

# Supercomputer Storage System Models for the Age of Exascale Computing

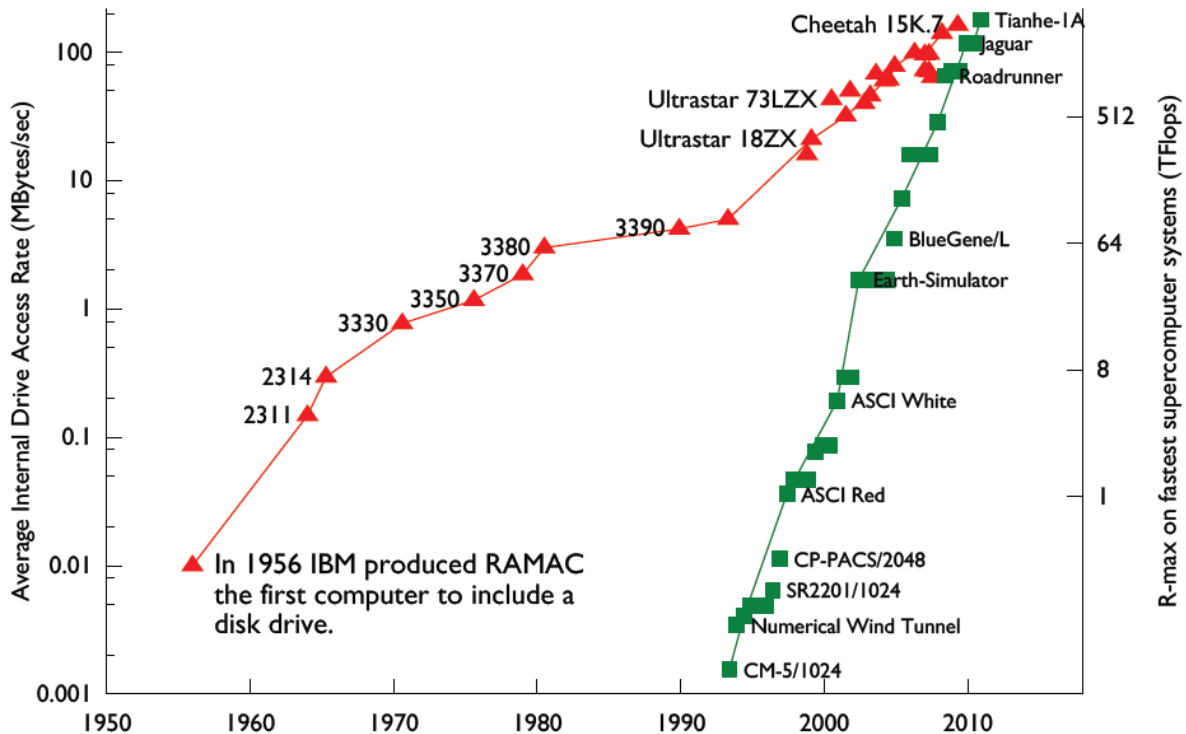
Ning Liu, Christopher Carothers, Jason Cope, Phil Carns,  
Robert Ross, Adam Crume, Carlos Maltzahn

{liun2, chrisc}@cs.rpi.edu

# Outline

- Motivation
- CODES Project
- Overview of the ALCF Blue Gene/P storage system
- Simulator/Models used for the ALCF Blue Gene/P storage system
- Experimental results
  - Intrepid filesystem storage model
  - Torus network model
- Future work

# Motivation: Parallel I/O is hard!

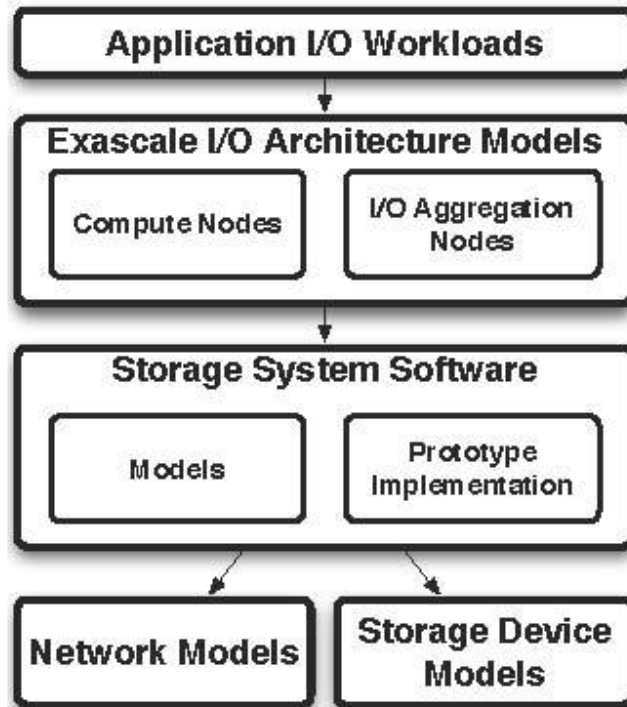


- Current Data:
  - Intrepid 478 Tflops but 60 GB/sec storage
  - Jaguar 2.2 Pflops but ~200 GB/sec storage
  - Storage BW/Flop is shrinking!!
- In practice...
  - 1/3 of app exec time consumed by I/O
  - Checkpointing & downward spiral of I/O
  - Kernel panic @ 600K files..

The gap between computation and I/O performance continues to increase.

# CODES Project

- CODES: Enabling Co-Design of Multilayer Exascale Storage Architectures
- **CODES GOAL: Develop a massively parallel simulation framework for evaluating exascale storage design challenges.**



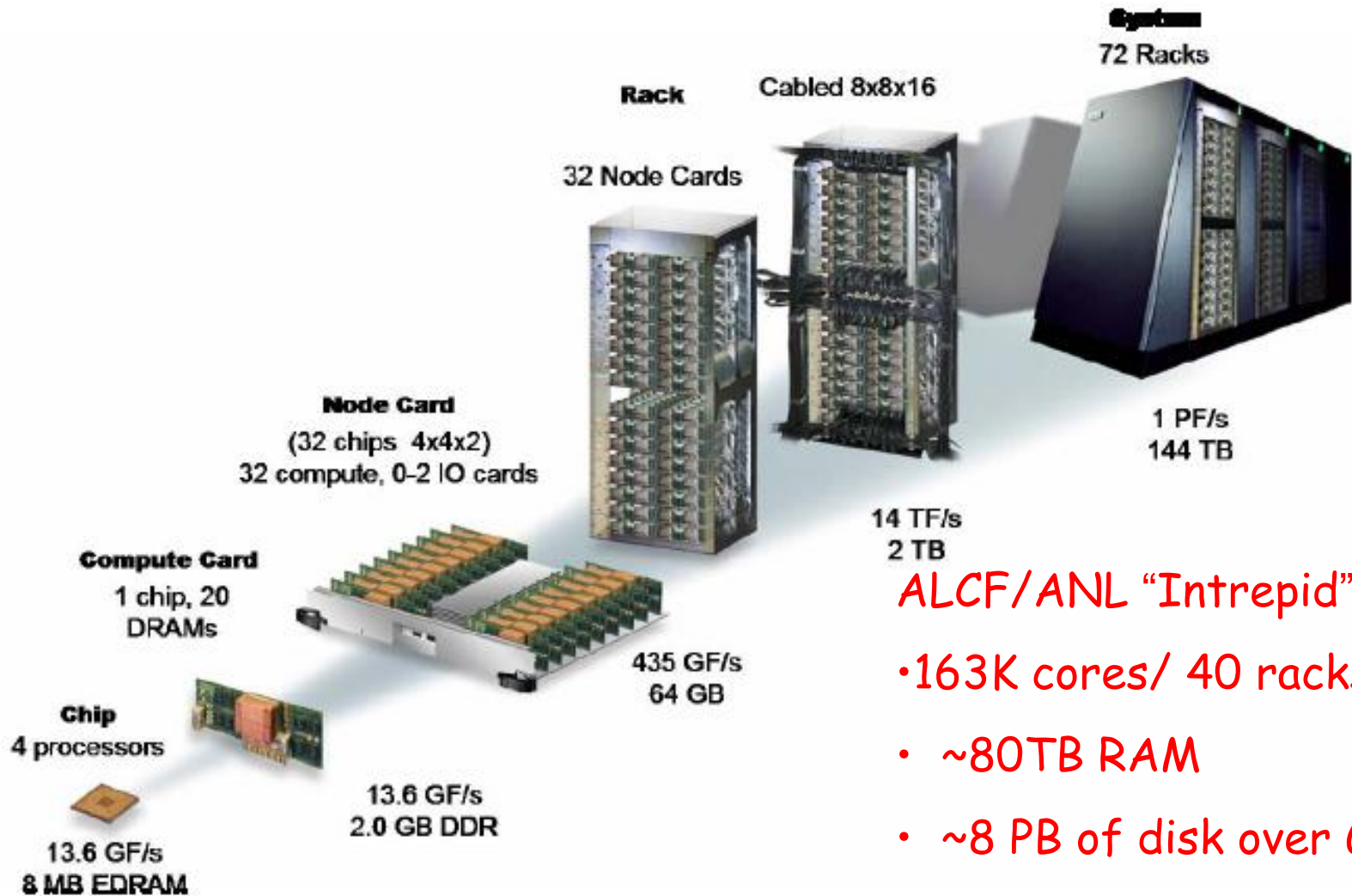
- **Application workload models:** Application I/O kernel models, I/O characterization models
- **I/O network models:** network cards, switches, and topologies
- **I/O storage node models:** storage software
- **I/O storage software:** models and prototype system software
- **I/O controller models:** RAID and enterprise storage devices
- **Disk models:** HDDs and SSDs

# Goals for the Study

- Model and simulate a leadership-class supercomputer storage system
- Validate the model using data collected from Intrepid[1]
- Simulate scalability of I/O system (software and hardware)
  - Does model capture system-wide runs scalability?
  - Where does model fidelity matter most?
- Understand particular I/O system characteristics
  - How do interactions of software and hardware components affect performance?
- Characterize challenges to I/O scalability of Leadership-class systems
  - Do current approaches scale on today's systems?
  - Anticipate problems likely to emerge in the next generation

[1]S. Lang, P. Carns, R. Latham, R. Ross, K. Harms, and W. Allcock, "**I/O Performance Challenges at Leadership Scale**," *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, Portland, Oregon, ACM, . Also Preprint ANL/MCS-P1648-0709, July 2009.

# Blue Gene /P Layout



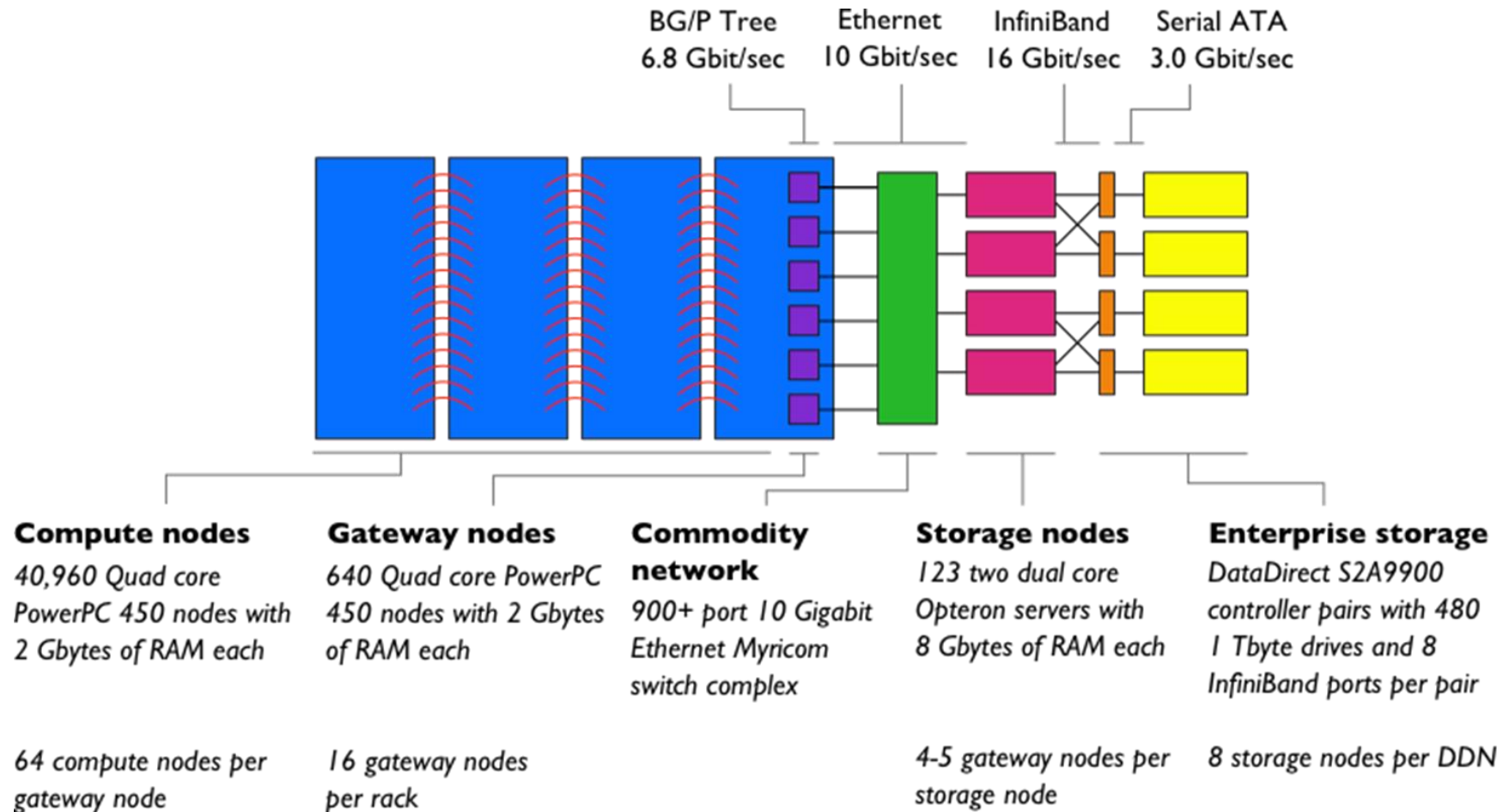
## ALCF/ANL "Intrepid"

- 163K cores/ 40 racks
- ~80TB RAM
- ~8 PB of disk over GPFS
- Custom OS kernel

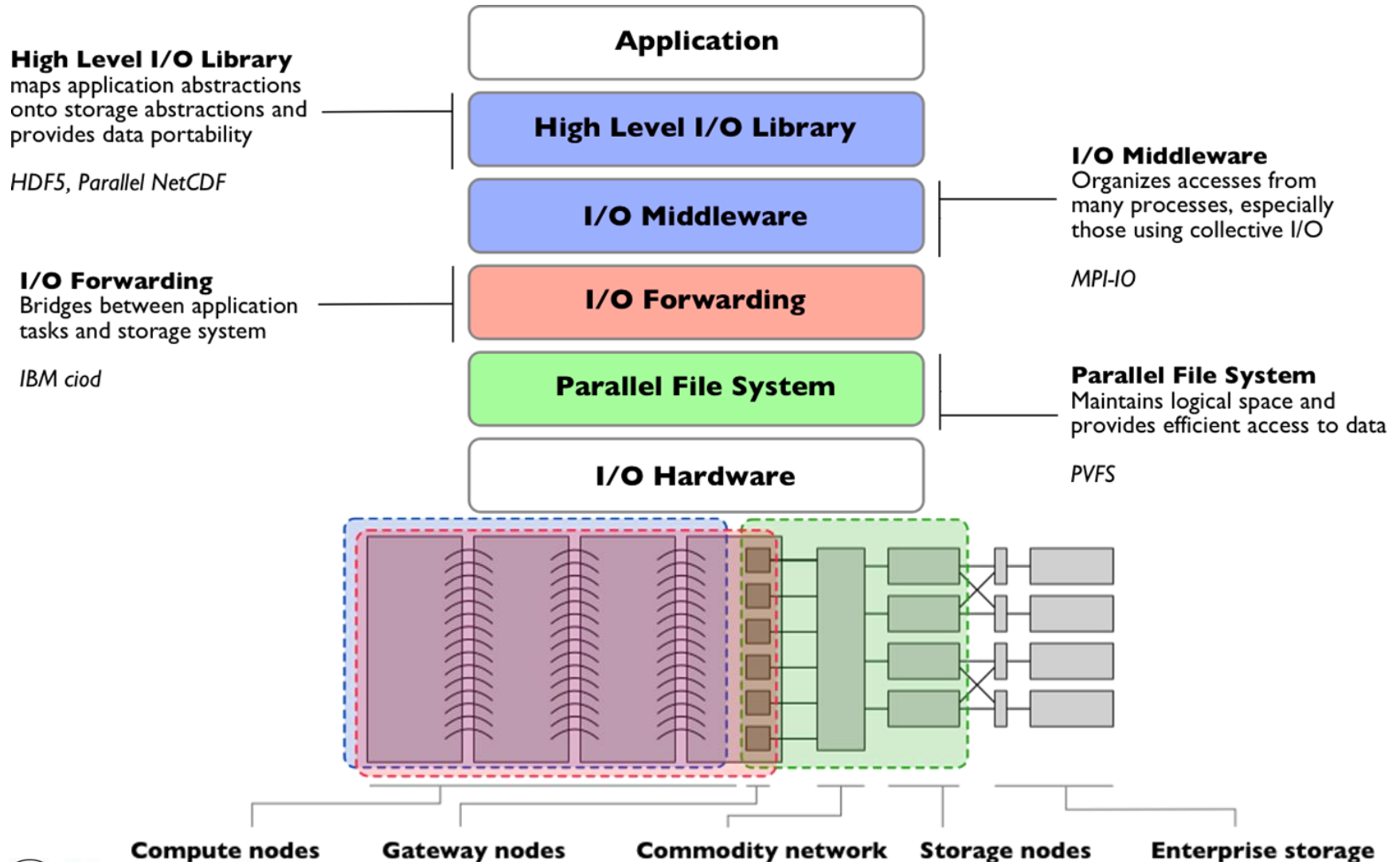
# Blue Gene/ P (vs. BG/L)

Property		BG/L	BG/P
Node Properties	Node Processors	2 * 440 PowerPC	4 * 450 PowerPC
	Processor Frequency	0.7 GHz	0.85 GHz
	Coherency	Software managed	SMP
	L1 Cache (private)	32KB / core	32KB / core
	L2 Cache (private)	14 stream prefetching	14 stream prefetching
	L3 Cache size (shared)	4 MB	8 MB
	Main Store/Node	512 MB, later 1 GB version	2 GB
	Main Store Bandwidth	5.6 GB/s (16B wide)	13.6 GB/s (2*16B wide)
	Peak Performance	5.6 GF/node	13.6 GF/node
Torus Network	Bandwidth	6*2*175MB/s = 2.1 GB/s	6*2*425MB/s = 5.1 GB/s
	Hardware Latency (Nearest Neighbor)	200 ns (32B packet) 1.6 us (256B packet)	100 ns (32B packet) 800 ns (256B packet)
	Hardware Latency (Worst Case)	6.4 us (64 hops)	3.0 us (64 hops)
Collective Network	Bandwidth	2*350MB/s = 700 MB/s	2*0.85GB/s = 1.7 GB/s
	Hardware Latency (Round Trip Worst Case)	5.0 us	3.0 us
System Properties	Peak Performance	360 TF (64k nodes)	1 PF (72k nodes)
	Total Power	1,7 MW	TBD ( about 2,3 MW)

# Intrepid's Storage System



# I/O Software Stack on Intrepid



# Overview of PVFS

- Open source parallel file system developed at Clemson University and Argonne National Laboratory
- File layout:
  - Each file consists of N data objects and 1 metadata object
  - Data is striped across all servers by default
  - Metadata is distributed on a per-file basis and throughout system or separately
- All data accessed concurrently, with byte-level granularity, with no distributed locking
- Underlying objects are stored in local files and Berkeley DB databases
- Object storage design, with each file consisting of:
  - Keeps metadata concise by allowing algorithmic distribution of data
  - Servers locally manage allocation of space on local disks
- Adaptive striping
  - Files begin life stored on a single server
  - Objects allocated on other servers in response to file growing beyond a threshold

# Discrete-Event Simulation

## What is Discrete-Event Simulation ?

- The operation of a system is represented as a chronological sequence of events. Each event occurs at an instant in time and marks a change of state in the system

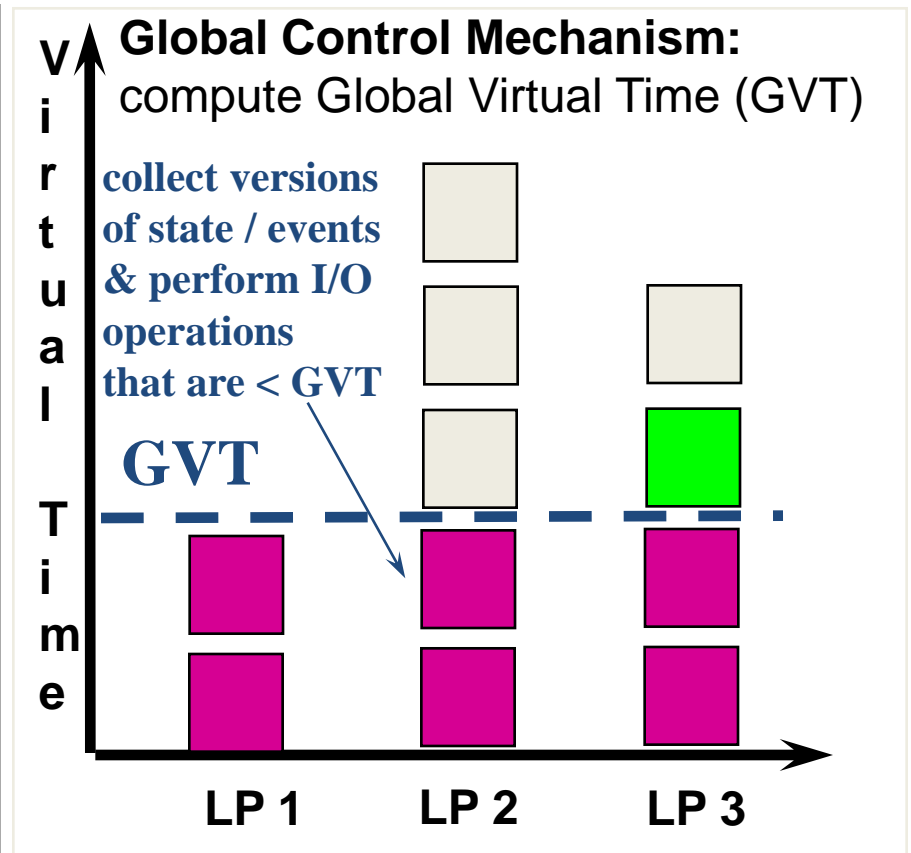
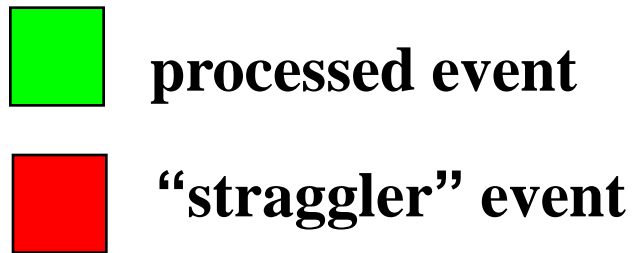
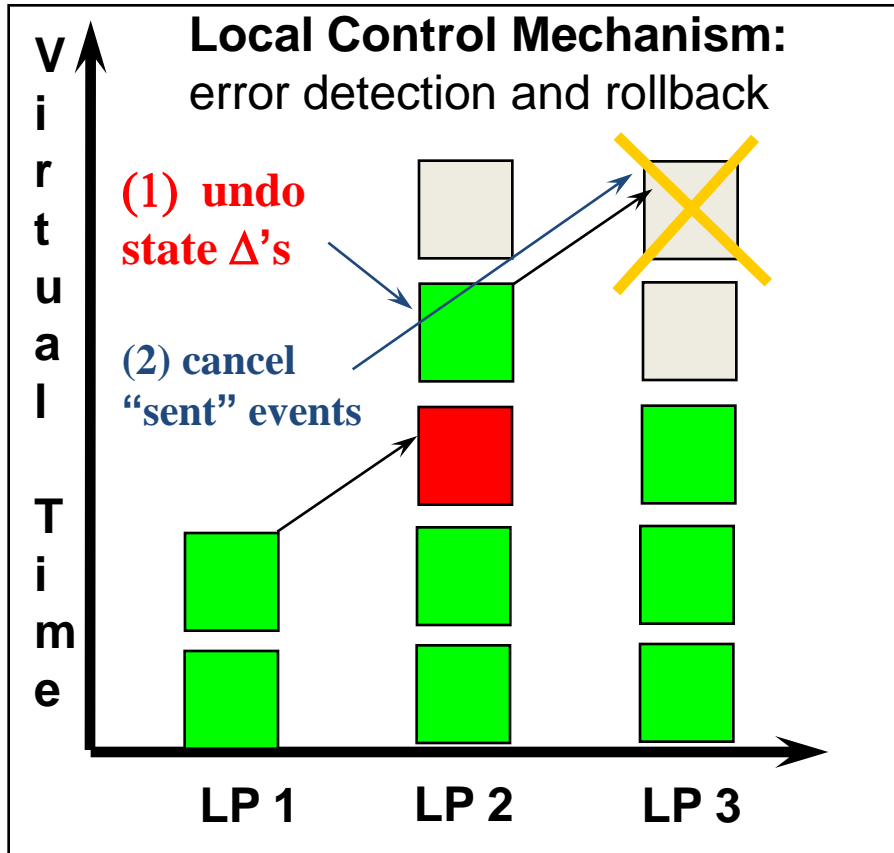
## Why Parallel Discrete-Event Simulation (DES)?

- Large-scale systems are difficult to understand
- Analytical models are often constrained

## Parallel DES simulation offers:

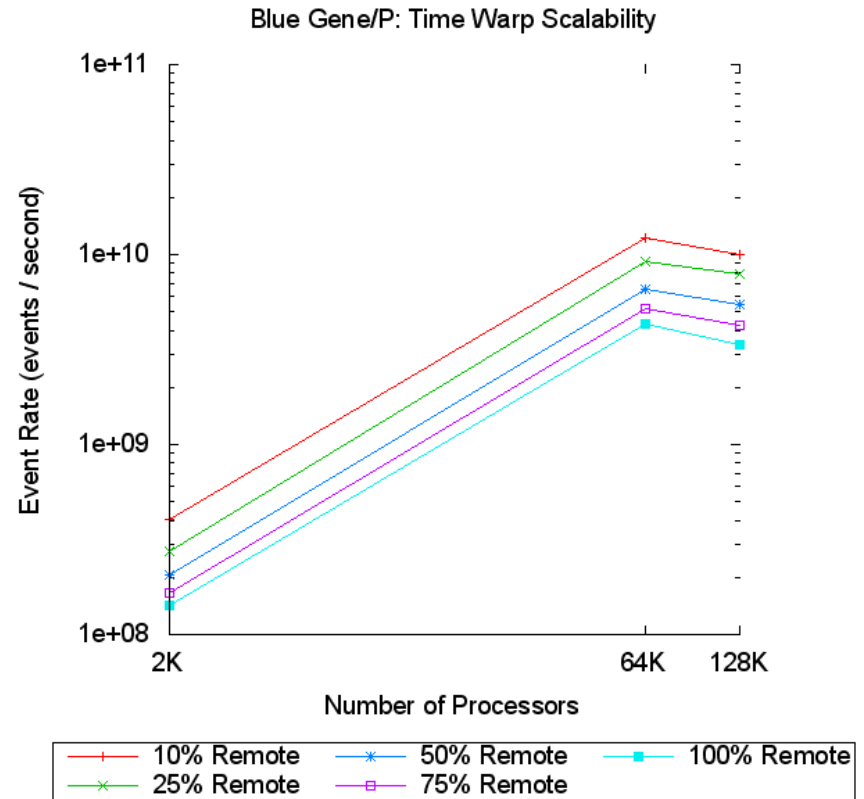
- Dramatically shrinks model's execution-time
- Prediction of future “what-if” systems performance
- Potential for real-time decision support
  - Minutes instead of days
  - Analysis can be done right away
- Example models: national air space (NAS), ISP backbone(s), distributed content caches, next generation supercomputer systems.

# Massively Parallel Discrete-Event Simulation Via Time Warp



# ROSS: Parallel Discrete-Event Simulator

- ROSS: Rensselaer Optimistic Simulation System
- Developed in ANSI C, API is simple and lean.
- Using Jefferson's Time Warp event scheduling mechanism.
  - Reverse computation for “undo”
  - Global virtual time algorithm exploits IBM Blue Gene's fast barrier and collective networks.
- Ross main page:  
[http://odin.cs.rpi.edu/ross/index.php/Main\\_Page](http://odin.cs.rpi.edu/ross/index.php/Main_Page)



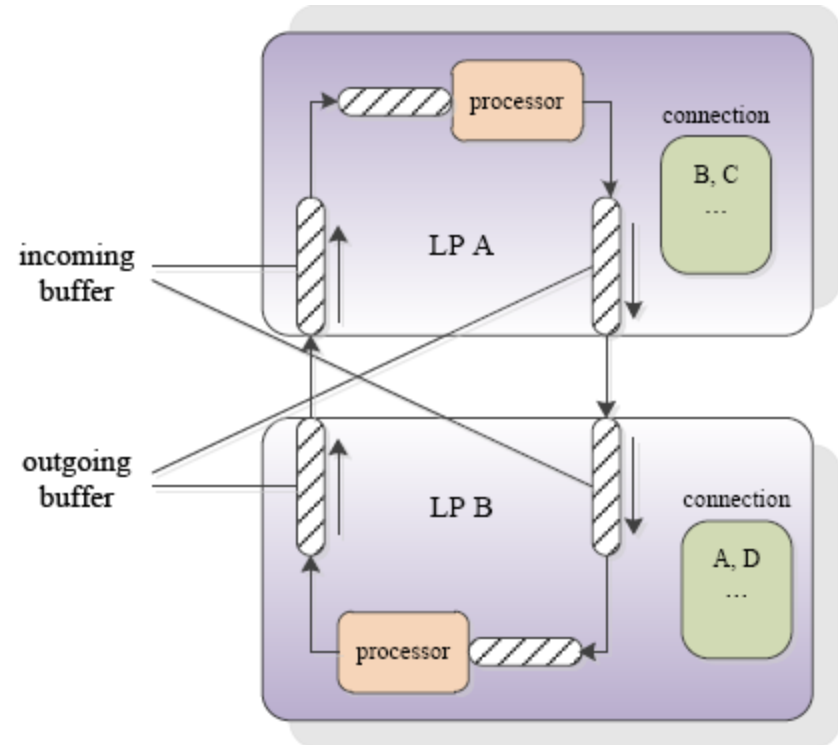
# LP as Basic Model Unit

Logical Process (LP) is the basic unit in the whole model, the following hardware components are modeled as LPs:

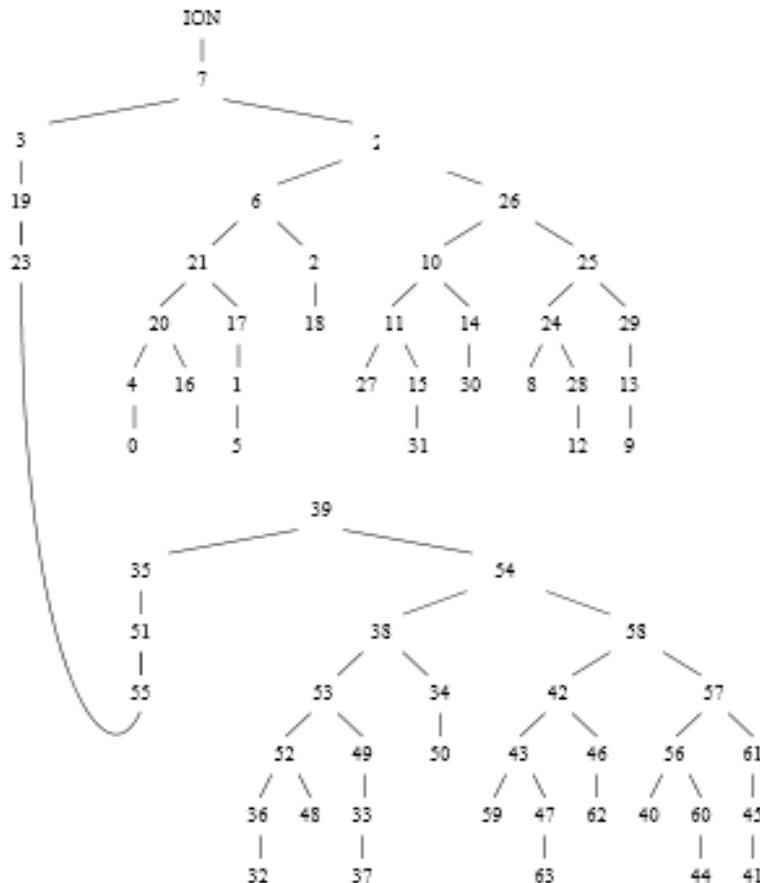
- Compute Nodes
- Gateway Nodes (I/O node)
- Storage Nodes (file server)
- Enterprise Storage (storage object)

Networks are modeled as the connection between LPs, configuration is recorded in a table in each LP unit.???

- Tree network is based on each pset
- Myricom network uses M to N mapping



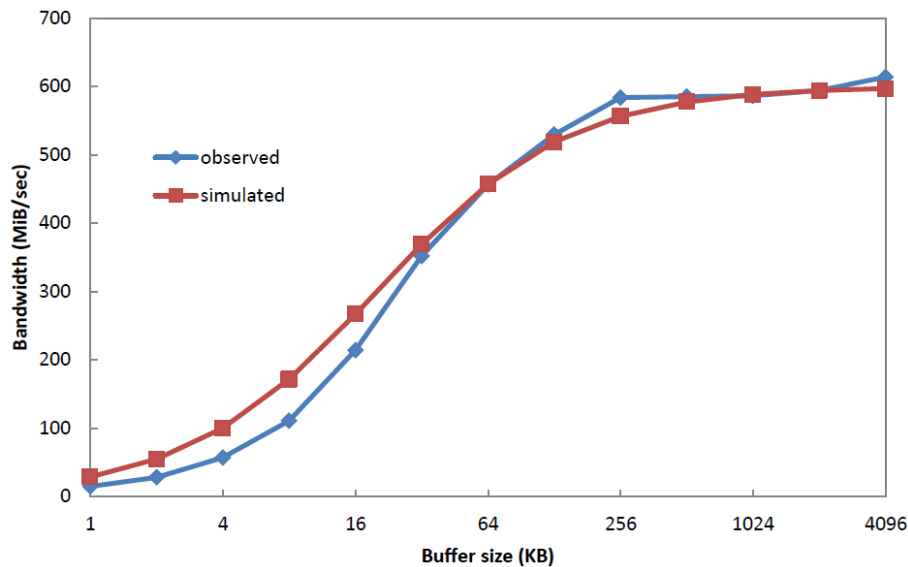
# IO-2-CN: Tree Network Topology



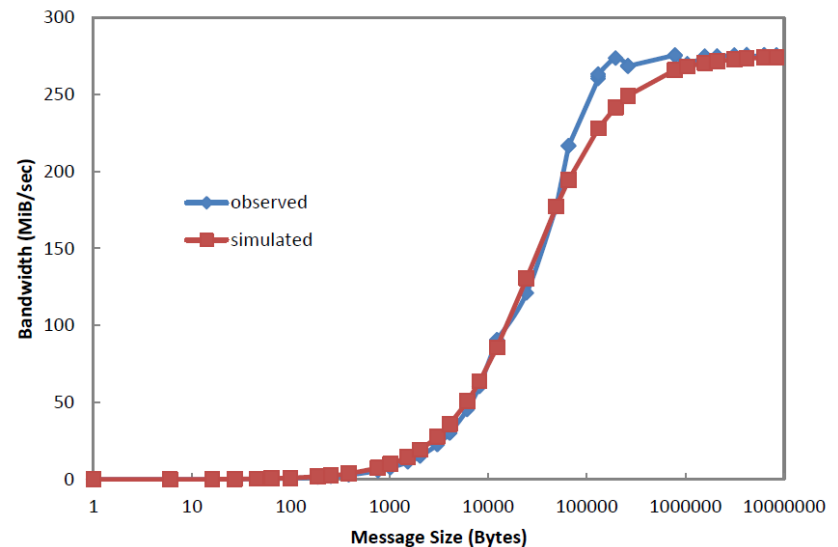
- Tree Network is based on each pset.
- Dedicated to I/O flow.
- Link bandwidth 700 MiB/Sec.
- Bottleneck is the link between ION and root CN.
- Tree network isn't the overall bottleneck in the storage system.
- Tree topology could vary in practice.

# Model for Component Connection

Model: Application bandwidth = payload / (handshake time + transmission time)

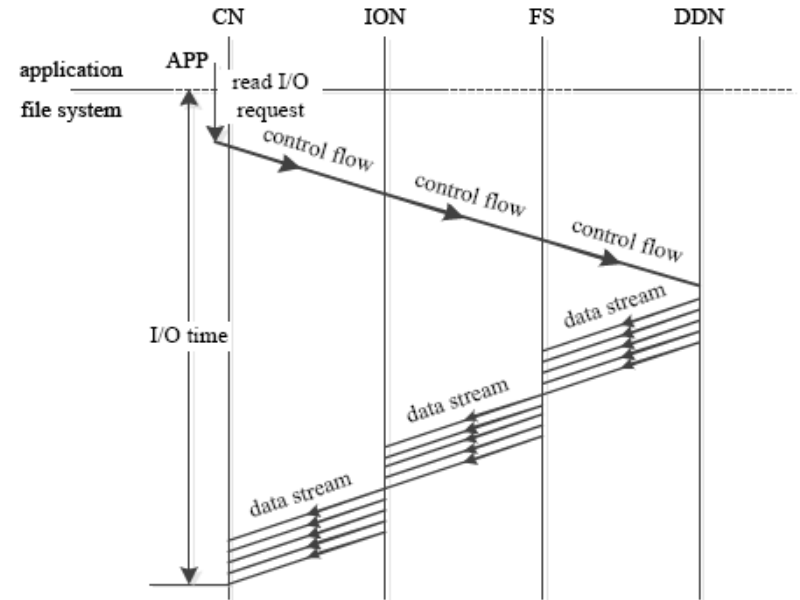
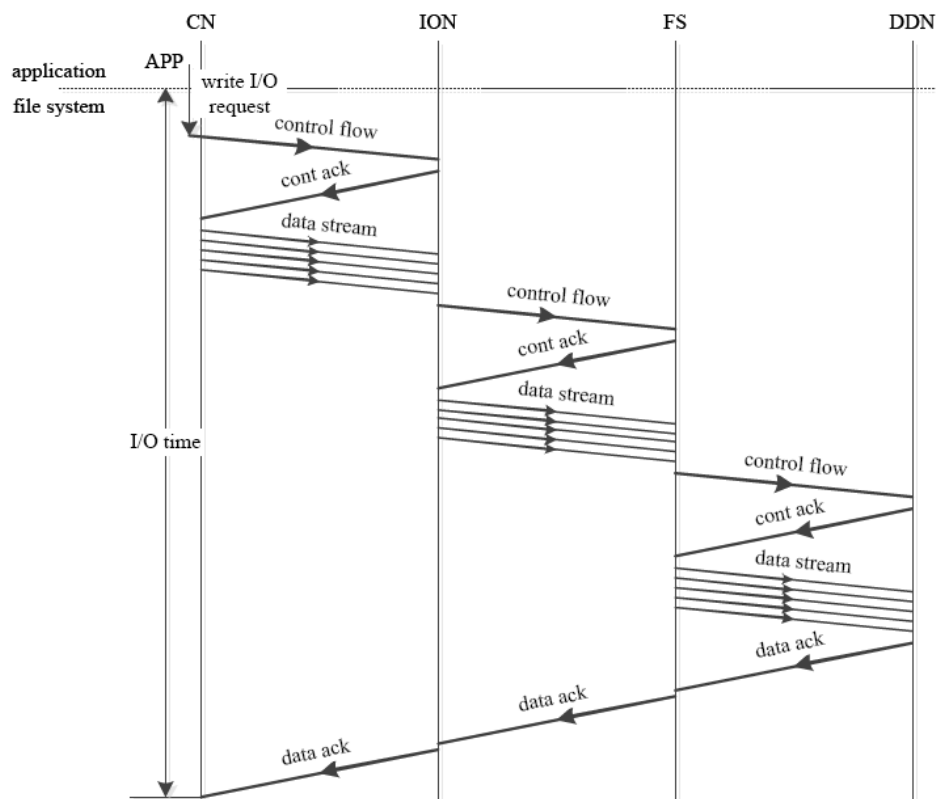


Bandwidth between CN and ION  
(64 processes write)

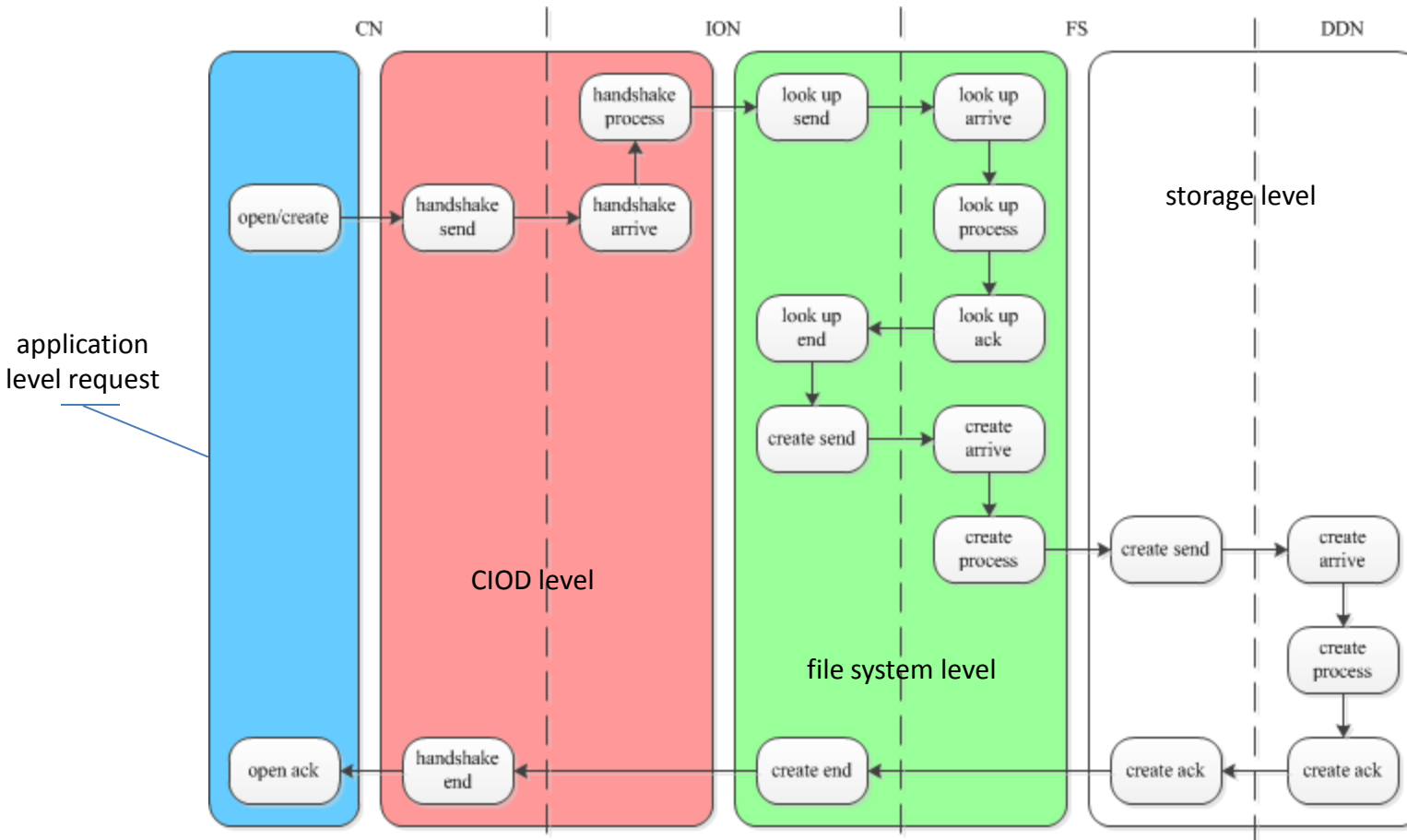


Bandwidth between ION and file  
server

# Basic Control and Data Flow

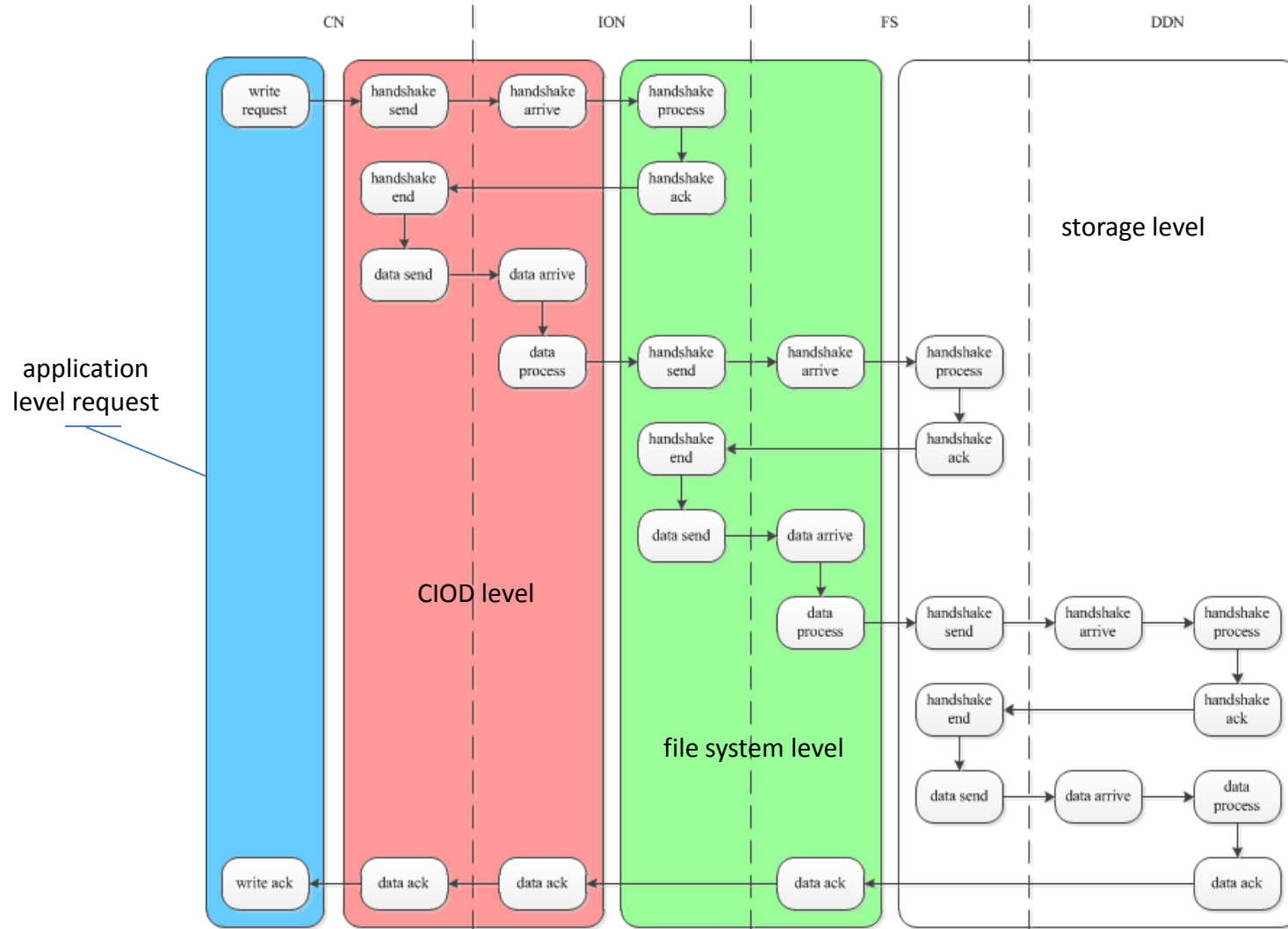


# File Open Request Model



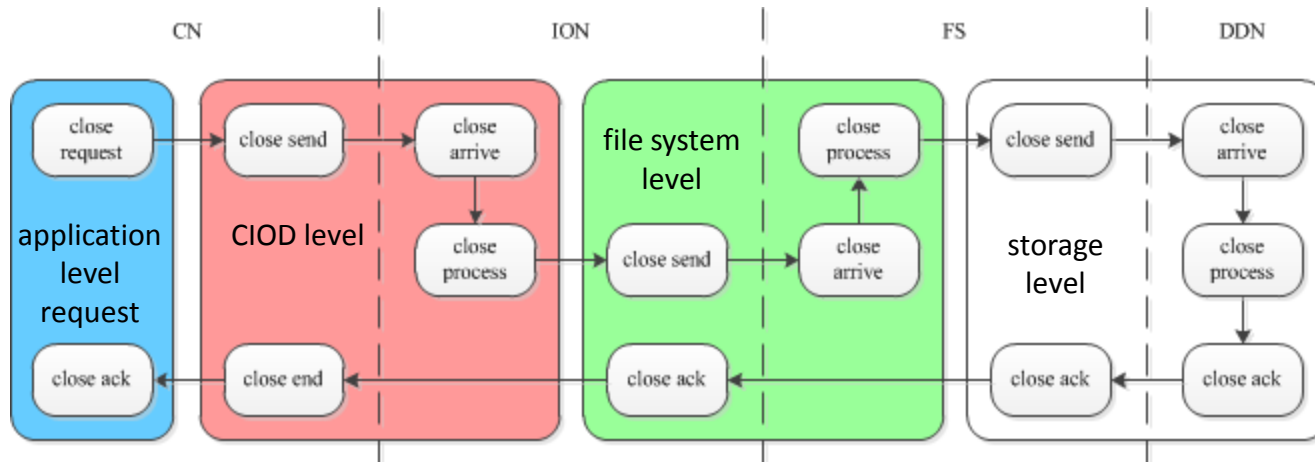
- PVFS clients talk to PVFS servers iteratively to find entry
- Randomly select a fileserver and create the object

# File Write Request Model

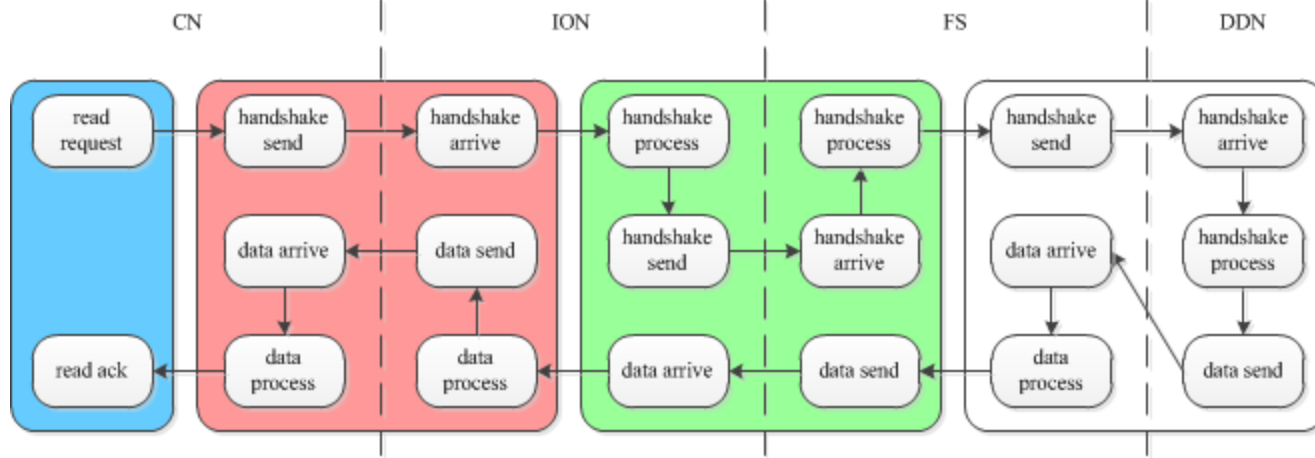


# Close/Read Request Model

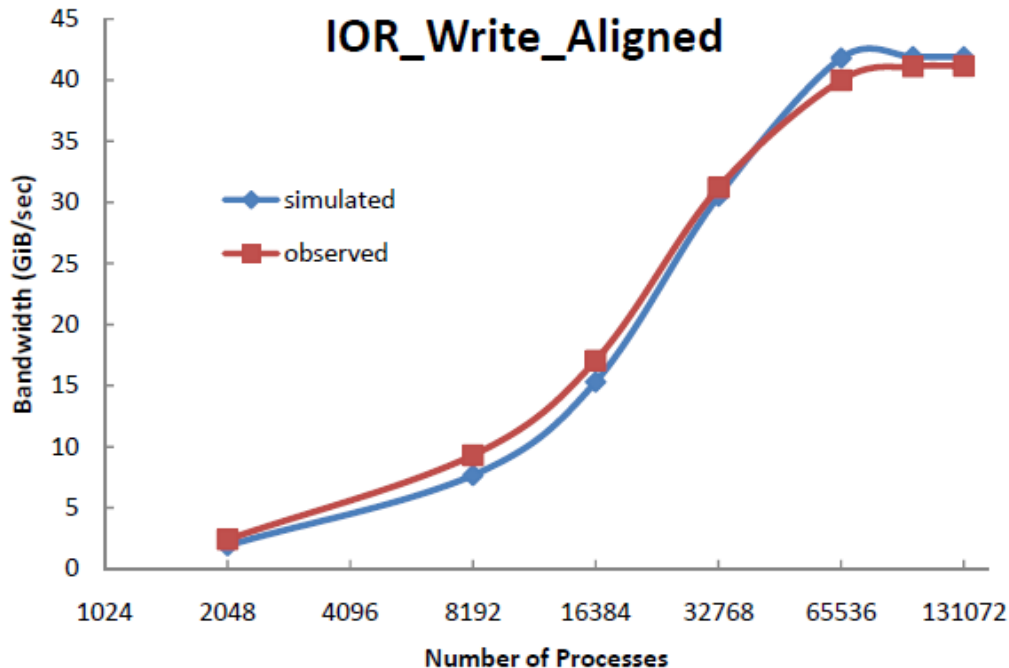
**CLOSE**



**READ**

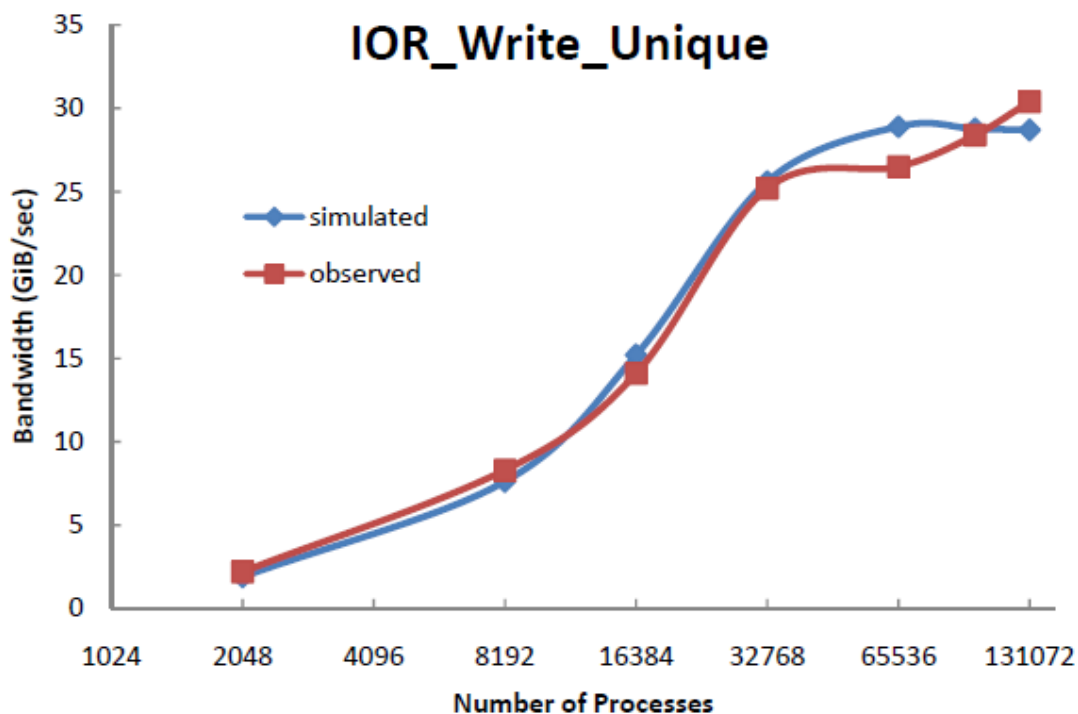


# IOR Shared Aligned Write Test



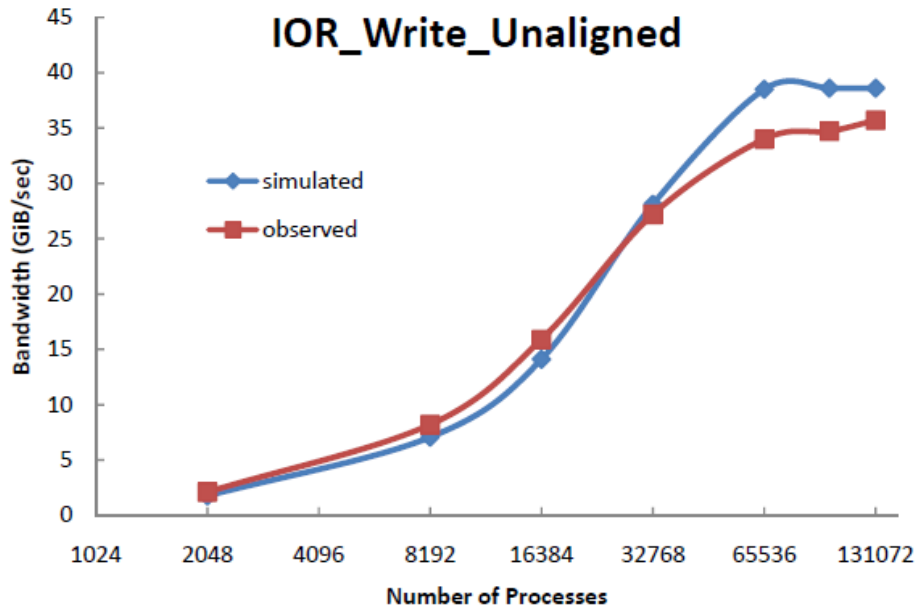
- Used Posix interface.
- Used 4 MB accesses for a total of 64 MB per process, align perfectly with the stripe of the file.
- Our model captures the bandwidth from 2048 to 128K processes
- Over predict at high processes count
- Possible reasons:
  - Lack of accuracy in the Myricom Network, didn't capture the payload packet queuing

# IOR Unique Aligned Write Test



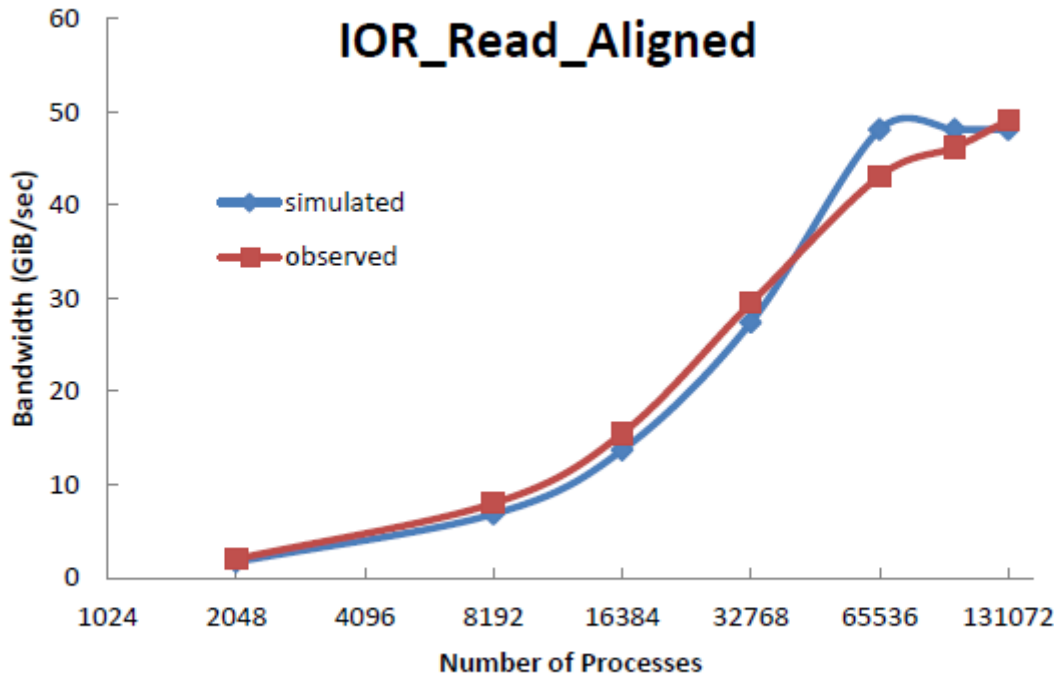
- Used Posix interface.
- Used 4 MB accesses for a total of 64 MB per process, align perfectly with the stripe of the file.
- More metadata request compared to shared file test.
- Curve tail goes down at high process counts.
- Possible reasons:
  - Model not accounting for queuing of meta-data at fileservers
  - Model not account Myricom network congestion

# IOR Shared Unaligned Write Test



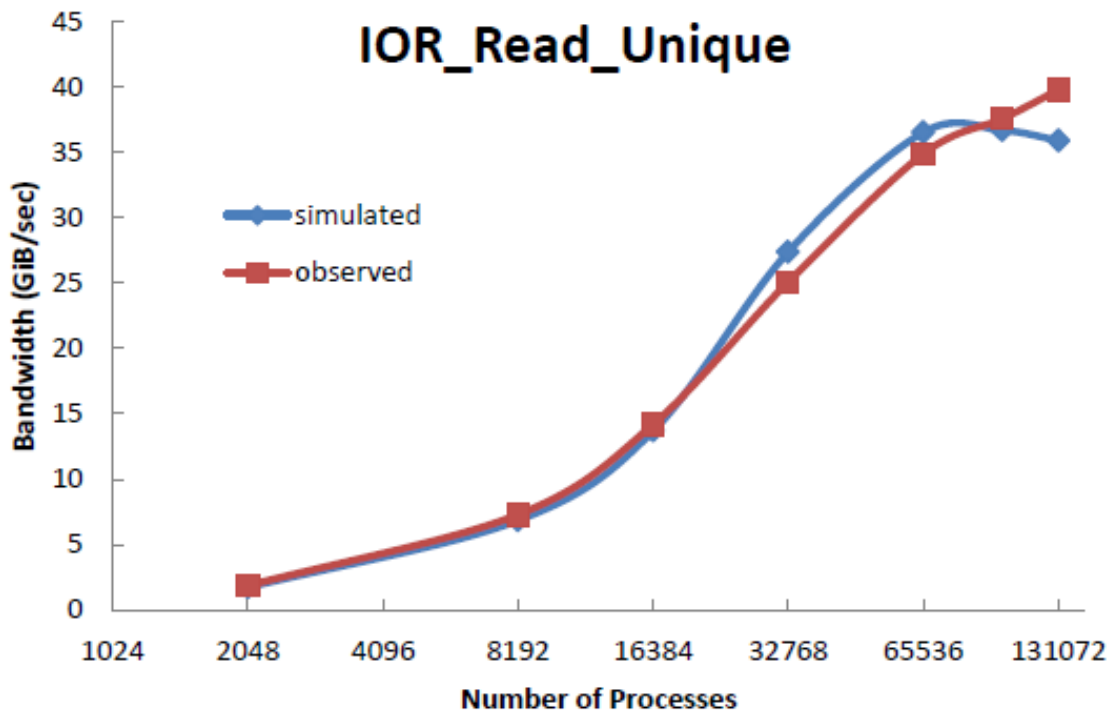
- Used 4MB ( $4 * 10^6$ ) accesses for a total of 64MB per process
- Requests span multiple file stripe units, requiring that requests serviced by 2 storage node rather than one
- Over predict (5-10%)
- Possible reasons
  - Bandwidth inaccuracy between IONs and FSs
  - Handshake overhead inaccuracy

# IOR Shared Aligned Read Test



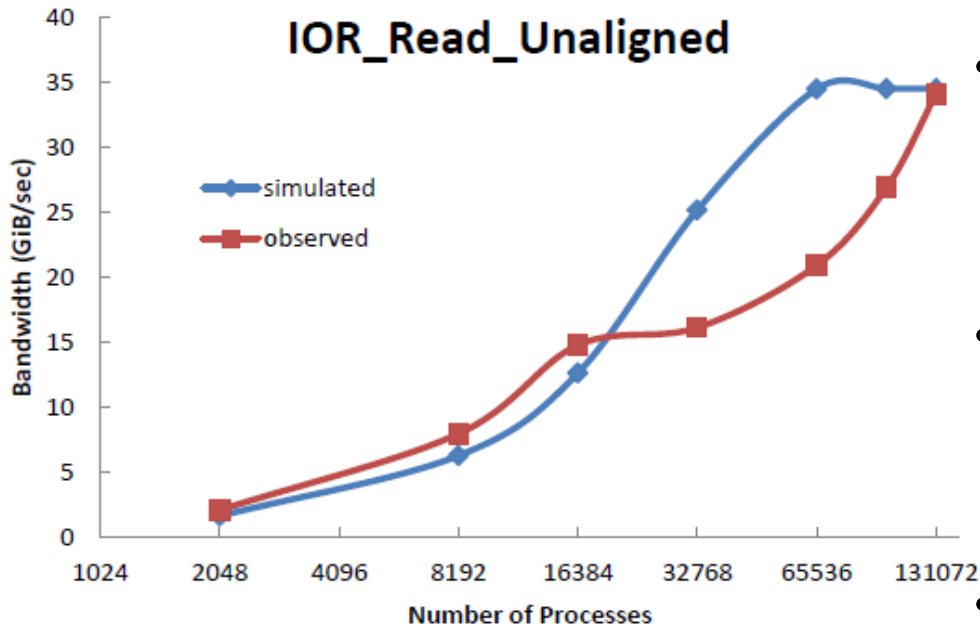
- Used Posix interface.
- Used 4 MB accesses for a total of 64 MB per process, align perfectly with the stripe of the file.
- Under-predict at high process counts.
- Possible reasons:
  - different read/write performance at the storage node
  - Model not accounting for Myricom network queueing

# IOR Unique Aligned Read Test



- Used Posix interface.
- Used 4 MB accesses for a total of 64 MB per process, align perfectly with the stripe of the file.
- More meta request compared to shared file test.
- Under-predict at high process counts
- Possible reasons:
  - Over-quantify the metadata conflicts
  - Different DDN read performance at storage levels due to increased cores

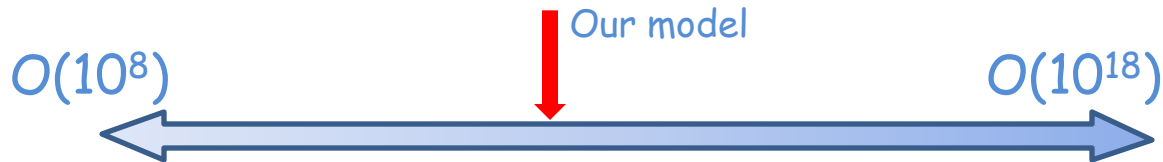
# IOR Shared Unaligned Read Test



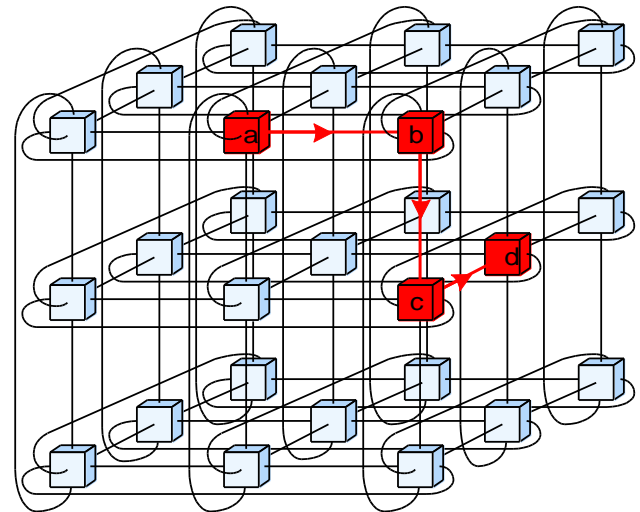
- Used Posix interface
- Used 4MB ( $4 * 10^6$ ) accesses for a total of 64MB per process
- Requests span multiple file stripe units, requiring that requests serviced by 2 storage node rather than one
- Simulated read performance curve is similar to simulated write performance curve because the striping algorithm is the same
- Max error is 30-40%
- Possible reasons:
  - Lack of queuing at fileserver and Myricom network layer

# Billion-Node Packet Model

- For accurate storage simulations, network is clearly important!
- So, can we model an “exascale” like network at the packet-level
  - Cycle accurate is clearly out of reach at this point...
- total number of generated packets is  $O(10^{11})$
- total number of events scheduled is  $O(10^{13})$



- Used Torus since it is a popular topology
- Each node is an LP in the parallel simulator
- Packet injection rate are 160 & 200 packets/ns
- Random destination selected
  - Yields larger avg. hop distances than typical for HPC
- Model consumes > 2 TB of RAM
- Validated model against Blue Gene/L Torus network
  - See PADS 2011 paper for details



# Billion-Node Torus Model on Blue Gene/L

TABLE I

Strong Scaling Performance of 1-Billion-Node Model at Configuration  $32^6$  with Packet Arrival Rate of 200 pkt/ns on Blue Gene/L.

number of processors	4,096	8,192	16,384
efficiency	97.05%	96.00%	81.90%
<b>event-rate (M/sec)</b>	<b>639</b>	<b>1,066</b>	<b>1,681</b>
remote event percentage	11.72%	12.41%	13.79%
secondary rollback rate	0.0286%	0.034%	0.220%

# Billion-Node Torus Model on Blue Gene/P

TABLE II

Strong Scaling Performance of 1-Billion-Node Model at Configuration  $32^6$  with Packet Arrival Rate of 160 pkt/ns on Blue Gene/P.

number of processors	4,096	8,192	16,384	32,768	65,536	131,072
efficiency	99.83%	99.90%	99.83%	99.55%	98.89%	97.51%
<b>event-rate (M/sec)</b>	<b>639</b>	<b>1,192</b>	<b>2,260</b>	<b>4,002</b>	<b>7,307</b>	<b>12,359</b>
remote event percentage	11.71%	12.39%	13.77%	16.53%	16.88%	17.22%
secondary rollback rate ( $10^{-5}$ )	1.06	0.254	0.0429	0.51	3.87	21.7

# Summary and Conclusions

- Built hardware and software models of Intrepid's PVFS storage system
  - Focused on developing a simple model with satisfactory fidelity
  - Provides near standard open(), read(), write(), close() interfaces at application level
  - Captures the necessary interactions between the storage system components
- Compared the model using several application I/O patterns generated by the IOR benchmark
  - Shared file vs. unique file access patterns
  - Stripe aligned vs. stripe unaligned access patterns
  - Read vs. write access patterns
- Provides a framework for evaluating the components of current Petascale storage systems
  - Identified critical components and interactions to model Intrepid's storage system
  - Confirmed simulation bottlenecks of the Intrepid storage system
  - Identified how much fidelity is required to accurately model the storage system
  - Provides basis for modeling and simulating Exascale storage systems

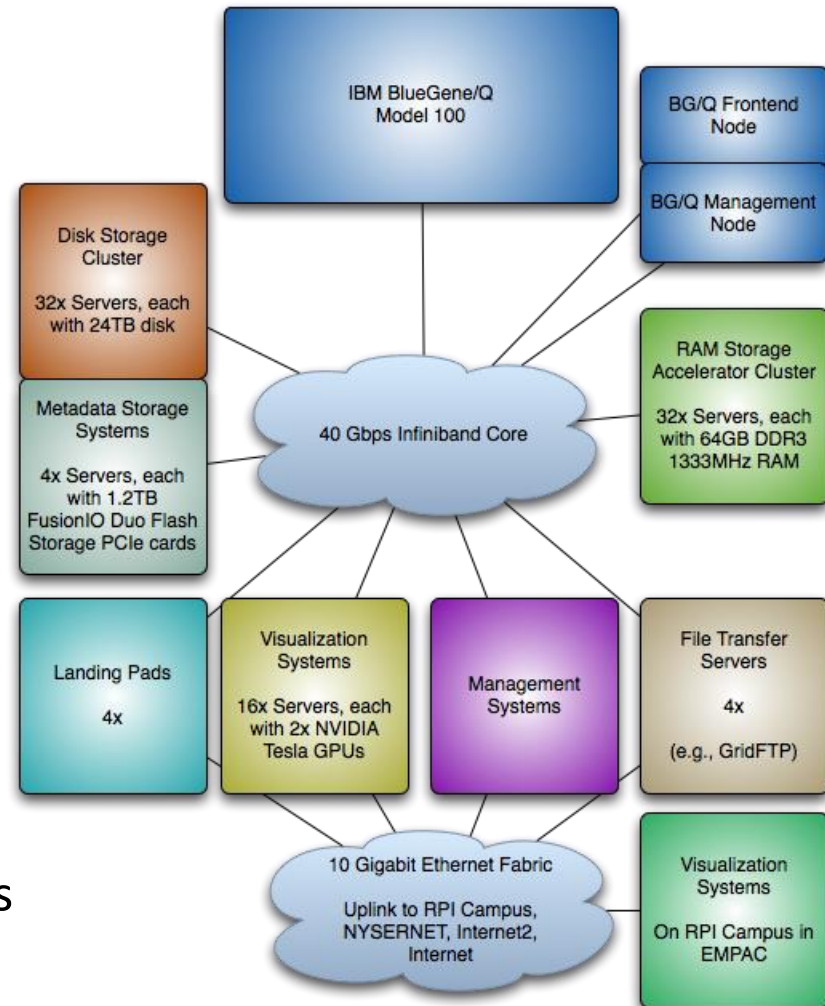
# Future Work

- Short term(1-2 months)
  - Add burst buffer to current model and capture its effects on periodical checkpoint
  - Modify parameters in hardware components and compare the difference
  - Run current model to exascale, e.g. 2000X larger
  - Validate the model using real applications (FLASH I/O, S3D I/O, Chombo I/O)
- Long term (6-12 months)
  - Model darshan workloads/traces
  - Model Myricom network, infiniband network
  - Add two-phase I/O features to current model, i.e. Combine the torus and tree network
  - Switch between different file systems and storage device
  - Run modified model to exascale, e.g. 2000X larger
  - Simulate a blue gene/Q: Mira storage system

# NSF MRI “Balanced” Cyberinstrument @ CCNI

Proposed RPI / CCNI Blue Gene/Q System Design

- Blue Gene/Q
  - 104 Tflops @ 2+ GF/watt
  - 32K threads/8K cores
  - 8 TB RAM
- RAM Storage Accelerator
  - 4 TB
  - 32 servers @ 128 GB each
- Disk storage
  - 32 servers @ 24 TB disk
  - 4 meta-data servers w/ SSD
- Viz systems
  - CCNI: 16 servers w/ dual GPUs
  - EMACS: display wall + servers



# Acknowledgement

- Special thanks to Jason Cope, Rob Ross, Chris Carothers, Phil Carns, Kevin Harms, Carlos Maltzahan and Adam Crume.
- Thanks everyone in Radix group.
- Codes project is funded through Department of Energy Office of Science
- Questions?