Can Parallel Replication Benefit Hadoop Distributed File System for High Performance Interconnects?

N. S. Islam, X. Lu, M. W. Rahman, and D. K. Panda

Network-Based Computing Laboratory
Department of Computer Science and Engineering
The Ohio State University, Columbus, OH, USA

Outline

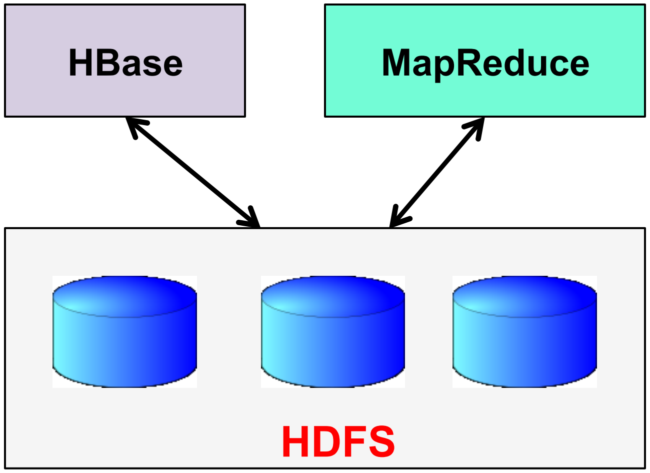
- Introduction and Motivation
- Problem Statement
- Design
- Performance Evaluation
- Conclusion & Future work

Introduction

- Big Data: provides groundbreaking opportunities for enterprise information management and decision making
- The rate of information growth appears to be exceeding Moore's Law
- The amount of data is exploding; companies are capturing and digitizing more information that ever
- 35 zettabytes of data will be generated and consumed by the end of this decade

Big Data Technology

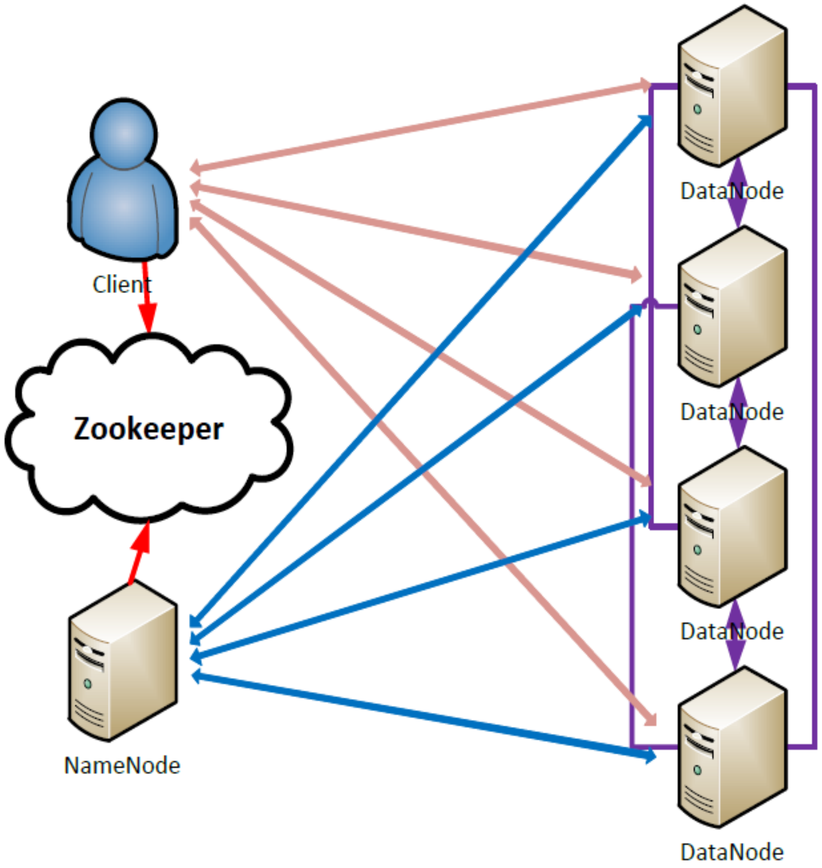
- Apache Hadoop is a popular Big Data technology
  - Provides framework for large-scale, distributed data storage and processing
- Hadoop is an open-source implementation of MapReduce programming model
- Hadoop Distributed File System (HDFS) (http://hadoop.apache.org/) is the underlying file system of Hadoop and Hadoop DataBase, HBase



Hadoop Framework

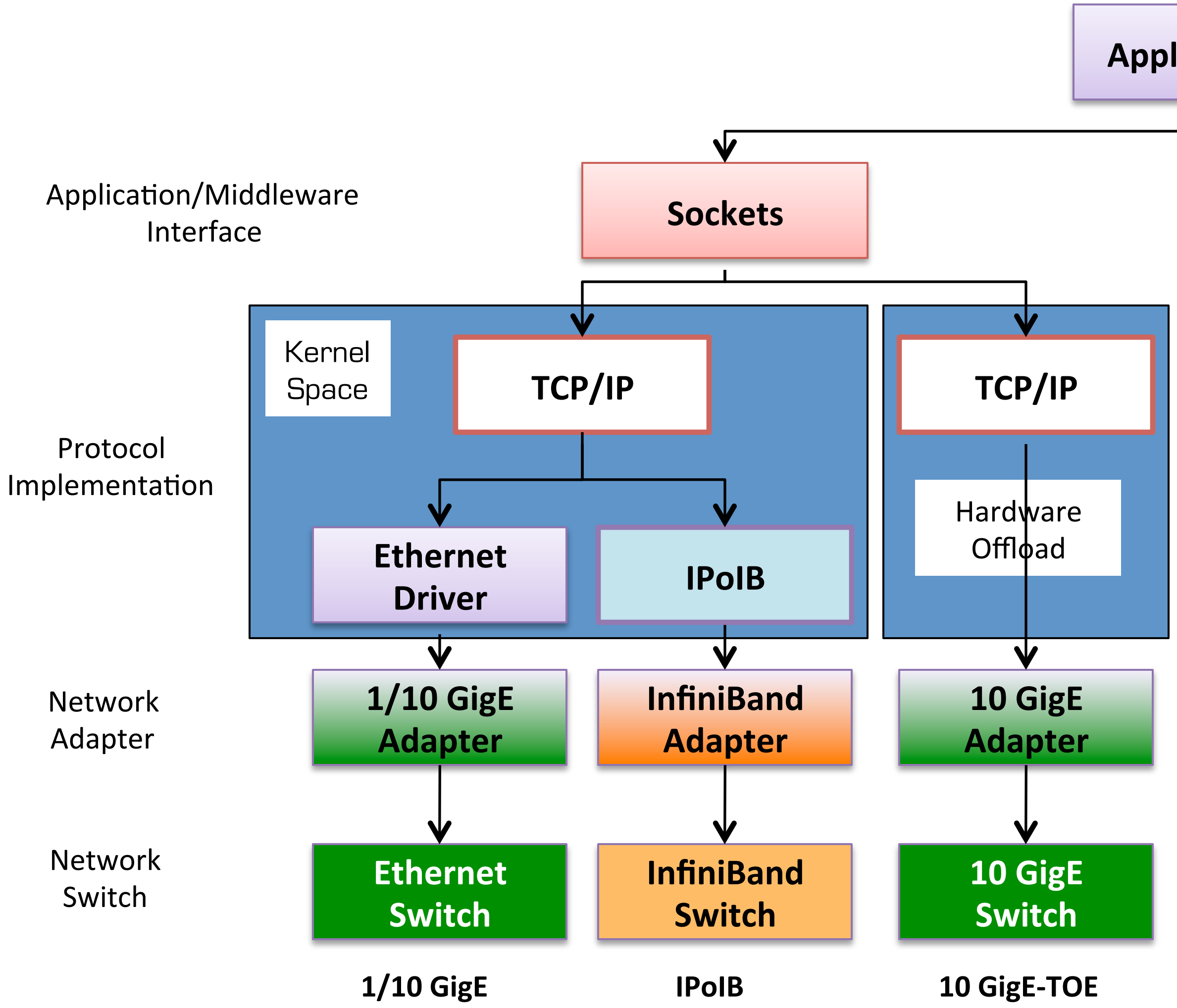
Hadoop Distributed File System (HDFS)

- Adopted by many reputed organizations
  - eg:- Facebook, Yahoo!
- Highly reliable and fault-tolerant - replication
- NameNode: stores the file system namespace
- DataNode: stores data blocks
- Developed in Java for platform-independence and portability
- Uses Java sockets for communication



(HDFS Architecture)

Modern High Performance Interconnects
(Socket Interface)

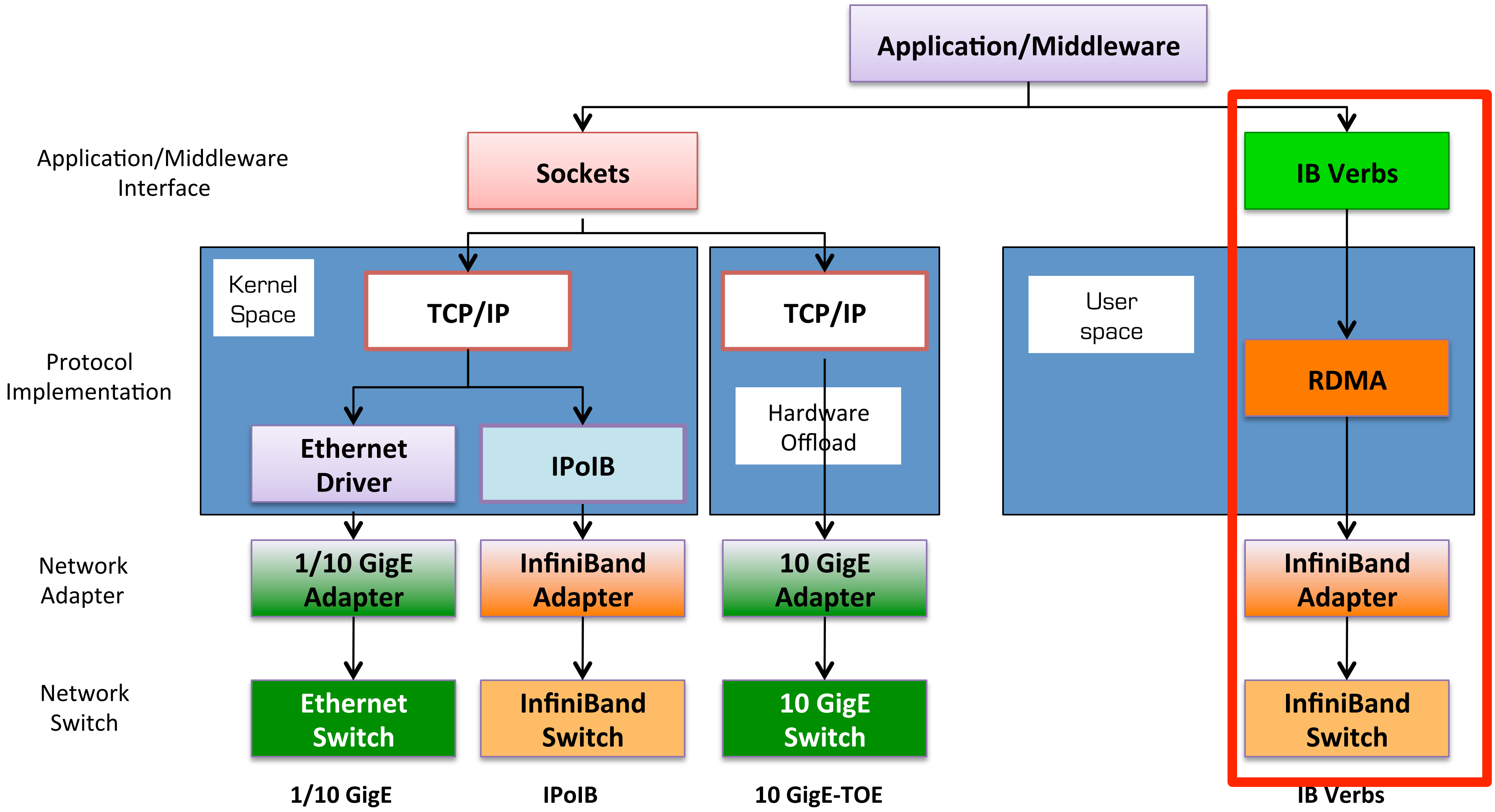


Cloud Computing systems are being widely used on High Performance Computing (HPC) Clusters

Commodity high performance networks like InfiniBand can provide low latency and high throughput data transmission

For data-intensive applications network performance becomes key component for HDFS

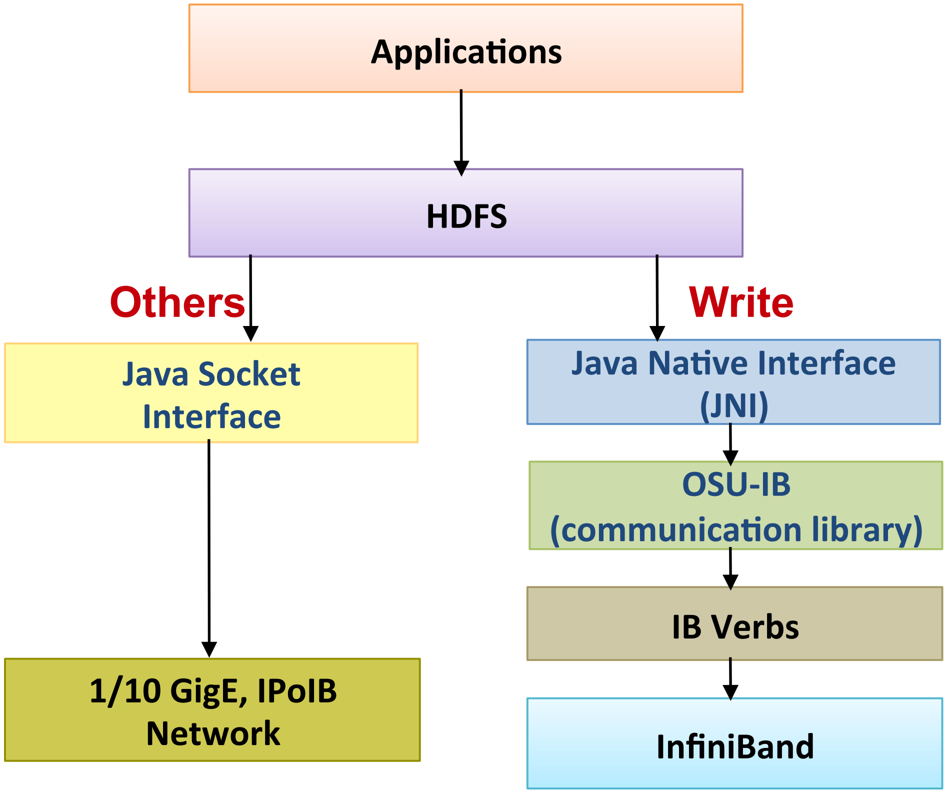
Modern High Performance Interconnects



RDMA-based Design of HDFS

Enables high performance RDMA communication, while supporting traditional socket interface

- JNI Layer bridges Java based HDFS with communication library written in native code
- **Only the communication part of HDFS Write is modified; No change in HDFS architecture**
- Available in Hadoop RDMA 0.9.1 release
  http://hadoop-rdma.cse.ohio-state.edu/



Applications

HDFS

Others

Write

Java Socket Interface

1/10 GigE, IPoIB Network

Java Native Interface (JNI)

OSU-IB (communication library)

IB Verbs

InfiniBand

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

Hadoop-RDMA Release

- High-Performance Design of Hadoop over RDMA-enabled Interconnects
  - High performance design with native InfiniBand support at the verbs-level for HDFS, MapReduce, and RPC components
  - Easily configurable for both native InfiniBand and the traditional sockets-based support (Ethernet and InfiniBand with IPoIB)
  - Current release: 0.9.1
    - Based on Apache Hadoop 0.20.2
    - Compliant with Apache Hadoop 0.20.2 APIs and applications
    - Tested with
      - Mellanox InfiniBand adapters (DDR, QDR and FDR)
      - Various multi-core platforms
      - Different file systems with disks and SSDs
  - http://hadoop-rdma.cse.ohio-state.edu

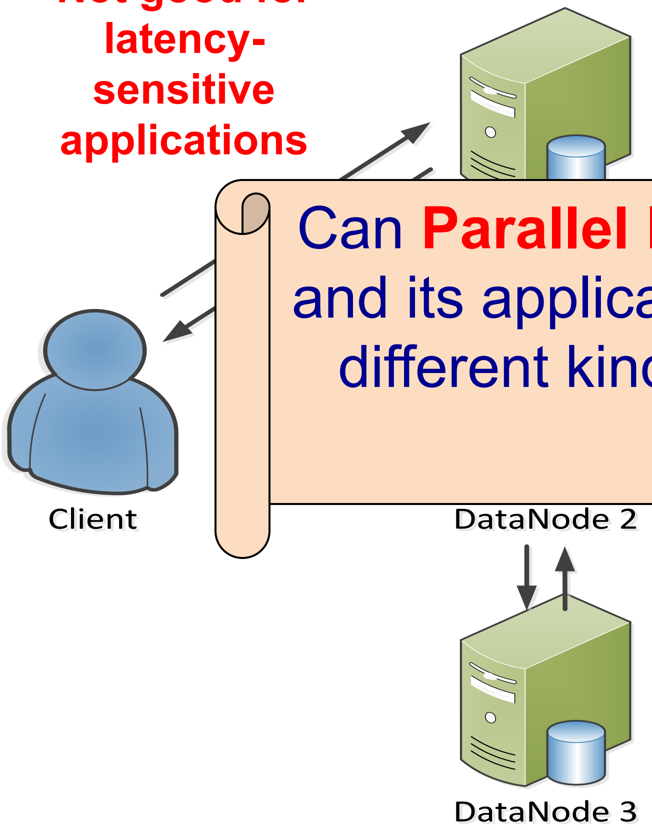
hadoop-RDMA

- Updated release with Hadoop stable version (1.2.1) coming soon

HDFS Replication: Pipelined vs Parallel

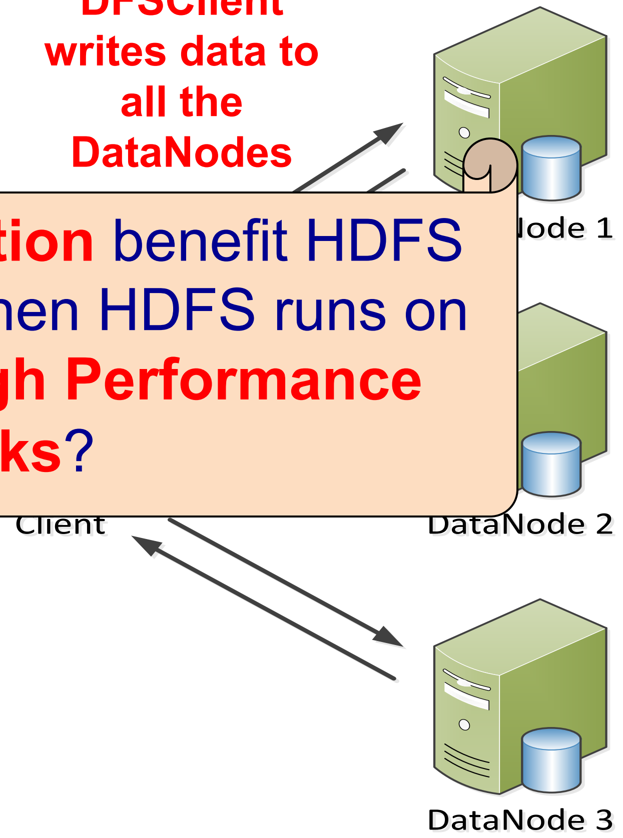
- Basic mechanism of HDFS fault tolerance is Data Replication
  - Replicates each block to multiple DataNodes

Can **Parallel Replication** benefit HDFS and its applications when HDFS runs on different kinds of **High Performance Networks**?



Not good for latency-sensitive applications

Pipelined replication



DFSClient writes data to all the DataNodes

Parallel replication

Outline

- Introduction and Motivation
- Problem Statement
- Design
- Performance Evaluation
- Conclusion & Future work

Problem Statement

- What are the challenges to introduce the parallel replication scheme in both the socket-based and RDMA-based design of HDFS?

- Can we re-design HDFS to take advantage of the parallel replication scheme over high performance networks and protocols?

- What will be the impact of parallel replication on Hadoop benchmarks over different interconnects and protocols?

- Can we observe performance improvement for other cloud computing middleware such as HBase with this replication technique?

Outline

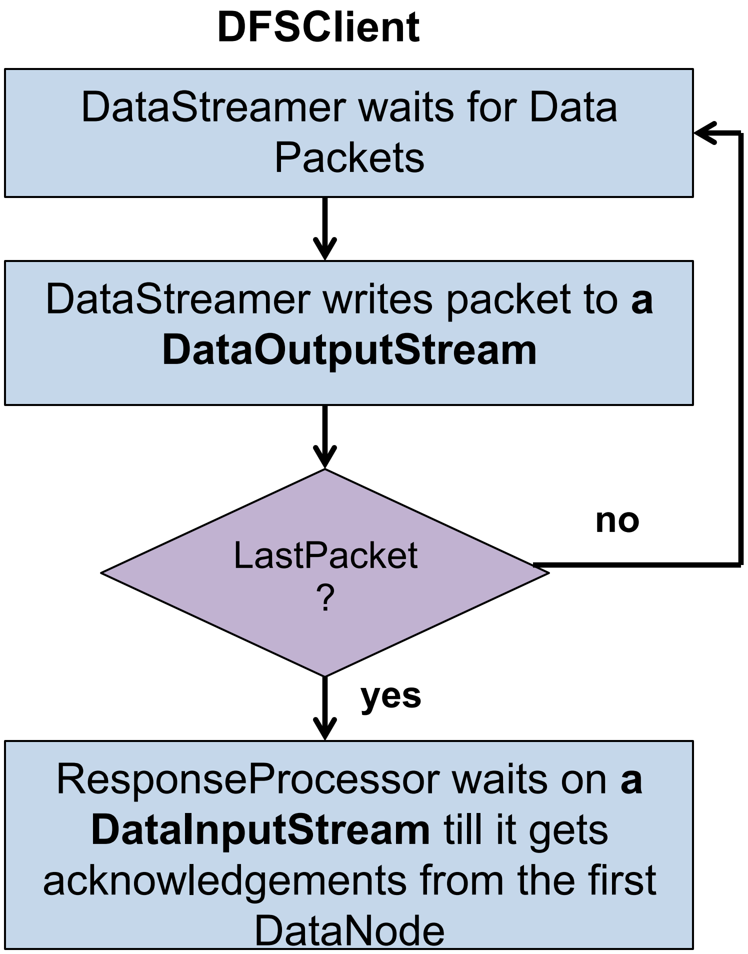
- Introduction and Motivation
- Problem Statement
- Design
  - Parallel Replication in Socket-based Design of HDFS
  - Parallel Replication in RDMA-based Design of HDFS
- Performance Evaluation
- Conclusion & Future work

Parallel Replication in Socket-based Design of HDFS (Issues and Challenges)

- In Pipelined Replication
  - DFSClient writes to the first DataNode in the pipeline
  - DFSClient receives acknowledgements from only the first DataNode in the pipeline

- In Parallel Replication
  - DFSClient writes to all (default is three) the DataNodes the block should be replicated to
  - DFSClient receives acknowledgements from all the DataNodes
  - MultiDataOutputStream and MultiDataInputStream

Parallel Replication in Socket-based Design of HDFS (Communication Flow)



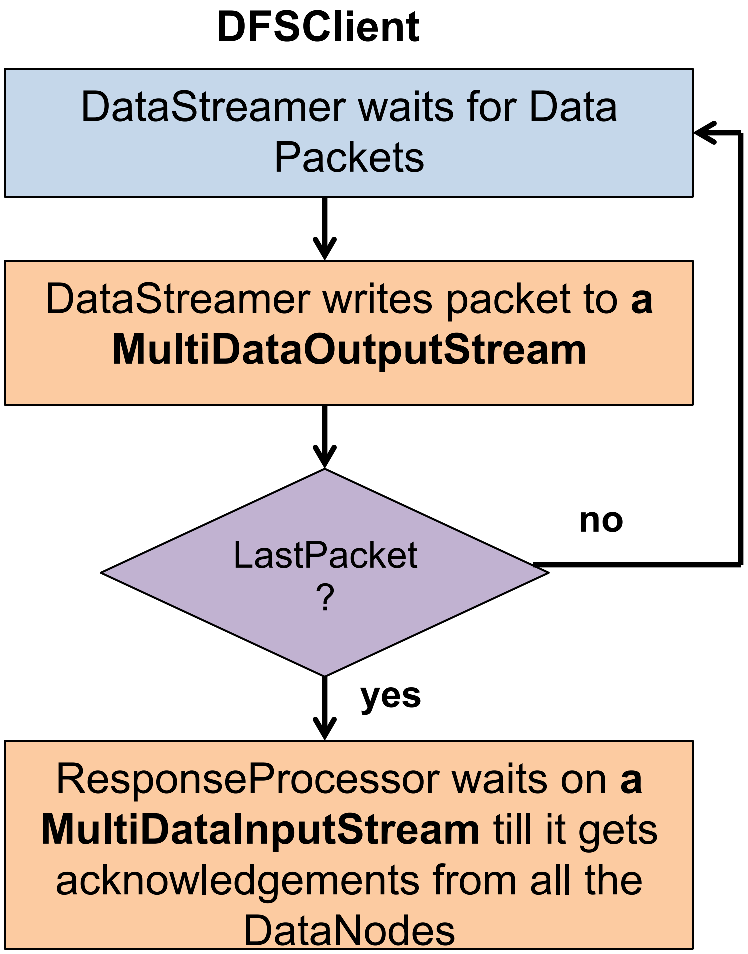
DFSClient

DataStreamer waits for Data Packets

DataStreamer writes packet to **a DataOutputStream**

ResponseProcessor waits on **a DataInputStream** till it gets acknowledgements from the first DataNode

Pipelined Replication



DFSClient

DataStreamer waits for Data Packets

DataStreamer writes packet to **a MultiDataOutputStream**

ResponseProcessor waits on **a MultiDataInputStream** till it gets acknowledgements from all the DataNodes

Parallel Replication

Outline

- Introduction and Motivation
- Problem Statement
- Design
  - Parallel Replication in Socket-based Design of HDFS
  - Parallel Replication in RDMA-based Design of HDFS
- Performance Evaluation
- Conclusion & Future work

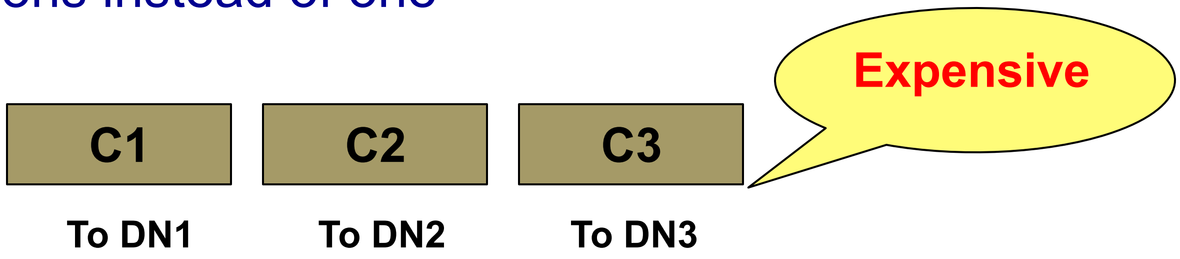
Parallel Replication in RDMA-based Design of HDFS (Issues and Challenges)

- The RDMA-based design of HDFS implements pipelined replication

- Challenges to incorporate parallel replication
  - Reducing connection creation overhead in DFSClient
  - Minimizing the polling overhead
  - Reducing the total wait time for acknowledgements in the DFSClient side

Parallel Replication in RDMA-based Design of HDFS (Connection Management)

- RDMA connection creation is expensive. DFSClient now needs three connections instead of one

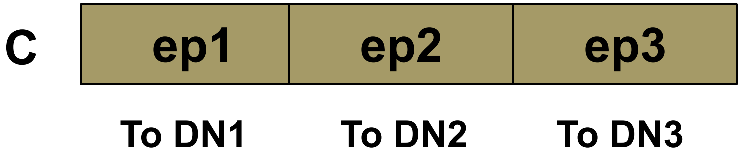


Expensive

C1 C2 C3

To DN1 To DN2 To DN3

- Single Connection object; different end-points to connect to different DataNodes



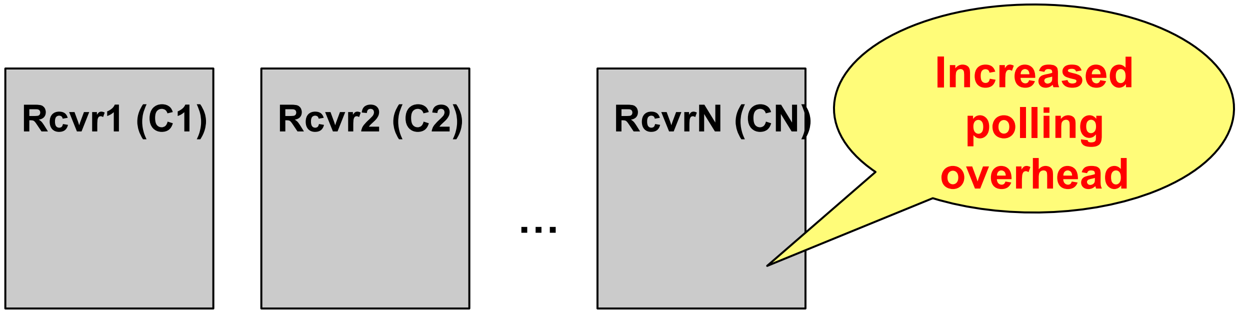
C ep1 ep2 ep3

To DN1 To DN2 To DN3

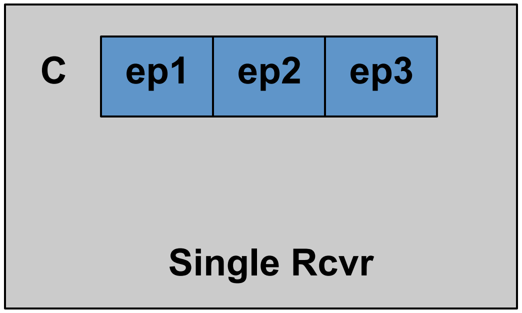
- It also reduces the total wait time for acknowledgements

Parallel Replication in RDMA-based Design of HDFS (Minimizing Polling Overhead)

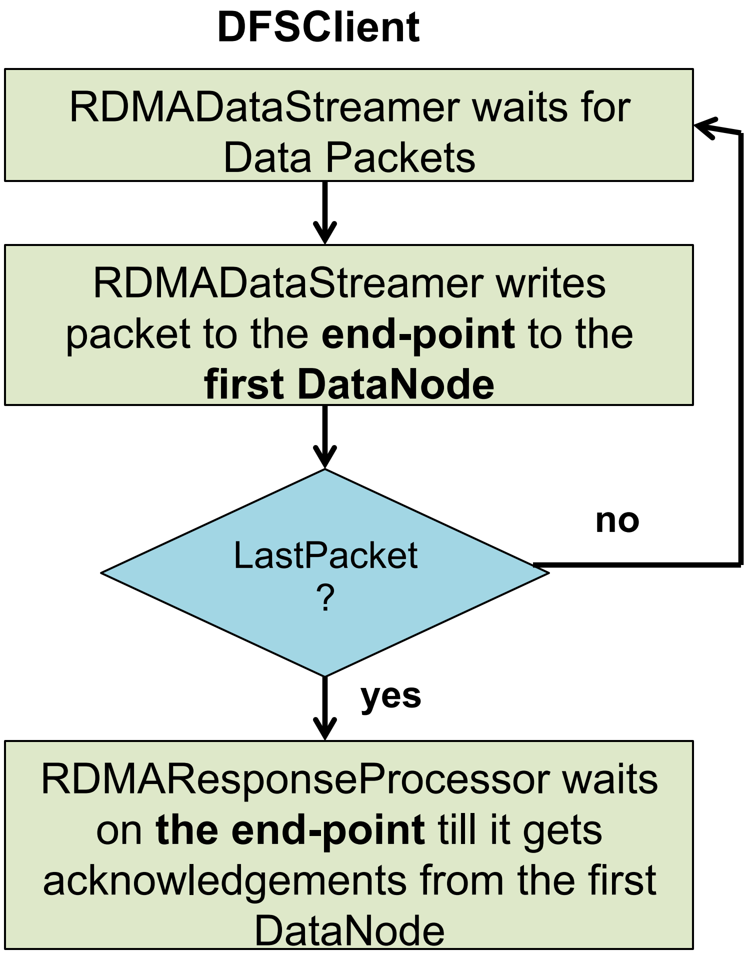
- One Receiver per client (or block) in the DataNode increases polling overhead



- Single Receiver in the DataNode; different clients connect to different end-points of the same connection object



Parallel Replication in RDMA-based Design of HDFS (Communication Flow)



DFSClient

RDMADataStreamer waits for Data Packets

RDMADataStreamer writes packet to the **end-point** to the **first DataNode**

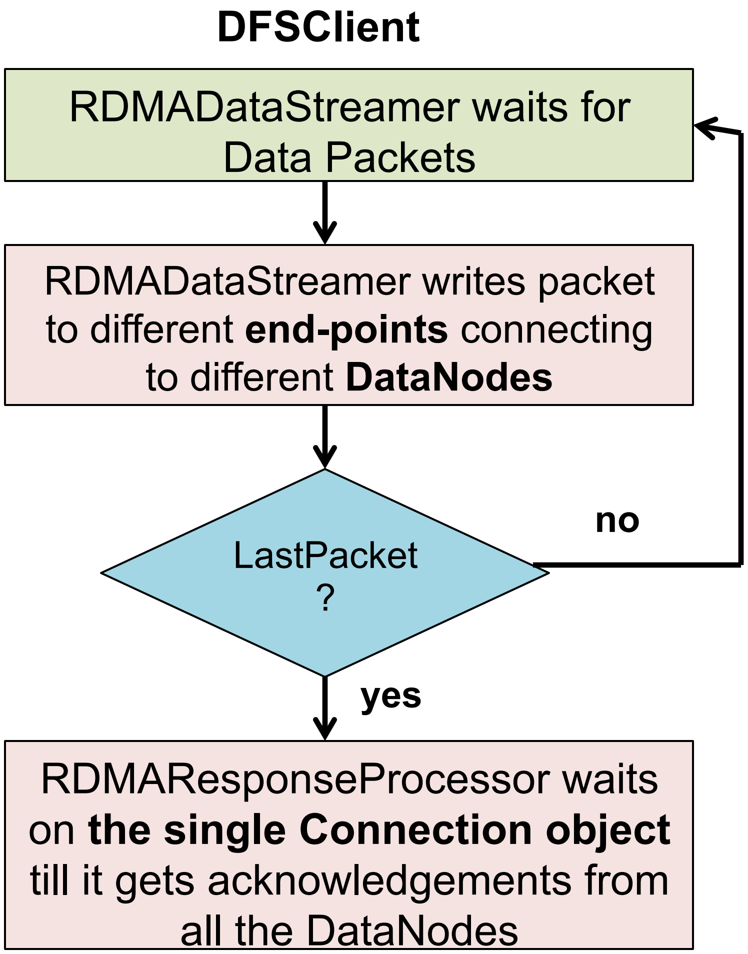
LastPacket ?

no

yes

RDMAResponseProcessor waits on **the end-point** till it gets acknowledgements from the first DataNode

Pipelined Replication



DFSClient

RDMADataStreamer waits for Data Packets

RDMADataStreamer writes packet to different **end-points** connecting to different **DataNodes**

LastPacket ?

no

yes

RDMAResponseProcessor waits on **the single Connection object** till it gets acknowledgements from all the DataNodes

Parallel Replication

Outline

- Introduction and Motivation
- Problem Statement
- Design
- Performance Evaluation
- Conclusion & Future Work

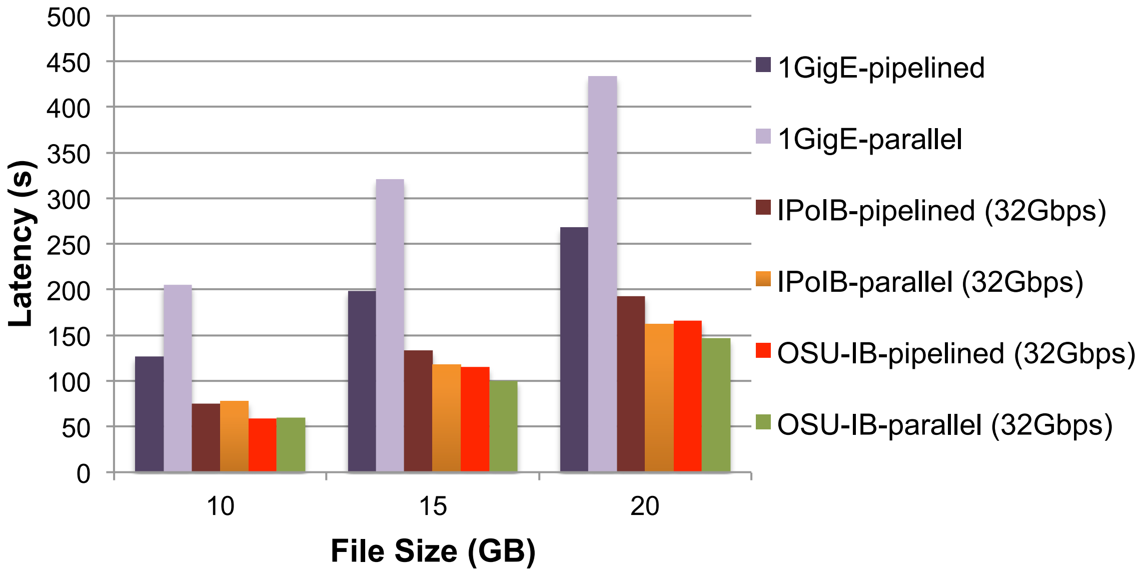
Experimental Setup

- Hardware
  - **Intel Westmere (Cluster A)**
    - Each node has 8 processor cores on 2 Intel Xeon 2.67 GHz Quad-core CPUs, 12 GB main memory, 160 GB hard disk
    - Network: 1GigE, IPoIB, and IB-QDR (32Gbps)
  - **Intel Westmere with larger memory (Cluster B)**
    - Nodes in this cluster has same configuration as Cluster A; 24GB RAM
    - 8 storage nodes with three 1 TB HDD per node
    - Network: 1GigE, 10GigE, IPoIB and IB-QDR (32Gbps)
- Software
  - Hadoop 0.20.2, HBase 0.90.3 and JDK 1.7
  - Yahoo! Cloud Serving Benchmark (YCSB)

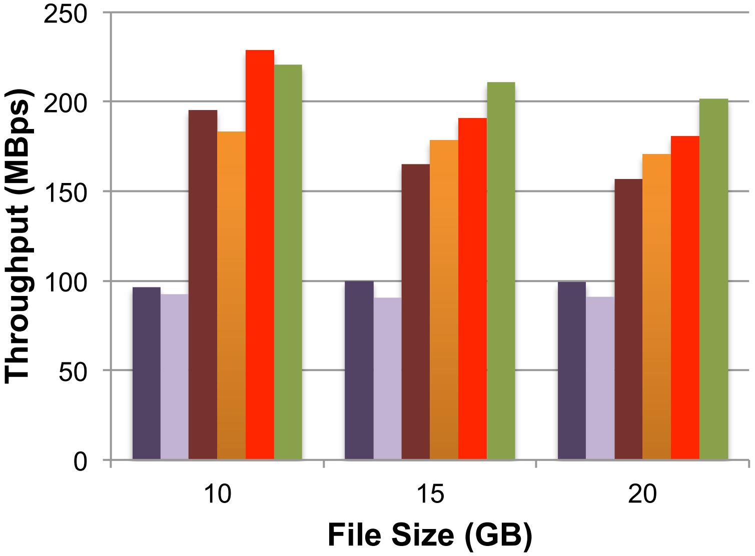
Outline

- Introduction and Motivation
- Problem Statement
- Design
- Performance Evaluation
  - Micro-benchmark level evaluations
  - Evaluations with TestDFSIO
  - Evaluations with TeraGen
  - Integration with HBase (TCP/IP)
- Conclusion & Future Work

Evaluations using Micro-benchmark



Micro-benchmark Latency



Micro-benchmark Throughput

- Cluster A with 8 HDD DataNodes
  - 15% improvement over IPoIB (32Gbps)
  - 12.5% improvement over OSU-IB (32Gbps)

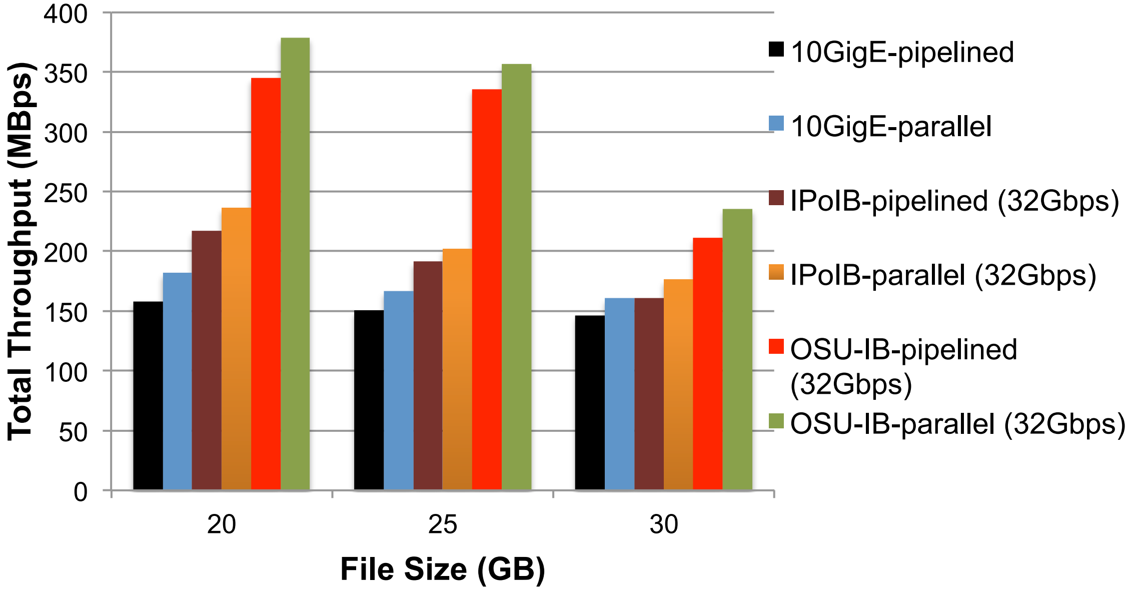
For 1GigE, NIC bandwidth is a bottleneck

Improvement for larger data size

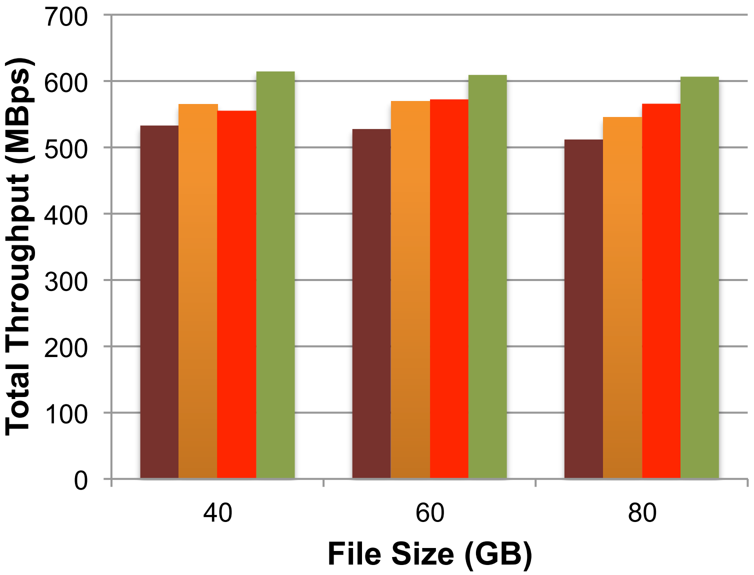
Outline

- Introduction and Motivation
- Problem Statement
- Design
- Performance Evaluation
  - Micro-benchmark level evaluations
  - Evaluations with TestDFSIO
  - Evaluations with TeraGen
  - Integration with HBase (TCP/IP)
- Conclusion & Future Work

Evaluations using TestDFSIO



Cluster B with 8 Nodes



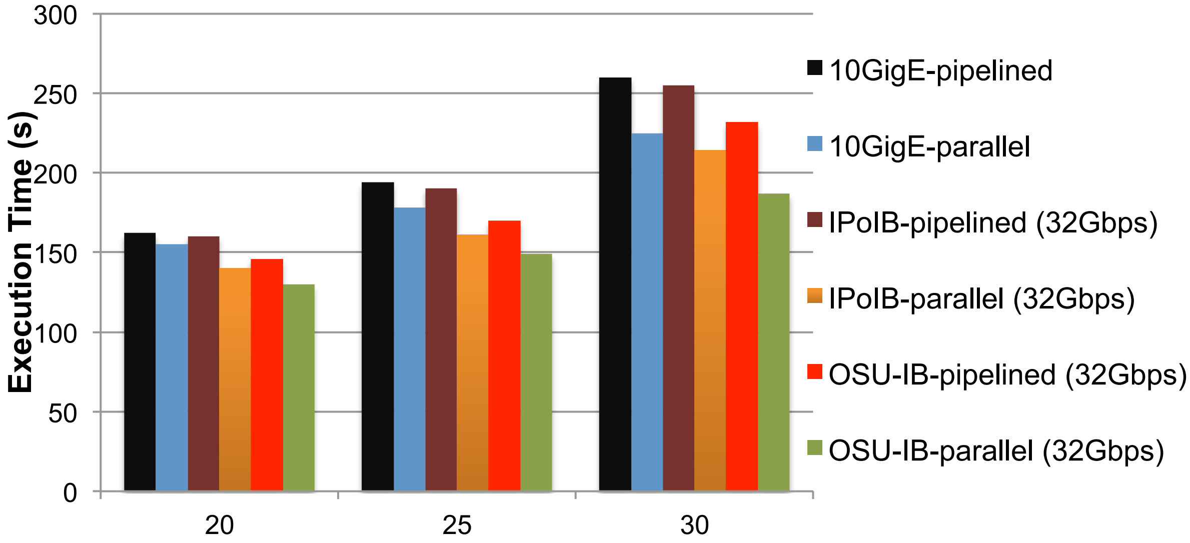
Cluster A with 32 Nodes

- Cluster B with 8 HDD DataNodes
  - **11%** improvement over 10GigE
  - **10%** improvement over IPoIB (32Gbps)
  - **12%** improvement over OSU-IB (32Gbps)
- Cluster A with 32 HDD DataNodes
  - **8%** improvement over IPoIB (32Gbps)
  - **9%** improvement over OSU-IB (32Gbps)

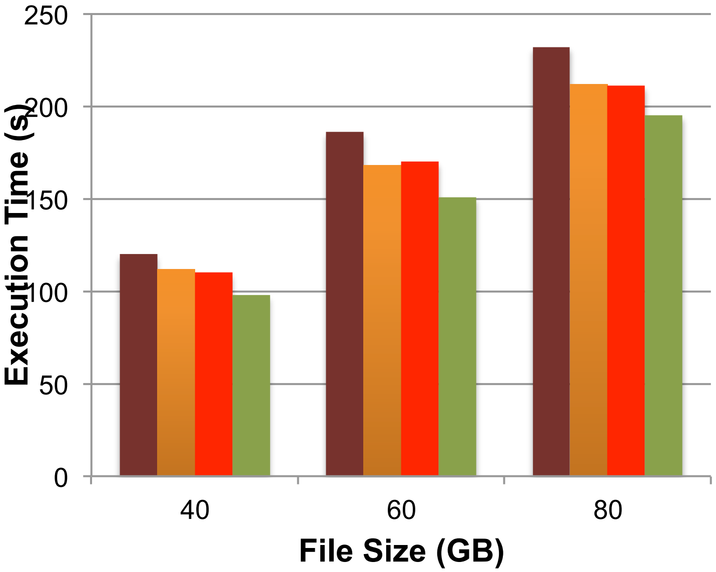
Outline

- Introduction and Motivation
- Problem Statement
- Design
- Performance Evaluation
  - Micro-benchmark level evaluations
  - Evaluations with TestDFSIO
  - Evaluations with TeraGen
  - Integration with HBase (TCP/IP)
- Conclusion & Future Work

Evaluations using TeraGen



Cluster B with 8 Nodes



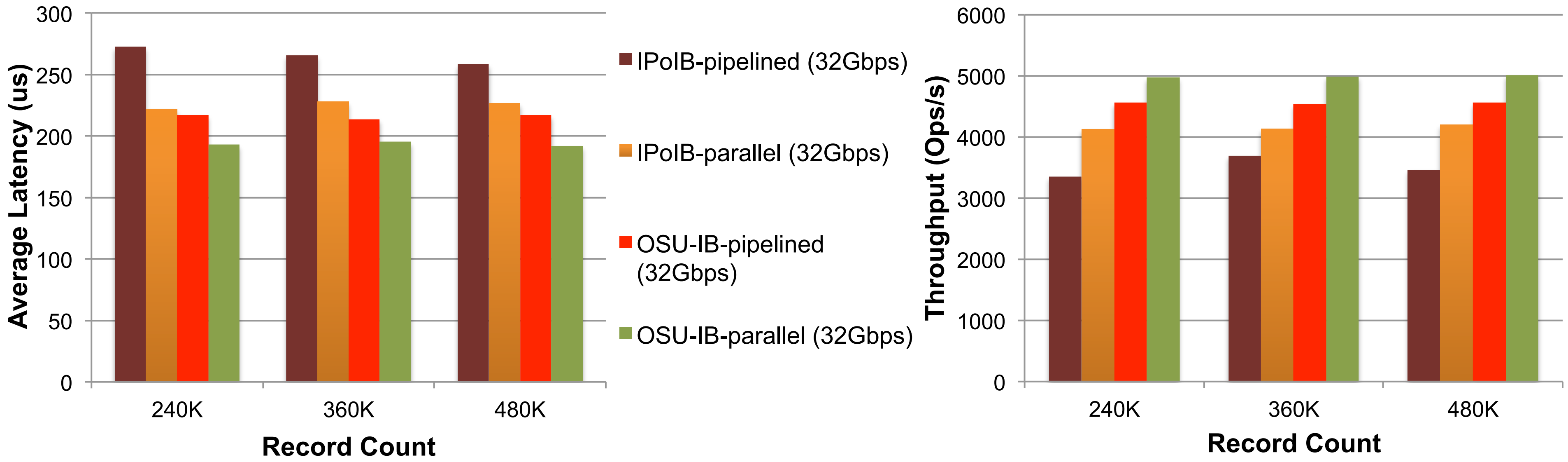
Cluster A with 32 Nodes

- Cluster B with 8 HDD DataNodes
  - **16%** improvement over IPoIB (32Gbps), 10GigE and OSU-IB (32Gbps)
- Cluster A with 32 HDD DataNodes
  - **11%** improvement over IPoIB (32Gbps) and OSU-IB (32Gbps)

Outline

- Introduction and Motivation
- Problem Statement
- Design
- Performance Evaluation
  - Micro-benchmark level evaluations
  - Evaluations with TestDFSIO
  - Evaluations with TeraGen
  - Integration with HBase (TCP/IP)
- Conclusion & Future Work

Evaluations using YCSB



Put Latency

Put Throughput

- HBase *Put* Operation with 4 Region Servers in Cluster A
  - **17%** improvement over IPoIB (32Gbps)
  - **10%** improvement over OSU-IB (32Gbps)

Outline

- Introduction and Motivation
- Problem Statement
- Design using Hybrid Transports
- Performance Evaluation
- Conclusion & Future Work

Conclusion and Future Works

- Introduced Parallel Replication in both Socket-based and RDMA-based design of HDFS over InfiniBand
- Comprehensive Evaluation regarding the impact of Parallel Replication on different Hadoop benchmarks
- Integration with HBase leads to performance improvement of HBase Put operation
- Identify architectural bottlenecks of higher level HDFS designs and propose enhancements to work with high performance communication schemes
- Integration with other Hadoop components designed over InfiniBand

Tutorial on August 23, 2013 (8:30 – 12:30)

Accelerating Big Data Processing with Hadoop and Memcached Using High Performance Interconnects: Opportunities and Challenges

by

D. K. Panda and Xiaoyi Lu

The Ohio State University

Thank You!

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

Network-Based Computing Laboratory

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

Hadoop Web Page

http://hadoop-rdma.cse.ohio-state.edu/