PERFORMING CLOUD COMPUTATION ON A PARALLEL FILE SYSTEM

Ellis Wilson (PSU), Garth Gibson (CMU)

OVERVIEW

- The line between HPC and Cloud compute is becoming blurry
  - Data-intensive science is increasingly looking to the MapReduce (MR) framework to expedite time-to-solution
  - MR comes coupled with the Hadoop Distributed File System (HDFS) to co-locate data and computation
- But, HPC often already has its own storage solutions in place
  - HPC Parallel File Systems (PFS) with RAID may be more cost effective than typical Cloud redundancy
  - Ability to grow compute and storage independently
  - Enterprise parts in specialized systems

Goal: Optimize MR computation using HPC PFS storage

ARCHITECTURE DETAILS

Traditional HDFS Pointed at PFS

- Configure HDFS with PFS paths rather than to local disks
  - Pros: Simplicity, HDFS replication as well as PFS reliability
  - Cons: Performance overhead, replication to same RAID set provides no added protection, replication limited to duplication, namespace outside of HDFS is useless

HDFS as a Wire Protocol in the PFS NAS Heads

- Run DataNodes (DNs) on NAS heads instead of all clients
  - Pros: Support HDFS as a protocol like others, no worries about replication to same RAID set
  - Cons: Bottleneck becomes NAS head, performance overheads, useless namespace outside of HDFS

No HDFS, MR Directly to PFS

- Run MR configured to send data directly to PFS
  - Pros: High-performance, namespace equivalence
  - Cons: Requires single path, no additional replication

MR through RainFS to PFS

- New Hadoop Filesystem designed specifically to intermediate between MR and PFS
  - Pros: High-performance, namespace equivalence, replication, multiple paths to distinct NAS systems usable
  - Cons: Non-standard Hadoop filesystem

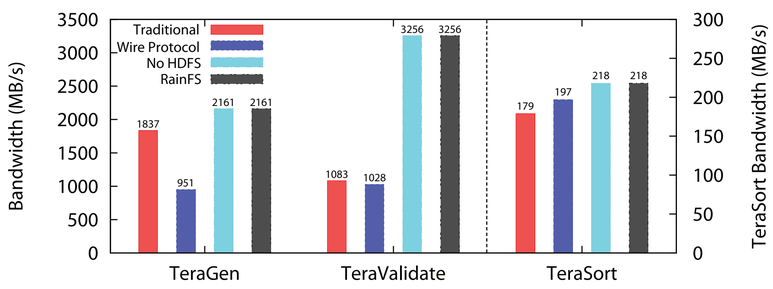
RELIABILITY POTENTIAL PER ARCHITECTURE

Disk-failure tolerance potentials differ between Architectures:

	RAID 5 Repl. 1	RAID 6 Repl. 1	RAID 5 Repl. 2	RAID 6 Repl. 2	RAID 5 Repl. 3	RAID 6 Repl. 3
Traditional	✔	✔	✔	✔	✘	✘
Wire Protocol	✔	✔	✔	✔	✔	✔
No HDFS	✔	✔	✘	✘	✘	✘
RainFS	✔	✔	✔	✔	✔	✔
Disk Failure Tolerance	1	2	3	5	5	8

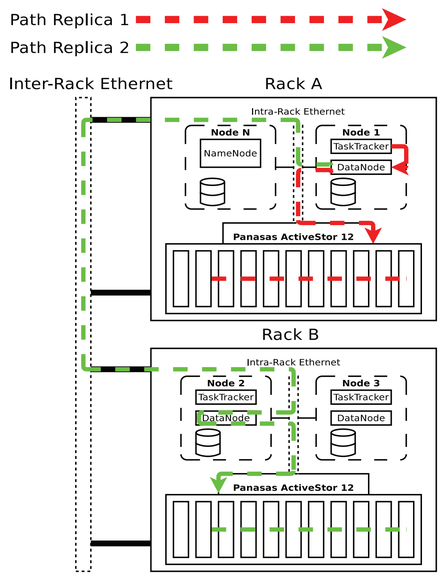
ARCHITECTURAL PERF. IMPLICATIONS

- Differences in architecture I/O paths results in wide performance variations, even without replication
- Graph compares user-perceived throughput for TeraGen (write-only), TeraValidate (read-only) and TeraSort (mixed)
  - Needless data pass-thru degrades performance significantly



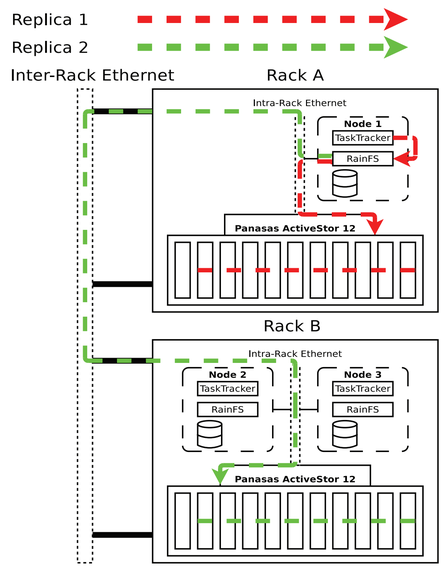
EXEMPLARY WRITE PATHS

Traditional HDFS at PFS



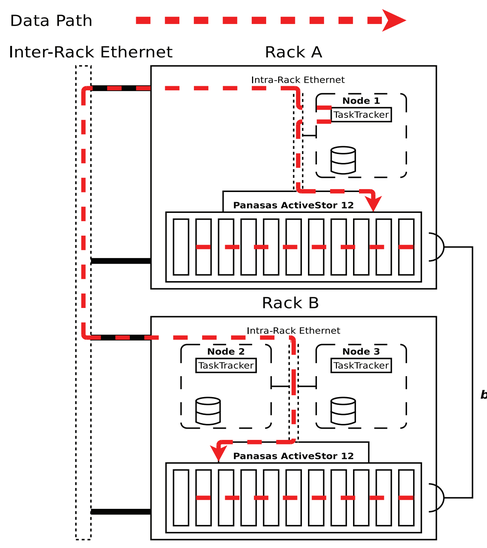
- Data forced through DNs
- Replica 2 incurs pass-thru
- One extra transit incurred

HDFS as a Wire Protocol



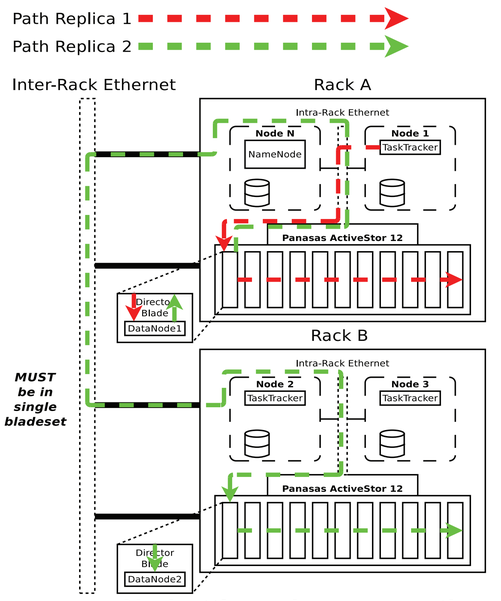
- Data forced through NAS head to storage blades
- Replica 2 incurs pass-thru

No HDFS, MR Direct to PFS



- No additional replication
- No bottlenecks or pass-thrus
- All NAS must be in single set

MR Through RainFS to PFS



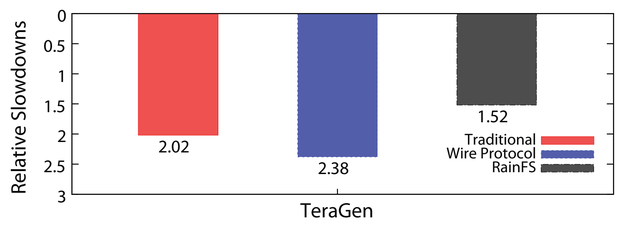
- No bottlenecks or pass-thrus
- NAS can be in multiple sets
- Replication again possible

EXPERIMENTAL FRAMEWORK

- 51 Nodes (VMs): 1 Master, 50 Slaves
  - 2 Cores, 4GB RAM, 1 SATA HDD, 1GbE NICs each
- 5 Panasas ActiveStor 12 Shelves
  - 20GbE per shelf, 40TB per shelf (20 disks)
- Benchmark Used: The Apache TeraSort Suite on 500GB

IMPACT OF REPLICATIONS

- Impact of increasing replication from none to duplication for TeraGen (write-only, bandwidth-limited benchmark)
  - Cannot compare "No HDFS," as no replication is possible
  - Cannot consider triplication, as not possible in "Traditional" (HDFS puts replica 1 and 2 on same "rack" or RAID set for triplication – this adds no additional reliability)



Intel Science & Technology Center for Cloud Computing

Carnegie Mellon University

Georgia Tech

intel

PRINCETON UNIVERSITY

UC Berkeley