Distributed Object Storage Rebuild Analysis *via* Simulation with GOBS

Justin M Wozniak, Seung Woo Son, and Robert Ross

Argonne National Laboratory

Presented at:

Workshop on Fault-Tolerance for HPC at Extreme Scale (FTXS 2010)
Chicago, IL – June 28, 2010

Outline

- Context
  - High-performance storage use cases
  - Parallel filesystems
  - Object storage motivation
- Problem statement
  - Disk size/fault rate dynamic
  - System model
- GOBS Simulator
  - Data placement for survivability
  - Simulation and analysis of rebuild performance (GOBS)
- Preliminary results
  - Hot spots
  - Eager rebuilds

Context

Uses of high-performance storage (1)

- Checkpoint
  - Write out all user memory to non-volatile storage
  - Basic survival strategy to avoid lost work
- Optimal checkpoint interval
  - First-order approximation to optimal checkpoint write interval

$$t_o = \sqrt{2 t_w t_f}$$

» t_o : checkpoint interval
» t_w : time to write checkpoint
» t_f : mean time to failure

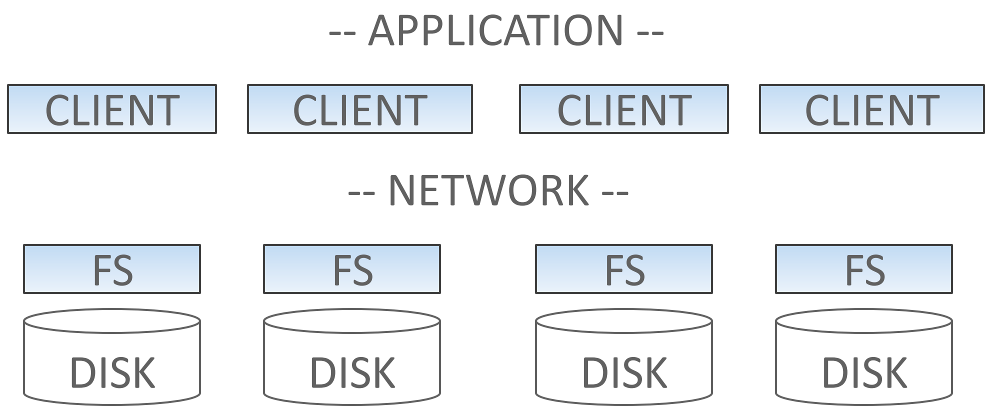
- Future trends
  - Bigger memory → longer writes
  - More components → more faults
  - Could reach a critical point

Uses of high-performance storage (2)

- Useful application data
  - MPI-IO
    - Parallel interface for file I/O operations
    - Allows I/O experts to implement optimizations
  - High-level libraries
    - Provide a variable-oriented view on data
    - PnetCDF, HDF5, ADIOS
    - Can use MPI-IO
  - POSIX I/O
    - Still prevalent in large-scale applications
    - Must maintain user expectations, portability, but make use of high-performance machines

Parallel filesystems

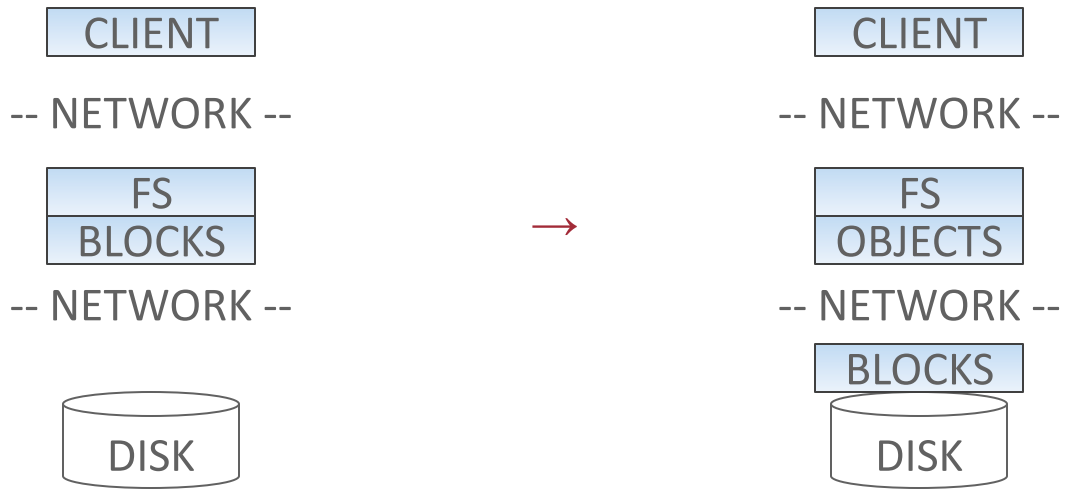
- Eliminate single bottlenecks in I/O



- PVFS – Clemson, ANL
  - Open source, community maintained
- GPFS – IBM
  - Licensed by IBM
- Lustre – Oracle/Sun
  - Open source but supported
- PanFS – Panasas
  - Software/hardware packages

Object storage

- Separation of concerns



- Employed by many modern systems – not “old news” either

Distributed object placement

- Object placement algorithm:

$$F(x_{i,j}) \rightarrow s_k$$

  - » $x_{i,j}$: object i, replica j
  - » s_k : server k

- Replicas must be placed on different servers
- Place whole objects
- Essentially distribute a hash table over multiple sites

Related work

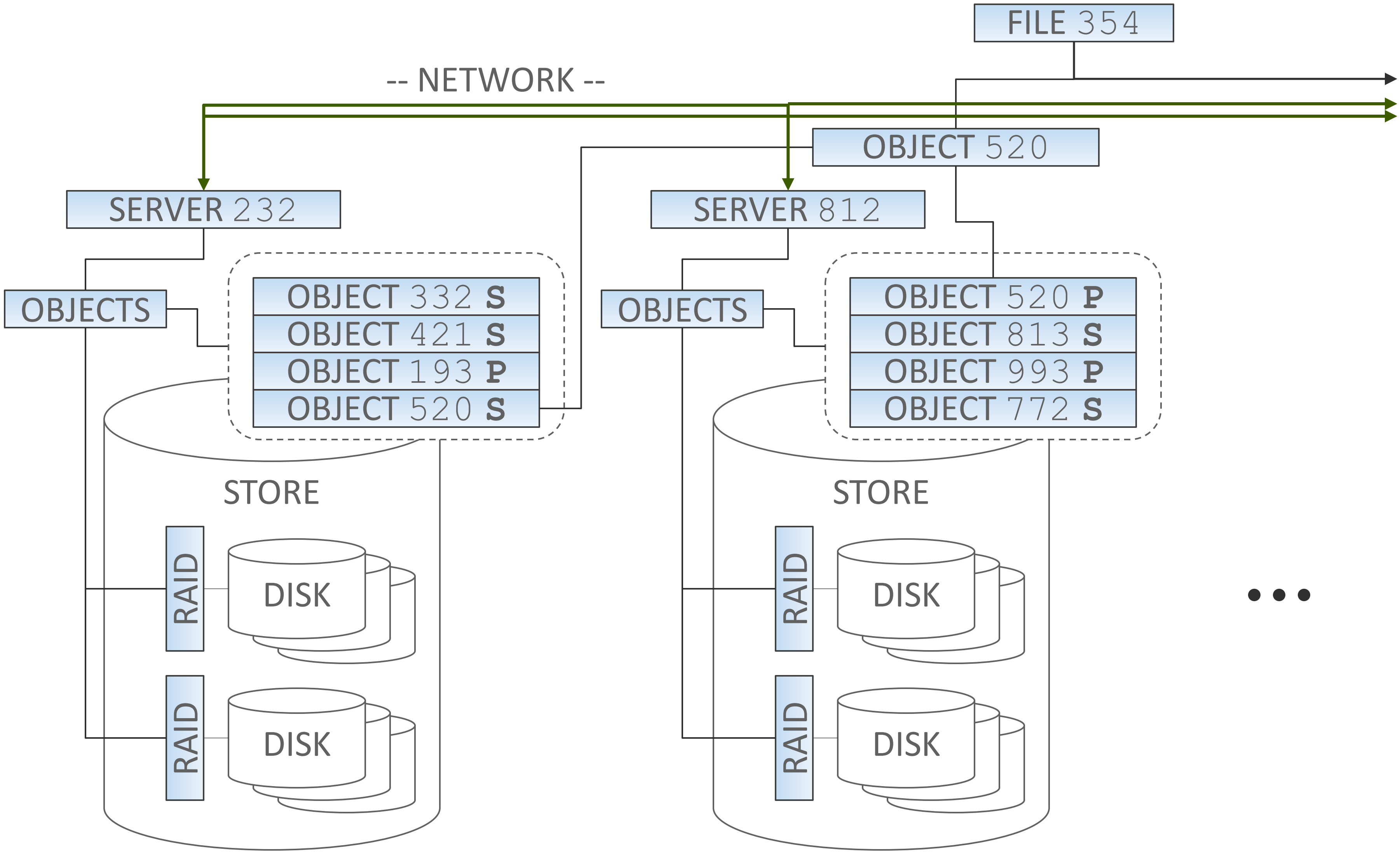
- RUSH
  - (Honicky and Miller, 2003) – described distribution of remaining replicas after loss of one replica server
  - (Weil et. al., 2006) – evaluated efficiency of reorganization
- Kinesis
  - (MacCormick et. al., 2009) – evaluated object load balancing of object placement, user accesses and rebuild parallelism
- PIO-SIM
  - (Bagrodia et. al., 1997) – analyze MPI-IO strategies such as collective operations, two-phase I/O, and cooperative caches
- PFSS
  - (Speth, 2005) – simulated PVFS with RAID under various fault conditions

Problem statement

Exascale storage challenges

- Number of disks
  - Speed: to satisfy checkpoint requirements, will need ~30,000 disks
  - Capacity: may use additional storage hierarchy for space
- Required bandwidth
  - ~12 TB/s
  - New ability to manage many clients
- Redundancy
  - Must plan to lose up to 10% of disks per year
  - That's 263 TB/day; 3.125 GB/s
- (Power)

System architecture



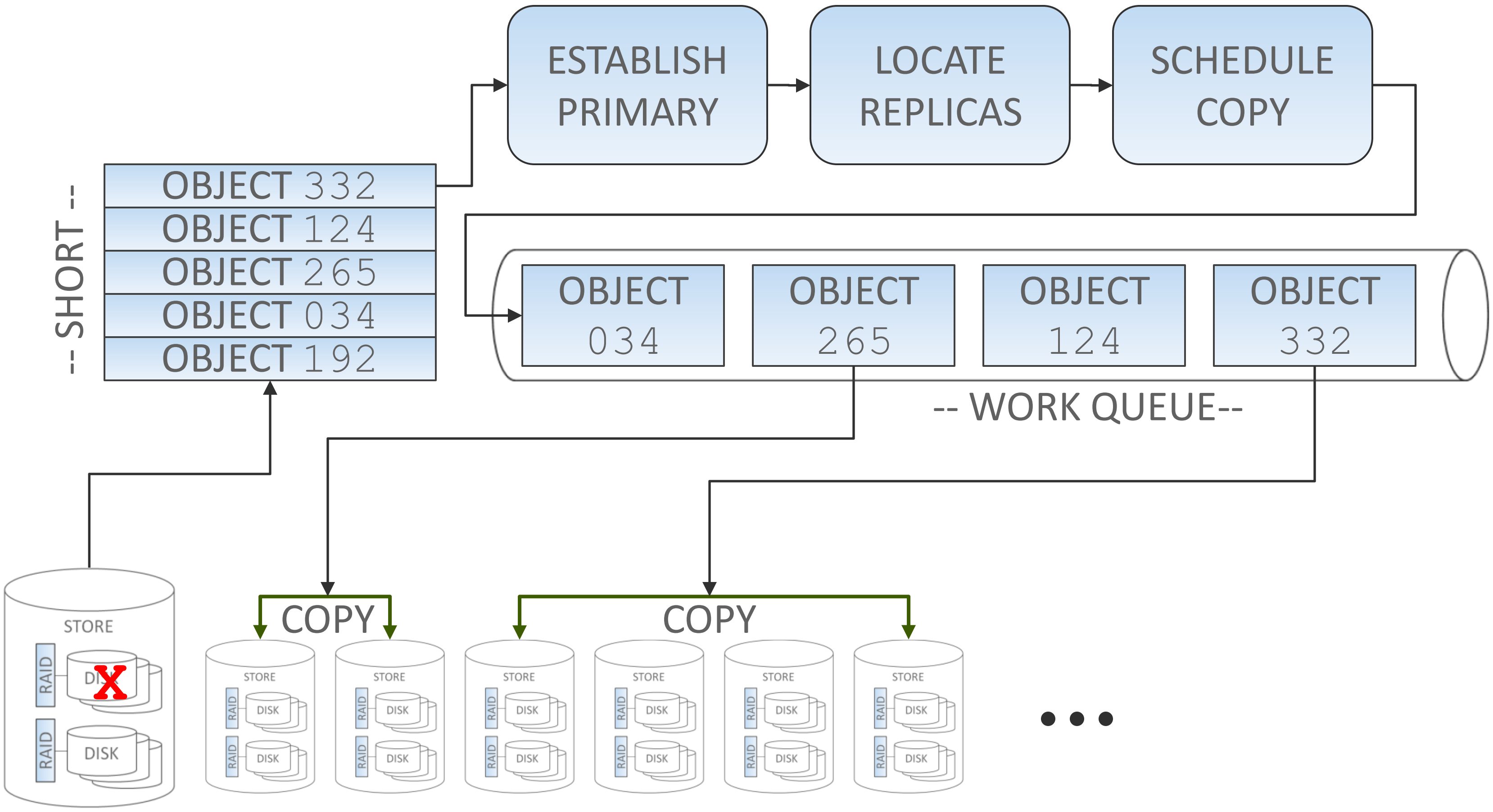
- Storage infrastructure modeled by GOBS

System design parameters

- Setup
  - `B` : address bit-length
  - `files` : number of files
  - `file.*` : file generation parameters, including `file.width`
  - `nodes` : number of servers (~700)
- Disks
  - `disk.size` : bytes
  - `disk.speed` : bytes/second
- User accesses
  - `reads`
  - `writes`
- Faults
  - `disk.mttf` : seconds
  - `mttc` : seconds
  - `mttr.reboot` : seconds (~1hr)
  - `mttr.disk` : seconds (~24hr)
- Redundancy
  - `replica.source` : (primary, etc.)
  - `file.replicas` : (~3)

RAID Reliability Formula
Chen et. al., 1994

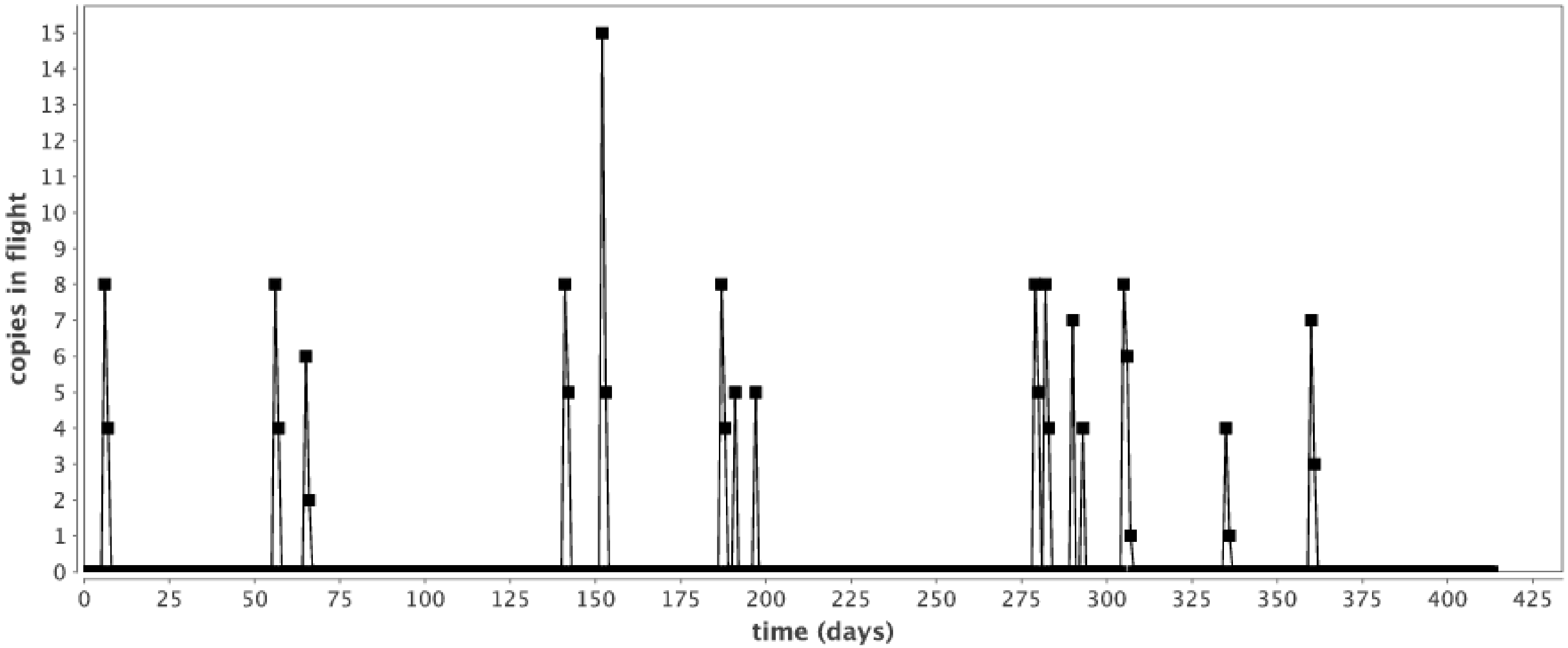
Fault response model



- Object copies scheduled by replica management routines
- One copy per server in flight

Disk failure rates

- CMU study
  - Typically ~5%/year
  - Up to 13%
- Google study
  - Below 5% in first year
  - Peaks near 10% in year 3



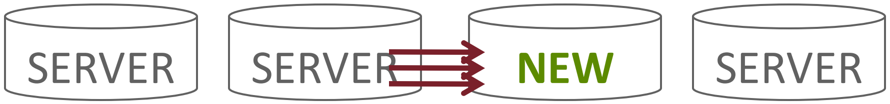
- GOBS simulation of 32,000 disks in RAID 5 (4+1)
  Plot shows inter-node traffic due to RAID loss

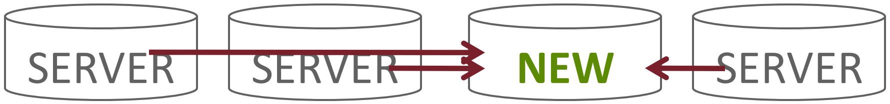
Simple data placement is problematic

- Combine local RAID with inter-node replication for availability
- Local RAID is relatively faster for read-modify-write operations
- Whole node loss – often temporary – managed with replicas

- Replica chaining
  - Simple, localized object placement
  - On rebuild, creates a hot spot of activity

- Large declustered RAIDs
  - Fully distributed
  - On rebuild, all nodes involved, all write to one new disk

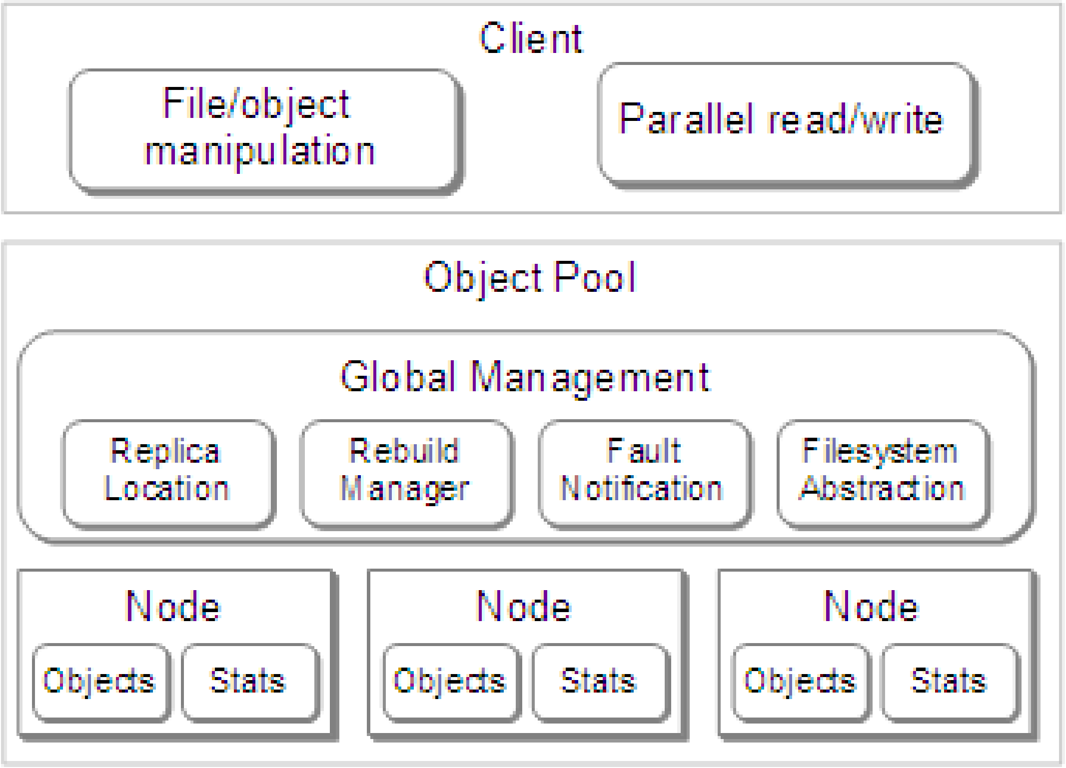




GOBS Simulator

Simulation as initial approach

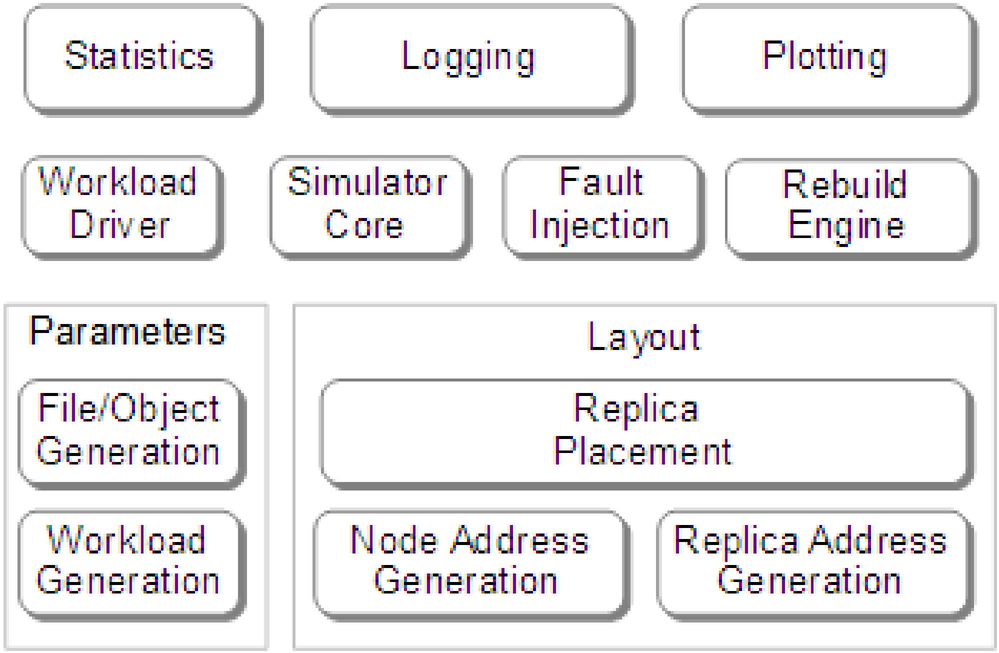
- Simulated system



- Workload simulation
- Idealized control
- Object servers

Software abstractions

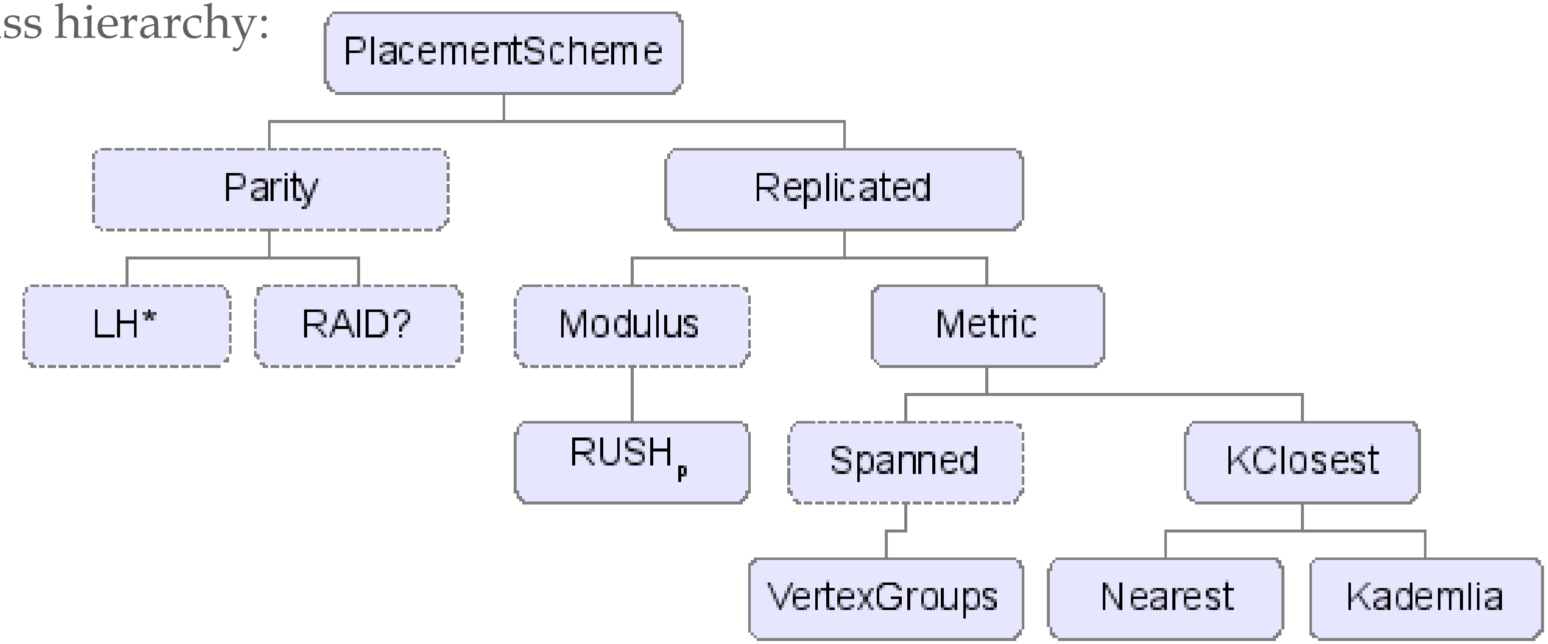
- General OBject Space (GOBS) simulator architecture



- User interface
- Core functionality
- Replaceable components

Simulator - extensibility

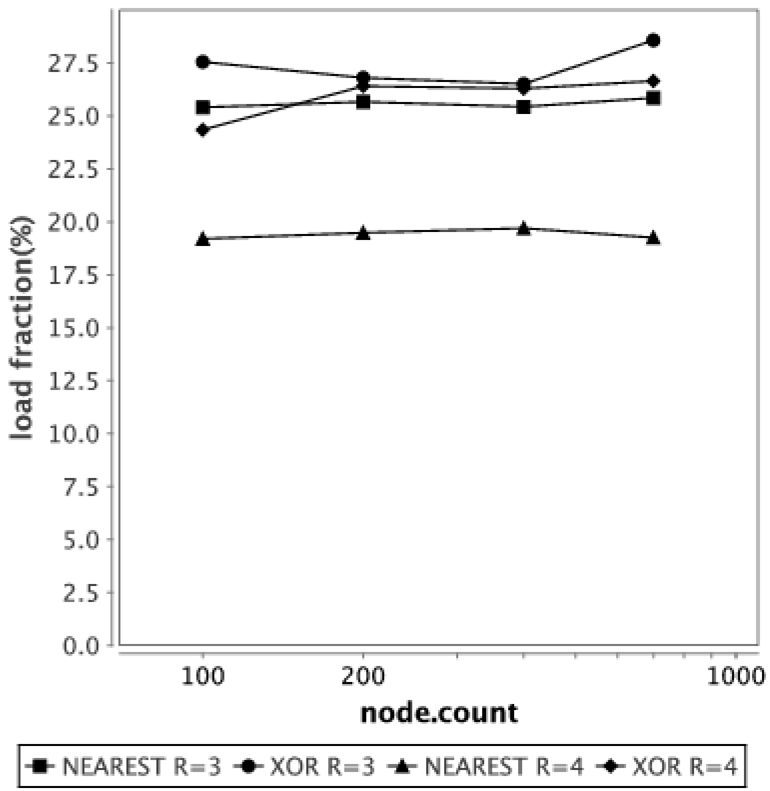
- Extensible Java simulator
  - Heavy use of inheritance
  - Enable easy implementation of new schemes
- Class hierarchy:



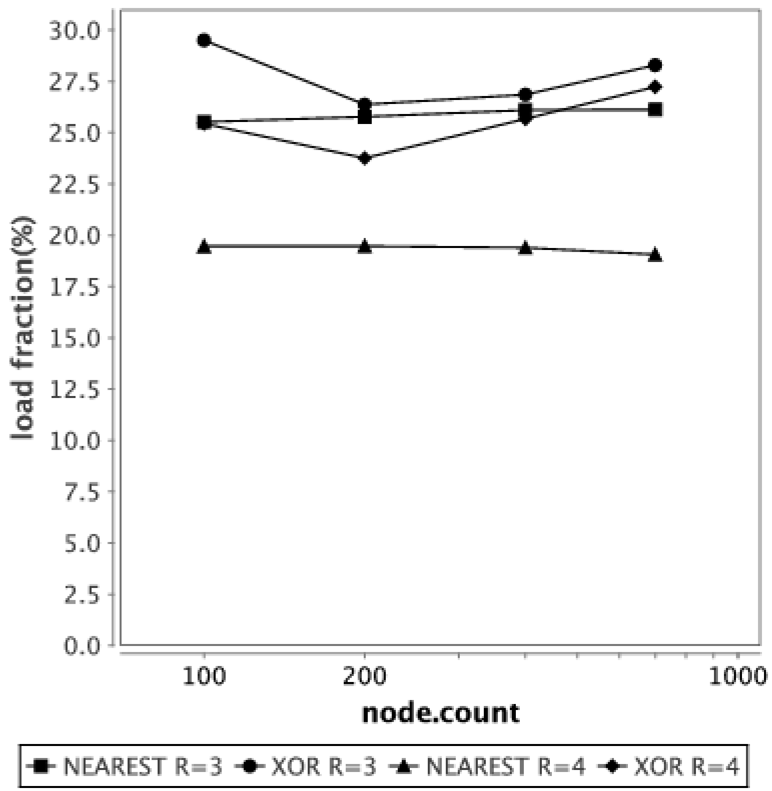
Preliminary results

GOBS results – rebuild hot spots

- 600 servers; 30 TB disks; RAID 5 (4+1); disk transfer rate 400 MB/s;
- 1EB filesystem
- Single fault induced – rebuild performed



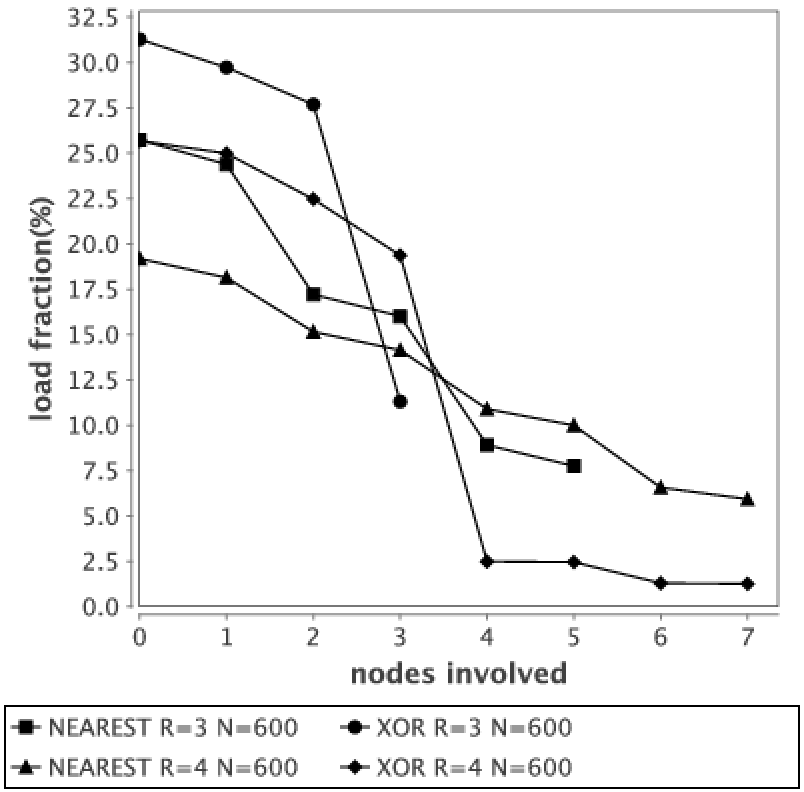
- Replica pulled from last in chain

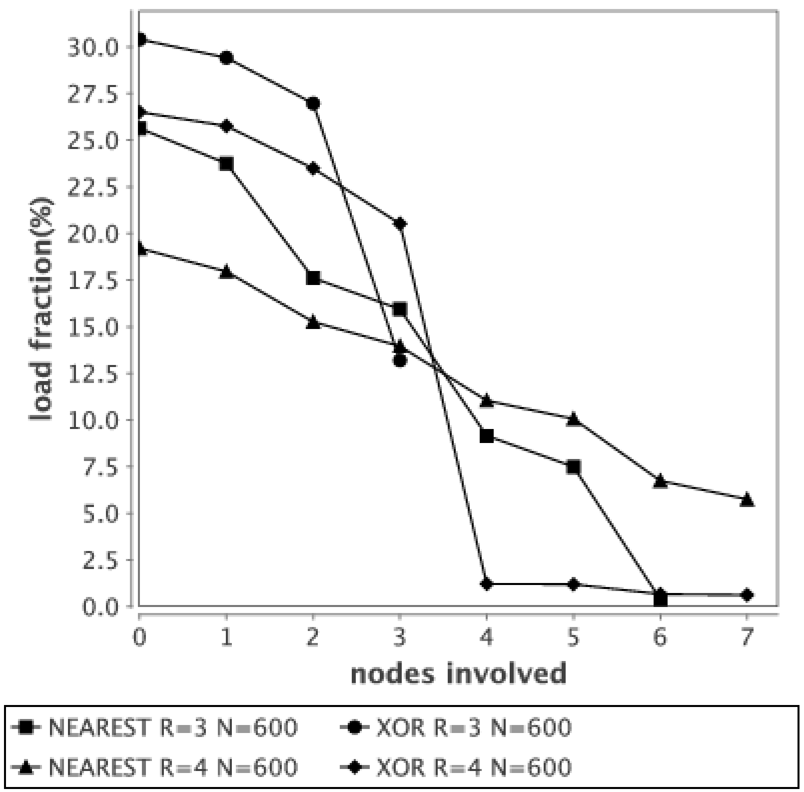


- Replica pulled from random node

GOBS results – rebuild curves

- Single fault induced – rebuild performed

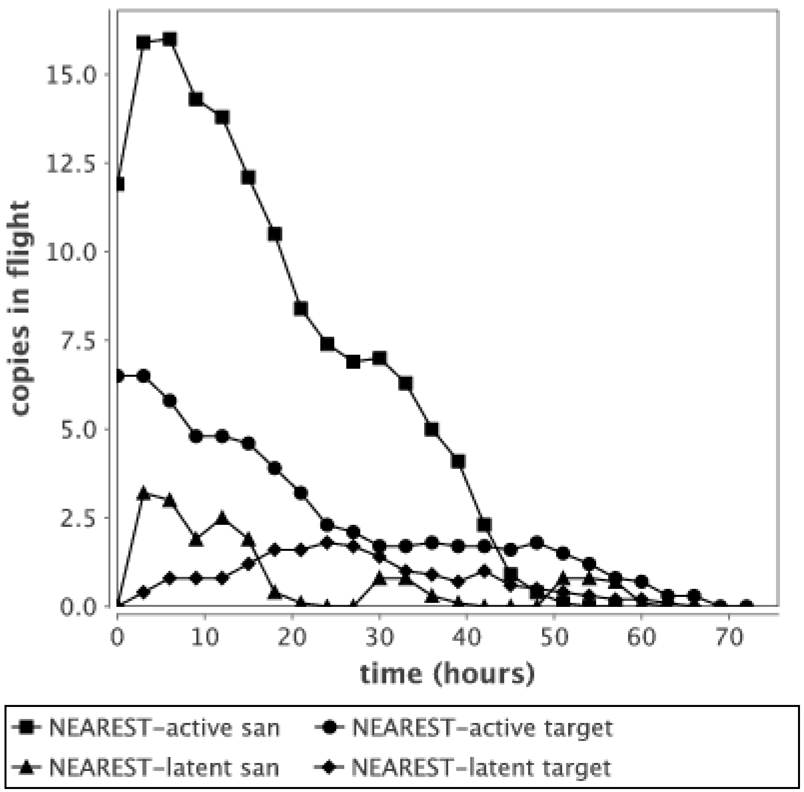




- Replica pulled from last in chain
- Replica pulled from random node

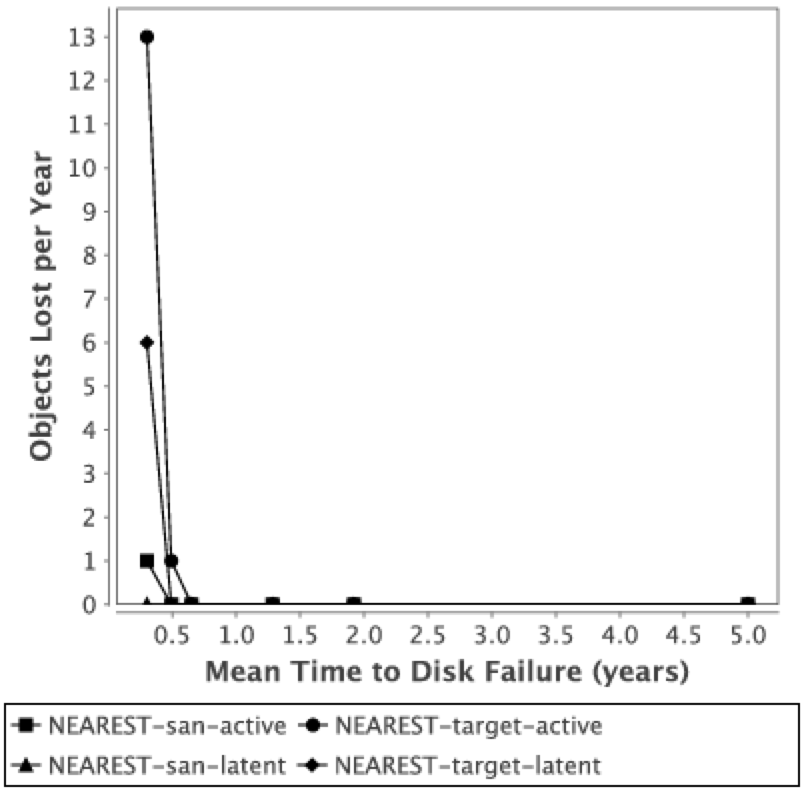
GOBS results – rebuild concurrency

- Multiple faults induced – average traffic recorded
- Replica pulled from primary
- “target” – RAID (4+1)
- “san” – RAID (8+2)
- “active” – begin copies immediately
- “latent” – wait until replacement is inserted



GOBS results – data loss

- Vary disk MTTF and report objects lost per year
- Neither scheme loses data unless MTTFs are extremely low
- Indicates that aggressive schemes may be used that favor user accesses
- (How does one quantify amount of data loss?)



GOBS: Summary

- Data placement strategies matter when performing rebuilds
- Rebuild time matters over long data lifetimes
- Simulation can help evaluate placement strategies
- Much more to do here…

Thanks

- Grants:
  This research is supported in part by NSF grant OCI-721939, NIH grants DC08638 and DA024304-02, the Office of Advanced Scientific Computing Research, Office of Science, U.S. Dept. of Energy under Contracts DE-AC02-06CH11357 and DE-AC02-06CH11357. Work is also supported by DOE with agreement number DE-FC02-06ER25777.

Questions