



Big Data Analytics on Traditional HPC Infrastructure Using Two-Level Storage

Dr. Feng Luo

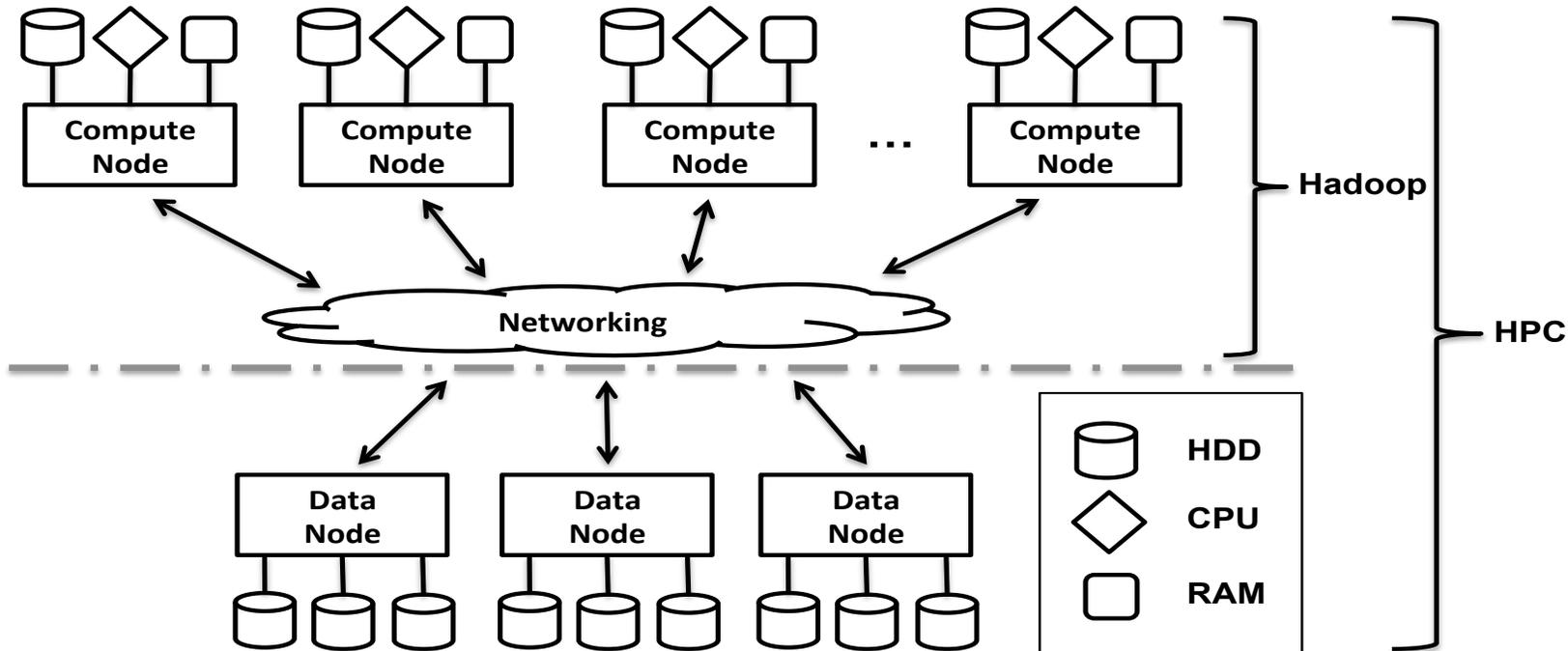
**School of Computing
Clemson University**





Architectures of Hadoop and HPC infrastructures

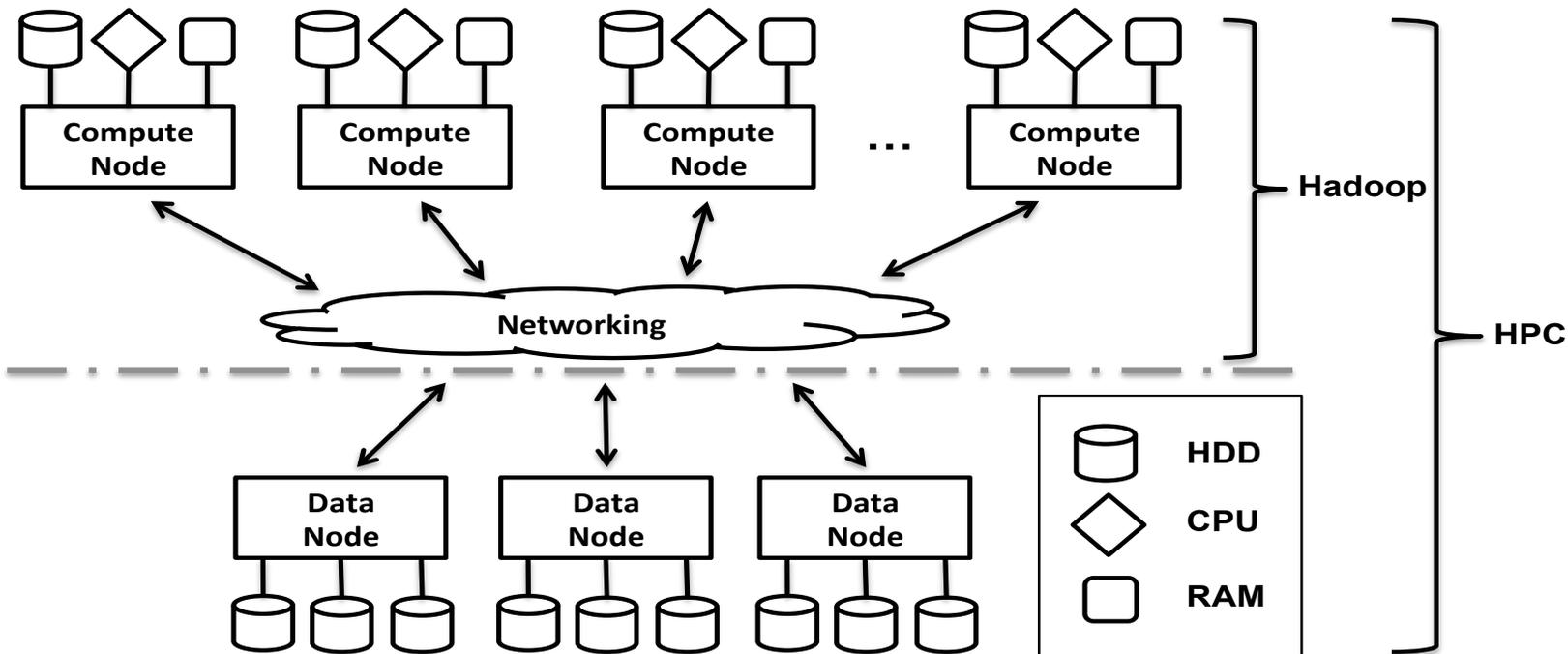
- In HPC architecture, data nodes are separated from compute nodes and they are connected via high-performance network.**
- two types of data storage services: persistent global-shared parallel file system on data nodes (large) and temporal local file system on compute nodes (small).





Architectures of Hadoop and HPC infrastructures

- In Hadoop architecture, compute node and data node are co-located in the same physical machine.**
 - The local storage device on each node is also used as part of the primary persistent data storage.
 - The computational task is scheduled to the physical machine where the required data is stored in order to achieve maximum data locality.





Data-intensive Computing on HPC: how to store the data

Using the parallel storage on HPC.

-  Provides high storage capacity with low cost data fault tolerance,
-  But has scalability issues limited by network and aggregate I/O bandwidth of storage nodes.

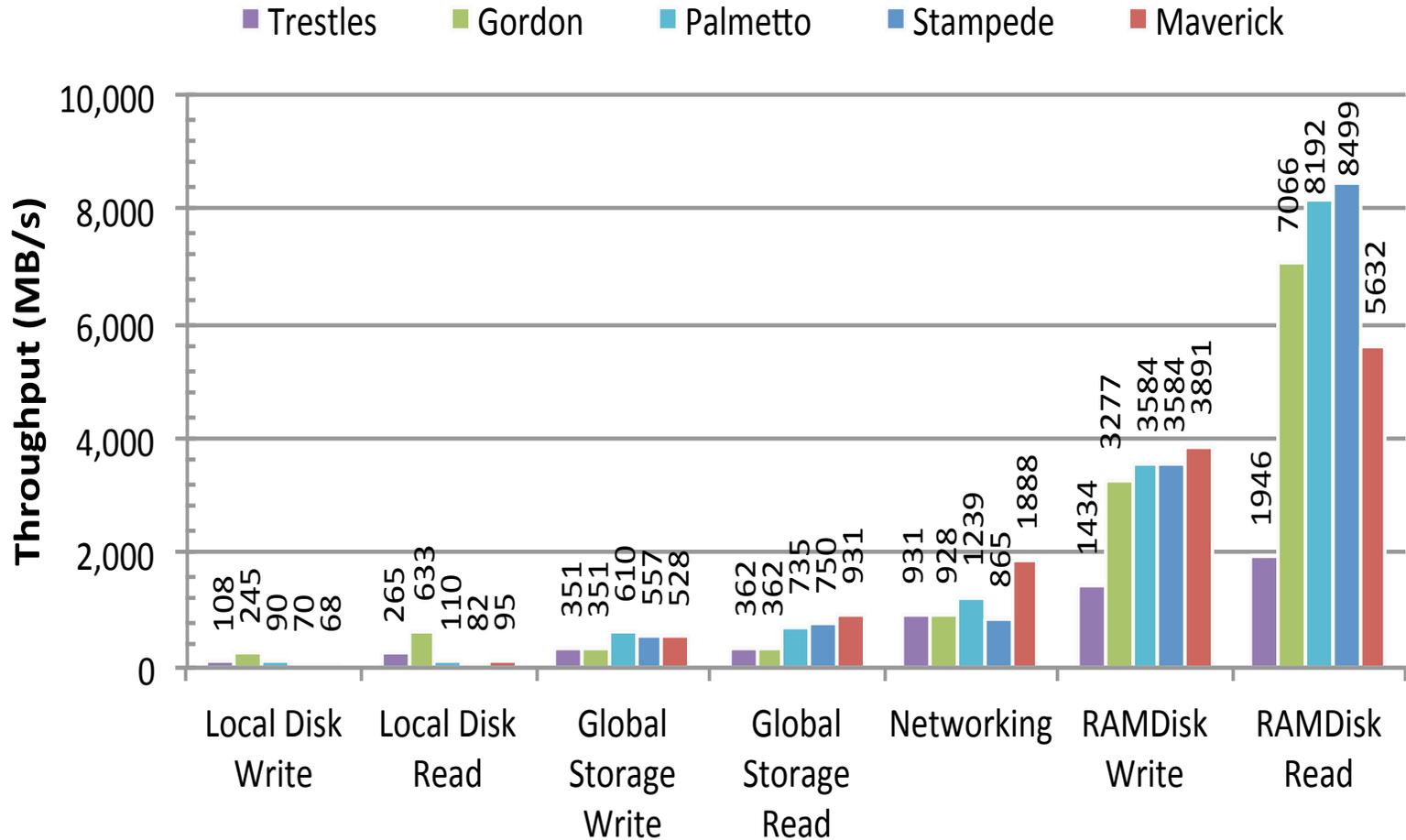
Deployed distributed file system for data-intensive computing, such as Hadoop distributed file system (HDFS), on compute nodes.

-  Delivers high aggregate I/O throughput,
-  But it suffers with a high cost for data fault tolerance and low data storage capacity.





I/O throughputs of a single compute node on national HPC clusters





Compute Node Storage Space Statistics on National HPC Clusters

HPC Cluster	Disk (GB)	RAM (GB)	PFS (GB)	CPU (Core)
Stampede	80	32	14×10^6	16
Maverick	240	256	20×10^6	20
Gordon	280	64	1.6×10^6	16
Trestles	50	64	1.4×10^6	32
Palmetto	900	128	0.2×10^6	20
Avg.	310	109	7.4×10^6	21





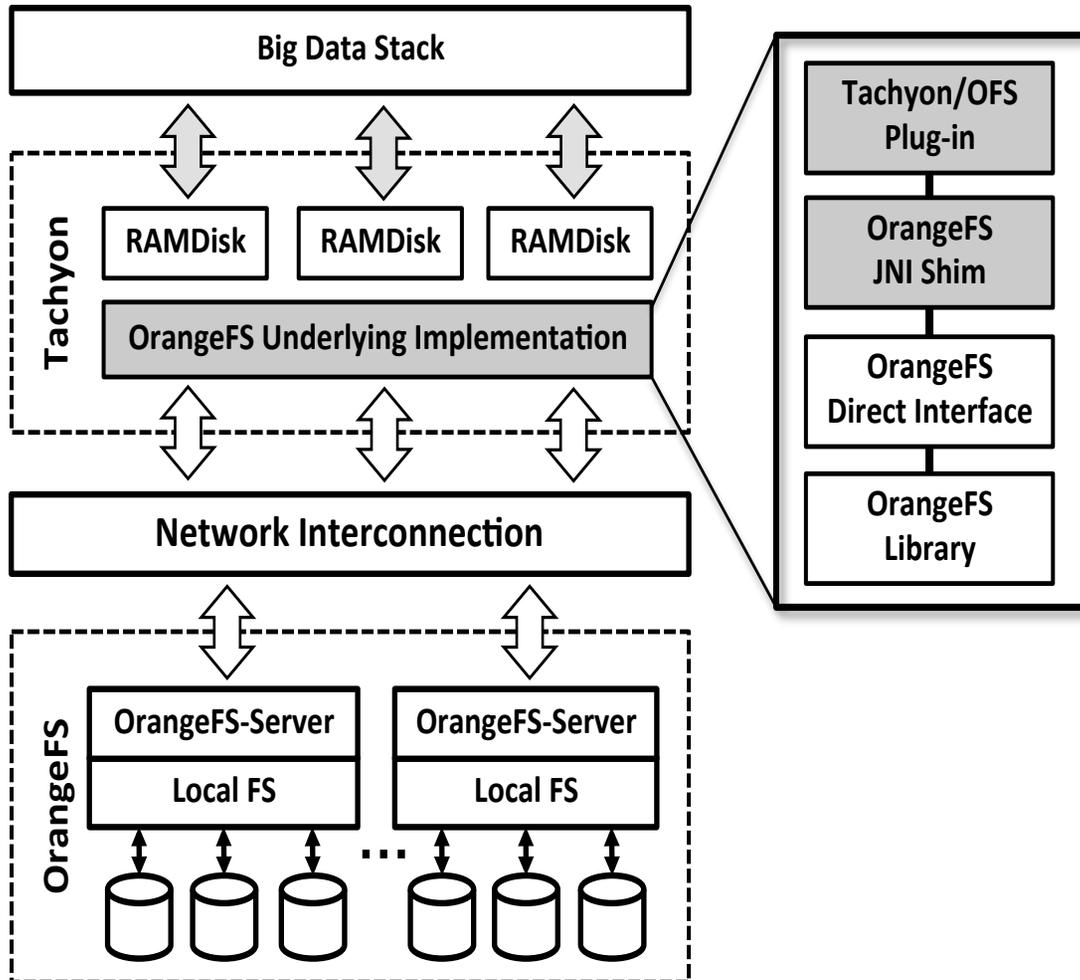
Our solution: Two-level Storage System

- 🐾 **Combined an in-memory file system on the compute nodes and a parallel file system on data nodes.**
 - 🐾 As the compute nodes on HPC clusters often have been equipped with large memory, the in-memory file system can also have storage capacity comparable to local storage-based HDFS.
 - 🐾 the I/O throughputs of in-memory file system are much faster than those of local disk.
 - 🐾 the parallel file system provides the data-fault tolerance and large storage capacity.
 - 🐾 the two-level storage will take advantages of both in-memory file system and parallel file system.





A prototype of two-level storage system

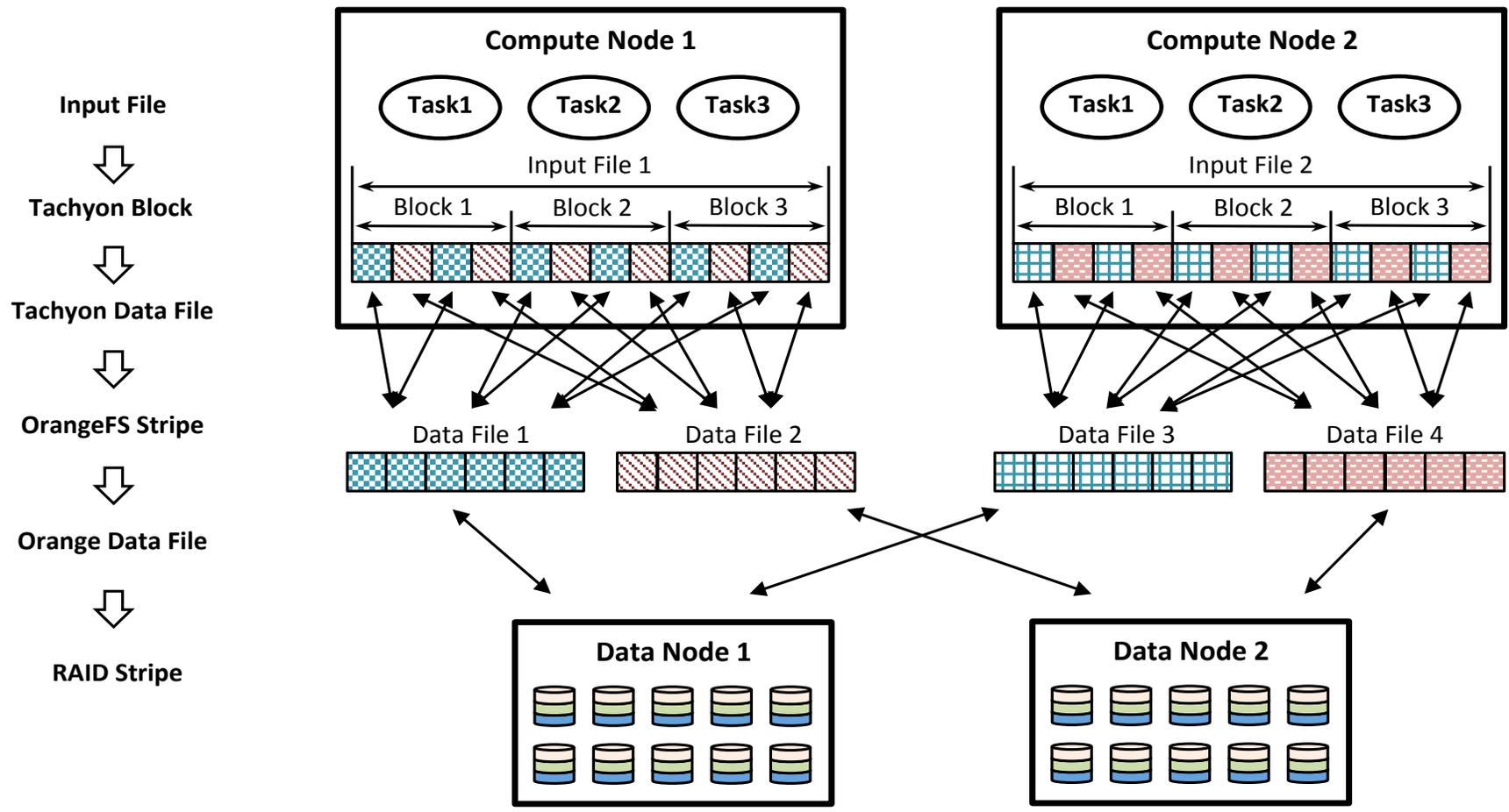


Integrating Tachyon-0.6.0, an in-memory with the OrangeFS-2.9.0, a parallel file system.

- Tachyon-OFS Plug-in: a Java plug-in that provides the interface to transform the functionalities of Tachyon in-memory file system to the functionalities of OrangeFS parallel file system.
- OrangeFS JNI Shim: a Java API that forwards all function calls from Tachyon-OFS Plug-in to the OrangeFS Direct Interface.



Data Layout Mapping





I/O Modes of Two-level Storage System

Three write modes

-  data is stored only in Tachyon
-  data is bypass Tachyon and written to OrangeFS
-  data is synchronously written to OrangeFS whenever data is created or updated in Tachyon

Three read modes

-  data is read from Tachyon only
-  data is read from OrangeFS directly without caching it in Tachyon
-  data is read from both Tachyon and OrangeFS.
 -  This is the primary usage pattern in data-intensive computing by caching reusable data to improve read performance with a matched data eviction policy, such as LRU/LFU.



MODELING I/O THROUGHPUTS OF DIFFERENT STORAGES

- ❃ We make the following simplifying assumptions in our modeling effort:
 - ❃ All nodes have identical hardware configurations and are connected via non-blocking switches.
 - ❃ The centralized switch and the bisection bandwidth of network are able to provide a non-blocking backplane throughput Φ and each node is attached by a full-duplex network interface with a bandwidth throughput ρ .
 - ❃ There is no network-level interference, such as TCP congestion, and Incast/Outcast.

Symbol	Significance
D	Data size
N	Number of compute nodes
M	Number of data nodes
f	The ratio of the size of data in Tachyon over the total size of data
Φ	Bandwidth of switch backplane, bisection bandwidth of network (MB/s)
ρ	Bandwidth of network interface of compute and data nodes (MB/s)
μ	I/O throughput of local hard drives on compute nodes (MB/s)
μ'	I/O throughput of local hard drives on data nodes (MB/s)
ν	I/O throughput of local memory (MB/s)
q	Average I/O throughput received on compute nodes (MB/s)





MODELING I/O THROUGHPUTS OF DIFFERENT STORAGES

I/O Modeling of HDFS

 The read throughput of each node, q_{read}^{HDFS} , is

$$q_{read}^{HDFS} = \begin{cases} \mu, & \text{\&local access} \\ \min\left(\rho, \frac{1}{N}\Phi, \mu\right), & \text{\&remote access} \end{cases} \quad (1)$$

 the write throughput of each node, q_{write}^{HDFS} , can be estimated as

$$q_{write}^{HDFS} = \min\left(\frac{1}{2}\rho, \frac{1}{2N}\Phi, \frac{1}{3}\mu\right) \quad (2)$$





MODELING I/O THROUGHPUTS OF DIFFERENT STORAGES

I/O Modeling of OrangeFS

-  the read throughput, q_{read}^{OFS} , and write throughput, q_{write}^{OFS} , of each compute node

$$q_{write}^{OFS} = q_{read}^{OFS} = \min\left(\rho, \frac{1}{N}\Phi, \frac{M}{N}\rho, \frac{M}{N}\mu'\right) \quad (3)$$

I/O Modeling of Tachyon

-  the read throughput of each node, $q_{read}^{Tachyon}$, is

$$q_{read}^{Tachyon} = \begin{cases} v, & local \\ \min\left(\rho, \frac{1}{N}\Phi, v\right), & remote \end{cases} \quad (4)$$

-  the write throughput of each compute node, $q_{write}^{Tachyon}$ is just limited by the throughput to memory:

$$q_{write}^{Tachyon} = v \quad (5)$$





MODELING I/O THROUGHPUTS OF DIFFERENT STORAGES

❁ I/O modeling of the Proposed Two-level Storage

- ❁ the write throughput of each compute node on two-level storage, q_{write}^{TLS} , is bounded by the write throughput to OrangeFS:

$$q_{write}^{TLS} = \min(q_{write}^{Tachyon}, q_{write}^{OFS}) = q_{write}^{OFS} \quad (6)$$

- ❁ Let f be the ratio of the size of data in Tachyon over the total size of data, D . the read throughput of each compute node is

$$q_{read}^{TLS} = 1 / \left(\frac{f}{v} + \frac{1-f}{q_{read}^{OFS}} \right) \quad (7)$$

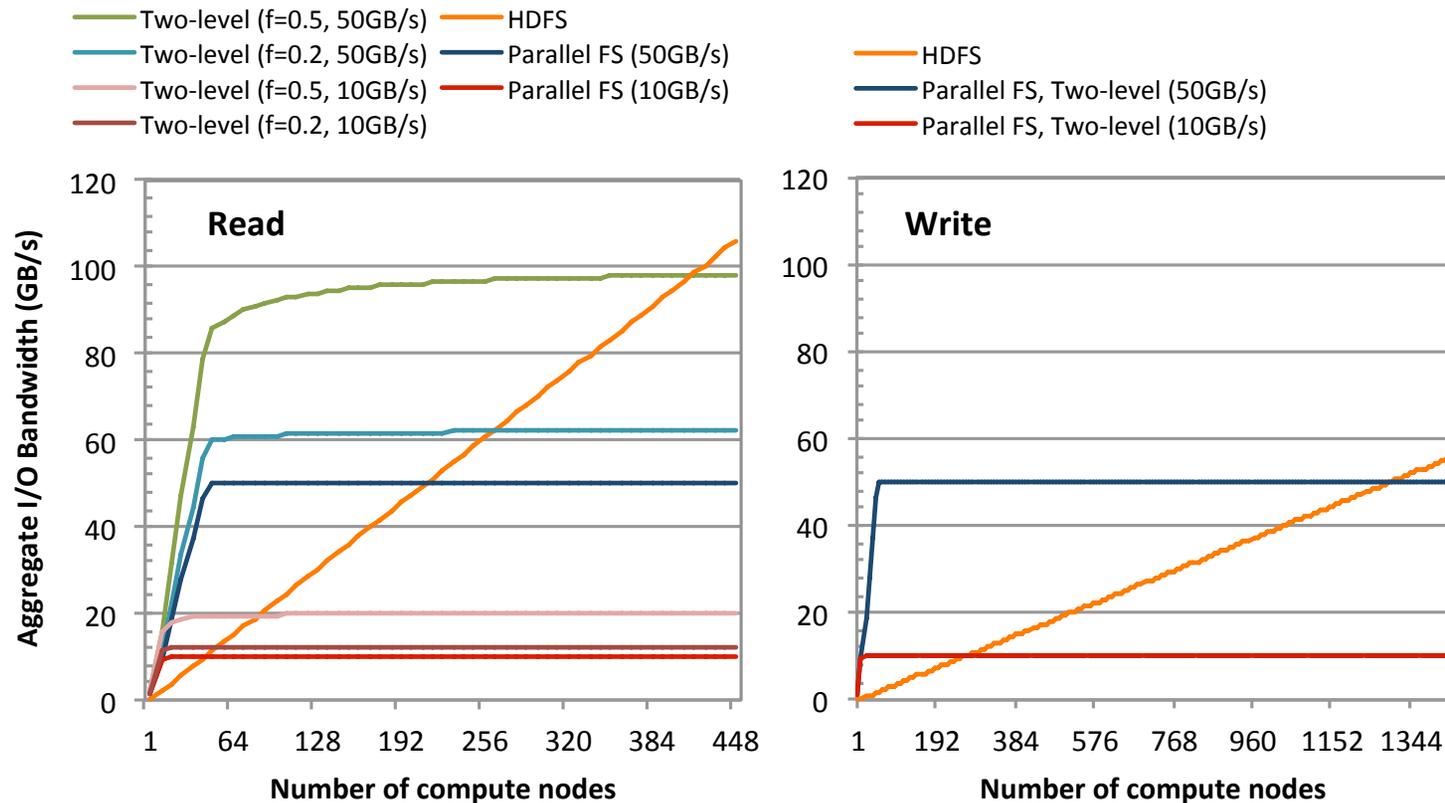
- ❁ If $f = 1$, all data is read from Tachyon only and if $f = 0$, all data is read from OrangeFS only. The higher the value of f , the higher read throughput the two-level system can provide.





Comparing Aggregate I/O Throughputs of Different Storages

- A case study:
 - network bandwidth is set to 1,170 MB/s per node.
 - local disk read throughput is 237 MB/s and the local disk write throughput is 116 MB/s.
 - local memory throughput is 6,267 MB/s.
 - two parallel file system aggregate throughputs: 10 GB/s and 50 GB/s.





Experiment Setup

- 🐾 All experiments are performed on Palmetto HPC cluster hosted at Clemson University.

CPU	Intel Xeon E5-2670 v2 20×2.50 GHz
HDD	1 TB 7200RPM SATA
RAID	12 TB LSI Logic MegaRAID SAS
RAM	128 GB DDR3-1600
Network	Intel 10 Gigabit Ethernet
Switch	Brocade MLXe-32 with 6.4 Tbps backplane

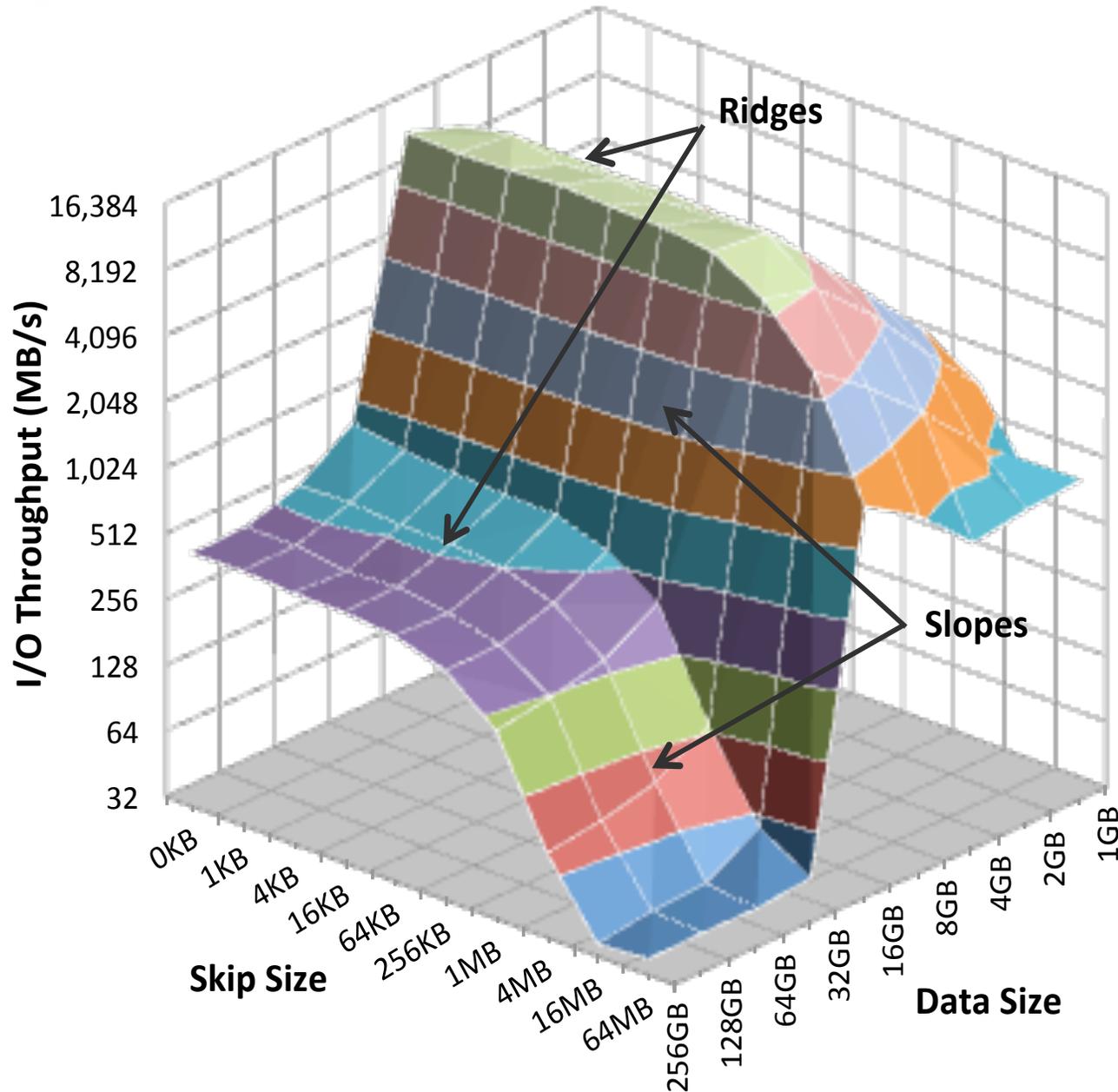


Experiment 1: Characterizing the I/O Performance of Two-level Storage

- 🐾 **Experiment 1: we use the Tachyon built-in performance evaluation program as the benchmark tool to measure the average read throughput received from two-level storage under a range of *data sizes* with different *skip sizes*.**
 - 🐾 We allocate 16 GB for Tachyon storage space



The storage mountain of two-level storage system



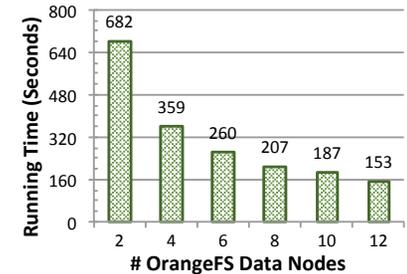
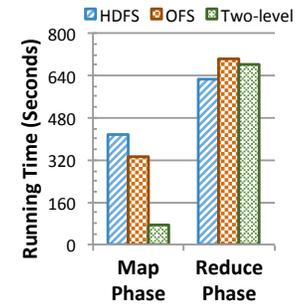
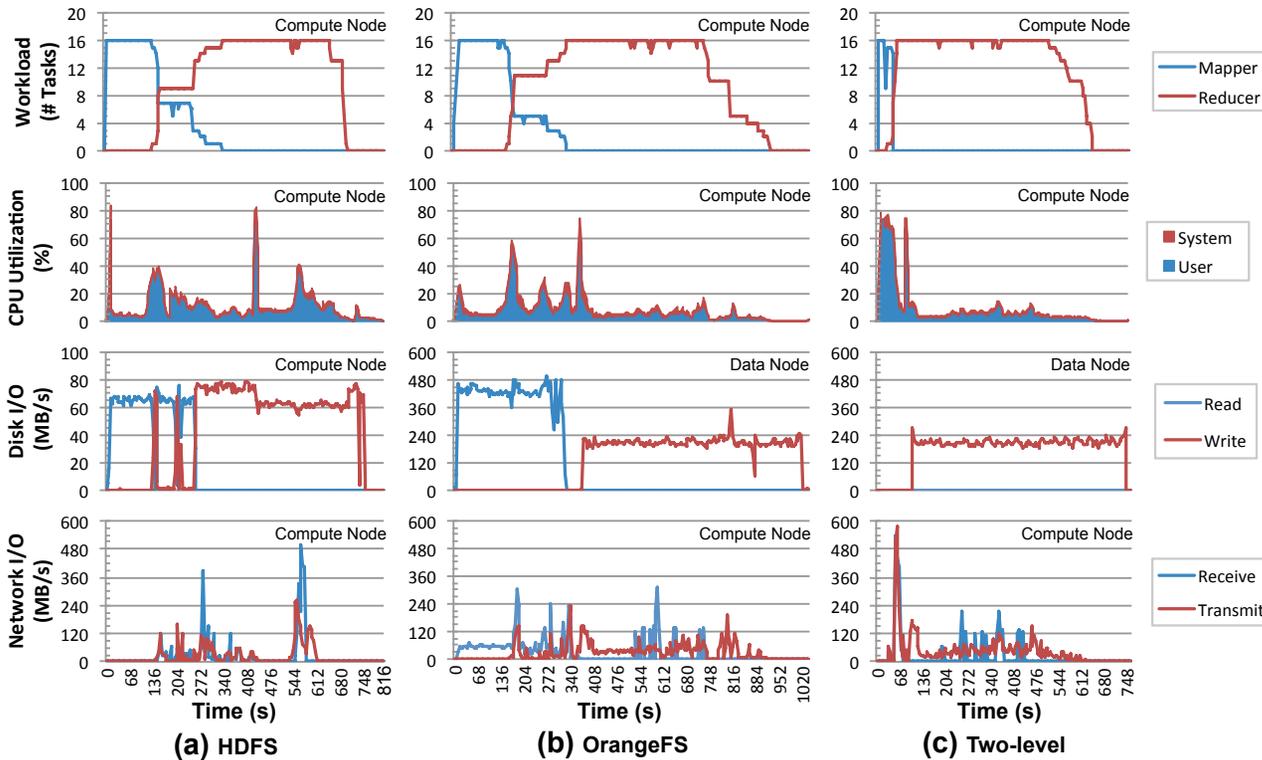


Experiment 2: Evaluate Performance using Terasort

- ❃ **We run the Terasort benchmark on a 17-node Hadoop cluster with 2-node OrangeFS as back-end storage system.**
 - ❃ One machine as the head node to host YARN's Resource Manager (RM) and Tachyon's Master service.
 - ❃ 16 compute nodes in Hadoop cluster. On each compute node, we assign 16 containers to occupy 16 CPU slots and leave the rest of 4 CPU slots to handle extra system overhead.
 - ❃ The capacity of Tachyon storage on each compute node is 32 GB.
- ❃ **We first run the TeraGen stage using a Map-only job to generate 256G data, then run the TeraSort stage using one Map/Reduce cycle. Mapper reads the data from storage and Reducer writes the sorted data back to storage.**
- ❃ **Before each test, we empty OS page caches to measure actual I/O costs.**

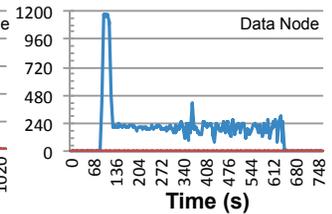
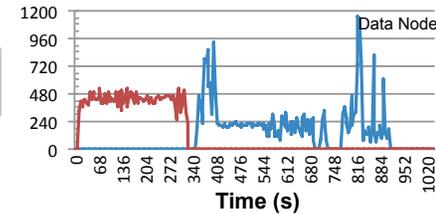


Experiment 2: Evaluate Performance using Terasort



(f)

(g)



(d)

(e)





CONCLUSSION AND DISCUSSION

- ❃ **Our theoretical modeling and experimental evaluation show that the two-level can increase read throughput, which will scale up with the number of compute nodes for Hadoop.**
- ❃ **In our two-level storage, local data always has a copy in OrangeFS; thus, OrangeFS provides fault-tolerance for Tachyon.**
- ❃ **Public HPC clusters are usually shared by a lot of users. Deploying Tachyon on HPC may lead to resource conflict.**





Team and ACKNOWLEDGMENTS

- 🐾 **Students: Pengfei Xuan, Jeffrey Denton**
- 🐾 **Faculties: Feng Luo, Rong Ge and Pradip Srimani**
- 🐾 **We thank the supporting from OrangeFS community led by Dr. Walter B. Ligon.**
- 🐾 **We are thankful to Haoyuan Li from Tachyon Nexus, and Zhao Zhang from UC Berkeley, AMPLab for providing helpful feedbacks.**
- 🐾 **HPC resources used in this research are supported by Omnibond Systems, LLC and the Clemson Computing and Information Technology (CCIT).**
- 🐾 **This work was partially supported by the NSF under Grant No. CCF-1551511.**





Questions?



Thomas Green Clemson (1807-1888)

