Accelerating Big Data Processing in the Cloud with Scalable Communication and I/O Schemes

Shashank Gugnani, Dhabaleswar K. Panda (Advisor), The Ohio State University

Overview

Current Trends in Big Data
- Huge increase in cloud deployments running Big Data analytics
- Analytics performed on data stored in cloud storage
- System and job sizes constantly increasing
- High-performance solutions for Big Data in the cloud essential

Importance of Big Data in Cloud
- Inherent flexibility and scalability
- Tremendous cost saving
- Built-in reliability and fault-tolerance

Communication Bottlenecks
- Not aware of topology and locality
- Slow TCP-based

Scalability Issues
- Cloud storage solutions have limited scalability
- Gateway or proxy servers limit throughput

Consistency Issues
- Cloud storage systems typically provide Eventual Consistency (EC)
- EC is not sufficient for traditional applications expecting POSIX-like consistency

QoS Issues
- Cloud storage systems lack application-oblivious service guarantees
- Emerging hardware-based QoS not exploited

Proposed Designs
Topology-aware communication[1]
- MapReduce-based automatic topology detection
- Locality and topology-aware communication and scheduling

High-performance Cloud Storage[2]
- RDMA-based communication
- Re-designed scalable architecture with client-based replication

POSIX-like consistent Cloud Storage
- Proposed use of atomic operations to provide consistency
- Implemented 2PC for write operations

QoS-aware Storage Runtime[3]
- Application-oblivious service guarantees
- Hardware-based NVMe request arbitration

Contributions
- Scalable automatic topology detection
- Efficient topology and locality-aware communication
- High-performance and consistent cloud storage
- QoS-aware storage runtime
- Ability to run version control, database, and big data applications directly on cloud storage

Publications
[1] Designing Virtualization-aware and Automatic Topology Detection Schemes for Accelerating Hadoop on SR-IOV-enabled Clouds. (S. Gugnani, X. Lu, and D.K. Panda, CloudCom'16)
[2] Swift-X: Accelerating OpenStack Swift with RDMA for Building an Efficient HPC Cloud. (S. Gugnani, X. Lu, D.K. Panda, CCGrid'17)
[3] Analyzing, Modeling, and Provisioning QoS for NVMe SSDs. (S. Gugnani, X. Lu, D.K. Panda, UCC'18)

More Information
- http://hibd.cse.ohio-state.edu/
- This research is supported in part by NSF grants #CNS-1513120, #IIS-1636846, and #CCF-1822987

Challenges

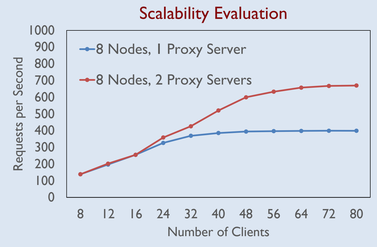
Inefficient Communication
- For large-sized clusters, topology-aware communication is paramount
- Existing topology-aware designs in Hadoop are not optimized for cloud environments
- No service that can automatically detect cluster topology and expose it to Hadoop

Process Location		Number of Hops	Latency (us)
Intra-Rack	Inter-Chassis	0 Hops in Leaf Switch	1.57
	Intra-Chassis	1 Hop in Leaf Switch	2.04
Inter-Rack	-	3 Hops in Leaf Switch	2.45
		5 Hops in Leaf Switch	2.85

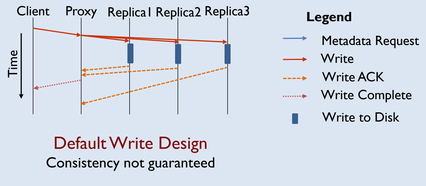
Reference: https://confluence.pegasus.isi.edu/download/attachments/5242944/topology-aware-poster.pdf

Communication Data from TACC Ranger System

Limited Scalability in Cloud Storage
- Proxy server design in Swift limits throughput since all operations are routed through the proxy server
- Server-side replication limits scalability
- Network communication is slow TCP-based

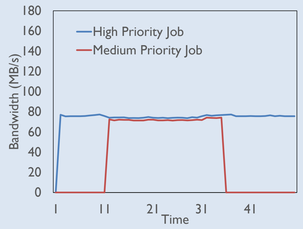


Consistency Issues
- Traditional applications reliant on POSIX-like consistency
- Cloud storage solutions provide Eventual Consistency
- Application migration to the cloud is not straightforward
- Consistency guarantees are required



Default Write Design
Consistency not guaranteed

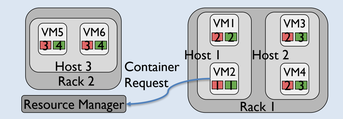
Lack of Service Guarantees
- Existing mechanisms not application-oblivious
- Workloads with varying request sizes not handled well
- Hardware-based arbitration not exploited



Default SPDK Design
Lack of Service Guarantees

Proposed Designs

Topology-aware Communication
- Automatic topology detection module can detect topology changes during runtime
- Maximize communication between co-located VMs
- Allocate Containers and Map tasks on a co-located VM before other VMs

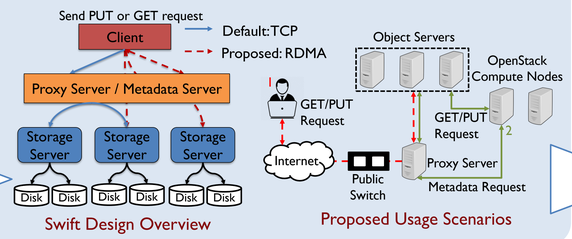


Default Hadoop Policy
1. Node local
2. Rack local
3. Off-rack

Proposed Policy
1. Node local
2. Host local
3. Rack local
4. Off-rack

Proposed Container Allocation Policy

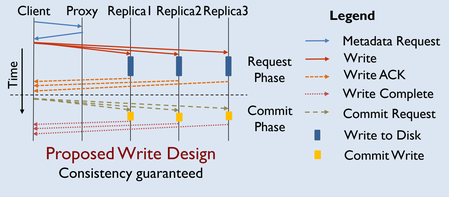
Scalable Cloud Storage: Swift-X
- Use proxy server only as a metadata server
- Client-based replication for scalability
- RDMA-based communication for high-performance
- Non-blocking semantics for efficient overlap between communication and I/O



Swift Design Overview

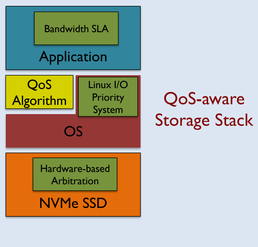
Proposed Usage Scenarios

POSIX-like consistent Cloud Storage
- Atomicity as a way to guarantee consistency
- Two-phase commit for atomic write operations
- Client-side caching to improve read/write performance
- Compatibility with HDFS API



Proposed Write Design
Consistency guaranteed

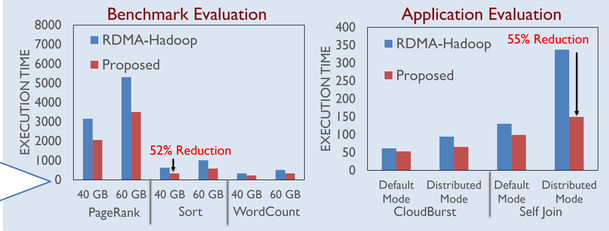
QoS-aware Storage Runtime
- Linux I/O priority system to transfer priority information to underlying runtime
- Hardware-based NVMe request arbitration
- Mechanisms to provide I/O bandwidth SLAs
- Request-size agnostic QoS algorithm



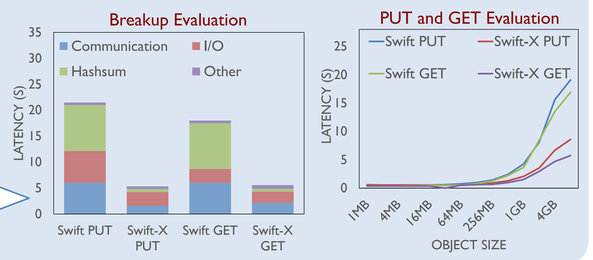
QoS-aware Storage Stack

Results

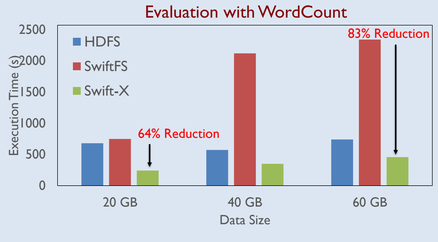
Evaluation with RDMA-Hadoop
- Up to 52% improvement over RDMA-Hadoop for benchmarks
- Up to 55% improvement over RDMA-Hadoop for applications
- Proposed design delivers the best performance and fault-tolerance



Evaluation with OpenStack Swift
- Up to 47% and 66% reduction in PUT and GET latencies
- Communication time reduced by up to 3.8x for PUT and up to 2.8x for GET
- Up to 7.3x improvement in read throughput



Evaluation with SwiftFS and HDFS
- Up to 83% improvement over SwiftFS
- Up to 64% improvement over HDFS
- With HDFS, data is copied from Swift
- Best performance and guaranteed consistency



Evaluation with SPDK
- Comparison using SPDK with Weighted Round Robbin NVMe arbitration
- Near desired job bandwidth ratios
- Stable and consistent bandwidth over time