Future Technologies

Mahidhar Tatineni
User Services, SDSC
Costa Rica Big Data School
December 7, 2017

Future Technologies

- **New software technologies, including several leveraging Spark**
- **Hardware evolution**
  - GPUs
  - Intel Nervana Neural Network Processor
  - Google Tensor Processing Units
- **Convergence of HPC and Big Data Hardware**
  - RDMA-Spark, Hadoop: Ongoing research at OSU and collaborative work between OSU, SDSC.

Spark Improvements

- Project Tungsten - optimizing Spark jobs for CPU and memory efficiency.
- Integration with deep learning frameworks, including TensorFlow and Intel's new BigDL library
- Cost-based optimizer framework for Spark SQL.
- Continuing improvements to the Python and R APIs.

Tungsten Optimization Features

- Off-Heap Memory Management using binary in-memory data representation
- Cache Locality: cache-aware computations with cache-aware layout for high cache hit rates
- Focuses on the hardware architecture of the platform being used by Spark (JVM, LLVM, GPU, NVRAM).
- Feasible to use certain non-JVM libraries (e.g BLAS) by mitigating cost of copying the data off-heap

Tungsten: Benefits

- Dataframes: Java, Scala, Python, R
- SparkSQL queries
- Some RDD API programs via general serialization and compression optimizations

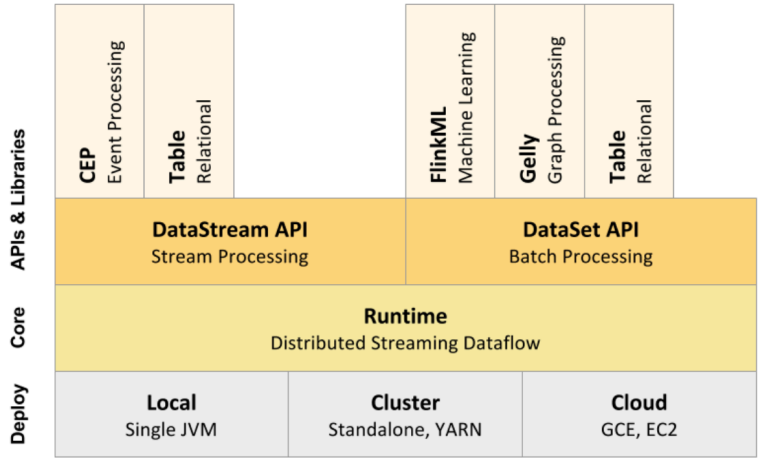
Ref : https://community.hortonworks.com/articles/72502/what-is-tungsten-for-apache-spark.html

New Projects leveraging Spark

- **Clipper -** Low latency online prediction serving system.
- **Drizzle -** execution engine for Spark - group scheduling, reduction in streaming and iterative operations.
- **Ray -** distributed execution engine for Spark
- **Apache SystemML -** library of machine learning algorithms design to scale on Apache Spark clusters.

Apache Flink

- **Open-source framework for distributed stream processing, unbounded datasets**
  - Can handle out-of-order or late-arriving data
  - Stateful and fault-tolerant
  - Performance at large scales



Ref:https://flink.apache.org/introduction.html

Apache Flink

- **Designed for handling**
  - Variety of (sometimes unreliable) data sources
  - Applications with state
  - Real-time or near real-time data processing

- **Example applications**
  - Optimization of search results in real time
  - Stream processing
  - Network/sensor monitoring, error detection
  - ETL for BI infrastructure

Ref: https://flink.apache.org/

Apache Kafka

- **Distribute streaming platform**

- **Typical applications**
  - Messaging
  - Website Activity Tracking
  - Operational monitoring and metrics
  - Log Aggregation
  - Stream Processing

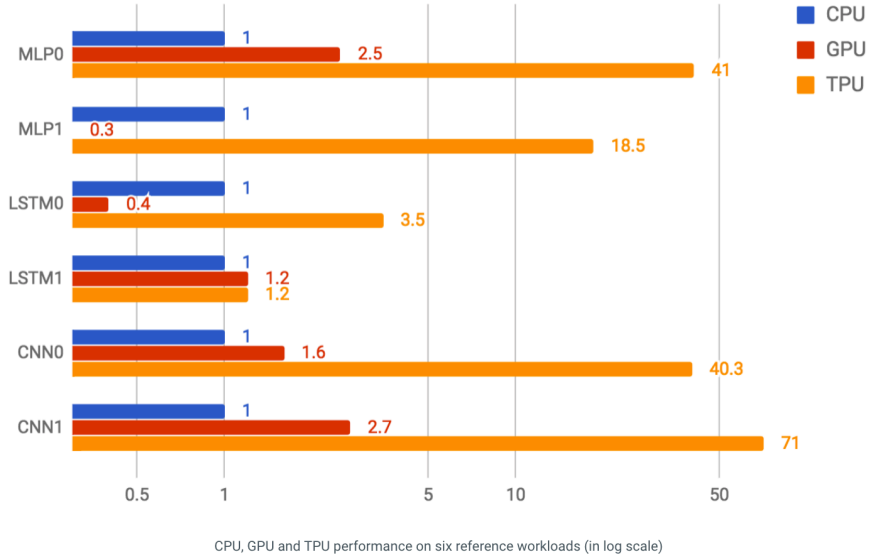
Ref: https://kafka.apache.org/

H_2O.ai

- **Machine learning and predictive analytics platform.**
  - Open source
  - In-memory, distributed, and highly scalable.
  - Python, R
- **Both Hadoop and Spark integration**
- **Algorithms**
  - Deep Learning
  - Gradient Boosting Machine (GBM)
  - Generalized Linear Model (GLM)
  - Kmeans
  - Distributed Random Forest (DRF)
- **Ref: https://www.h2o.ai/**

Google Tensor Processing Unit (TPU)

- **Uses Quantization:** optimization technique that uses an 8-bit integer to approximate an arbitrary value between a preset minimum and a maximum value.
- TPU contains 65,536 8-bit integer multipliers
- High level instructions defined neural network inference.
- Matrix multiplier unit (MXU) can do **hundreds of thousands of operations** per clock cycle.
- Systolic Array architecture designed to accelerate MXU operations.



CPU, GPU and TPU performance on six reference workloads (in log scale)

Ref: https://cloudplatform.googleblog.com/2017/04/quantifying-the-performance-of-the-TPU-our-first-machine-learning-chip.html

Intel Nervana Neural Network Processor

- Purpose built for deep learning
- Does not have standard cache hierarchy and on-chip memory managed by software directly.
- Takes advantage of apriori knowledge of operations/data movement
- High speed on- and off-chip interconnects.
- New numeric format (Flexpoint). Allows for increase in parallelism and lowers power requirement.
- Aims for 100x increase in deep learning performance by 2020.
- https://www.intelnervana.com/intel-nervana-neural-network-processor-architecture-update/

HPC and Big Data Convergence

- Existing HPC users already have “big data” problems! Scales of simulations have been increasing with larger and higher resolution problems.
- Rapid rise in data driven science (e.g. Bio-informatics, high energy physics) on HPC machines.
- Hardware like GPUs are optimal for segments of both HPC and Big Data workloads leading to HPC designs that can be effectively used for Big Data.
- HPC machines feature a rich set of storage and memory options - High Bandwidth Memory (HBM), SSDs, NVMe, and high performance parallel filesystems like Lustre, GPFS.

=> Opportunity for convergence of HPC and Big Data resources - both hardware and software.

RDMA Spark/Hadoop - HPC+Big Data example

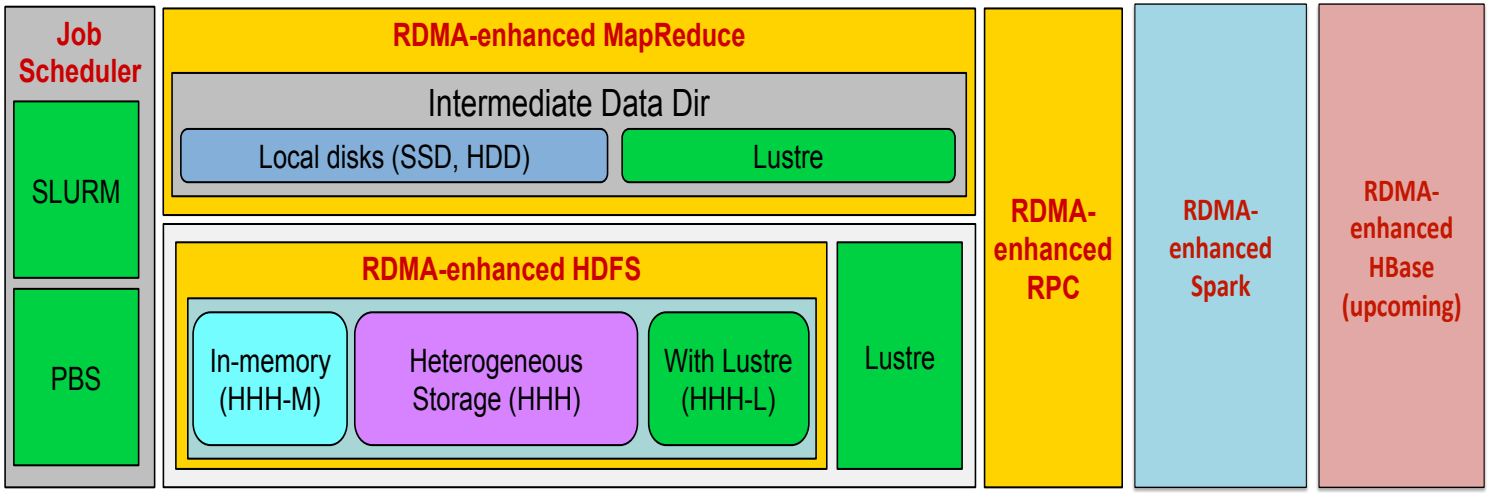
- HPC architectures feature hardware designed for large scale processing with advanced processors, large and high performance parallel filesystems, advanced networks.
- RDMA technology allows processes to access remote memory locations with very low CPU load on the remote process.
- SSDs with high data reliability and performance available on several HPC machines
- Standard big data middleware and tools like Apache Hadoop and Spark are currently not able to fully take advantage of these capabilities

⇒ ***Motivation*** to develop optimized versions of Apache Hadoop and Spark to fully realize the benefits of the architecture.

- SDSC Comet is an excellent test-bed for such tools, with advanced processors, filesystems, SSDs, and high performance networks.

RDMA-Hadoop and Spark Design

- Exploit performance on modern clusters with RDMA-enabled interconnects for Big Data applications.
- Hybrid design with in-memory and heterogeneous storage (HDD, SSDs, Lustre).
- Keep compliance with standard distributions from Apache.



RDMA-Hadoop and RDMA-Spark

Network-Based Computing Lab, Ohio State University

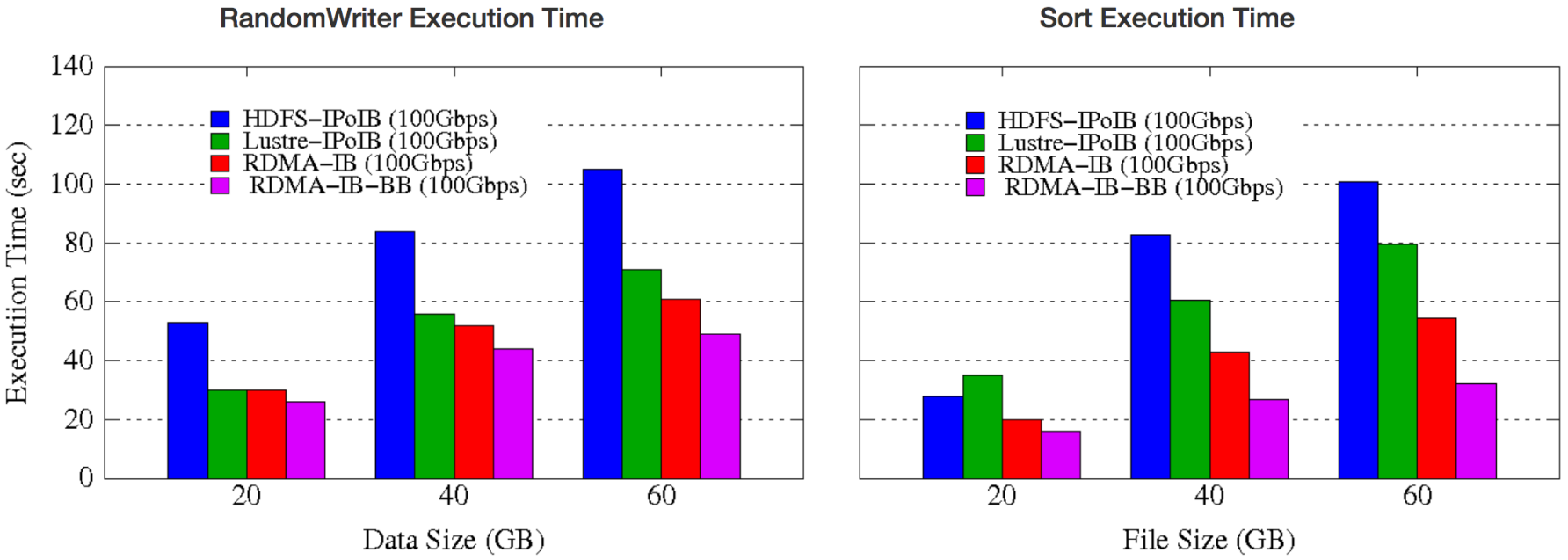
NSF funded project in collaboration with Dr. DK Panda

- HDFS, MapReduce, and RPC over native InfiniBand and RDMA over Converged Ethernet (RoCE).
- Based on Apache distributions of Hadoop and Spark.
- Version RDMA-Apache-Hadoop-2.x 1.1.0 (based on Apache Hadoop 2.6.0)
- Version RDMA-Spark 0.9.4 (based on Apache Spark 2.1.0)
- Version RDMA-based Apache HBase 0.9.1 (RDMA-HBase) based on Apache HBase 1.1.2
- More details on the RDMA-Hadoop and RDMA-Spark projects at:
  - http://hibd.cse.ohio-state.edu/

RDMA Hadoop – Scheduler Integration

- **Directly integrated with Slurm/PBS**
- **Several running modes:**
  - HHH: Heterogeneous storage devices with hybrid replication schemes for fault tolerance and performance.
  - HHH-M: A high-performance in-memory based setup.
  - HHH-L: Integrated with Lustre parallel filesystem.
  - MapReduce over Lustre, with/without local disks: Provides support to run MapReduce jobs on top of Lustre alone.

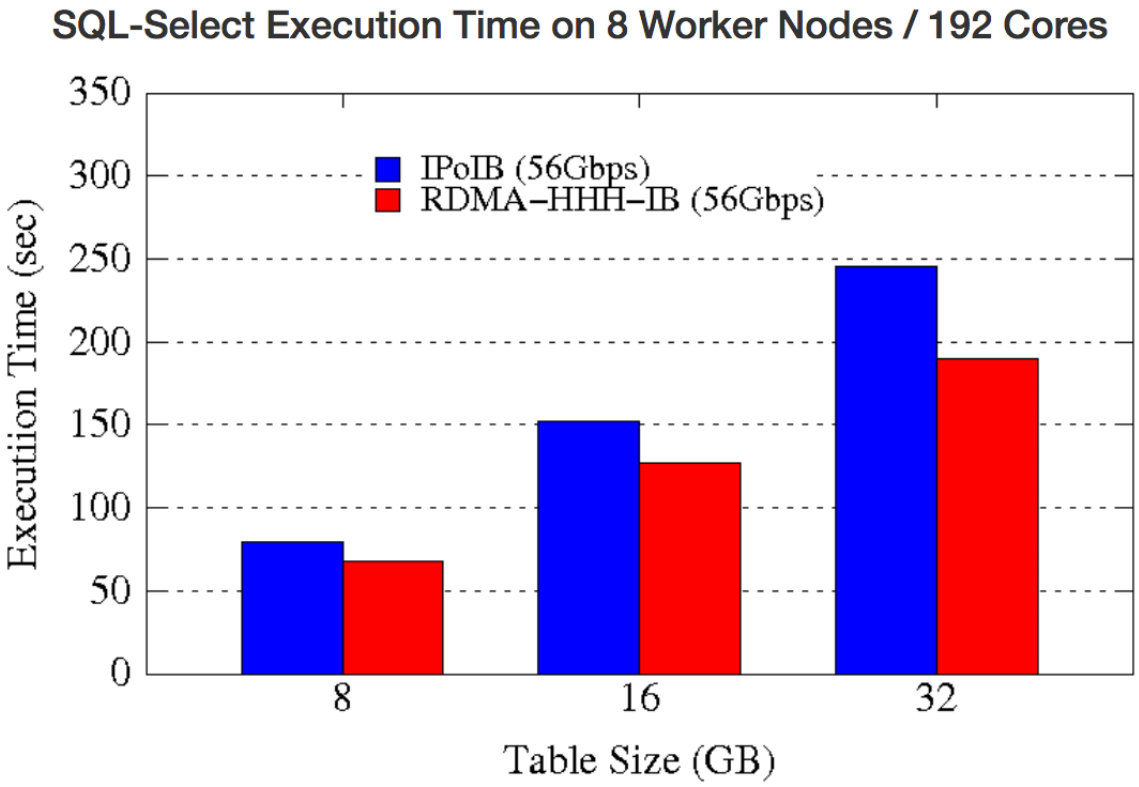
RDMA Hadoop: Random Writer and Sort



RDMA-IB design improves the job execution time of RandomWriter by 43% comapred to HDFS-IPoIB (100Gbps) and 14% compared to Lustre-IPoIB (100Gbps). The RDMA-IB-BB design improves the job execution time of RandomWriter by 53% comapred to HDFS-IPoIB (100Gbps) and 31% compared to Lustre-IPoIB (100Gbps). For Sort, RDMA-IB has an improvement of 46% compared to HDFS-IPoIB (100Gbps) and 43% compared to Lustre-IPoIB (100Gbps). The RDMA-IB-BB design has an improvement of 68% compared to HDFS-IPoIB (100Gbps) and 59% compared to Lustre-IPoIB (100Gbps).

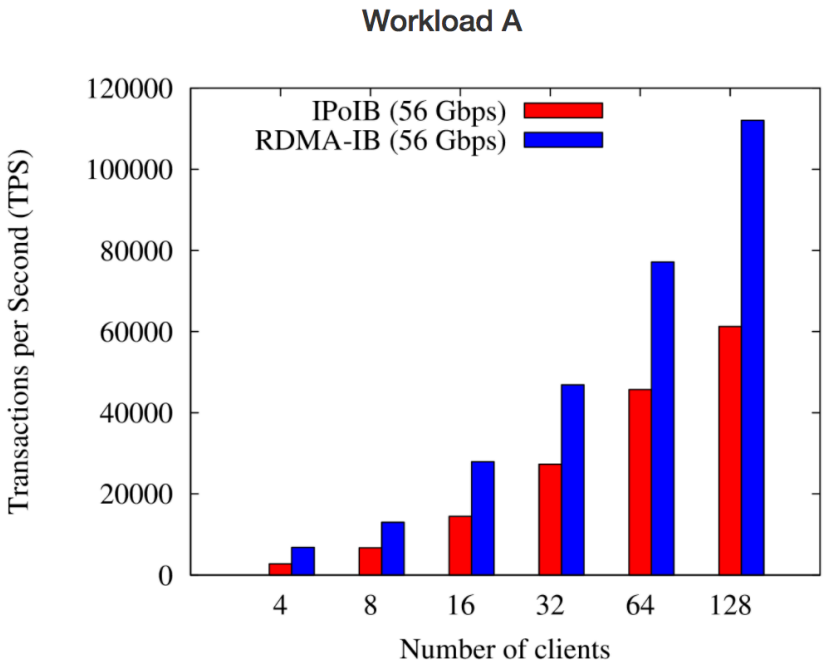
Ref: http://hibd.cse.ohio-state.edu/performance/withlustre_hdp

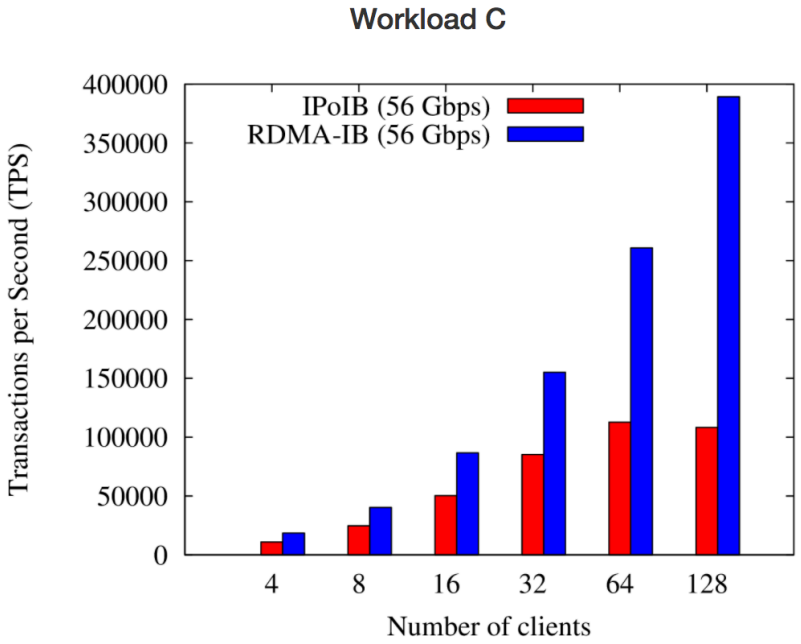
RDMA Spark, HDFS : SQL-Aggregation



RDMA-enhanced Heterogenous HDFS design HHH as the underlying filesystem that stores the Hive tables. Spark is run in Standalone mode. SSD is used for Spark Local and Work data. The RDMA-IB design with HHH improves the job execution time of SQL-Select by 14% - 22% compared to IPoIB (56Gbps).

RDMA-Hbase: Yahoo Cloud Serving Benchmark (YCSB)





These experiments are run with 4 Regionservers and upto 16 client nodes, each running 1-8 HBase client instances. Each client uses 1 thread for communication and performs 25000 operations on the HBase cluster. RDMA-IB design improves the throughput for Workload A, B, and C by up to 145.51%, 164.82%, and 259.41% over IPoIB (56Gbps), respectively.

Ref: http://hibd.cse.ohio-state.edu/performance/ycsb/

RDMA Hadoop: Benchmark Results

- HDFS I/O performance tested using TestDFSIO and RandomWriter.
- Hadoop performance was tested using Sort and TeraSort benchmarks.
- Performance of RDMA version compared with standard Apache version using the same hardware (Comet).
- Detailed results are available on the HiBD website (http://hibd.cse.ohio-state.edu/)

RDMA Hadoop: Benchmark Results

Benchmark Name	*Configuration*	*Max. Benefit over Apache Hadoop using IPoIB*
TestDFSIO Throughput	*RDMA Hadoop HHH-M, 16 Data Nodes*	*1.82x*
RandomWriter	*RDMA Hadoop HHH, 16 Data Nodes*	*1.32x*
Sort	*RDMA Hadoop HHH, 16 Data Nodes*	*1.48x*
TeraSort	*RDMA Hadoop HHH, 16 Data Nodes*	*1.23x*

RDMA Spark: Benchmark Results

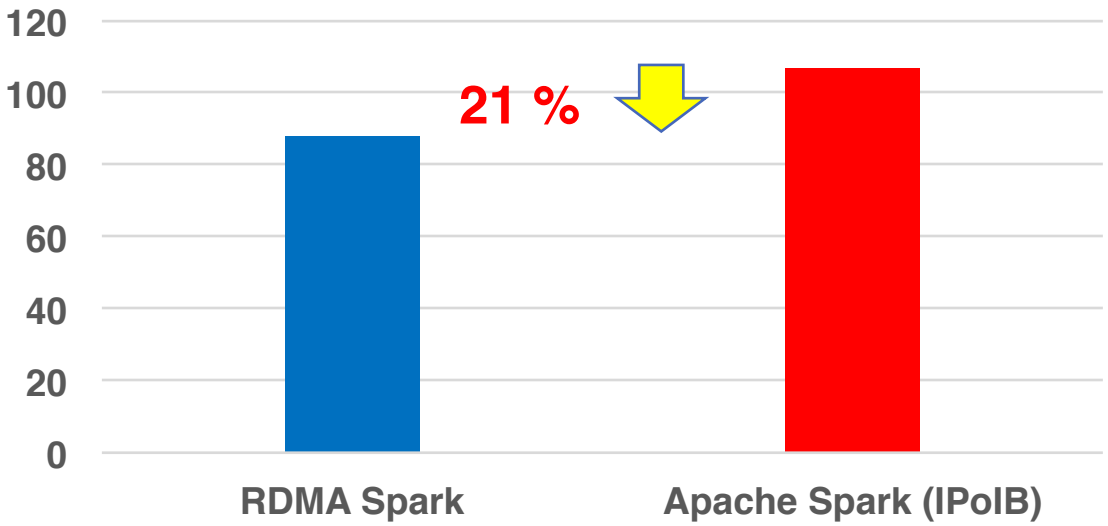
Benchmark Name	*Configuration*	*Max. Benefit over Apache Hadoop using IPoIB*
GroupBy	*RDMA Spark 64 nodes, 1536 maps/reduces*	*1.57x*
SortBy	*RDMA Spark 64 nodes, 1536 maps/reduces*	*1.8x*
HiBench TeraSort	*RDMA Spark 64 nodes, 1536 cores*	*1.22x*
HiBench PageRank	*RDMA Spark 64 nodes, 1536 cores*	*1.46x*

RDMA-Spark: Applications

- **Kira Toolkit[1]: Distributed astronomy image processing toolkit implemented using Apache Spark.**
- **Source extractor application, using a 65GB dataset from the SDSS DR2 survey that comprises 11,150 image files.**
- **Compare RDMA Spark performance with the standard apache implementation using IPoIB.**

1. Z. Zhang, K. Barbary, F. A. Nothaft, E.R. Sparks, M.J. Franklin, D.A. Patterson, S. Perlmutter. Scientific Computing meets Big Data Technology: An Astronomy Use Case. *CoRR, vol: abs/1507.03325*, Aug 2015.

RDMA-Spark: Applications



Execution times (s) for Kira SE benchmark using 65 GB dataset, 48 cores.

RDMA-Spark: Applications

- **Keystone ML* :** Large scale machine learning pipelines using Spark.
- Benchmark used: RandomPatchCifar pipeline is used with the CIFAR-10 dataset
- Compare RDMA Spark performance with the standard apache implementation using IPoIB.
- With 16 compute nodes an improvement of **4.7%** was seen with the RDMA version.

*http://keystone-ml.org/
Developed by UC Berkeley AMPLab

RDMA Spark: Applications

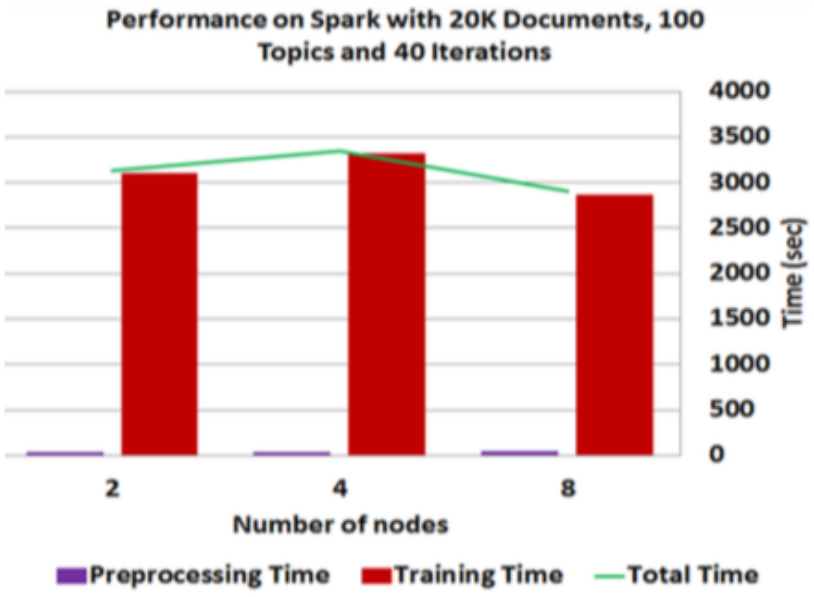
- Application in Social Sciences: Topic modeling using big data middleware and tools*.
- Latent Dirichlet Allocation (LDA) for unsupervised analysis of large document collections.
- Computational complexity increases as the volume of data increases.
- RDMA Spark enabled simulation of largest test cases.

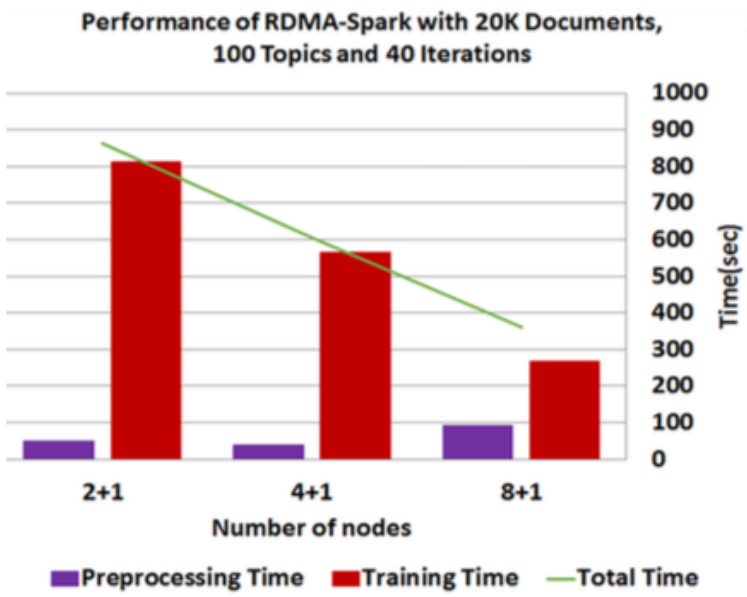
***See XSEDE16 Poster:**

Investigating Topic Models for Big Data Analysis in Social Science Domain

Nitin Sukhija, Nicole Brown, Paul Rodriguez, Mahidhar Tatineni, and Mark Van Moer

RDMA Spark: Topic Modeling Application

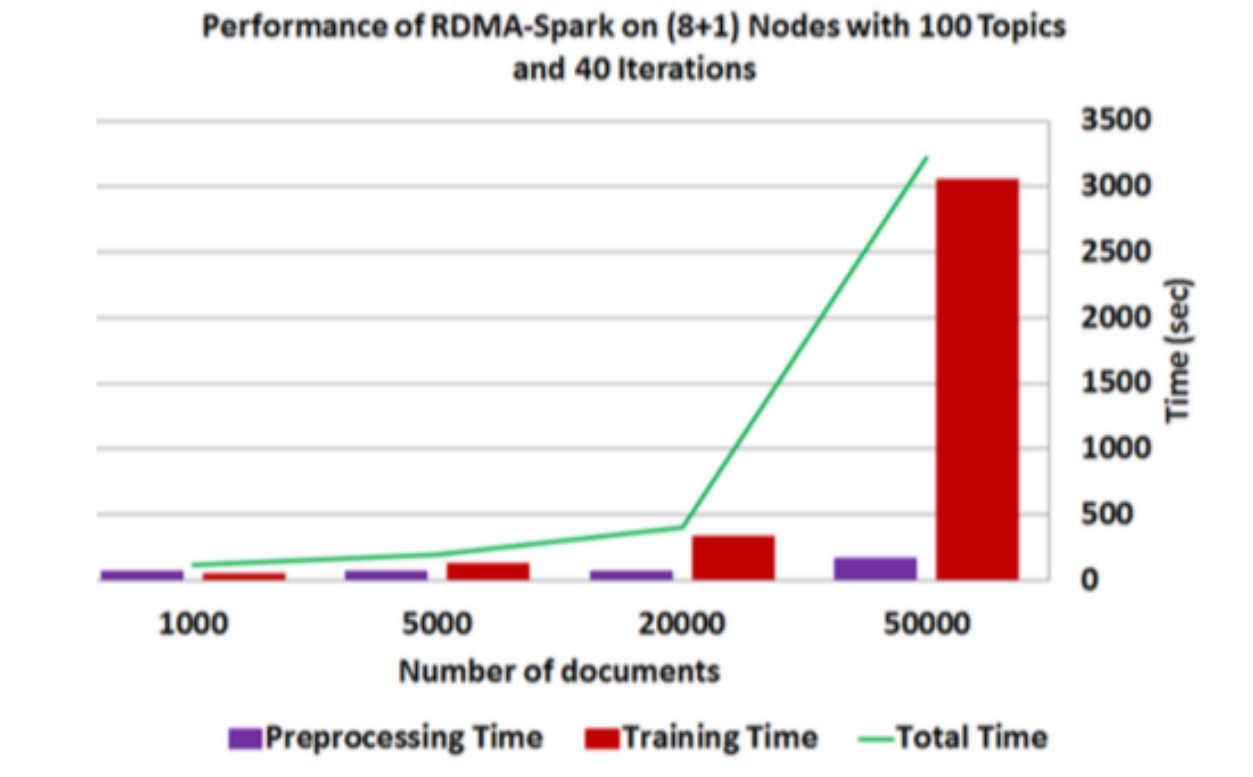




***Reference: XSEDE16 Poster:**

Investigating Topic Models for Big Data Analysis in Social Science Domain
Nitin Sukhija, Nicole Brown, Paul Rodriguez, Mahidhar Tatineni, and Mark Van Moer

RDMA Spark: Topic Modeling Application



***Reference: XSEDE16 Poster:**

Investigating Topic Models for Big Data Analysis in Social Science Domain
Nitin Sukhija, Nicole Brown, Paul Rodriguez, Mahidhar Tatineni, and Mark Van Moer

Summary

- Big data technologies are rapidly evolving to significantly increase capabilities
- Developments in both software and hardware
- Several projects underway to improve existing frameworks like Spark
- Aggressive optimizations in hardware design with specific computational characteristics in mind - e.g. Google TPU, Intel Nervana Neural Network Processors
- Large scale high performance computing (HPC) and Big Data architectures can converge with performance improvements.
- RDMA-Hadoop, Spark from HiBD group at OSU are examples of such converged solutions that leverage existing HPC hardware to obtain performance improvements.