vPFS+: Managing I/O Performance for Diverse HPC Applications

Ming Zhao, Arizona State University

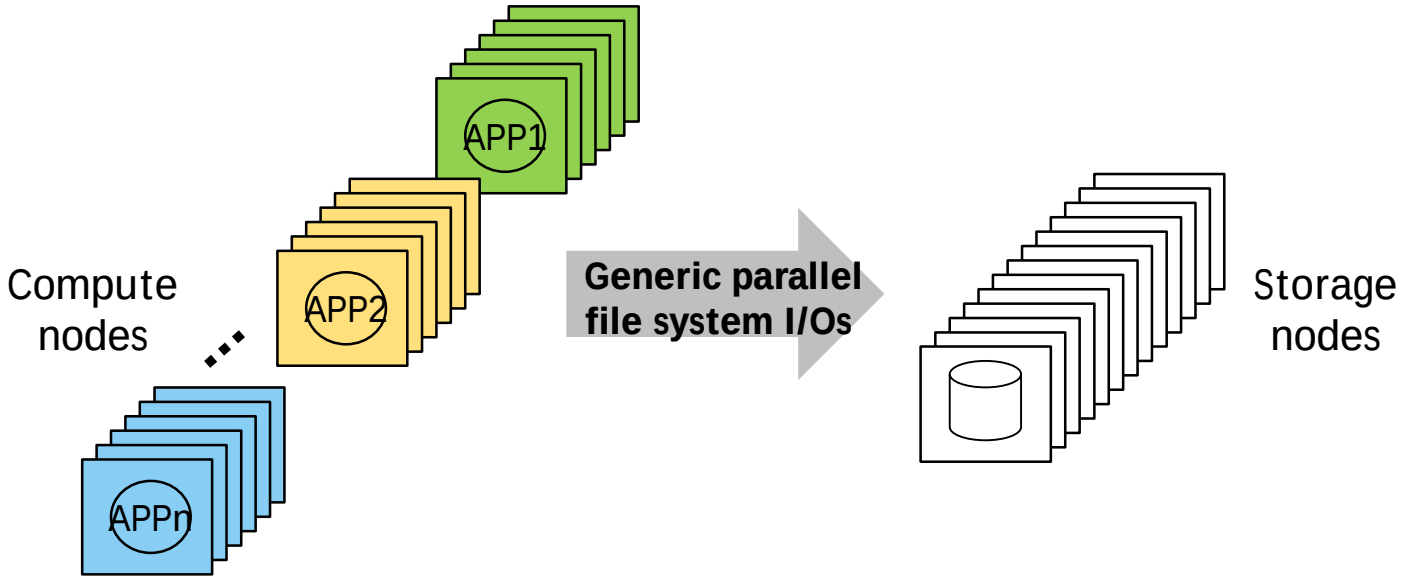
Yiqi Xu, VMware

http://visa.lab.asu.edu

Background: HPC I/O Management

- Increasing diverse HPC applications on shared storage
  - Different I/O rates, sizes, and data/metadata intensities
- Lack of I/O QoS differentiation

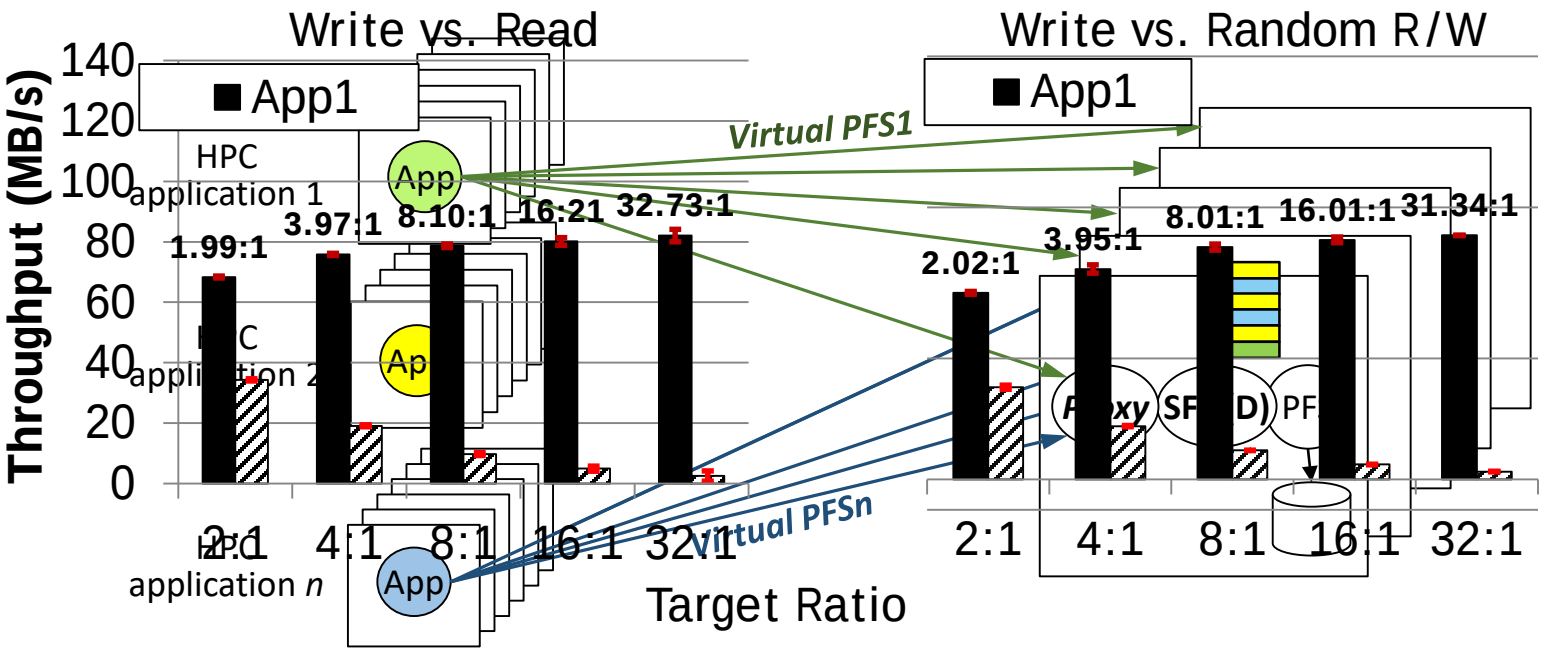
Mismatch!



Background: vPFS

- Proxy-based interposition of application data requests
  - Transparent to applications, support different setups
- Proportional I/O bandwidth scheduling using SFQ(D)
  - Work conserving, strong fairness

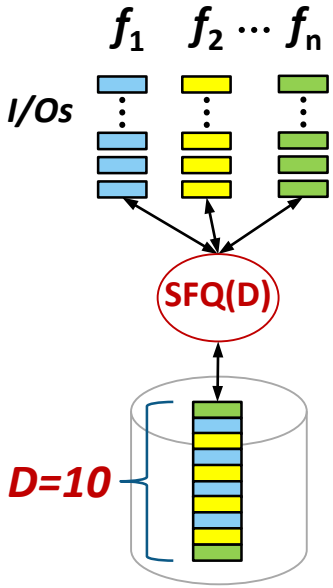
Limitations?



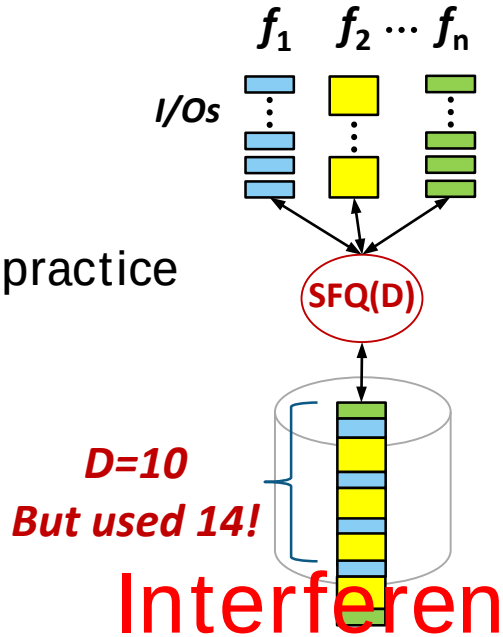
Limitations

- Lack of isolation between large and small workloads
- SFQ (D): start-time fair queueing with I/O depth ***D***
  - Start times capture each flow's service usage
    - Dispatch requests in the increasing order of their start times
  - ***D*** captures the available I/O parallelism
    - Allow up to *D* of outstanding requests

In theory



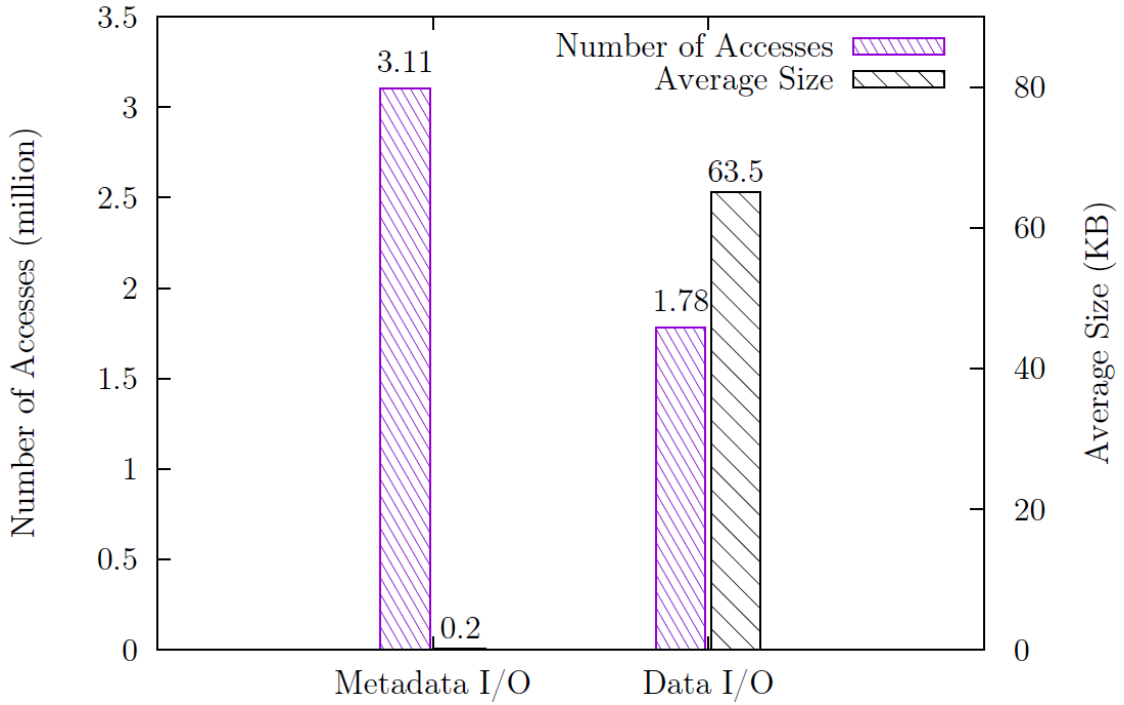
In practice



Interference!

Limitations

- Lack of Metadata I/O scheduling
- Many HPC applications are metadata intensive
  - Metadata I/O performance is important

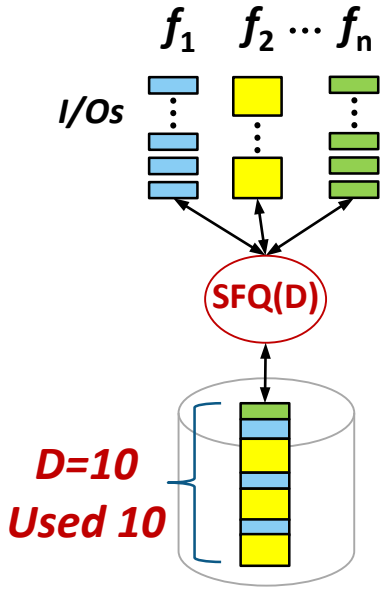


Solution: vPFS+

- SFQ(D)+
  - A new scheduler to support diverse I/O sizes
- Metadata I/O management
  - An extension to support distributed scheduling of metadata requests
- PVFS2-based real prototype
- Comprehensive experimental evaluation

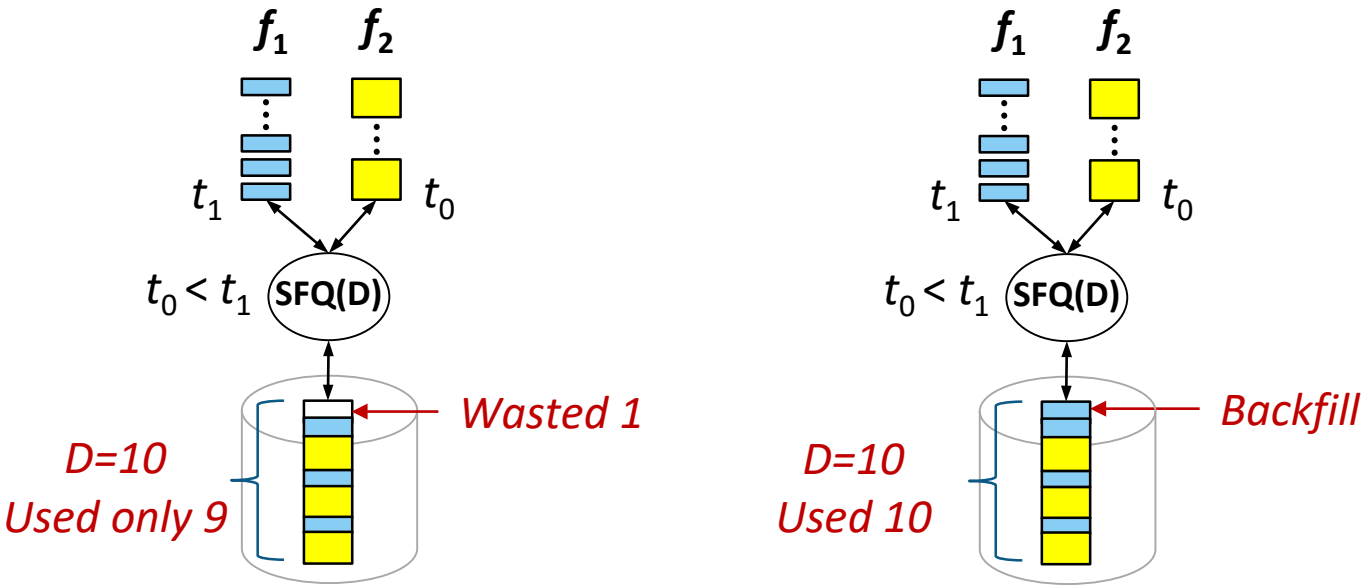
SFQ(D)+: Variable Cost I/O Depth Allocation

- Allocate the limited I/O depth ***D*** to outstanding requests based on their sizes
  - Consider ***D*** as the number of available I/O slots
    - Each slot represents the cost of the smallest I/Os
  - Each outstanding request occupies one or multiple slots based on its size
    - Stop dispatching when ***D*** is used up
- Effectively protect small I/O workloads
  - Low-rate I/Os wait less for large outstanding I/Os to complete
  - Small I/Os are less affected by large I/Os after dispatched



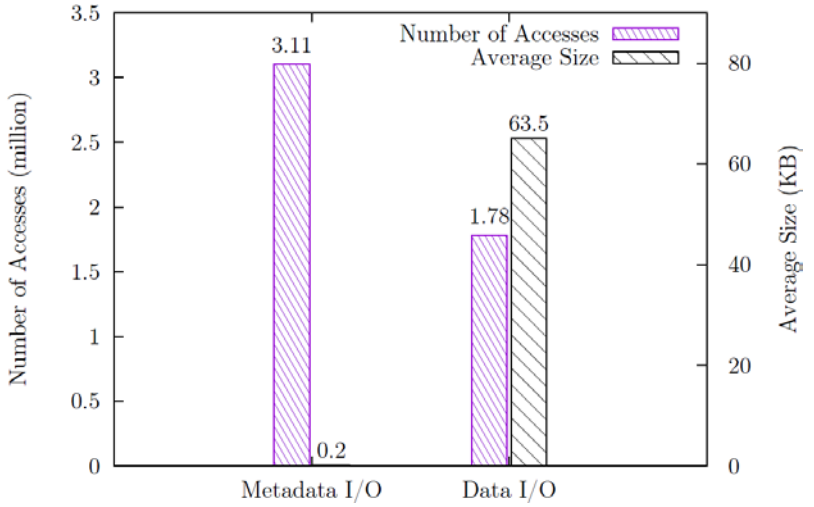
SFQ(D)+: I/O Backfilling

- Large I/Os at the head of queue have to wait till there are enough slots
  - Waste the currently available slots
- Backfill promotes small I/Os to utilize the available slots
  - Similar to the backfill of small jobs in batch scheduling



Metadata I/O Scheduling

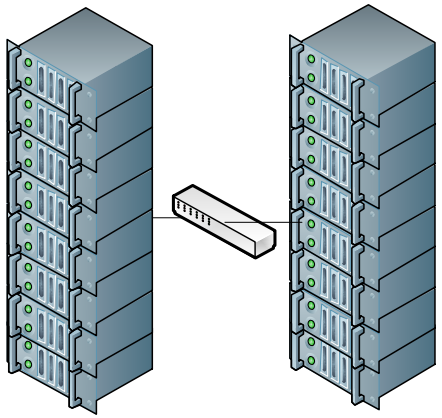
- Extends the scheduling to both data and metadata requests
  - Apply SFQ(D)+ to schedule metadata I/Os on each server
  - Treat metadata I/Os as small I/Os



- Achieve total-metadata-service fair sharing for distributed metadata servers
  - Coordinate scheduling across distributed metadata servers
  - Each scheduler adjusts its scheduling of local metadata requests based on global metadata service distribution

Evaluation

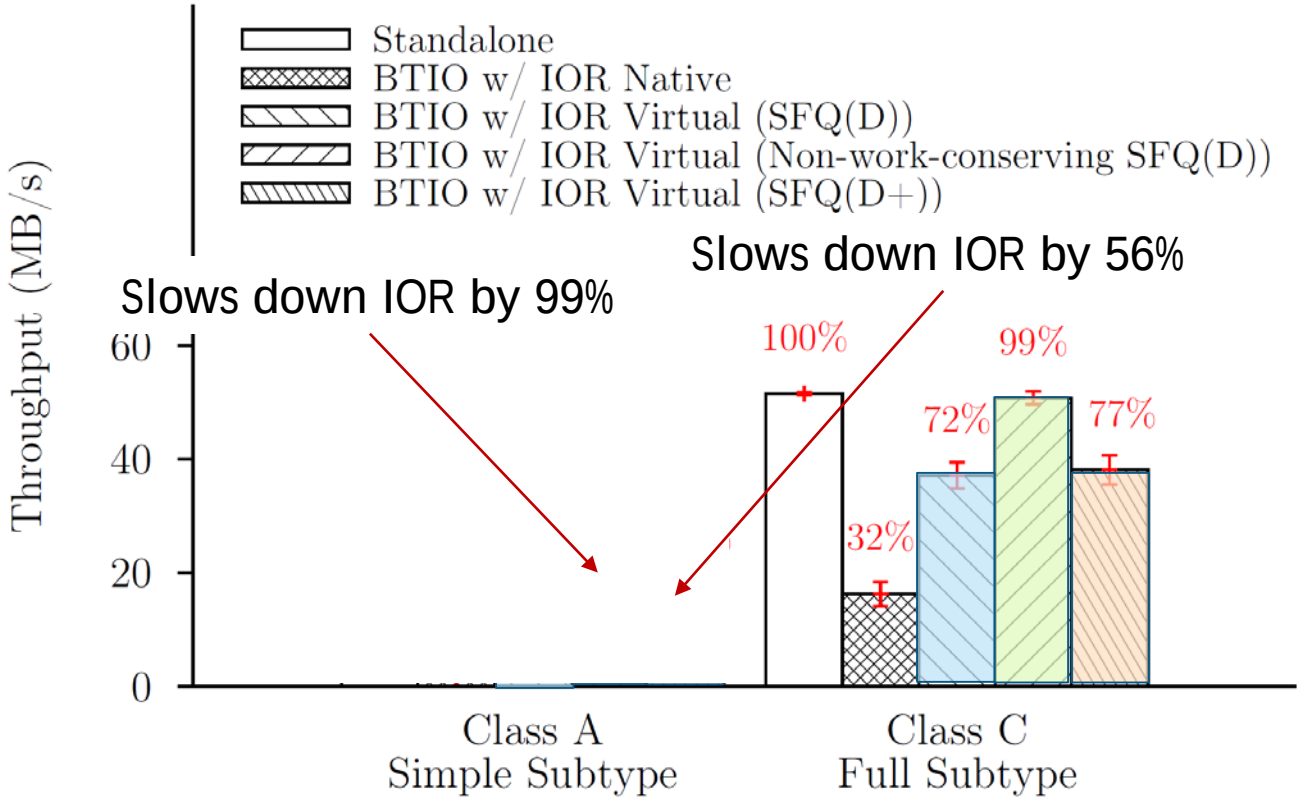
- Testbed
  - vPFS+ implemented for PVFS2
  - 8 Clients & 8 Servers, 1 gigabit switch



- Workloads
  - IOR: intensive checkpointing I/Os
  - multi-md-test: intensive metadata I/Os
  - BTIO: scientific application benchmark
  - WRF: real-world scientific application

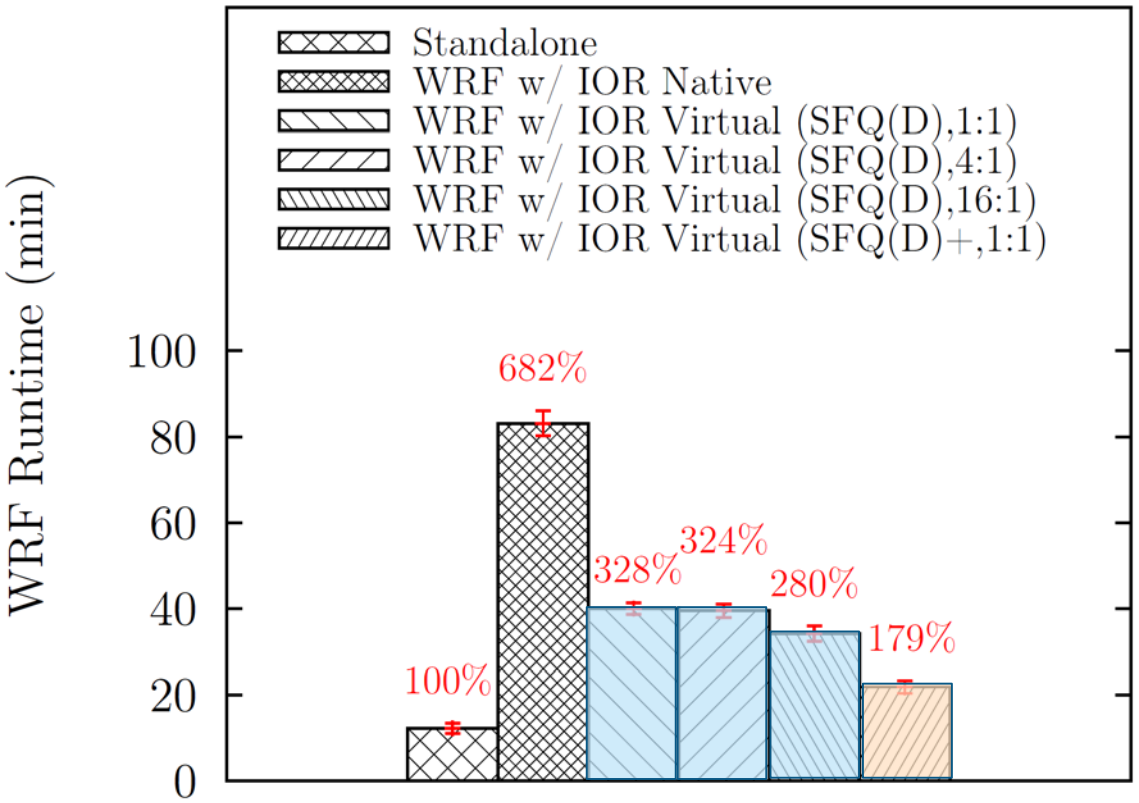
BTIO vs. IOR

- BTIO—Class C (4MB-16MB I/Os), Class A (320B I/Os)
- vPFS+ substantially reduces BTIO slowdown



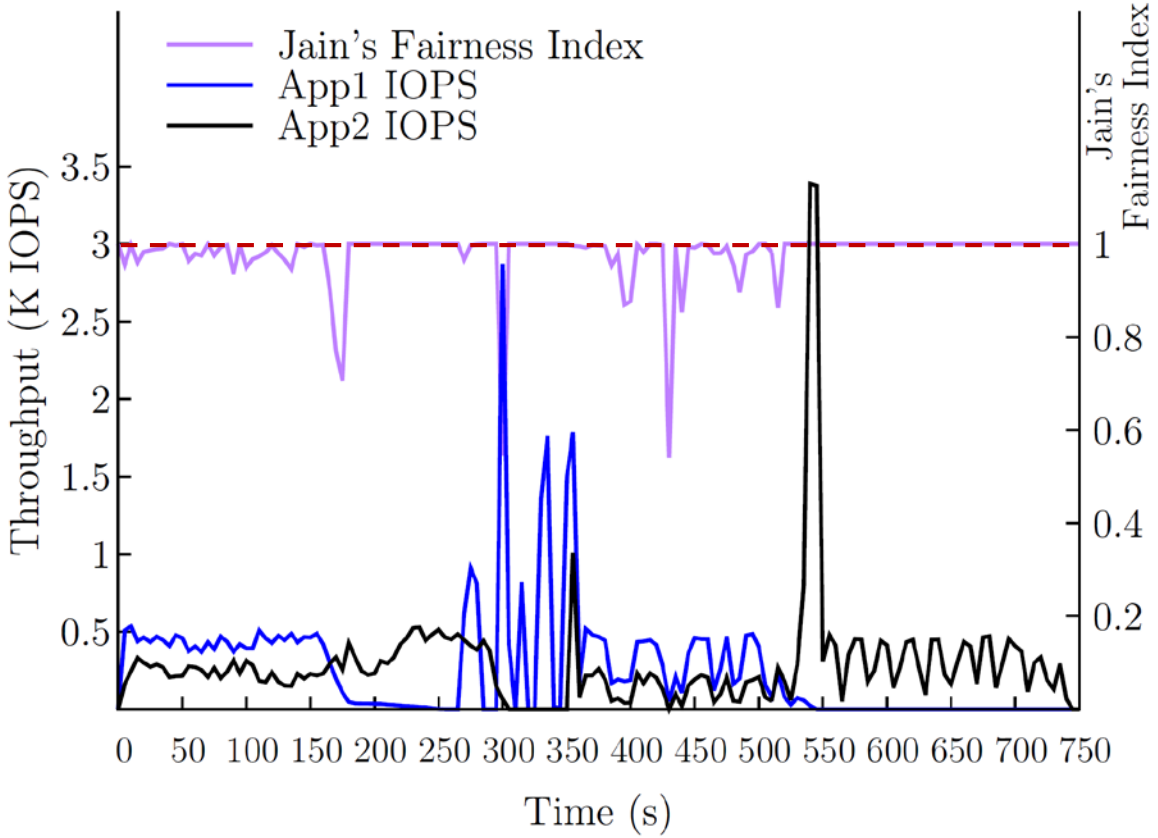
WRF vs. IOR

- WRF—a large number of small I/Os and intensive metadata requests
- vPFS+ achieves 80% and 281% better performance for WRF than Native and vPFS, respectively



Metadata I/O Scheduling

- multi-md-test—mktestdir, create, write, readdir, read, close, rm, rmtestdir
- vPFS+ achieves nearly perfect fairness despite dynamic metadata demands for two metadata-intensive apps



Conclusions

- I/O diversity is becoming a top concern
  - Different types of requests (POSIX vs. MPI-IO, data vs. metadata)
  - Different I/O rates and sizes
- vPFS+ manages I/O performance for diverse apps
  - SFQ(D)+ recognizes the variable cost of different I/Os and takes it under control
  - Distributed metadata scheduling supports metadata-intensive applications

Future Work

- Implement SFQ(D)+ directly into data/metadata servers
  - Proxy-based scheduling may incur extra latency
  - But its impact to throughput is small (< 1%)
- Evaluate vPFS+ in larger and more diverse environments
  - Performance isolation is even more important on larger systems with more diverse workloads
  - Faster storage does not eliminate performance isolation → the gap between processor and I/O performance is still increasing

Acknowledgement

- National Science Foundation
  - CNS-1629888, CNS-1619653, CNS-1562837, CNS-1629888, CMMI-1610282, IIS-1633381

- VISA Lab @ ASU

- *Thank you!*