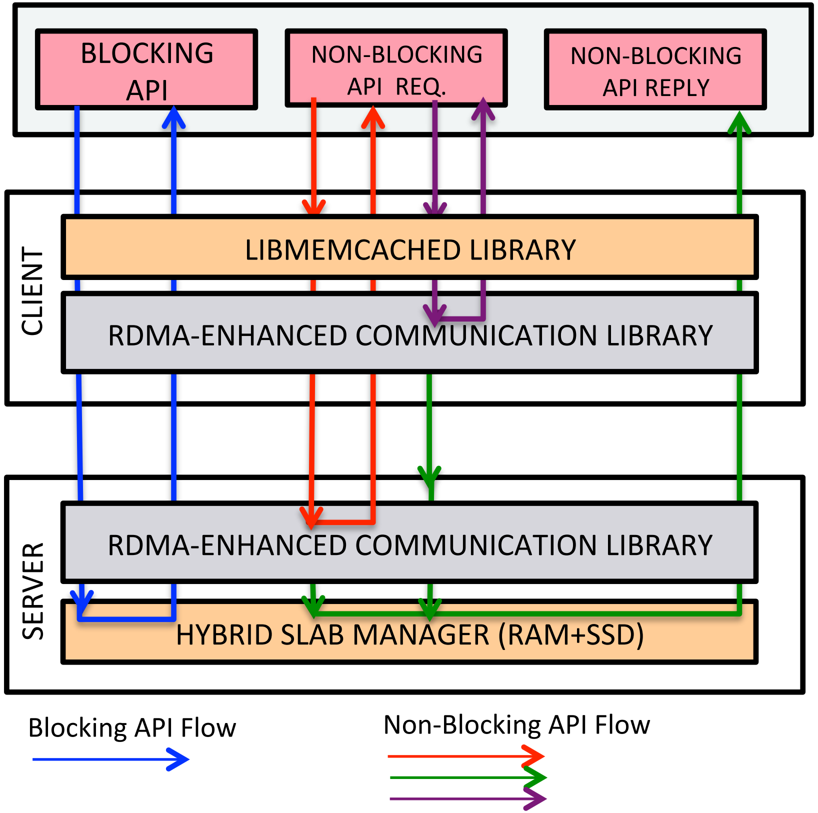
D. Shankar, X. Lu, J. Jose, M. W. Rahman, N. Islam, and D. K. Panda, Can RDMA Benefit On-Line Data Processing Workloads with Memcached and MySQL, ISPASS'15

Performance Evaluation on IB FDR + SATA/NVMe SSDs



- Memcached latency test with Zipf distribution, server with 1 GB memory, 32 KB key-value pair size, total size of data accessed is 1 GB (when data fits in memory) and 1.5 GB (when data does not fit in memory)
- When data fits in memory: RDMA-Mem/Hybrid gives 5x improvement over IPoIB-Mem
- When data does not fit in memory: RDMA-Hybrid gives 2x-2.5x over IPoIB/RDMA-Mem

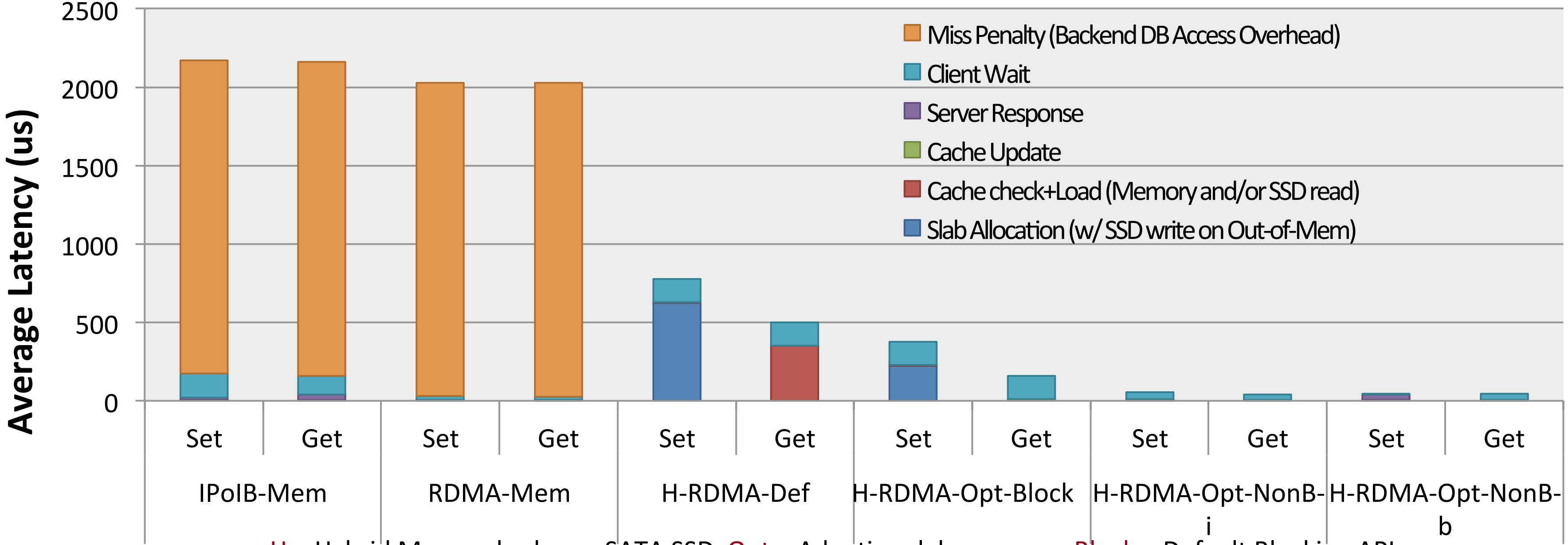
Accelerating Hybrid Memcached with RDMA, Non-blocking Extensions and SSDs



- RDMA-Accelerated Communication for Memcached Get/Set
- Hybrid 'RAM+SSD' slab management for higher data retention
- **Non-blocking API extensions**
  - **memcached_(iset/iget/bset/bget/test/wait)**
  - Achieve near in-memory speeds while hiding bottlenecks of network and SSD I/O
  - Ability to exploit communication/computation overlap
  - Optional buffer re-use guarantees
- Adaptive slab manager with different I/O schemes for higher throughput.

D. Shankar, X. Lu, N. S. Islam, M. W. Rahman, and D. K. Panda, High-Performance Hybrid Key-Value Store on Modern Clusters with RDMA Interconnects and SSDs: Non-blocking Extensions, Designs, and Benefits, IPDPS, May 2016

Performance Evaluation with Non-Blocking Memcached API



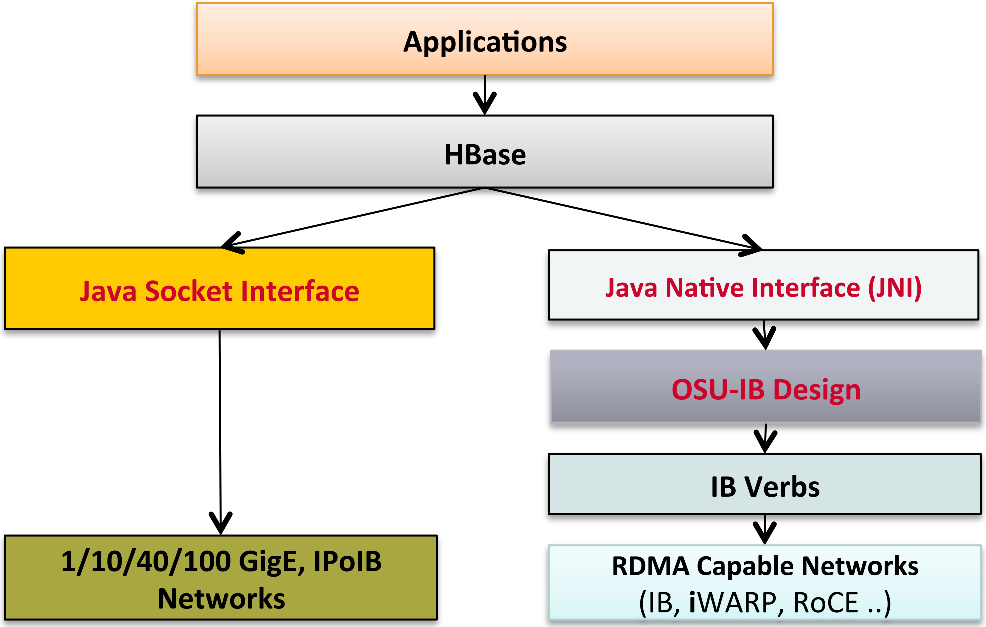
H = Hybrid Memcached over SATA SSD Opt = Adaptive slab manager Block = Default Blocking API
NonB-i = Non-blocking iset/iget API   NonB-b = Non-blocking bset/bget API w/ buffer re-use guarantee

- Data does not fit in memory: Non-blocking Memcached Set/Get API Extensions can achieve
  - >16x latency improvement vs. blocking API over RDMA-Hybrid/RDMA-Mem w/ penalty
  - >2.5x throughput improvement vs. blocking API over default/optimized RDMA-Hybrid
- Data fits in memory: Non-blocking Extensions perform similar to RDMA-Mem/RDMA-Hybrid and >3.6x improvement over IPoIB-Mem

Presentation Outline

- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

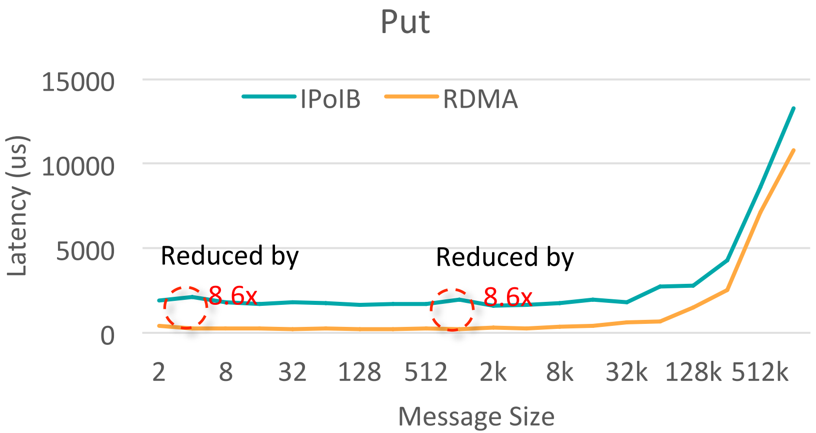
HBase-RDMA Design Overview

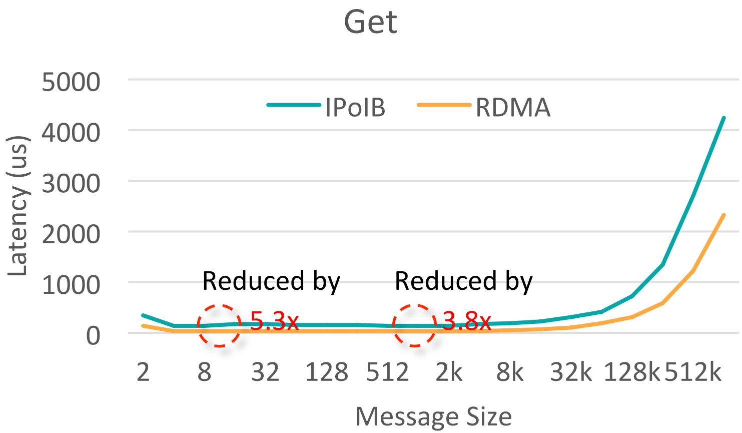

Applications
HBase
Java Socket Interface
Java Native Interface (JNI)
OSU-IB Design
IB Verbs
1/10/40/100 GigE, IPoIB Networks
RDMA Capable Networks (IB, iWARP, RoCE ..)

J. Huang, X. Ouyang, J. Jose, M. W. Rahman, H. Wang, M. Luo, H. Subramoni, Chet Murthy, and D. K. Panda, High-Performance Design of HBase with RDMA over InfiniBand, IPDPS'12

- JNI Layer bridges Java based HBase with communication library written in native code
- Enables high performance RDMA communication, while supporting traditional socket interface

Performance Numbers of RDMA for Apache HBase – OHB in SDSC-Comet



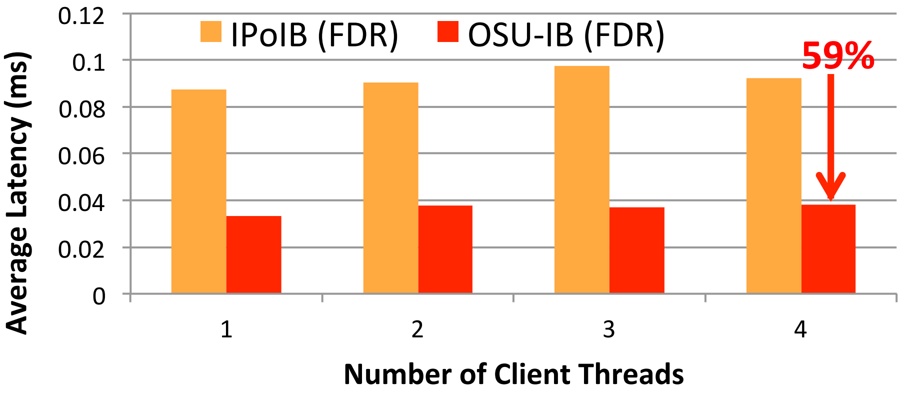


Evaluation with OHB Put and Get Micro-Benchmarks (1 Server, 1 Client)

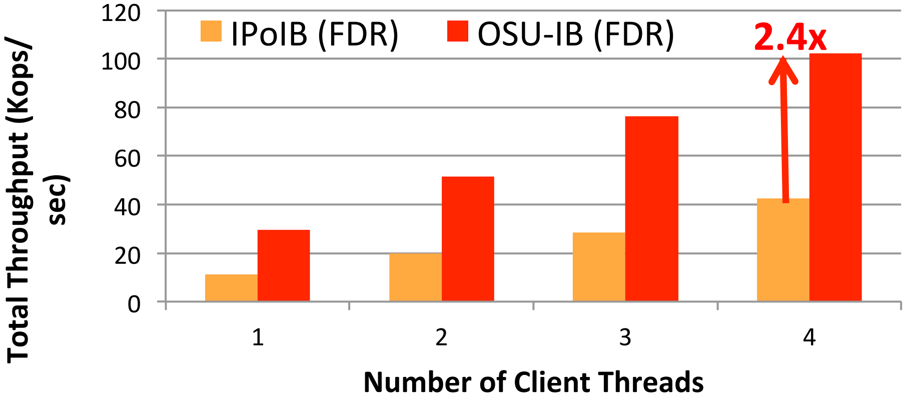
- Up to **8.6x** improvement over IPoIB

- Up to **5.3x** improvement over IPoIB

HBase – YCSB Get Latency and Throughput on SDSC-Comet



Get Latency



Get Throughput

- HBase Get average latency (FDR)
  - 4 client threads: 38 us
  - 59% improvement over IPoIB for 4 client threads
- HBase Get total throughput
  - 4 client threads: 102 Kops/sec
  - 2.4x improvement over IPoIB for 4 client threads

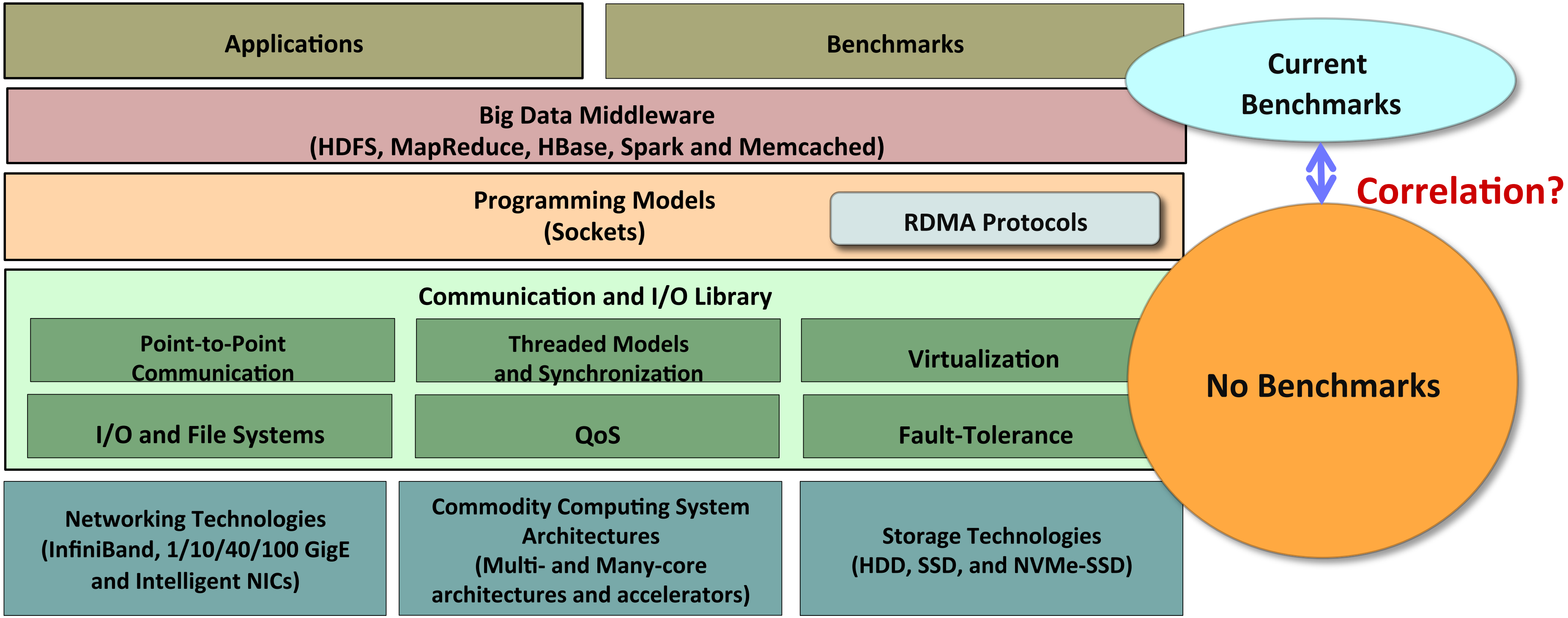
Presentation Outline

- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

Are the Current Benchmarks Sufficient for Big Data?

- The current benchmarks provide some performance behavior
- However, do not provide any information to the designer/developer on:
  - What is happening at the lower-layer?
  - Where the benefits are coming from?
  - Which design is leading to benefits or bottlenecks?
  - Which component in the design needs to be changed and what will be its impact?
  - Can performance gain/loss at the lower-layer be correlated to the performance gain/loss observed at the upper layer?

Challenges in Benchmarking of RDMA-based Designs



Applications

Benchmarks

Current Benchmarks

Correlation?

No Benchmarks

Big Data Middleware
(HDFS, MapReduce, HBase, Spark and Memcached)

Programming Models
(Sockets)

RDMA Protocols

Communication and I/O Library

Point-to-Point Communication

Threaded Models and Synchronization

Virtualization

I/O and File Systems

QoS

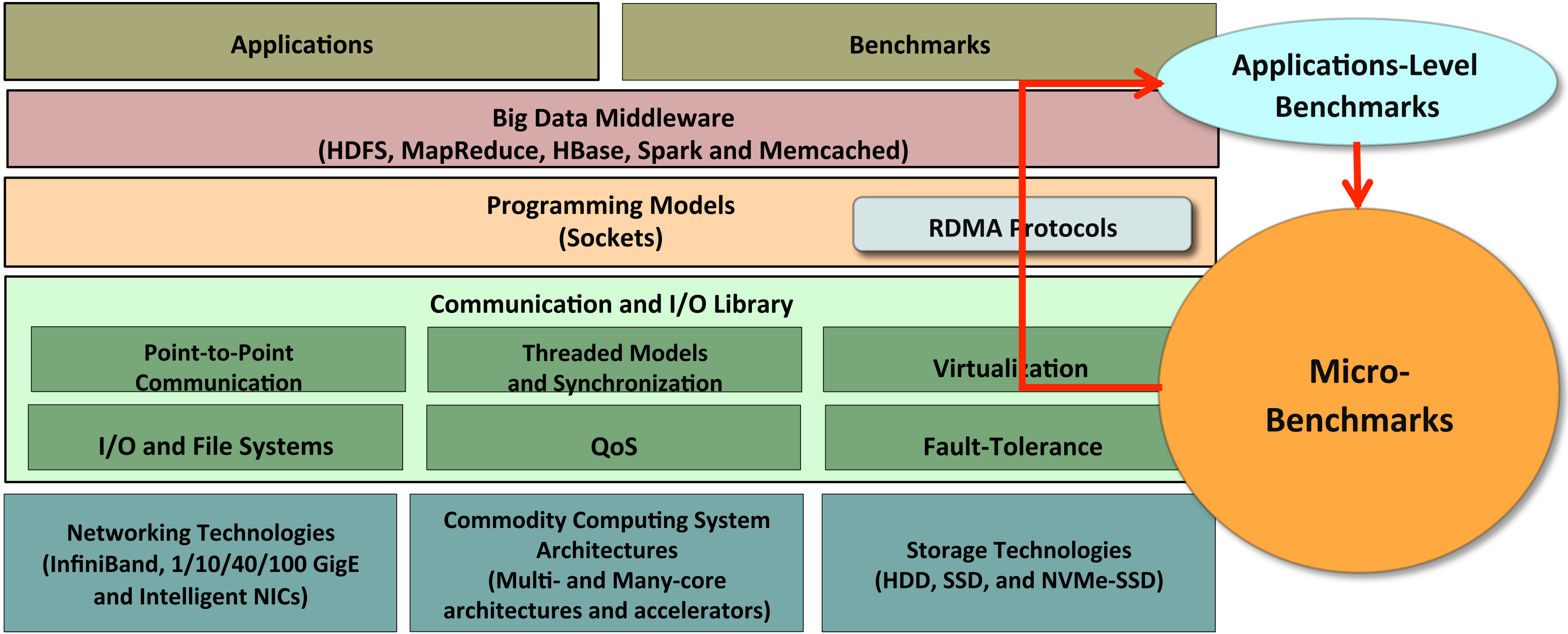
Fault-Tolerance

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

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

Storage Technologies
(HDD, SSD, and NVMe-SSD)

Iterative Process – Requires Deeper Investigation and Design for Benchmarking Next Generation Big Data Systems and Applications


Applications
Benchmarks
Applications-Level Benchmarks
Big Data Middleware (HDFS, MapReduce, HBase, Spark and Memcached)
Programming Models (Sockets)
RDMA Protocols
Communication and I/O Library
Point-to-Point Communication
Threaded Models and Synchronization
Virtualization
I/O and File Systems
QoS
Fault-Tolerance
Micro-Benchmarks
Networking Technologies (InfiniBand, 1/10/40/100 GigE and Intelligent NICs)
Commodity Computing System Architectures (Multi- and Many-core architectures and accelerators)
Storage Technologies (HDD, SSD, and NVMe-SSD)

OSU HiBD Micro-Benchmark (OHB) Suite - HDFS

- Evaluate the performance of standalone HDFS
- Five different benchmarks
  - Sequential Write Latency (**SWL**)
  - Sequential or Random Read Latency (**SRL** or **RRL**)
  - Sequential Write Throughput (**SWT**)
  - Sequential Read Throughput (**SRT**)
  - Sequential Read-Write Throughput (**SRWT**)

N. S. Islam, X. Lu, M. W. Rahman, J. Jose, and D. K. Panda, A Micro-benchmark Suite for Evaluating HDFS Operations on Modern Clusters, Int'l Workshop on Big Data Benchmarking (WBDB '12), December 2012

Benchmark	File Name	File Size	HDFS Parameter	Readers	Writers	Random/ Sequential Read	Seek Interval
SWL	√	√	√				
SRL/RRL	√	√	√			√	√ (RRL)
SWT		√	√		√		
SRT		√	√	√			
SRWT		√	√	√	√		

OSU HiBD Micro-Benchmark (OHB) Suite - MapReduce

- Evaluate the performance of stand-alone MapReduce
- Does not require or involve HDFS or any other distributed file system
- Models shuffle data patterns in real-workload Hadoop application workloads
- Considers various factors that influence the data shuffling phase
  - underlying network configuration, number of map and reduce tasks, intermediate shuffle data pattern, shuffle data size etc.
- Two different micro-benchmarks based on generic intermediate shuffle patterns
  - **MR-AVG:** intermediate data is evenly distributed (or approx. equal) among reduce tasks
    - **MR-RR** i.e., round-robin distribution and **MR-RAND** i.e., pseudo-random distribution
  - **MR-SKEW:** intermediate data is unevenly distributed among reduce tasks
    - Total number of shuffle key/value pairs, max% per reducer, min% per reducer to configure skew

D. Shankar, X. Lu, M. W. Rahman, N. Islam, and D. K. Panda, Characterizing and benchmarking stand-alone Hadoop MapReduce on modern HPC clusters, The Journal of Supercomputing (2016)

D. Shankar, X. Lu, M. W. Rahman, N. Islam, and D. K. Panda, A Micro-Benchmark Suite for Evaluating Hadoop MapReduce on High-Performance Networks, BPOE-5 (2014)

OSU HiBD Micro-Benchmark (OHB) Suite - RPC

- Two different micro-benchmarks to evaluate the performance of standalone Hadoop RPC
  - Latency: Single Server, Single Client
  - Throughput: Single Server, Multiple Clients
- A simple script framework for job launching and resource monitoring
- Calculates statistics like Min, Max, Average
- Network configuration, Tunable parameters, DataType, CPU Utilization

Component	Network Address	Port	Data Type	Min Msg Size	Max Msg Size	No. of Iterations	Handlers	Verbose
lat_client	√	√	√	√	√	√		√
lat_server	√	√					√	√

Component	Network Address	Port	Data Type	Min Msg Size	Max Msg Size	No. of Iterations	No. of Clients	Handlers	Verbose
thr_client	√	√	√	√	√	√			√
thr_server	√	√			√		√	√	√

X. Lu, M. W. Rahman, N. Islam, and D. K. Panda, A Micro-Benchmark Suite for Evaluating Hadoop RPC on High-Performance Networks, Int'l Workshop on Big Data Benchmarking (WBDB '13), July 2013

OSU HiBD Micro-Benchmark (OHB) Suite - Memcached

- Evaluates the performance of stand-alone Memcached in different modes
- Default API Latency benchmarks for Memcached in-memory mode
  - **SET Micro-benchmark**: Micro-benchmark for memcached set operations
  - **GET Micro-benchmark**: Micro-benchmark for memcached get operations
  - **MIX Micro-benchmark**: Micro-benchmark for a mix of memcached set/get operations (Read:Write ratio is 90:10)
- Latency benchmarks for Memcached hybrid-memory mode
- Non-Blocking API Latency Benchmark for Memcached (both in-memory and hybrid-memory mode)
- Calculates average latency of Memcached operations in different modes

D. Shankar, X. Lu, M. W. Rahman, N. Islam, and D. K. Panda, Benchmarking Key-Value Stores on High-Performance Storage and Interconnects for Web-Scale Workloads, IEEE International Conference on Big Data (IEEE BigData '15), Oct 2015

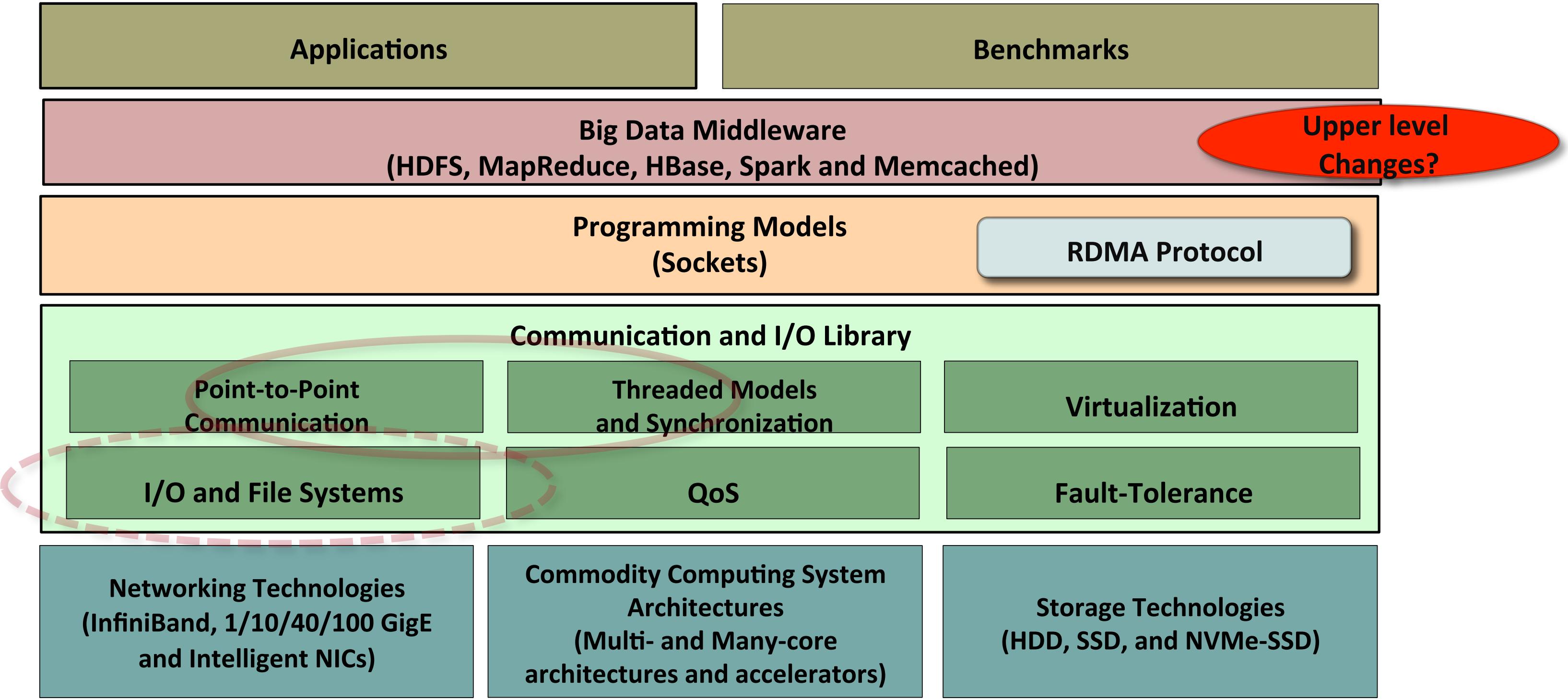
Presentation Outline

- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

Other Existing Hadoop Acceleration Solutions

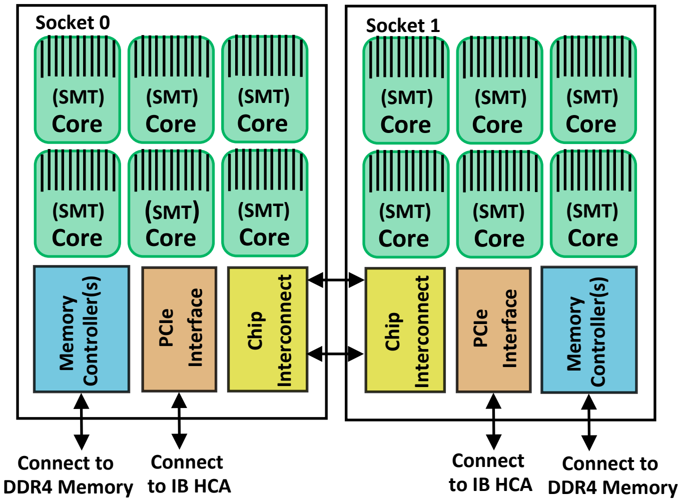
- Mellanox UDA and R4H
  - UDA (Unstructured Data Accelerator): RDMA-based implementation of Hadoop MapReduce shuffle engine
  - R4H (RDMA for HDFS): A plugin for Hadoop Distributed File System (HDFS) which accelerates HDFS by using RDMA (Remote Direct Memory Access) technology.
  - http://www.mellanox.com/page/products_dyn?product_family=144
  - https://github.com/Mellanox/R4H
- Cloudera Distributions of Hadoop (CDH)
  - Cloudera Standard: CDH with automated cluster management
  - Cloudera Enterprise: Cloudera Standard + enhanced management capabilities and support
  - Ethernet/IPoIB
  - http://www.cloudera.com/content/cloudera/en/products.html
- Hortonworks Data Platform (HDP)
  - Developed as projects through the Apache Software Foundation (ASF), NO proprietary extensions or add-ons
  - Functional areas: Data Management, Data Access, Data Governance and Integration, Security, and Operations.
  - Ethernet/IPoIB
  - http://hortonworks.com/hdp
- Hadoop over other file systems
  - OrangeFS: http://www.orangefs.org/
  - Ceph: http://ceph.com/

Designing Communication and I/O Libraries for Big Data Systems: Solved a Few Initial Challenges

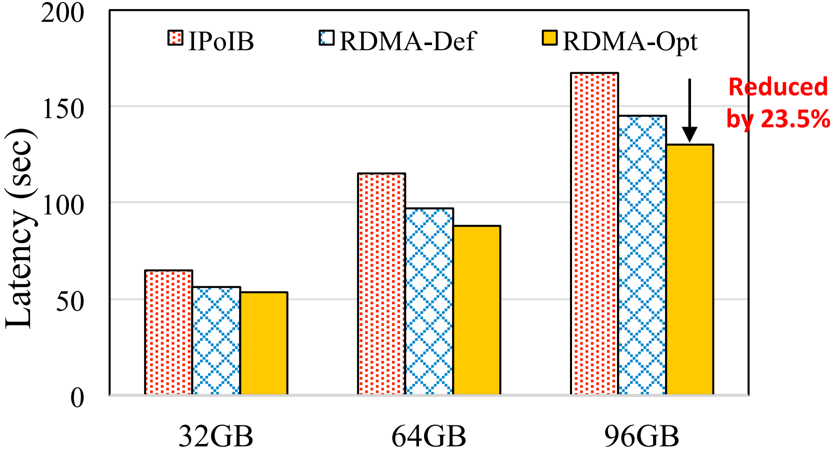

Applications
Benchmarks
Big Data Middleware
(HDFS, MapReduce, HBase, Spark and Memcached)
Upper level Changes?
Programming Models
(Sockets)
RDMA Protocol
Communication and I/O Library
Point-to-Point Communication
Threaded Models and Synchronization
Virtualization
I/O and File Systems
QoS
Fault-Tolerance
Networking Technologies
(InfiniBand, 1/10/40/100 GigE and Intelligent NICs)
Commodity Computing System Architectures
(Multi- and Many-core architectures and accelerators)
Storage Technologies
(HDD, SSD, and NVMe-SSD)

Accelerating Hadoop and Spark with RDMA over OpenPOWER

IBM POWER8 Architecture



Spark SortBy



Reduced by 23.5%

- Challenges
  - Performance characteristics of RDMA-based Hadoop and Spark over OpenPOWER systems with IB networks
  - Can RDMA-based Big Data middleware demonstrate good performance benefits on OpenPOWER systems?
  - Any new accelerations for Big Data middleware on OpenPOWER systems to attain better benefits?
- Results
  - For the Spark SortBy benchmark, RDMA-Opt outperforms IPoIB and RDMA-Def by 23.5% and 10.5%

X. Lu, H. Shi, D. Shankar and D. K. Panda, Performance Characterization and Acceleration of Big Data Workloads on OpenPOWER System, IEEE BigData 2017.

More Challenges

- Multi-threading and Synchronization
  - Multi-threaded model exploration
  - Fine-grained synchronization and lock-free design
  - Unified helper threads for different components
  - Multi-endpoint design to support multi-threading communications
- QoS and Virtualization
  - Network virtualization and locality-aware communication for more Big Data middleware
  - Hardware-level virtualization support for End-to-End QoS
  - I/O scheduling and storage virtualization
  - Live migration
- Support of Accelerators
  - Efficient designs for Big Data middleware to take advantage of NVIDA GPGPUs and Intel MICs
  - Offload computation-intensive workload to accelerators
  - Explore maximum overlapping between communication and offloaded computation

On-going and Future Plans of OSU High Performance Big Data (HiBD) Project

- Upcoming Releases of RDMA-enhanced Packages will support
  - Upgrades to the latest versions of Hadoop and Spark
  - OpenPOWER Support
  - Streaming (RDMA-Kafka)
  - MR-Advisor
  - Deep Learning (gRPC and TensorFlow)
- Upcoming Releases of OSU HiBD Micro-Benchmarks (OHB) will support
  - MapReduce, Hadoop RPC, and gRPC
- Advanced designs with upper-level changes and optimizations
  - Boldio (Burst Buffer over Lustre for Big Data I/O Acceleration)
  - Efficient Indexing

Presentation Outline

- Overview of Modern Clusters, Interconnects and Protocols
- Challenges for Accelerating Big Data Processing
- The High-Performance Big Data (HiBD) Project
- RDMA-based designs for Apache Hadoop and Spark
  - Case studies with HDFS, MapReduce, and Spark
  - RDMA-based MapReduce on HPC Clusters with Lustre
- RDMA-based designs for Memcached and HBase
  - RDMA-based Memcached
  - RDMA-based HBase
- Challenges in Designing Benchmarks for Big Data Processing
  - OSU HiBD Benchmarks
- On-going and Future Activities
- Conclusion and Q&A

Concluding Remarks

- Presented an overview of Big Data, Hadoop (MapReduce, HDFS, HBase, Spark, RPC) and Memcached
- Provided an overview of Networking Technologies
- Discussed challenges in accelerating Hadoop and Memcached
- Presented initial designs to take advantage of InfiniBand/RDMA for HDFS, MapReduce, HBase, Spark, RPC and Memcached
- Results are promising
- Many other open issues need to be solved
- Will enable Big Data community to take advantage of modern HPC technologies to carry out their analytics in a fast and scalable manner

Thank You!

luxi@cse.ohio-state.edu

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/