Use of Hadoop File System for Nuclear Physics Analyses in STAR

EVAN SANGALINE

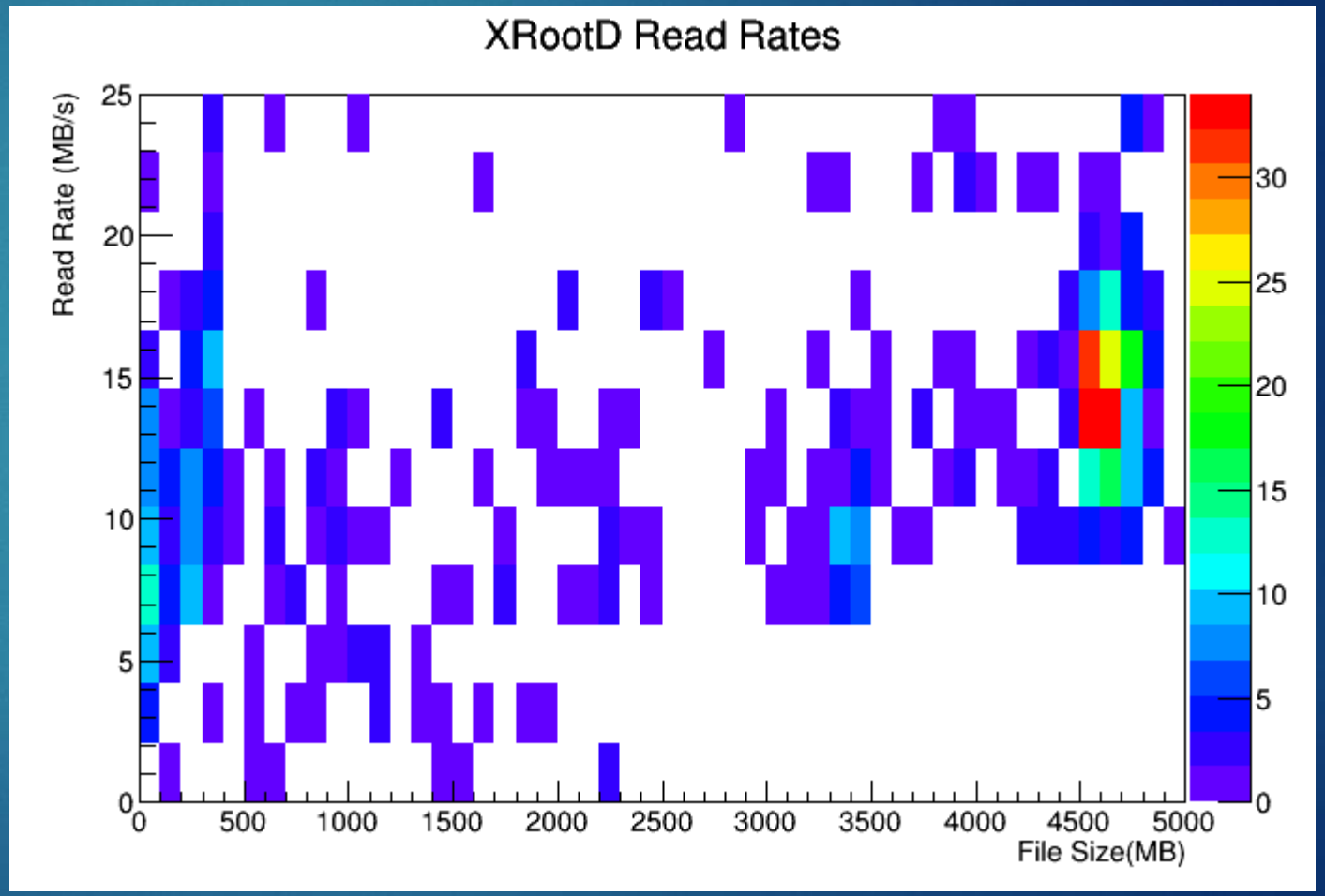
UC DAVIS

Motivations

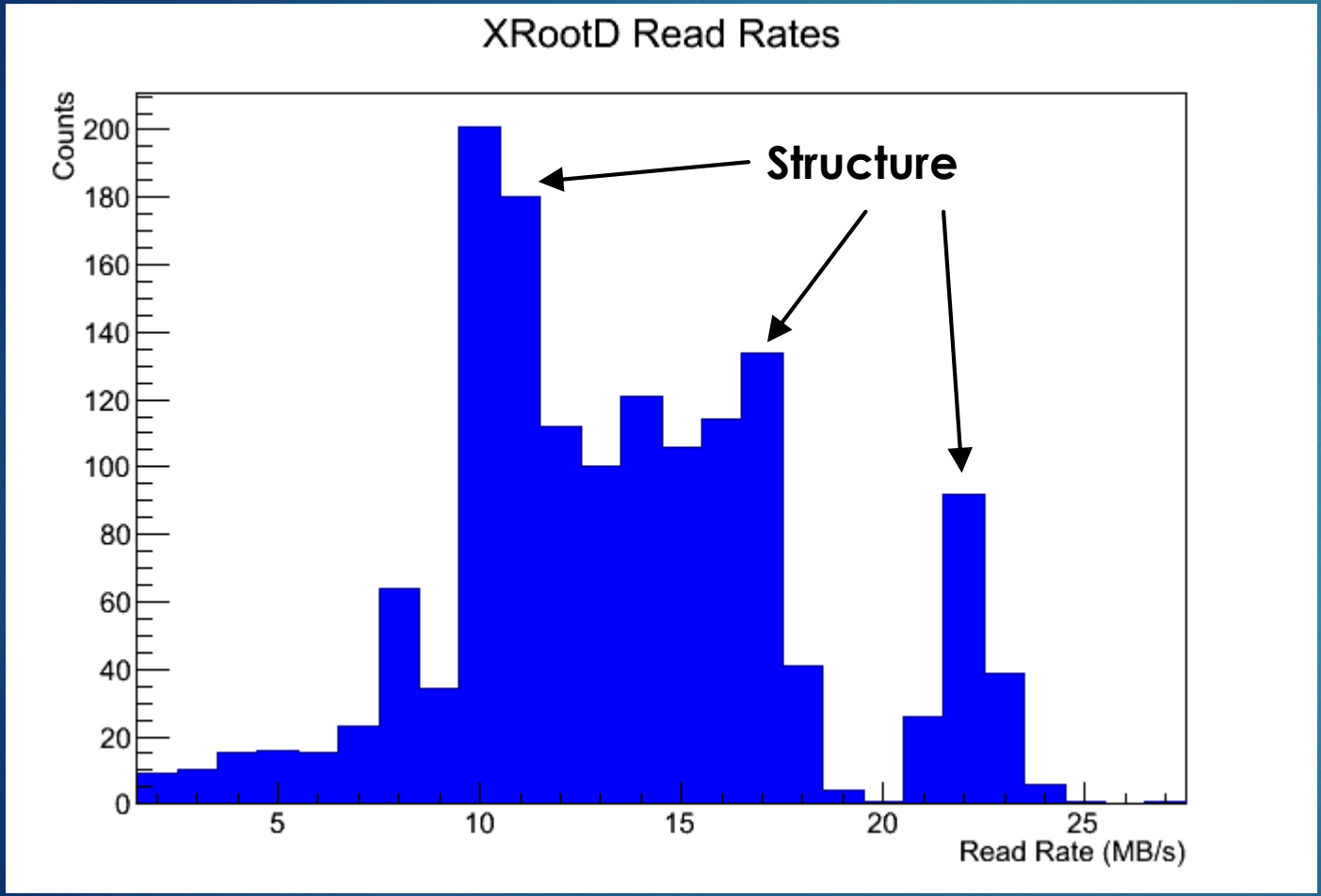
- Data storage a key component of analysis requirements
  - Transmission and storage across diverse resources
  - Large quantities of data
- XRootD offers a robust solution for STAR and other experiments
  - Works well but not designed for dynamic configuration
    - Utilization of on availability resources
  - Difficult to deploy on temporary/changing cloud resources
- Hadoop File System offers a possible alternative
  - Strong track record of performance in dynamic environments

XRootD Performance

- XRootD Read Rates
  - ROOT analysis jobs
  - Same hardware as test nodes
  - Mean read of 13.5 MB/s
- Baseline to compare Hadoop performance

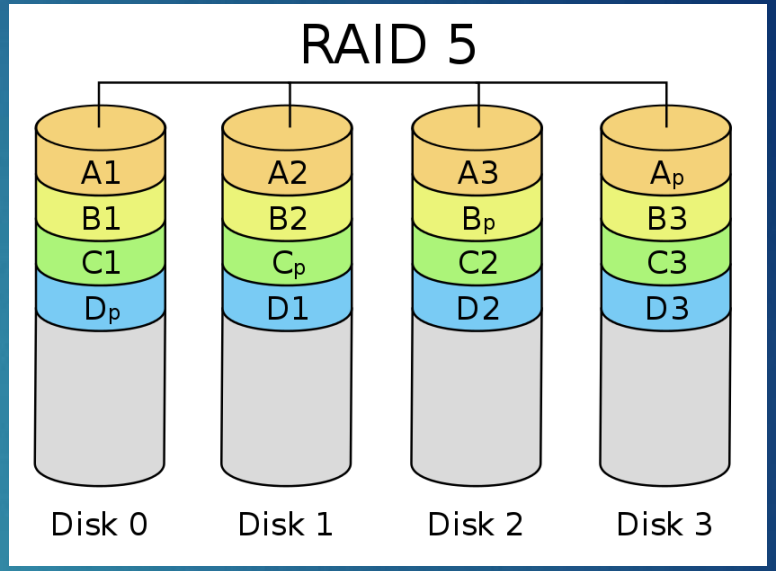


XRootD Performance



- Same analysis task
  - Filling two histograms
- Interesting structure
  - Due to different classes of files
    - Different triggers, etc
  - Different analyses would effect structure as well

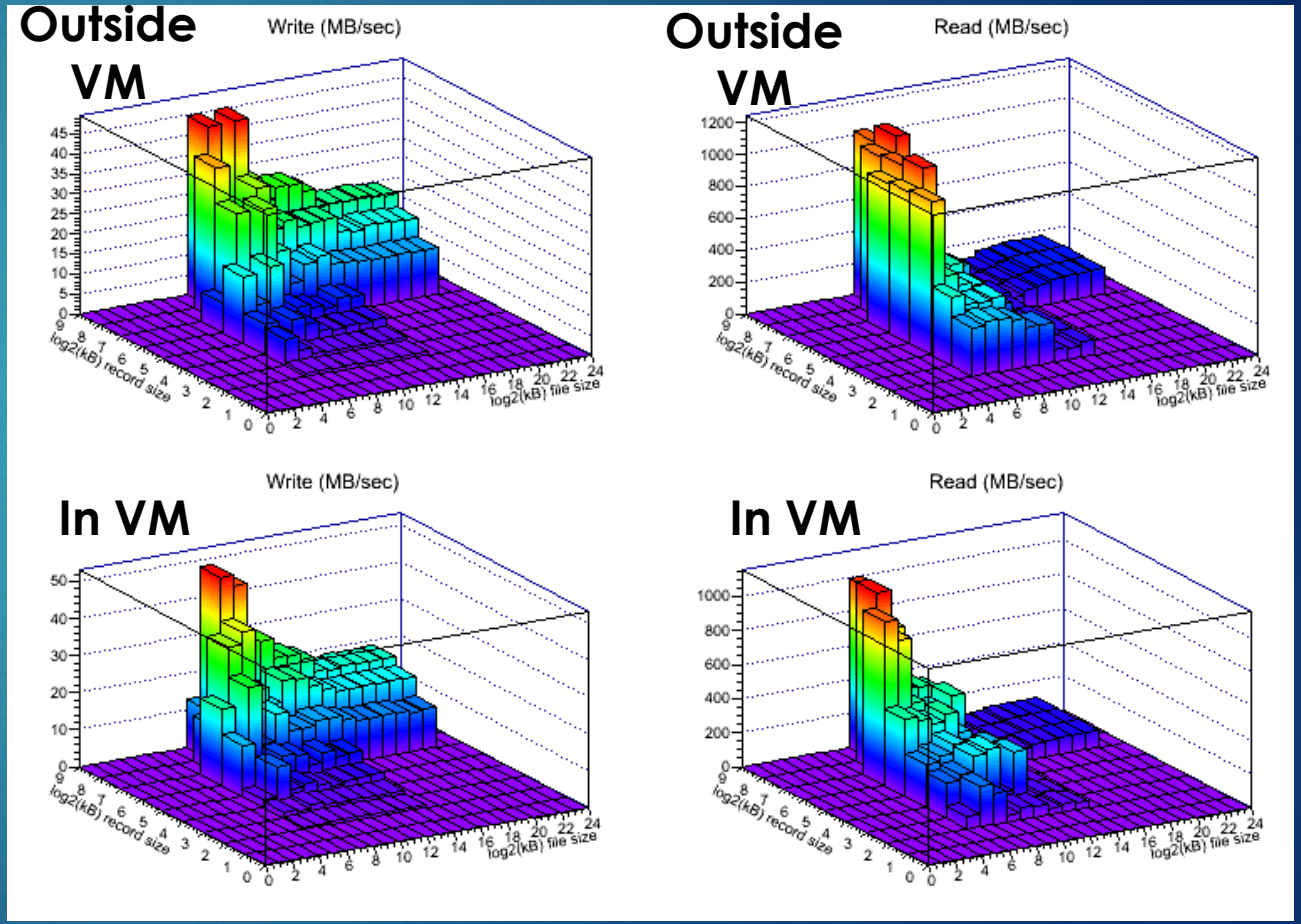
Test Bed for Performance Evaluation



- 25 virtual node cluster was constructed
  - OS: CentOS 5.9
  - Hypervisor: Xen
  - Storage: Four 2 TB drives in a RAID 5 array
  - RAM: 4GB
  - CPU: 1 core of a dual core 1.8 GHz AMD Opteron Processor
- Hadoop 1.1.2 was deployed with a 64 MB bock size
  - three methods of access
    - FUSE DFS interface
    - Cloudera NFS Proxy
    - Direct Hadoop client

VM Overhead in Disk Access

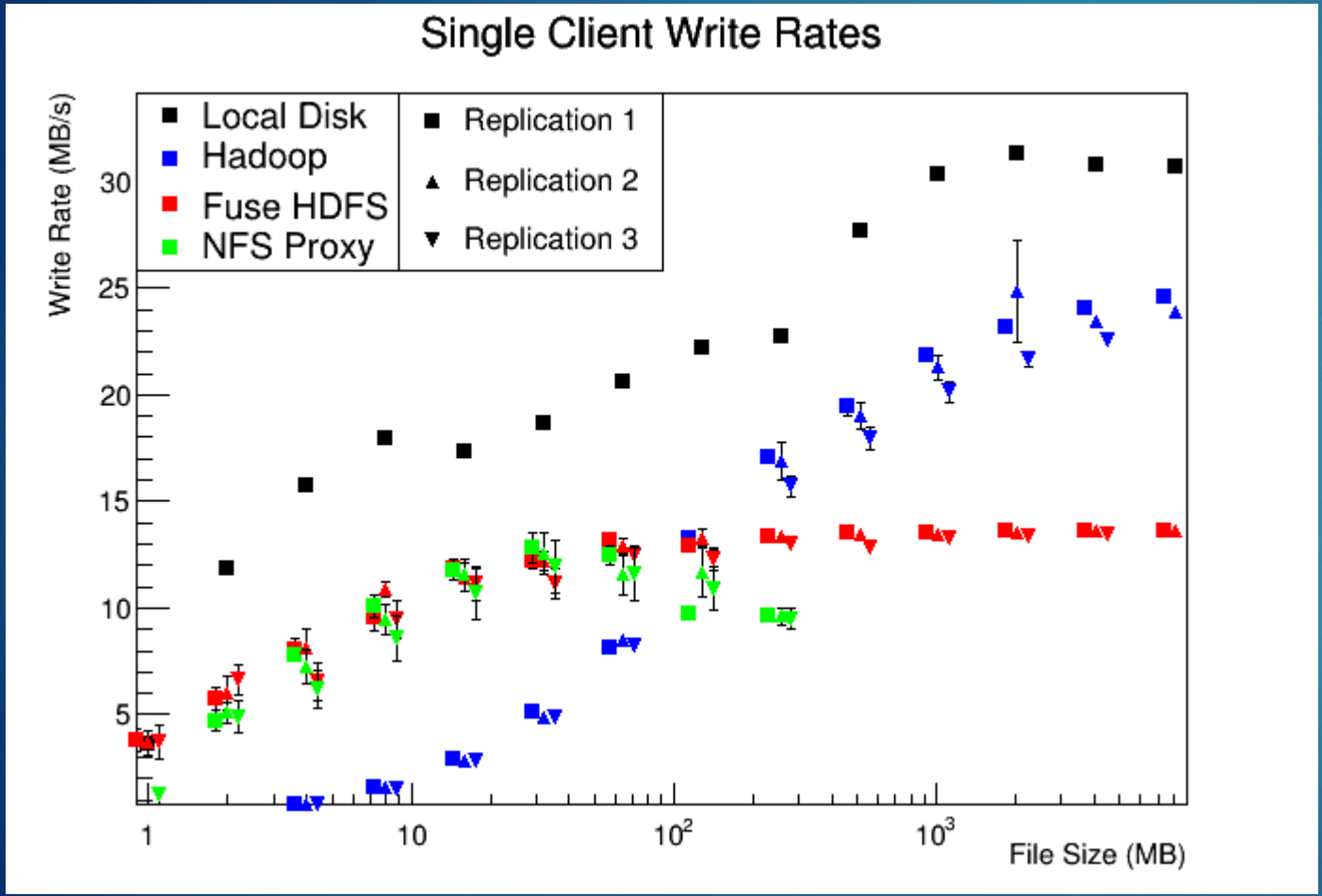
- Small overhead for writes
- More significant for reads
- Separate issue than Hadoop vs disk or XRootD
  - All comparisons done within a consistent environment



What is  ?

- Apache open source framework in Java
  - Based on Google's MapReduce and Google File System papers
  - Allows data to be redundantly stored across many nodes
  - A single job can be split across the nodes and analyze in parallel
  - Rack awareness allocates blocks and jobs intelligently
- Hadoop File System
  - Designed for sequential reading, usually text
  - No direct POSIX interface
    - FUSE DFS and Cloudera NFS Proxy
  - Easy to configure and to add/remove nodes
  - Robust against node failure

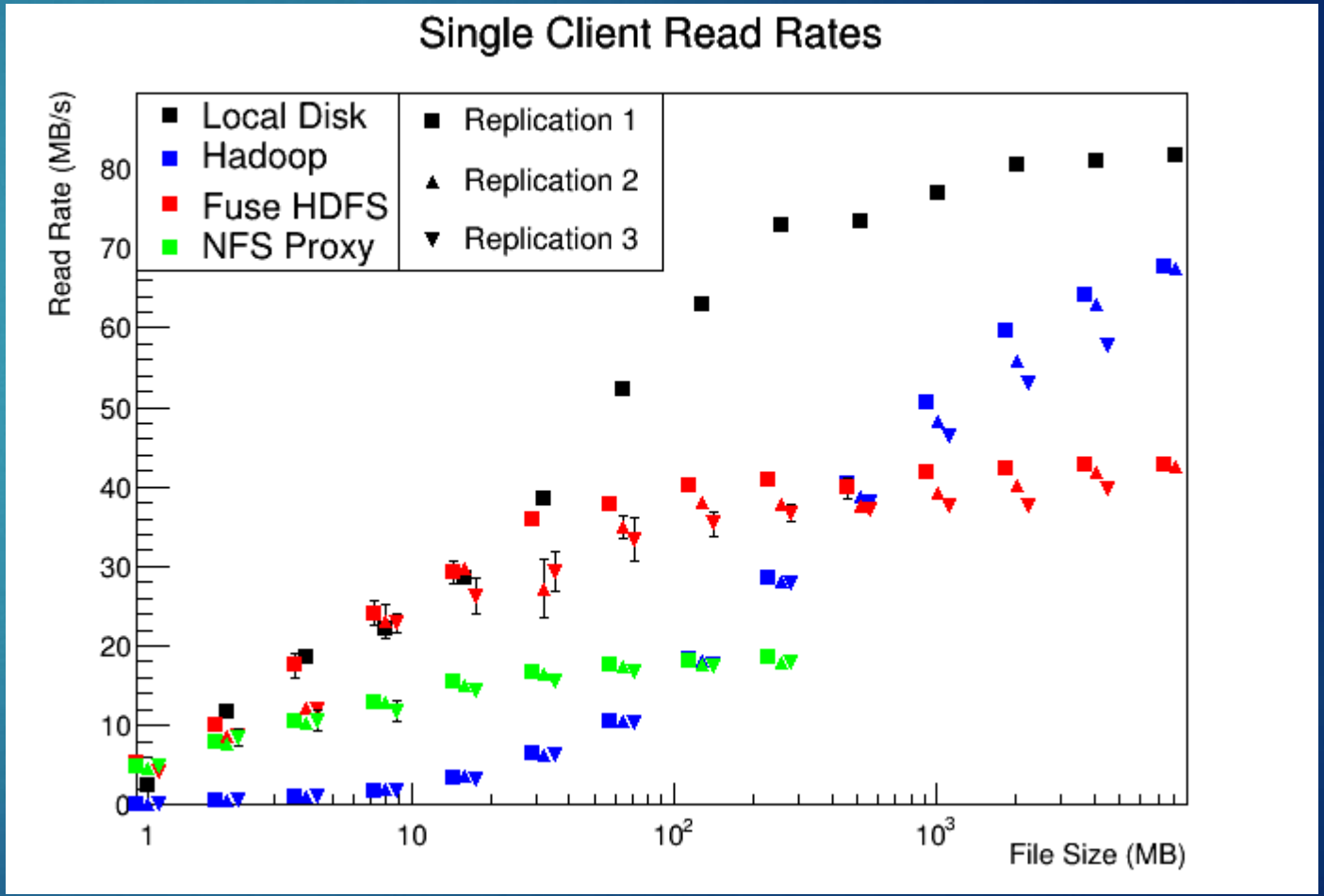
Write Rates for Various File Sizes



- FUSE DFS and NFS Proxy have similar performance
- NFS Proxy writes break for files larger than 300 MB
- Hadoop client has high overhead at low file sizes, does well with larger
- Low additional upfront cost for higher replications
- Hadoop at ~85% of disk rate for large files

Read Rates for Various File Sizes

- Only small gain in read rates for higher replications
- NFS rates are very low, break for large file sizes
- Hadoop again near 85% of disk for large files
- Large overhead for Hadoop client at low file sizes



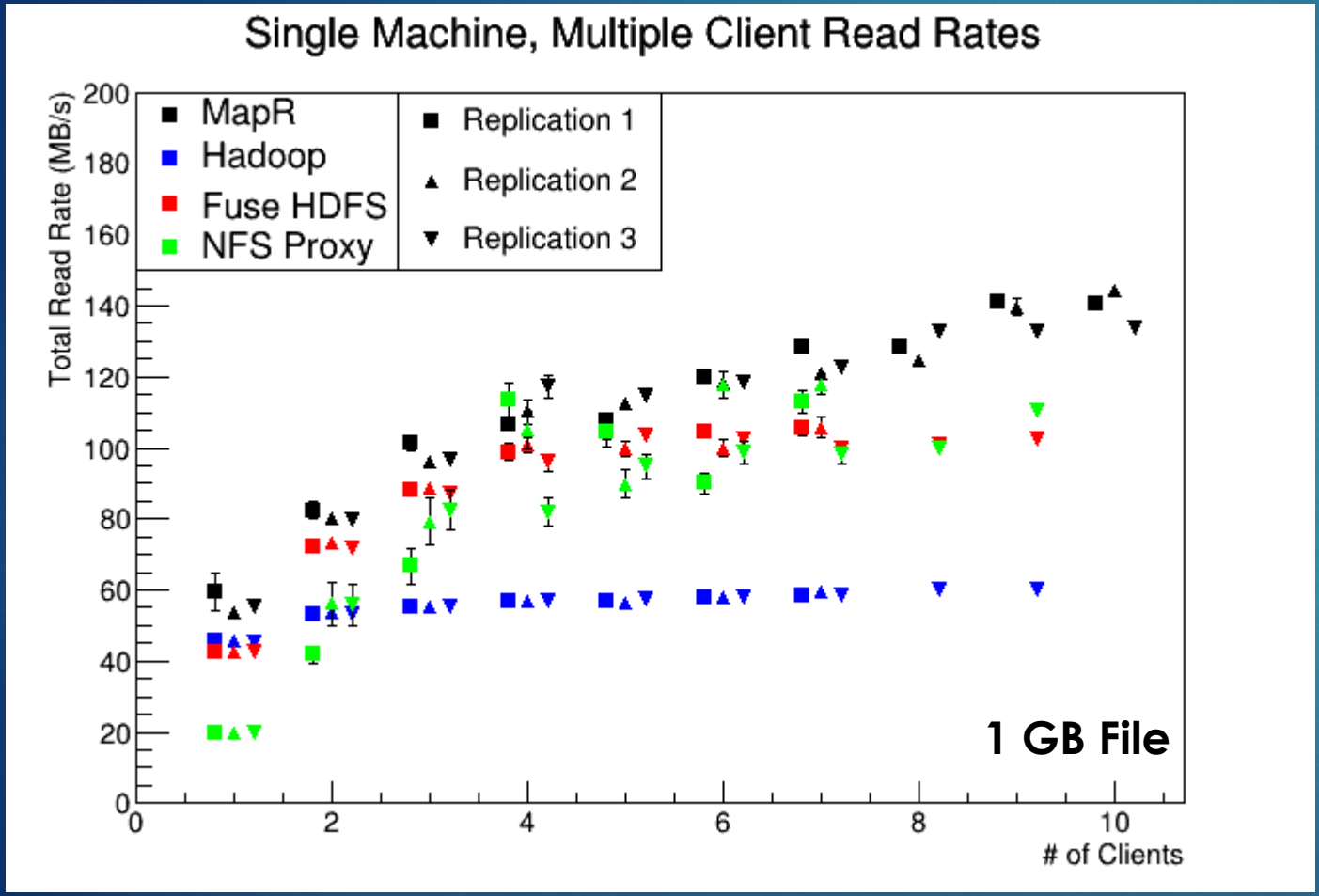
General Performance

- For transferring large amounts of data into the cluster Hadoop client allows for ~80% of local write performance
  - Replication comes basically for free
- Read rates through FUSE DFS are ~50% that of local reads
  - Might not matter for CPU bottlenecked analyses
- Fuse DFS outperforms NFS Proxy
  - NFS Proxy has issues with larger files
    - Not a viable candidate

What is MAPR™ TECHNOLOGIES ?

- A commercial distribution based on Apache Hadoop
- Uses a custom file system
  - Accesses partitions directly, not through another file system
  - Stripes across multiple partitions, analogous to the RAID 5
- Includes an NFS proxy
  - Only for one node in the free version (M3)
  - Multiple nodes requires a commercial license (M5)
- Good documentation and packages for installation

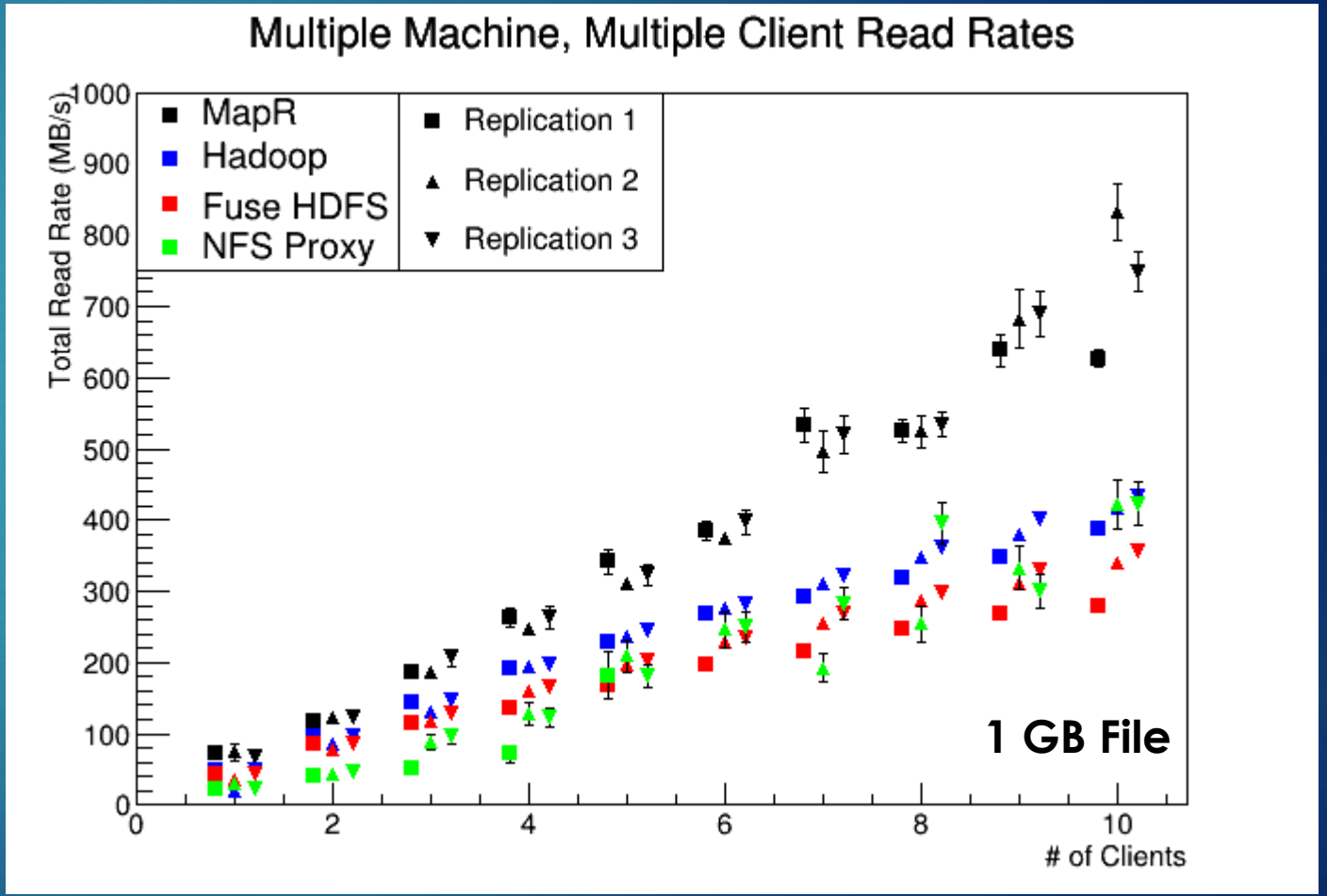
Multiple Clients Reading One File



- Little dependence on replication
- MapR is fastest but Fuse HDFS and NFS Proxy are comparable
  - Likely local caching
- Total throughput for Hadoop client flattens earlier
  - No local caching

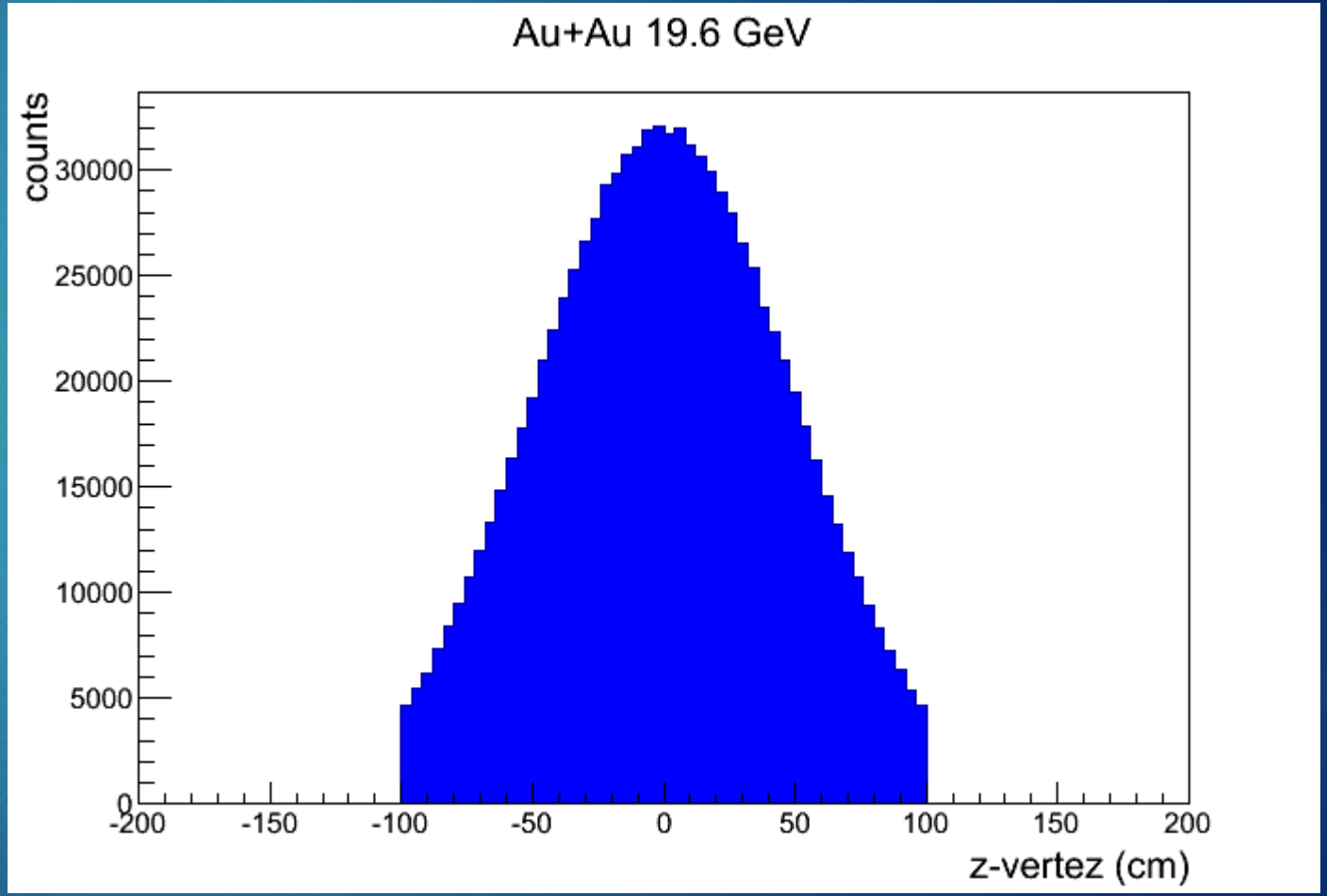
Multiple Clients Reading One File

- Little dependence on replication
- MapR is nearly twice as fast as Hadoop for many clients
- Total throughput scales almost linearly even for replication one
  - 64 MB blocks stored on separate machines



Actual Analysis Task

- Run over 1 GB ROOT file containg a TTree of event and track data
- Fill histograms with different track and event quantities
- Comparable to XRootD tests, simpler tree structure



Analysis Rates

	Fuse DFS	MapR	Disk
Read Rate (MB/s)	9.9+/-0.1	3.1+/-0.1	9.7+/-0.6

- FUSE DFS performance is consistent with local disk access
  - Comparable to XRootD performance (13.5 MB/s)
- MapR tests ran at ~30% the speed of disk or fuse
- NFS Proxy was unstable
- Running ROOT with libhdfs support was attempted
  - Issues with consistency, progress ongoing

Summary

- Reasonable performance for importing data files
  - 80% of writing to local disk using the Hadoop client
  - Little to no initial penalty for higher replication rates
- Fuse DFS performs as well as local disk in a typical ROOT analysis
  - 9.9 MB/s
- Total throughput of reads scales almost linearly
- MapR outperforms Hadoop at sequential reads but not in the analysis test
- NFS Proxy has issues with larger files and is not a robust option

Conclusions

- HDFS performs well for importing data files and for reading files for analysis through Fuse DFS
- It is easy to configure and to update dynamically
- It is well documented and is an active project
- Overall, it seems a well suited alternative to XRootD for use in dynamic cloud environments
  - More study is needed of high concurrency scaling