BLUE WATERS

SUSTAINED PETASCALE COMPUTING

May 9, 2013

Analysis of the Blue Waters File System Architecture for Application I/O Performance

Robert Sisneros

Kalyana Chadalavada

GREAT LAKES CONSORTIUM FOR PETASCALE COMPUTATION

This Talk

- Goal: Investigate I/O tuning specific to Blue Waters
- Blue Waters file systems
  - Disk subsystem
  - Network connectivity
- Design implications
- Hypothesis testing
- Application of results to application
- Conclusion

Analysis of the Blue Waters File System Architecture for Application I/O Performance

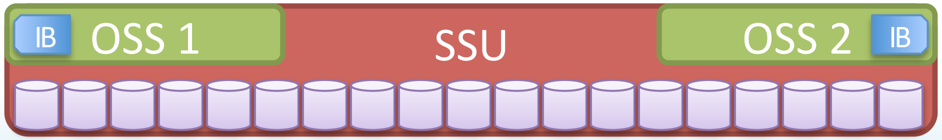
BLUE WATERS FILE SYSTEMS OVERVIEW

Blue Waters File Systems

- Three distinct file systems
  - home, project, scratch
  - Three distinct metadata servers
  - Jobs and users don't interfere with each other
- home, project – 98 GB/s, 2 PB each
- scratch – 980 GB/s, >21 PB usable

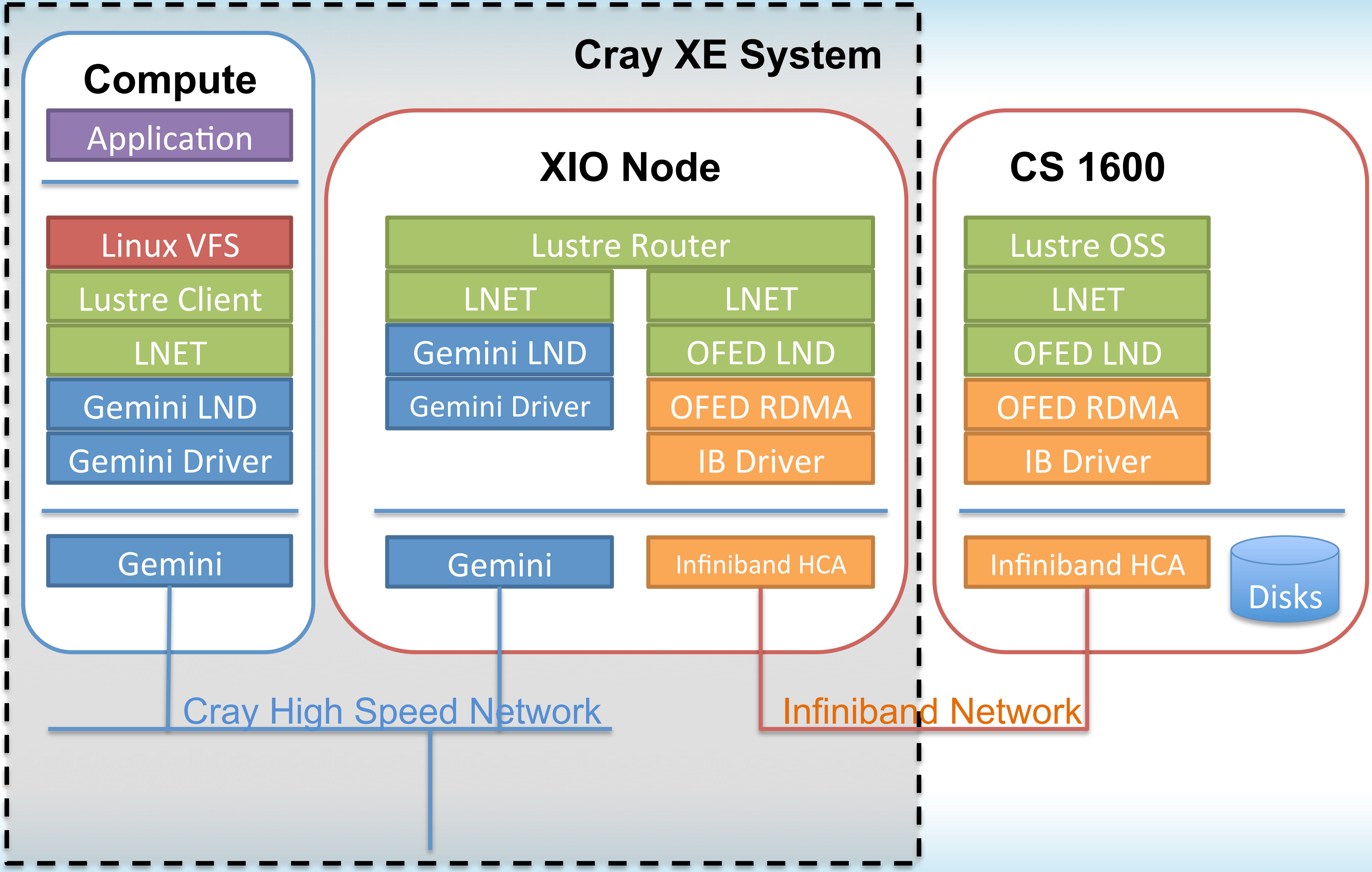
Blue Waters File System Disk Subsystem

- Cray Sonexion-1600 Lustre appliance
- Infiniband networking
- Two OSS units, one SSU per CS-1600
  - 84 disks per SSU (Scalable Storage Unit)
  - OSTs: 8x (8+2) RAID 6 volumes
  - Two disk RAID 1 volume for OSS failover

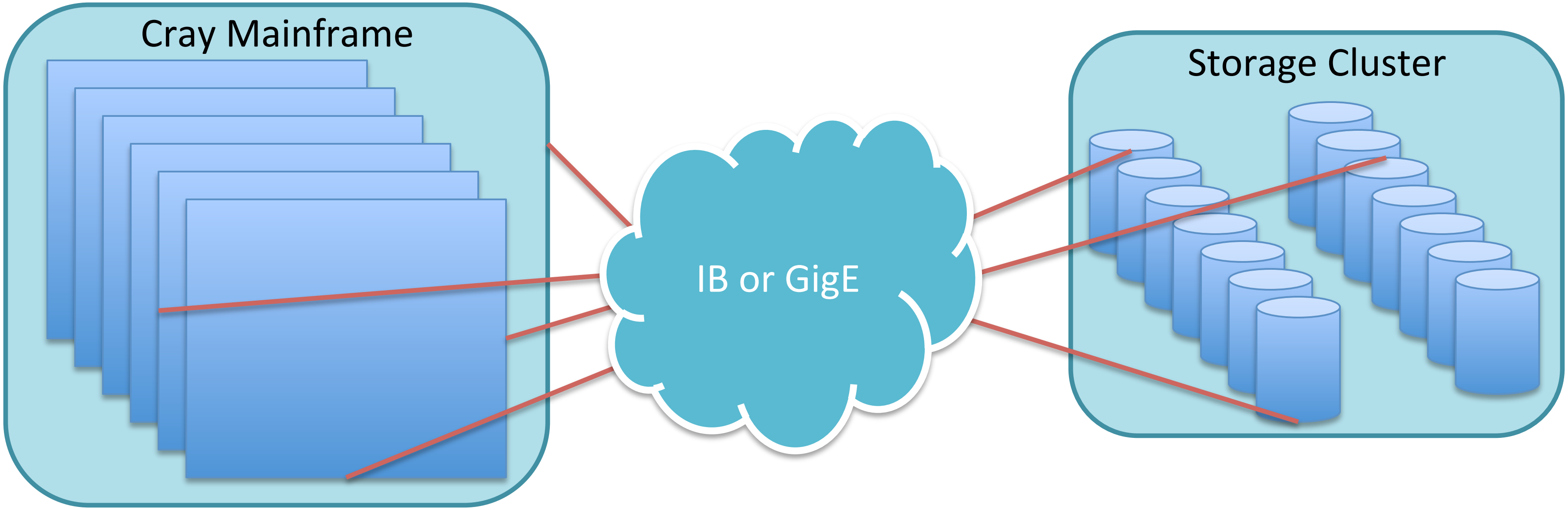


Blue Waters File System Connectivity

- LNET – Lustre Networking subsystem
  - Routers route LNET packets between Gemini & IB
  - Cray system uses XIO nodes as LNET routers
  - Each LNET router provides 2GB/s
- 576 total LNET routers
  - scratch
    - 480 for OSS units, 2 for MDS units
  - home, project
    - 48 for OSS units, 2 for MDS units each


Cray XE System
Compute
Application
Linux VFS
Lustre Client
LNET
Gemini LND
Gemini Driver
Gemini
XIO Node
Lustre Router
LNET
LNET
Gemini LND
OFED LND
Gemini Driver
OFED RDMA
IB Driver
Gemini
Infiniband HCA
CS 1600
Lustre OSS
LNET
OFED LND
OFED RDMA
IB Driver
Infiniband HCA
Disks
Cray High Speed Network
Infiniband Network

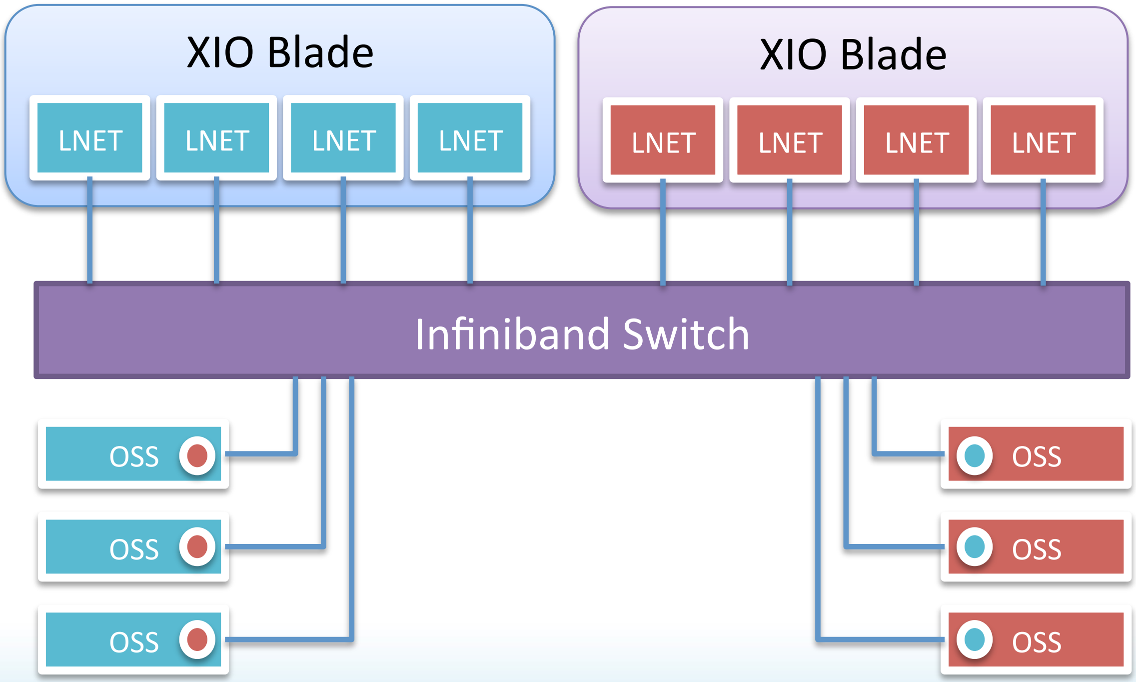
Typical Connectivity



- Lustre traffic exits Gemini HSN at the nearest XIO node
- Further switching happens on the dedicated storage fabric
- A compute node is at a uniform distance from any disk

Blue Waters File System Connectivity

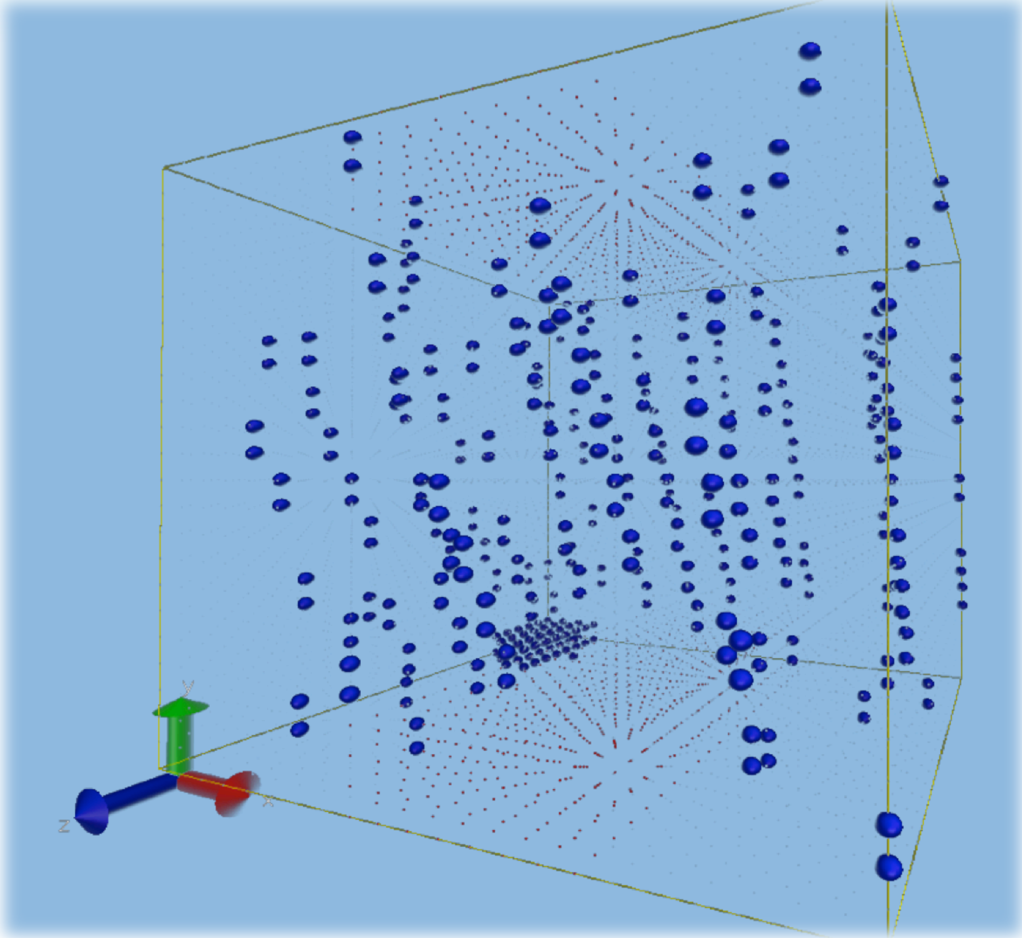
- Secondary LNET groups for availability
- Peer takes over disks in case of OSS, network failures
- 4:3 LNET: OSS



Distribution of LNET Routers on Blue Waters

- Compute node to OST traffic
  - Routed on the Gemini HSN to the primary LNET router group
  - Does not exit Gemini at the nearest XIO node
- LNET routers not 100% uniformly distributed across the HSN
- Disks are at varying distances from a compute node
- Layout changes possible, studies in progress

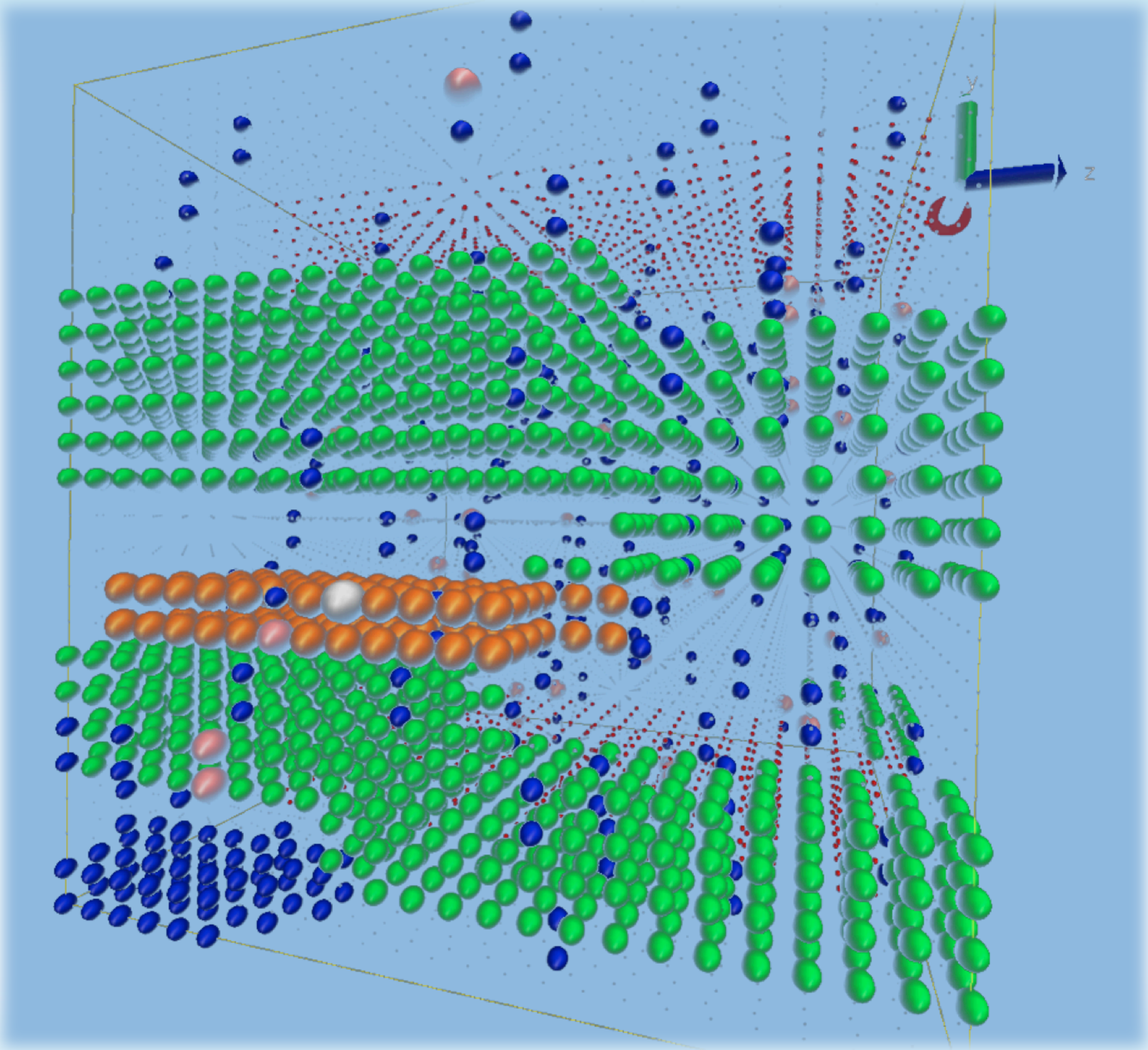
Distribution of LNET routers on Blue Waters



Implications of the File System Design

- Non-uniform disk access
  - Between runs, the distance between a process and a file varies
  - Does this impact overall I/O bandwidth?
- Concurrent MPI & I/O traffic on the Gemini HSN
  - Could contention between Lustre and MPI get ugly?
  - Ex: Large collectives or a large I/O bound job
- *Yes* to either leads to possible
  - Pronounced dependency on machine state
  - Inconsistent runtimes

A Real
(and possibly unfortunate)
Layout



Lustre Tuning for Parallel I/O on Blue Waters

- HPC I/O best practices
  - Reduce OST contention
  - Align I/O operations
  - Stripe count + stripe size
- What about offset (lfs -o)?
  - The typical (always route through nearest LNET): won't help
  - What about on Blue Waters?

Analysis of the Blue Waters File System Architecture for Application I/O Performance

EXPERIMENTS & RESULTS

Experiments*: What We Used

IOR

- Customizable I/O Benchmark
- Common for HPC
- Developed at LLNL

Enzo

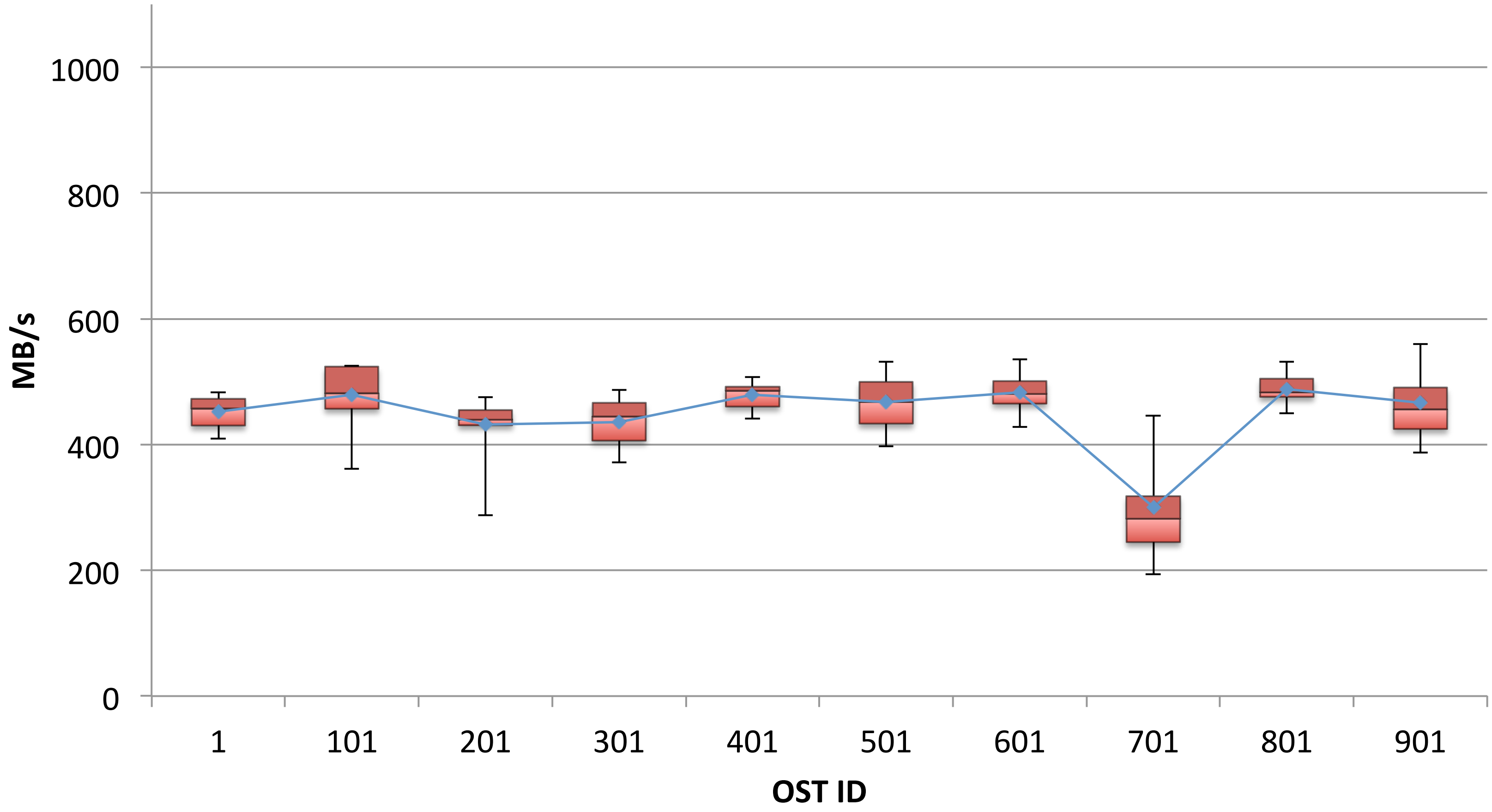
- Galaxy formation simulation
- Grid-based + AMR
- HDF5
- In use on Blue Waters

* Experiments not designed to extract the best possible throughput, but focus on assessing the impact of OST distance in a typical use case

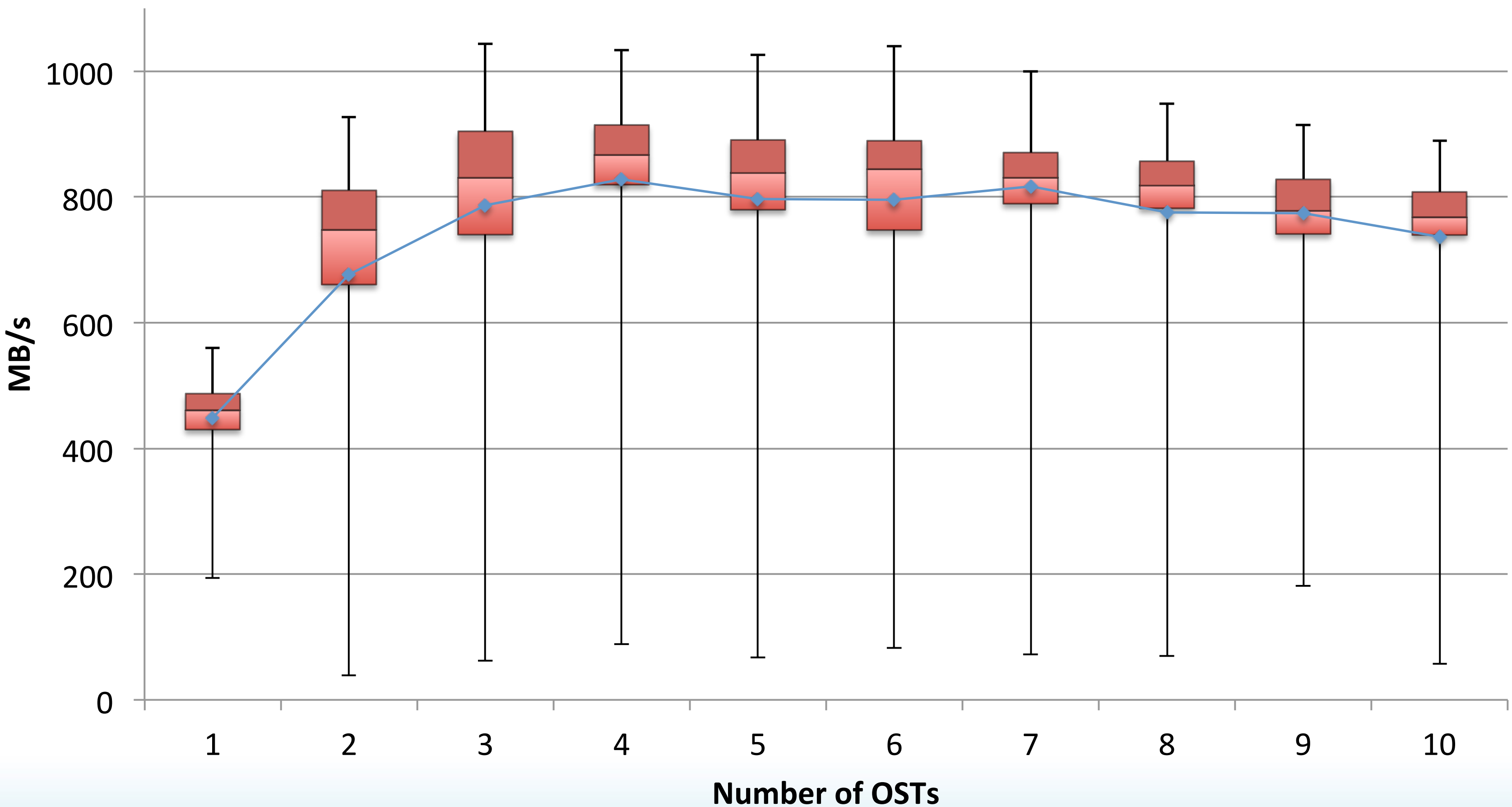
Experiment 1: Does OST Selection Really Matter?

- Mimic default system OST selection
  - Utilization based, but random to user
  - Arbitrarily select 10
- Parallel 1GB reads/writes
  - I/O is unaligned
  - 64 cores (4 nodes)
- Vary striping from 1 to 10
- 10 iterations per test

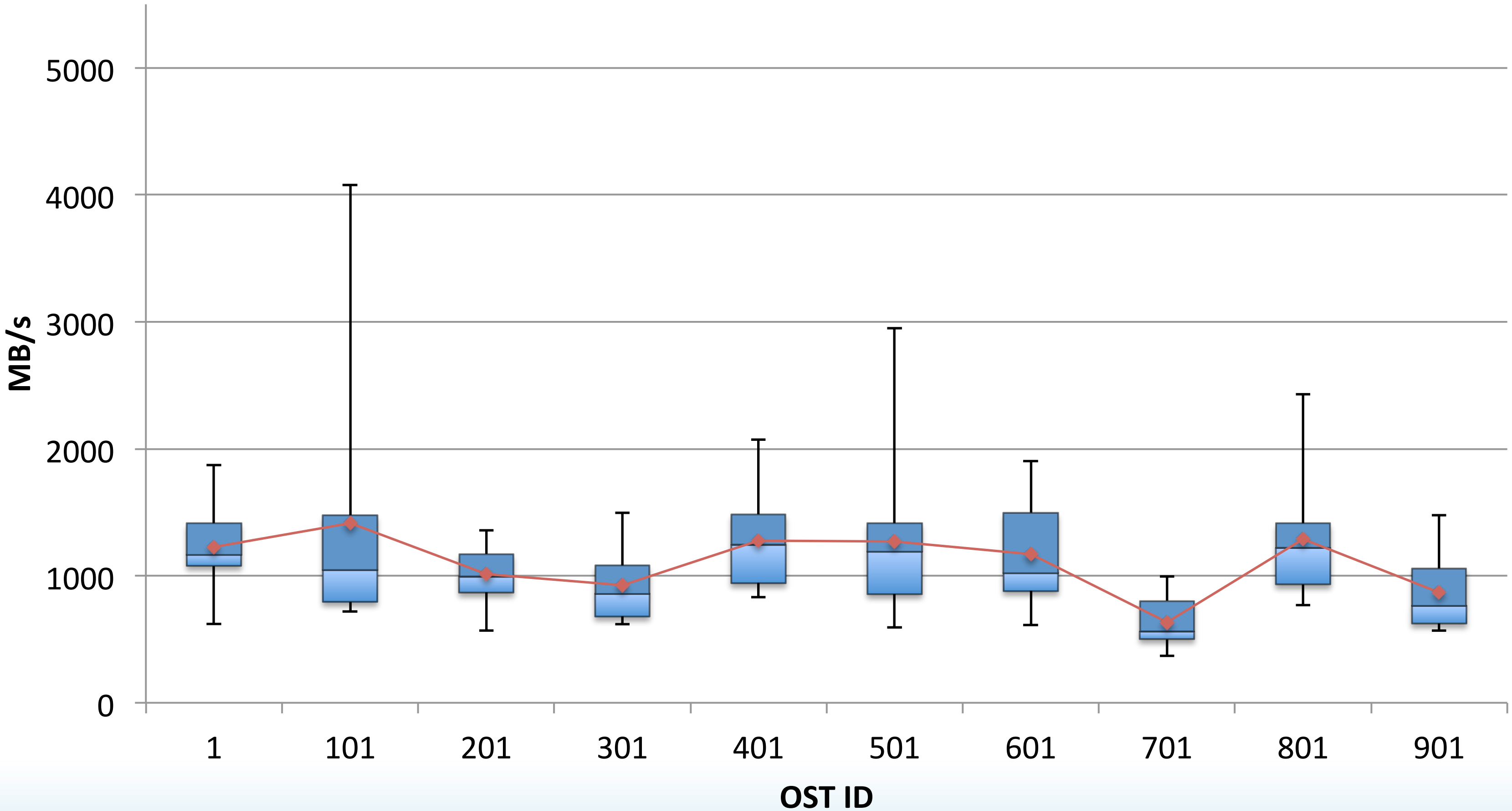
Writing to a Single OST (100 total IOR runs)



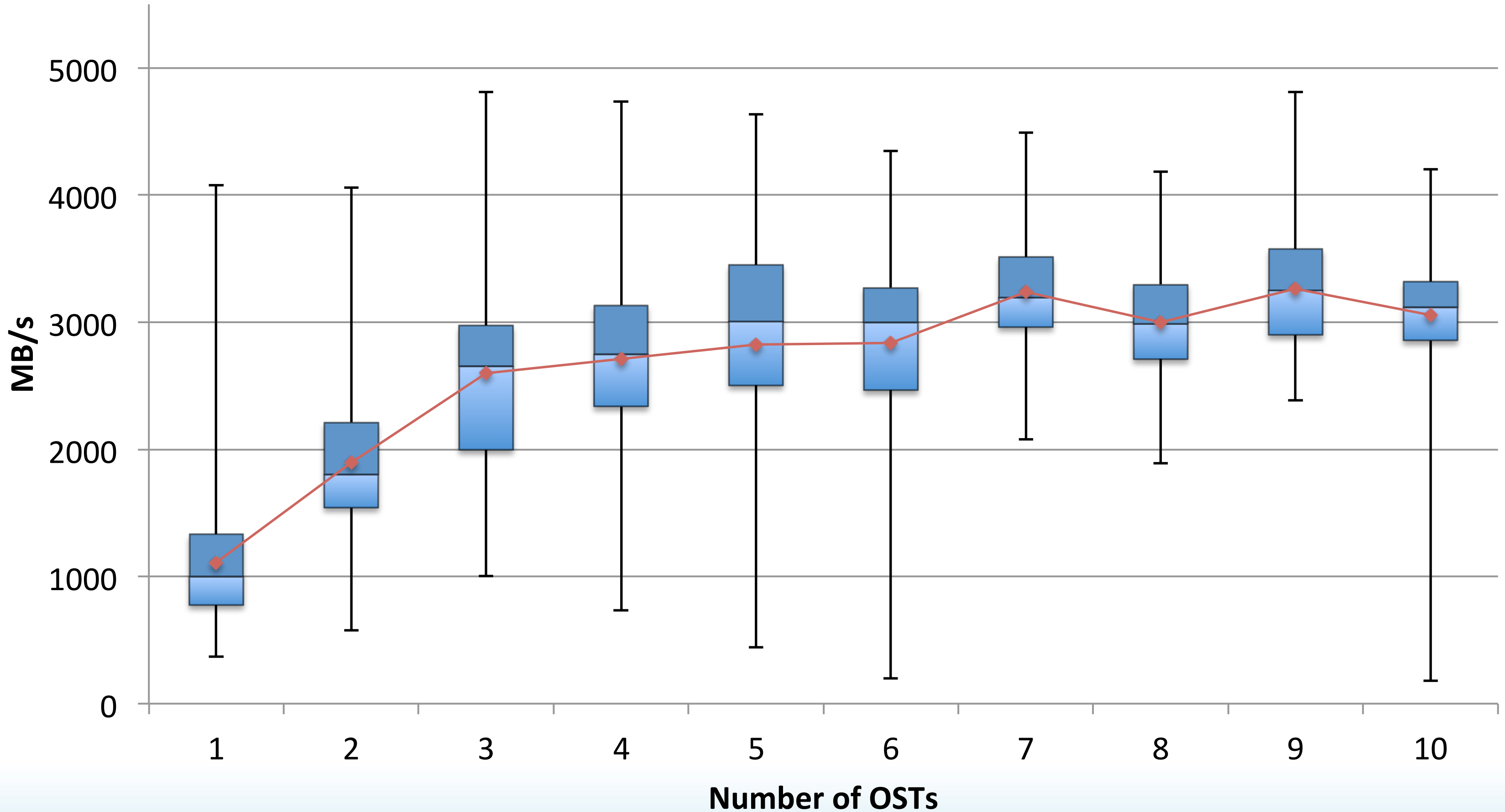
Varying Stripe Count (1000 total IOR runs)



Single OST Read (100 total IOR runs)



Varying Stripe Count (1000 total IOR runs)



Experiment 1: Conclusions

- Offset for throughput
  - No discernible relationship
  - I/O variability: outlier cases represent poor user experience
- Offset for consistency
  - Increasing stripe count increases throughput (as expected for parallel I/O)
  - Also increases variability (would be expected, if offset mattered)

Experiment 2: Offsets by Throughput *and* Distance

- Serial I/O
  - Worst case (for seeing variability)
  - Root privilege necessary for specifying a group of OSTs (pool) rather than a single OST (offset)
- IOR runs from one node to every OST
  - Stripe count of 1
  - 1440 OSTs in total
  - 768MB reads/writes
  - 2 iterations per test

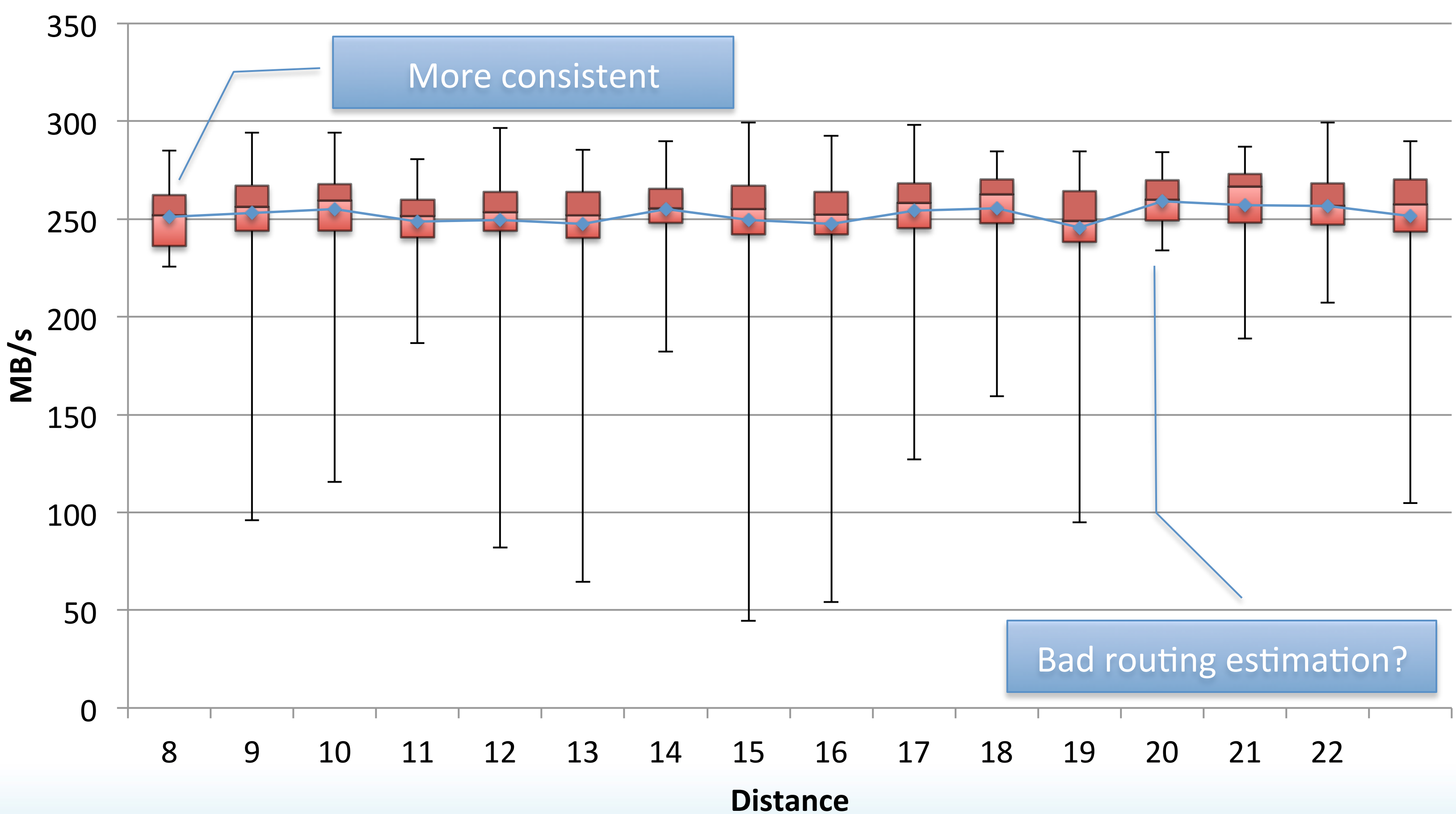
Experiment 2: Offsets by Throughput *and* Distance

- Calculating distance between a node and an OST
  - Must find necessary LNET router
  - Must replicate system routing
- Our distance approximation
  - Assume primary LNET is always used, looked up from table
  - Unweighted node-to-node "hop count"
    - Simple Manhattan distance from node to primary LNET
    - Torus wrap-around is considered

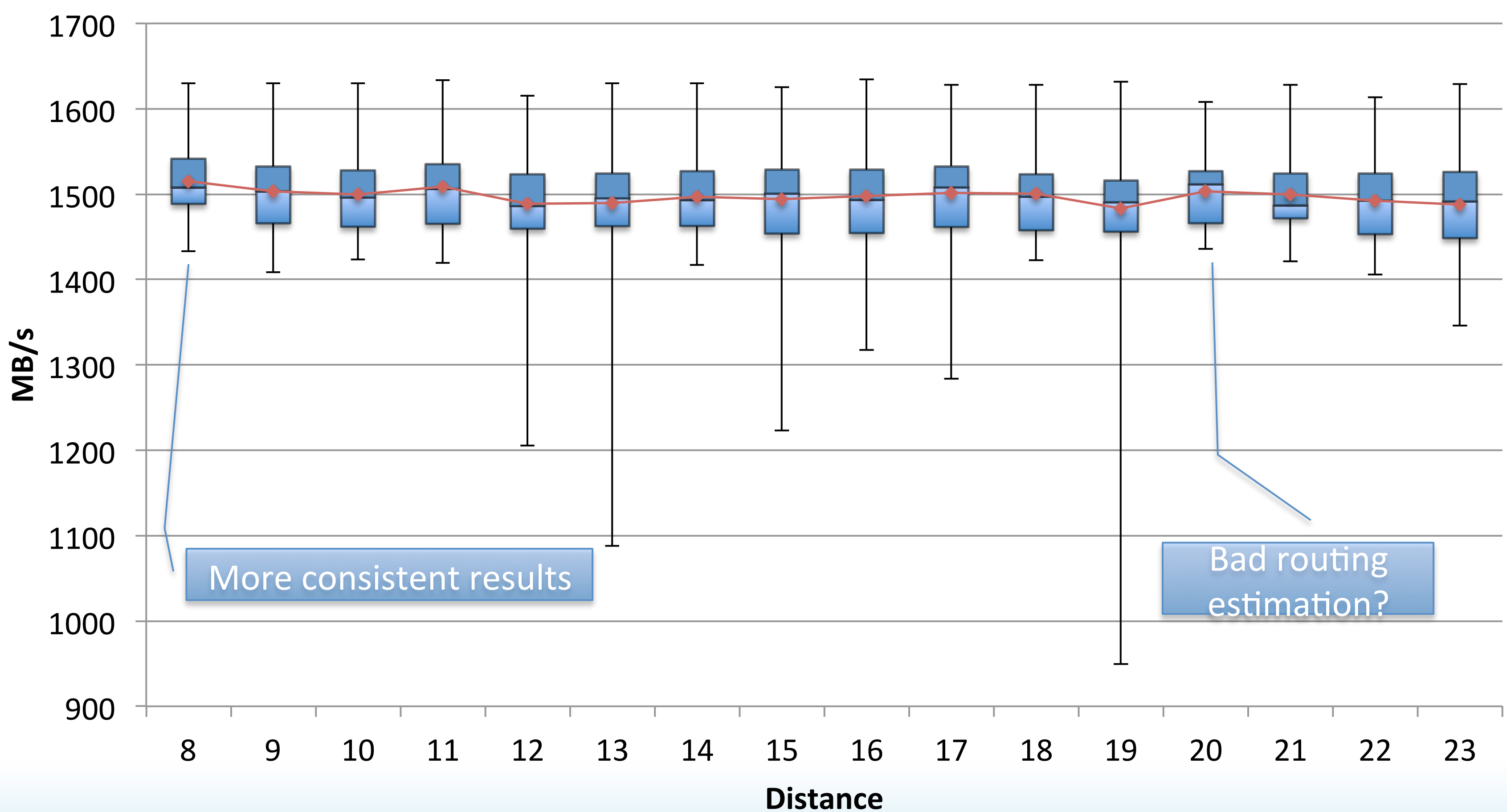
Limitations of the Metric

- Primary LNET usage not a guarantee
- Not all torus dimensions are created equally
- Actual system routing algorithm more complex
    - Utilizes low-level situational information
    - “Same” cases handled in different ways

Impact of OST-Node Distance, Single OST Write



Impact of OST-Node Distance, Single OST Read



Experiment 2: Conclusions

- Offset does not affect throughput
  - No correlation
  - Outliers *still* represent poor user experience
- Offset selection may
  - Improve consistency
  - Reduce outliers
- Metric may need improvement

Experiment 3: Enzo

- Compare two types of runs for a real application
  - Default offset
  - Optimized offset
    - Select OSTs per file based on writer location
    - Pre-configure striping, offset per file using lfs

Enzo Configuration

- Standard Input Set
  - Simulating dark matter with AMR
  - 128x128x128 Grid on 128 processes
- Output
  - ~11000 files
  - ~100 directories

Enzo Results

	Runtimes		% improvement
	Default Offset	Tuned Offset	
Run1	2571.37	2305.21	10.4%
Run2	2968.69	2355.28	20.7%
Run3	2594.47	2325.53	10.4%
Variability	223.02	25.18	88.7%

- More consistent runtimes
- Improvement in total runtime
- Minimal to no code modifications

Analysis of the Blue Waters File System Architecture for Application I/O Performance

CONCLUSIONS

Conclusions and Future Work

- Location-based offset selection appropriate target for I/O optimization on Blue Waters
  - Minimizes interference from/on other jobs
  - Minimizes network traversal
  - Provides boost to application performance
- Future work
  - Improve distance approximation
  - Control OST selection for larger stripe counts
  - Provide library to automate OST selection through API calls

Special Thanks

- Michelle Butler and Alex Parga from the Blue Waters storage team
- Manisha Gajbe from the Blue Waters Scientific & Engineering Applications team
- You (if you listened (or at least faked it at times))

Compliments?: sisneros@illinois.edu
Questions/complaints?: kalyan@illinois.edu