Portable, Scalable, and High-Performance I/O Forwarding on Massively Parallel Systems

Jason Cope
copej@mcs.anl.gov

Computation and I/O Performance Imbalance

- Leadership-class computational scale:
  - >100,000 processes
  - Multi-core architectures
  - Lightweight operating systems on compute nodes
- Leadership-class storage scale:
  - >100 servers
  - Cluster file systems
  - Commercial storage hardware
- Compute and storage imbalance in current leadership-class systems hinders application I/O performance
  - 1 GB/s of storage throughput for every 10TF of computation performance gap
  - The gap has increased by a factor of 10 in recent years

DOE FastOS2 I/O Forwarding Scalability Layer (IOFSL) Project

Goal: Design, build, and distribute a scalable, unified high-end computing I/O forwarding software layer that would be adopted by the DOE Office of Science and NNSA.

- Reduce the number of file system operations that the parallel file system handles
- Provide function shipping at the file system interface level
- Offload file system functions from simple or full OS client processes to a variety of targets
- Support multiple parallel file system solutions and networks
- Integrate with MPI-IO and any hardware features designed to support efficient parallel I/O

Outline

- I/O Forwarding Scalability Layer (IOFSL) Overview
- IOFSL Deployment on Argonne's IBM Blue Gene/P Systems
- IOFSL Deployment on Oak Ridge's Cray XT Systems
- Optimizations and Results
  - Pipelining in IOFSL
  - Request Scheduling and Merging in IOFSL
  - IOFSL Request Processing
- Future Work and Summary

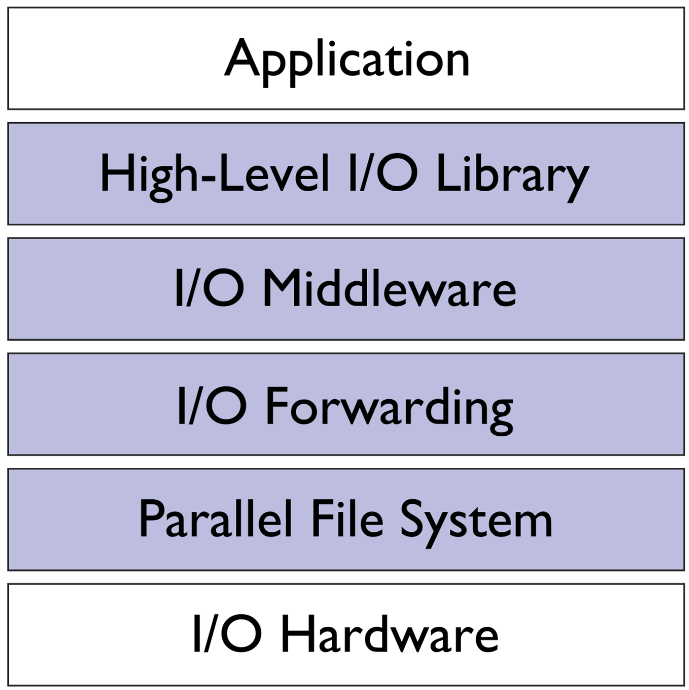
HPC I/O Software Stack

High-Level I/O Library
maps application abstractions onto storage abstractions and provides data portability.

HDF5, Parallel netCDF, ADIOS

I/O Forwarding
bridges between app. tasks and storage system and provides aggregation for uncoordinated I/O.

IBM ciod



I/O Middleware
organizes accesses from many processes, especially those using collective I/O.

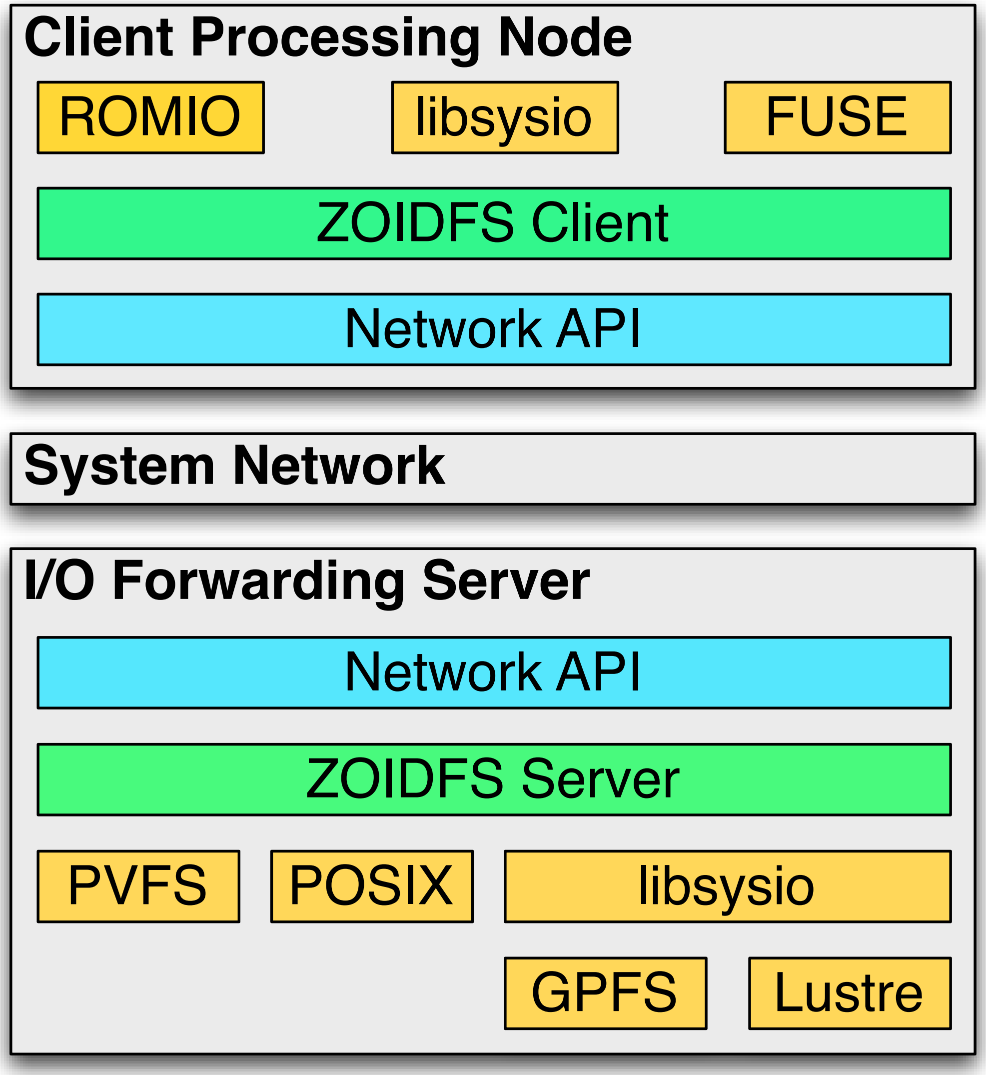
MPI-IO

Parallel File System
maintains logical space and provides efficient access to data.

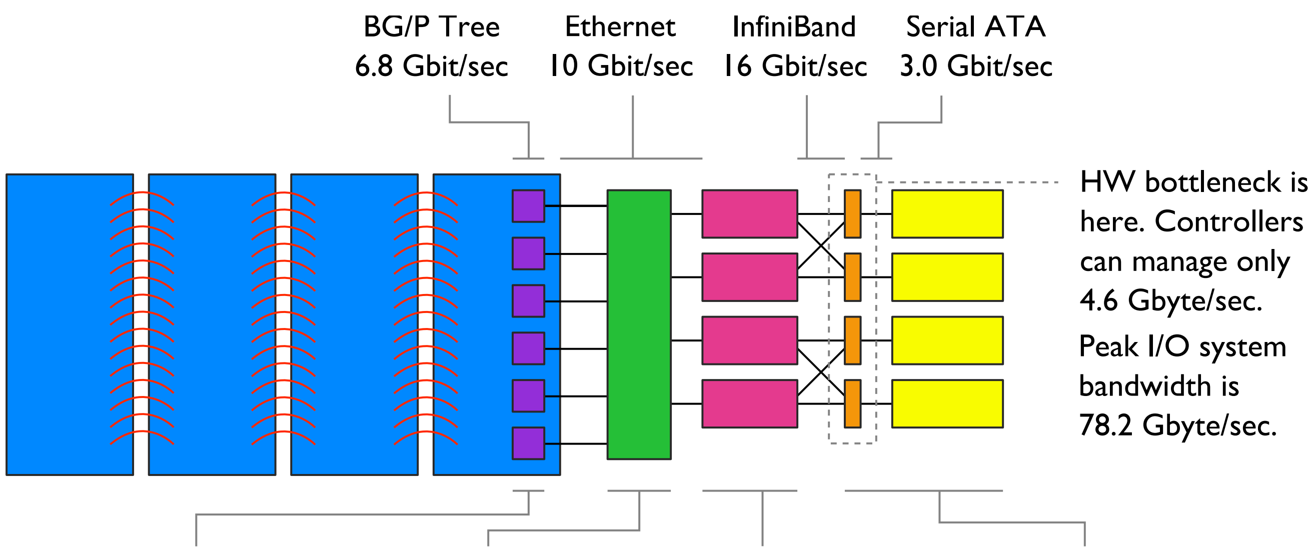
PVFS, PanFS, GPFS, Lustre

IOFSL Architecture

- Client
  - MPI-IO using ZoidFS ROMIO interface
  - POSIX using libsysio or FUSE
- Network
  - Transmit message using BMI over TCP / IP, MX, IB, Portals, and ZOID
  - Messages encoded using XDR
- Server
  - Delegates IO to backend file systems using native drivers or libsysio



Argonne’s IBM Blue Gene/P Systems



Gateway nodes run parallel file system client software and forward I/O operations from HPC clients.

640 Quad core PowerPC 450 nodes with 2 Gbytes of RAM each

Commodity network primarily carries storage traffic.

900+ port 10 Gigabit Ethernet Myricom switch complex

Storage nodes run parallel file system software and manage incoming FS traffic from gateway nodes.

136 two dual core Opteron servers with 8 Gbytes of RAM each

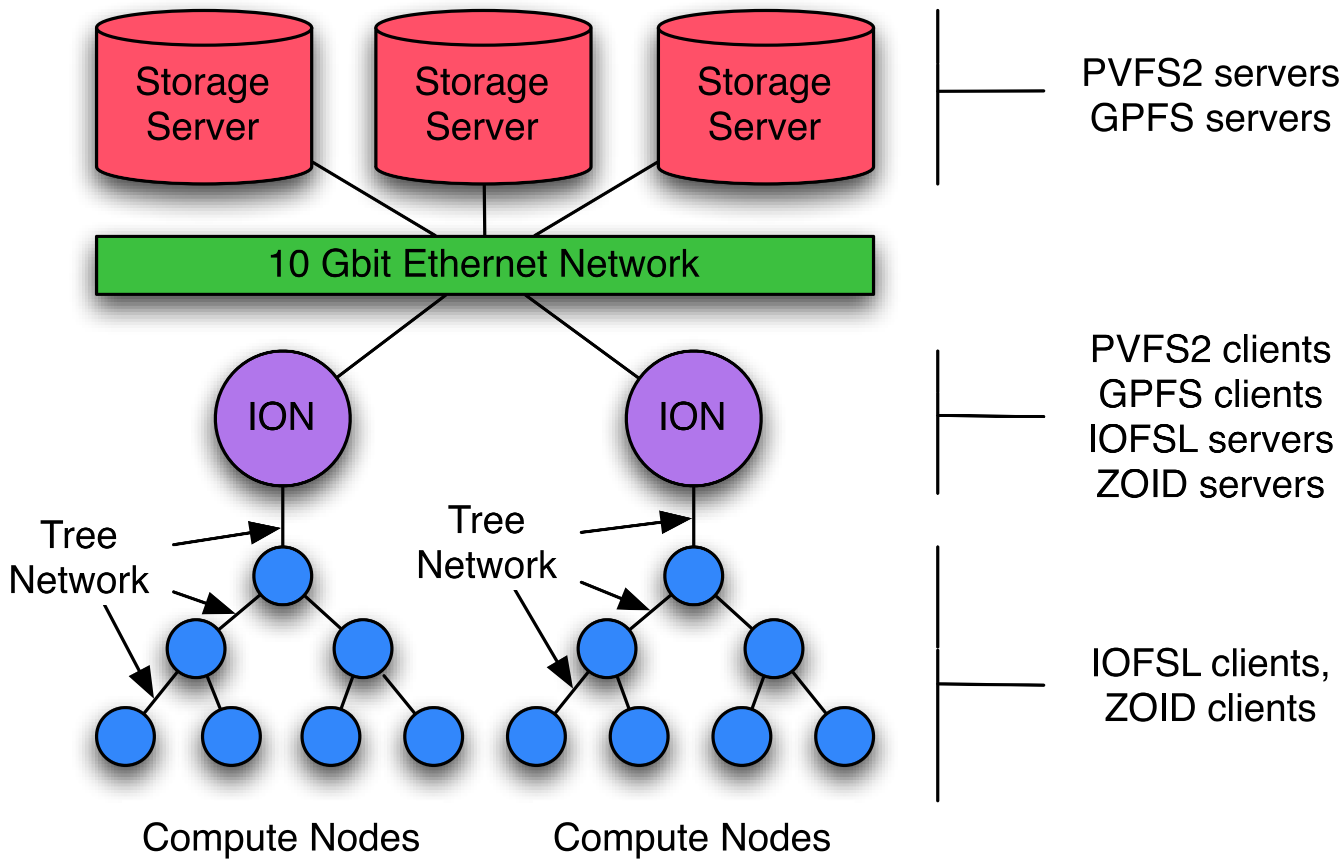
Enterprise storage controllers and large racks of disks are connected via InfiniBand or Fibre Channel.

17 DataDirect S2A9900 controller pairs with 480 1 Tbyte drives and 8 InfiniBand ports per pair

Architectural diagram of the 557 TFlop IBM Blue Gene/P system at the Argonne Leadership Computing Facility.

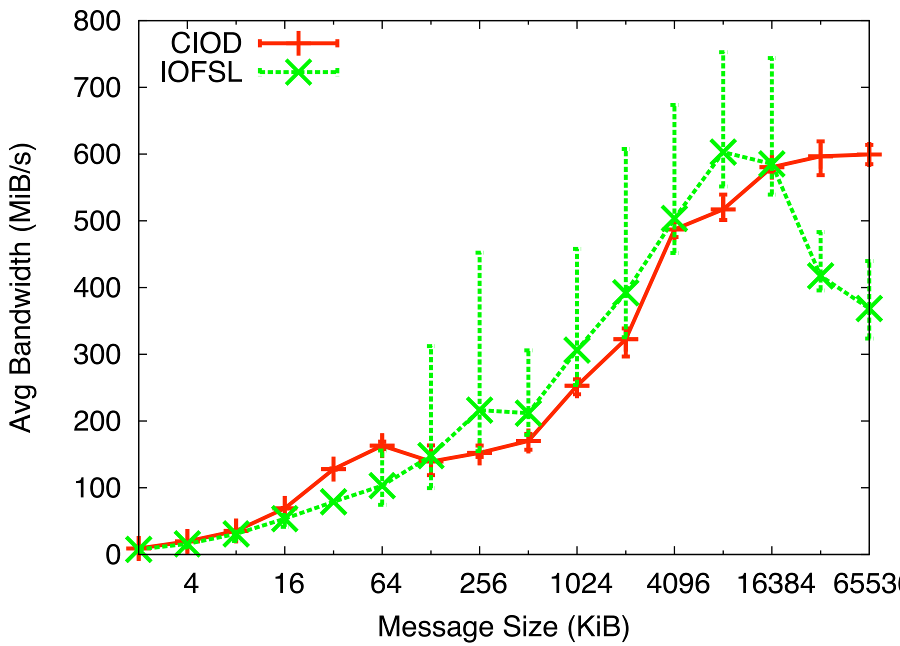
Figure Courtesy of Robert Ross, ANL

IOFSL Deployment on Argonne's IBM Blue Gene/P Systems

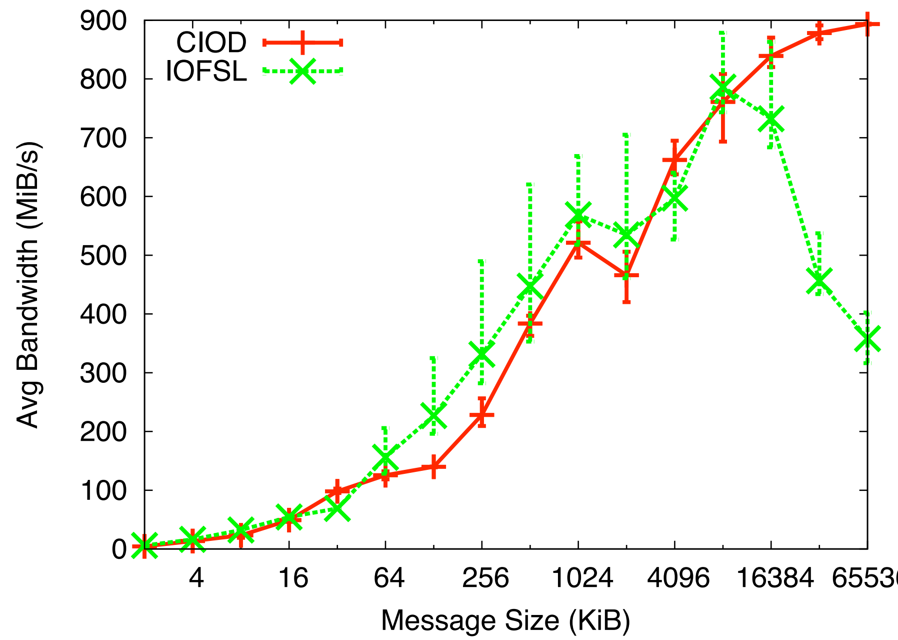


Initial IOFSL Results on Argonne's IBM Blue Gene/P Systems

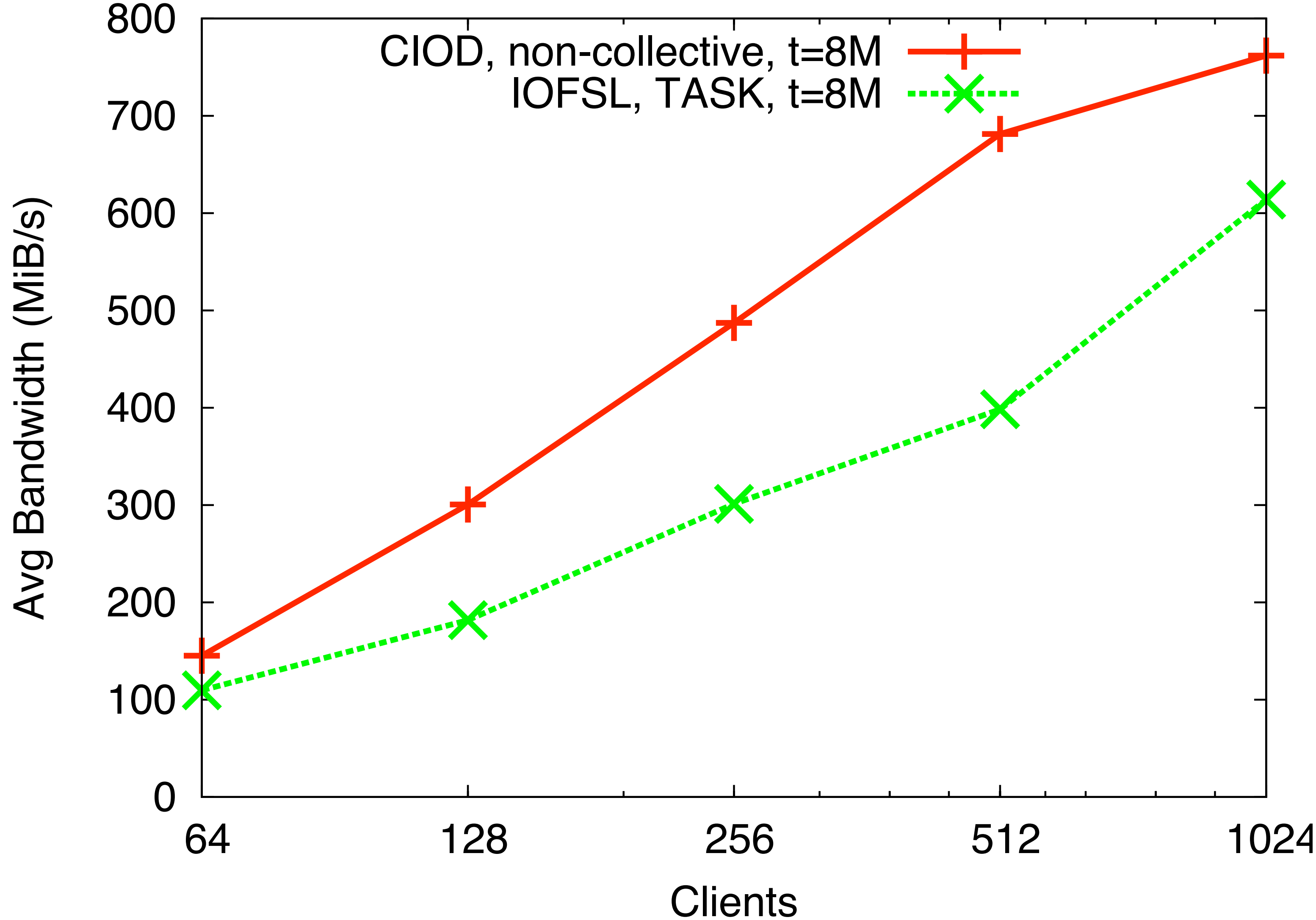
IOR Read



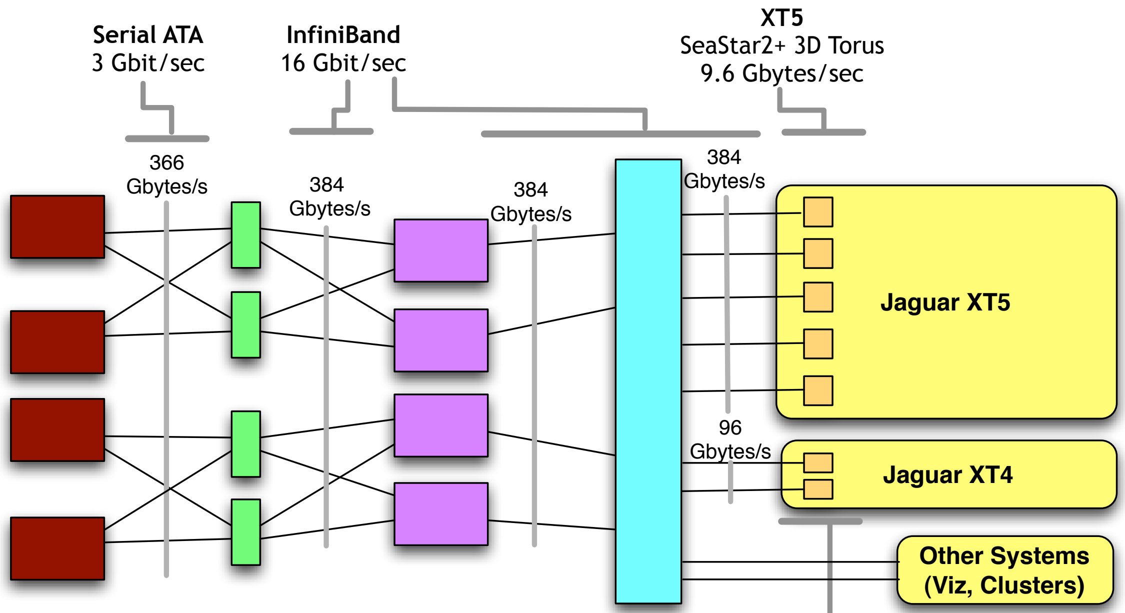
IOR Write



Initial IOFSL Results on Argonne's IBM Blue Gene/P Systems



Oak Ridge's Cray XT Systems



Enterprise Storage
controllers and large racks of disks are connected via InfiniBand.

48 DataDirect S2A9900 controller pairs with 1 Tbyte drives and 4 InifiniBand connections per pair

Storage Nodes
run parallel file system software and manage incoming FS traffic.

192 dual quad core Xeon servers with 16 Gbytes of RAM each

SION Network
provides connectivity between OLCF resources and primarily carries storage traffic.

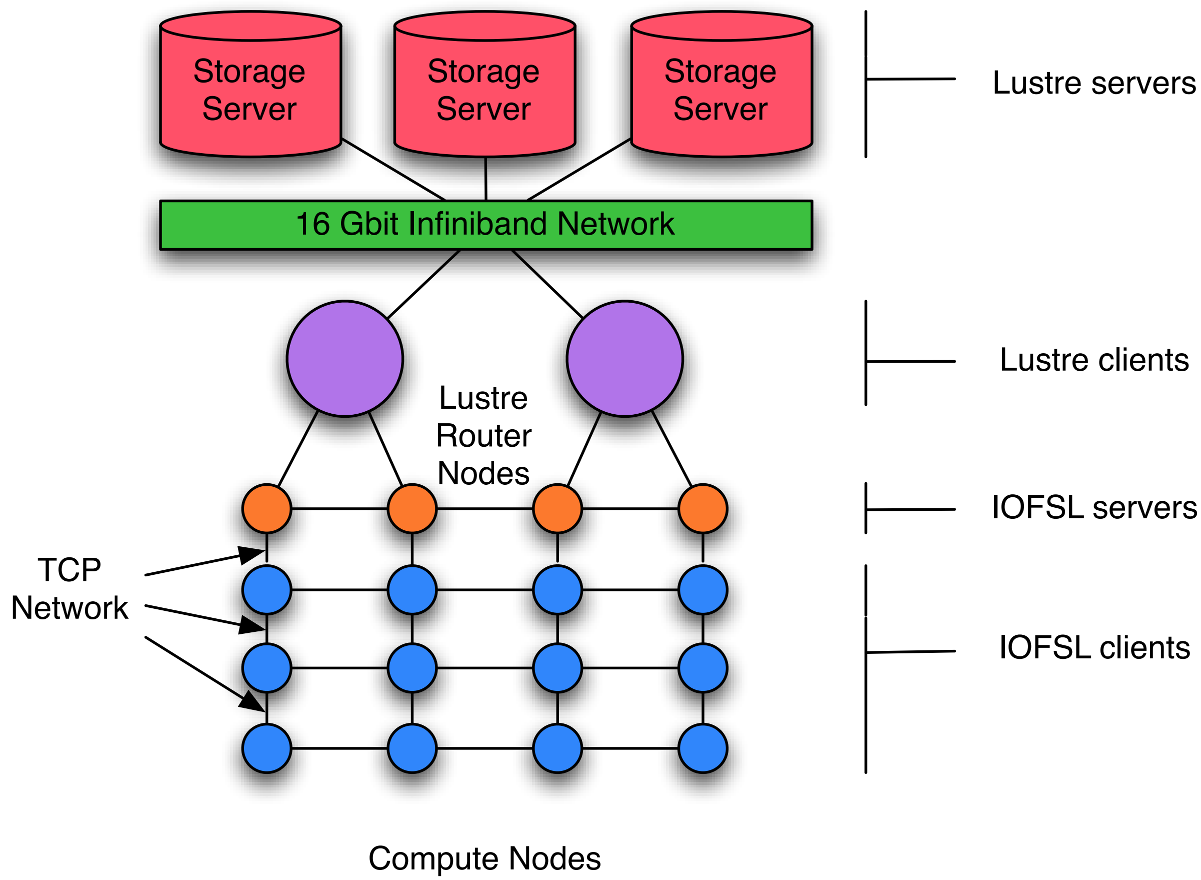
3000+ port 16 Gbit/sec InfiniBand switch complex

Lustre Router Nodes
run parallel file system client software and forward I/O operations from HPC clients.

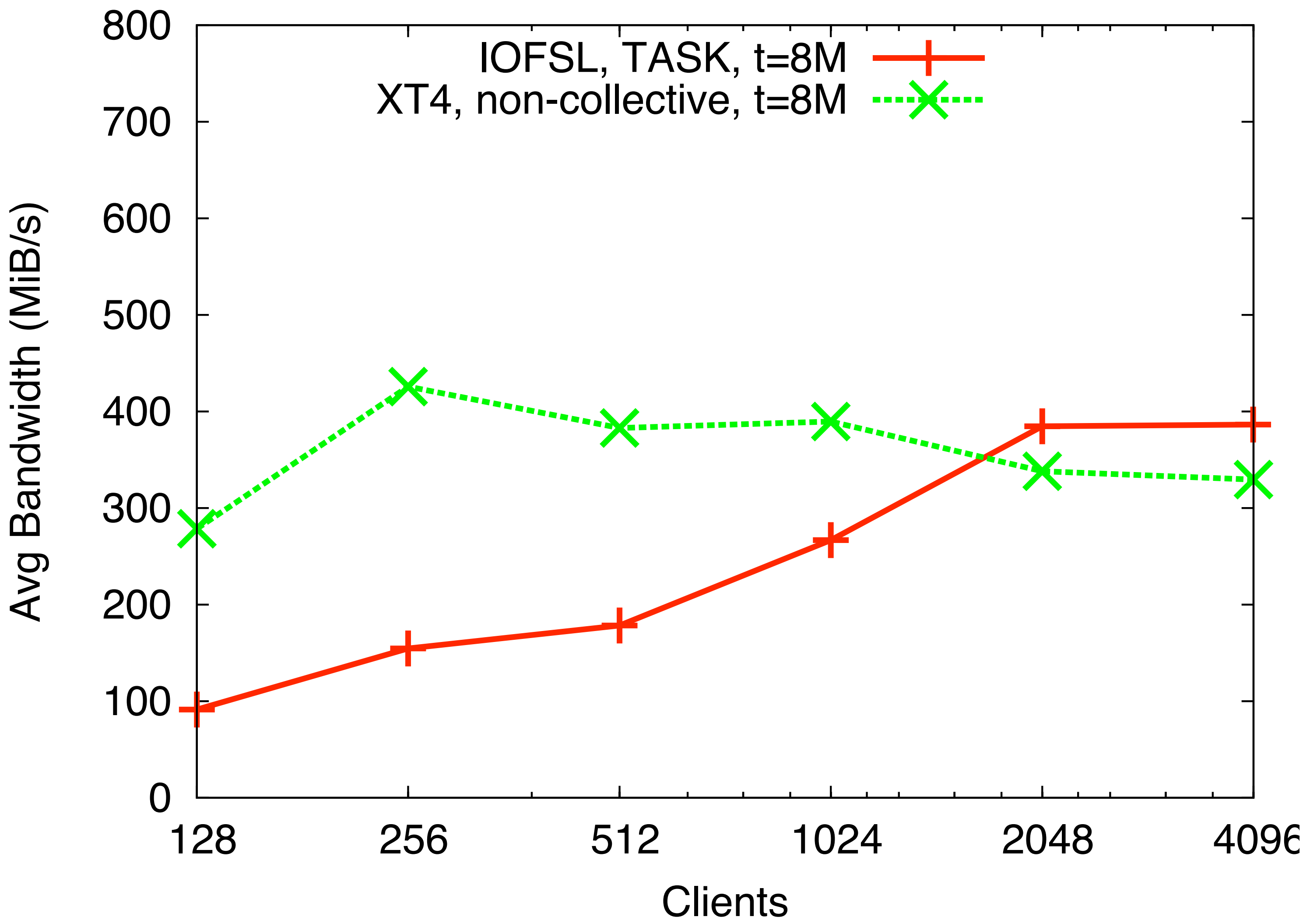
192 (XT5) and 48 (XT4) one dual core Opteron nodes with 8 GB of RAM each

Figure Courtesy of Galen Shipman, ORNL

IOFSL Deployment on Oak Ridge's Cray XT Systems



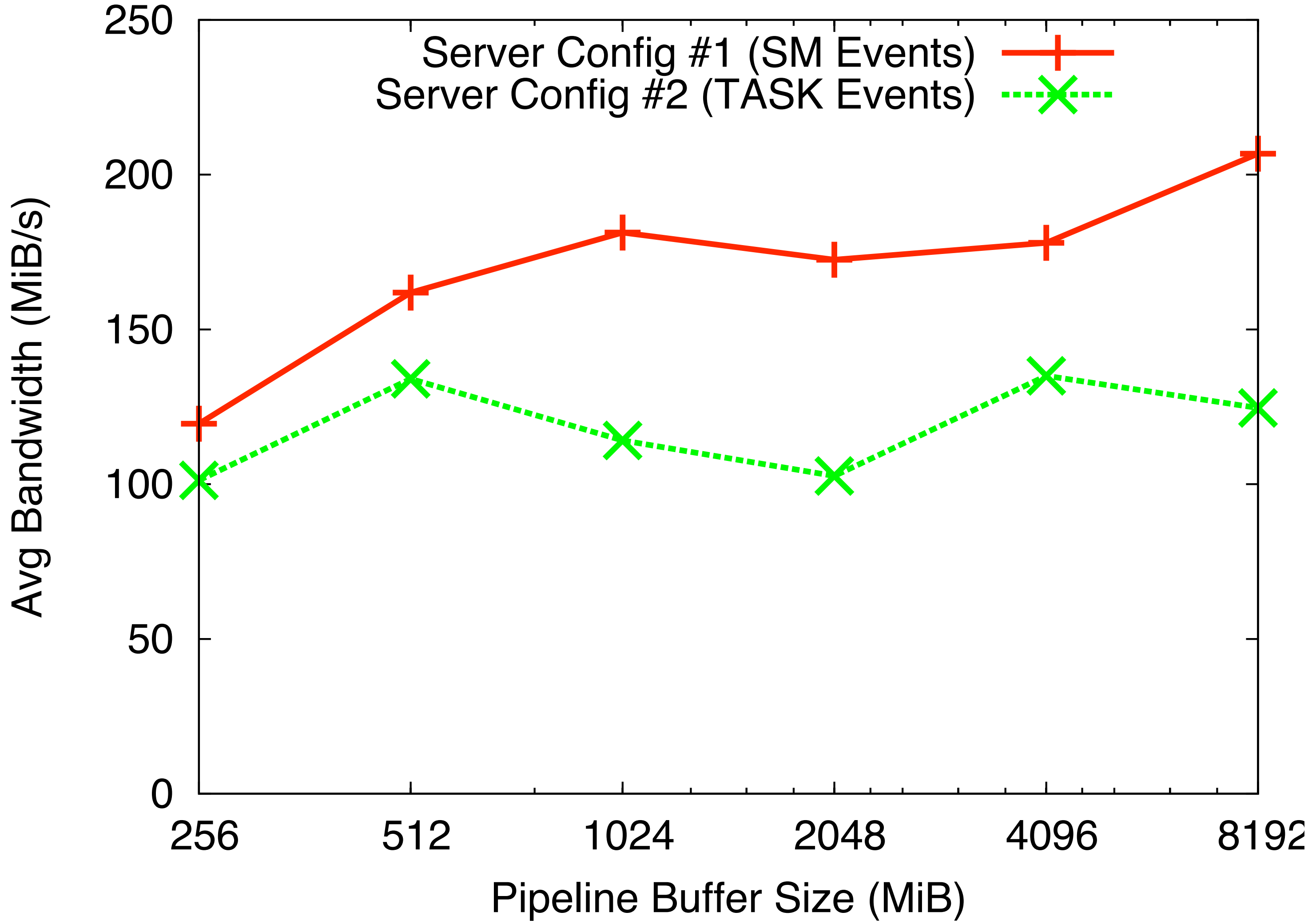
Initial IOFSL Results on Oak Ridge's Cray XT Systems



IOFSL Optimization #1: Pipeline Data Transfers

- Motivation
  - Limits on the amount of memory available on I/O nodes
  - Limits on the amount of posted network operations
  - Need to overlap network operations and file system operation for sustained throughput
- Solution: Pipeline data transfers between the IOFSL client and server
  - Negotiate the pipeline transfer buffer size
  - Data buffers are aggregated or segmented at the negotiated buffer size
  - Issue network transfer requests for each pipeline buffer
  - Reformat pipeline buffers into the original buffer sizes
- Currently serial and parallel pipeline modes

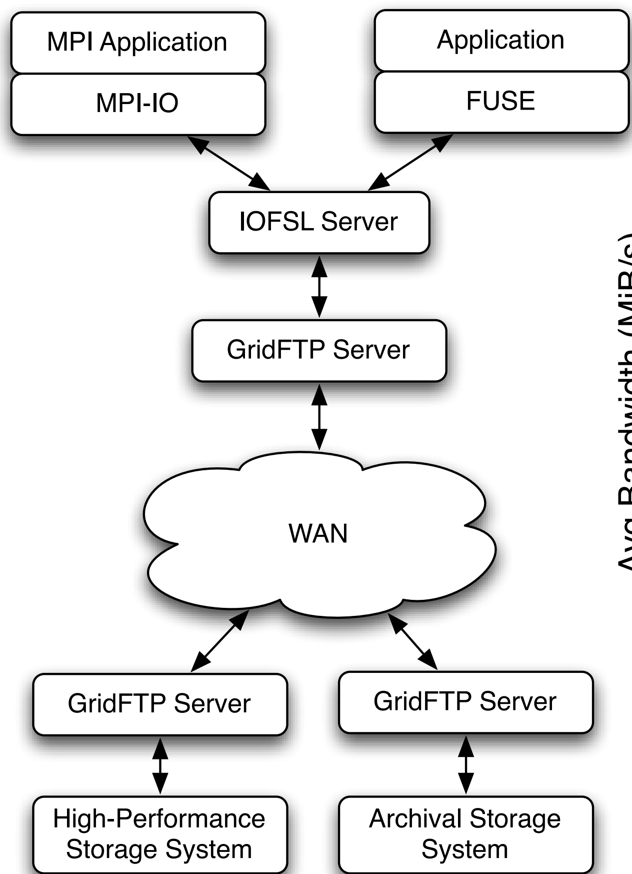
Pipeline Data Transfer Results for Different IOFSL Server Configurations

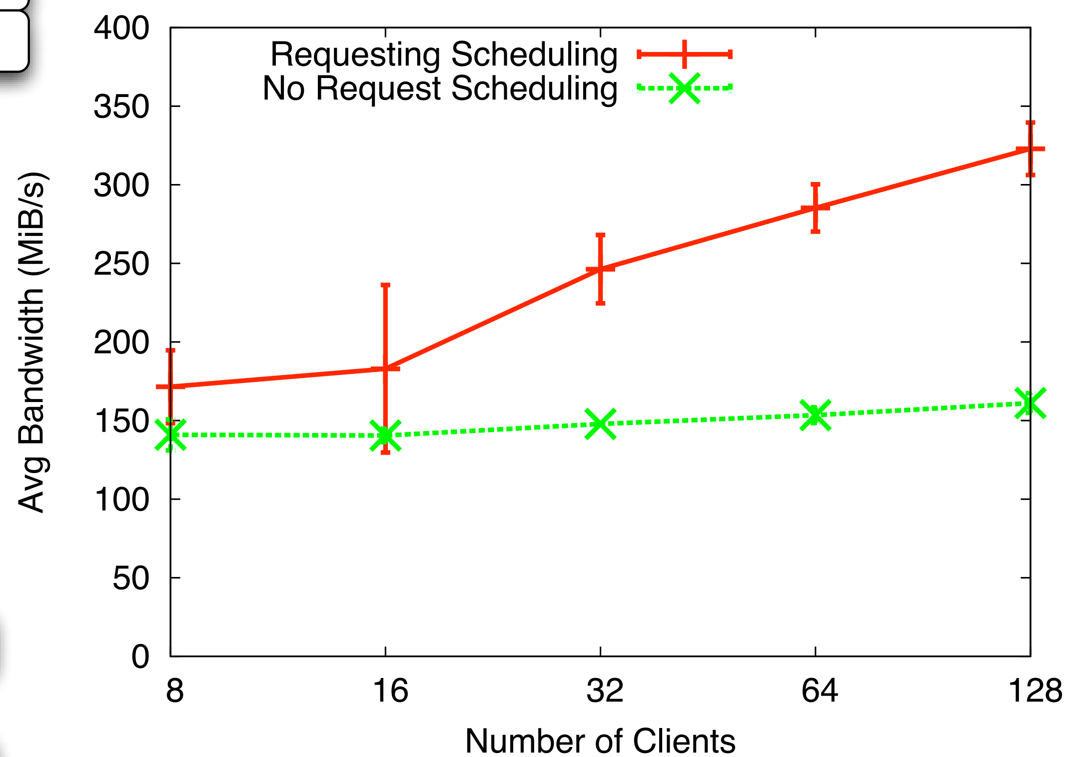


IOFSL Optimization #2: Request Scheduling and Merging

- Request scheduling aggregates several requests into a bulk IO request
  - Reduces the number of client accesses to the file systems
  - With pipeline transfers, overlaps network and storage IO accesses
- Two scheduling modes supported
  - FIFO mode aggregates requests as they arrive
  - Handle-Based Round-Robin (HBRR) iterates over all active file handles to aggregate requests
- Request merging identifies aggregates noncontiguous requests into contiguous requests
  - Brute Force mode iterates over all pending requests
  - Interval Tree mode compares requests that are on similar ranges

IOFSL Request Scheduling and Merging Results with the IOFSL GridFTP Driver

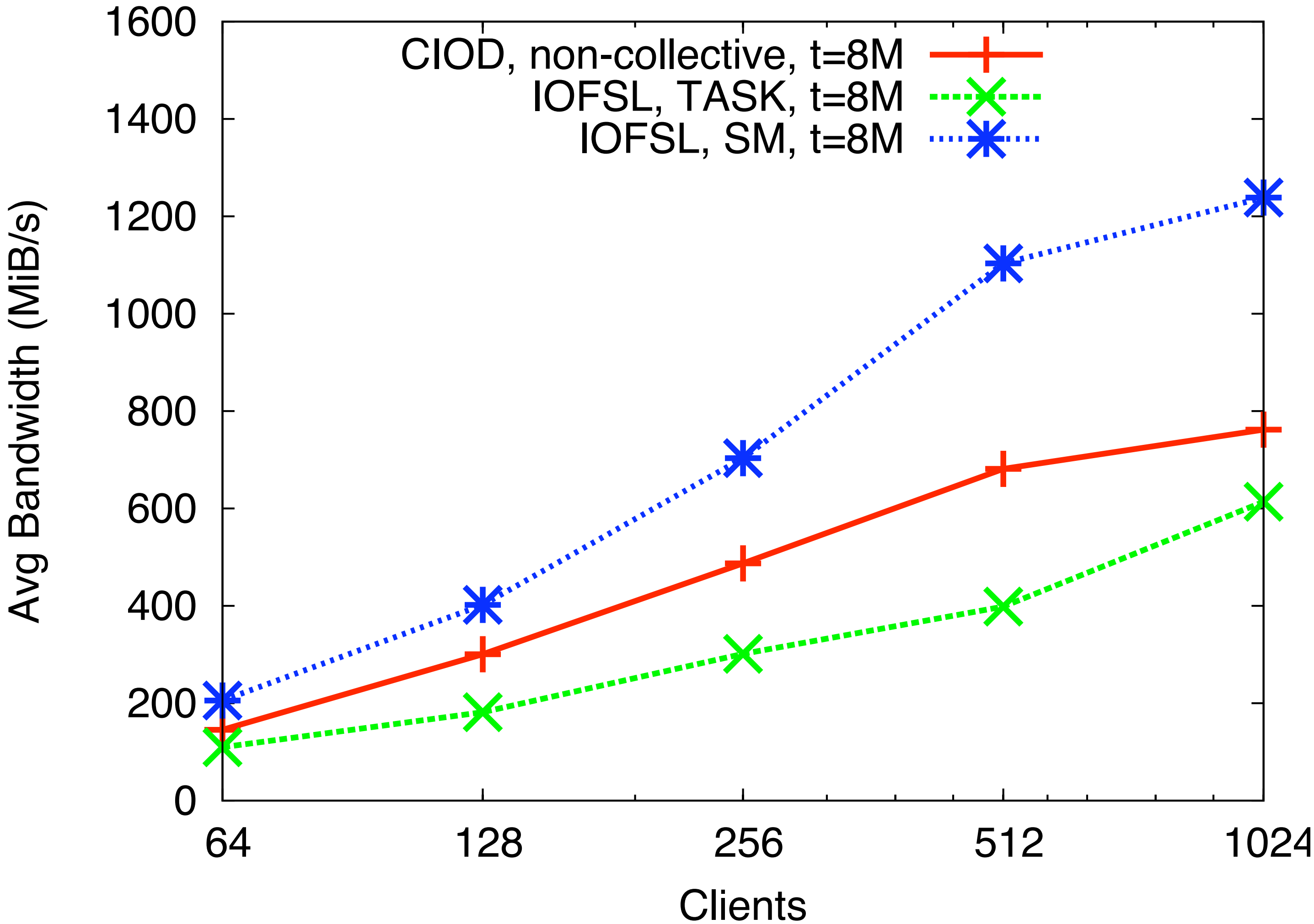




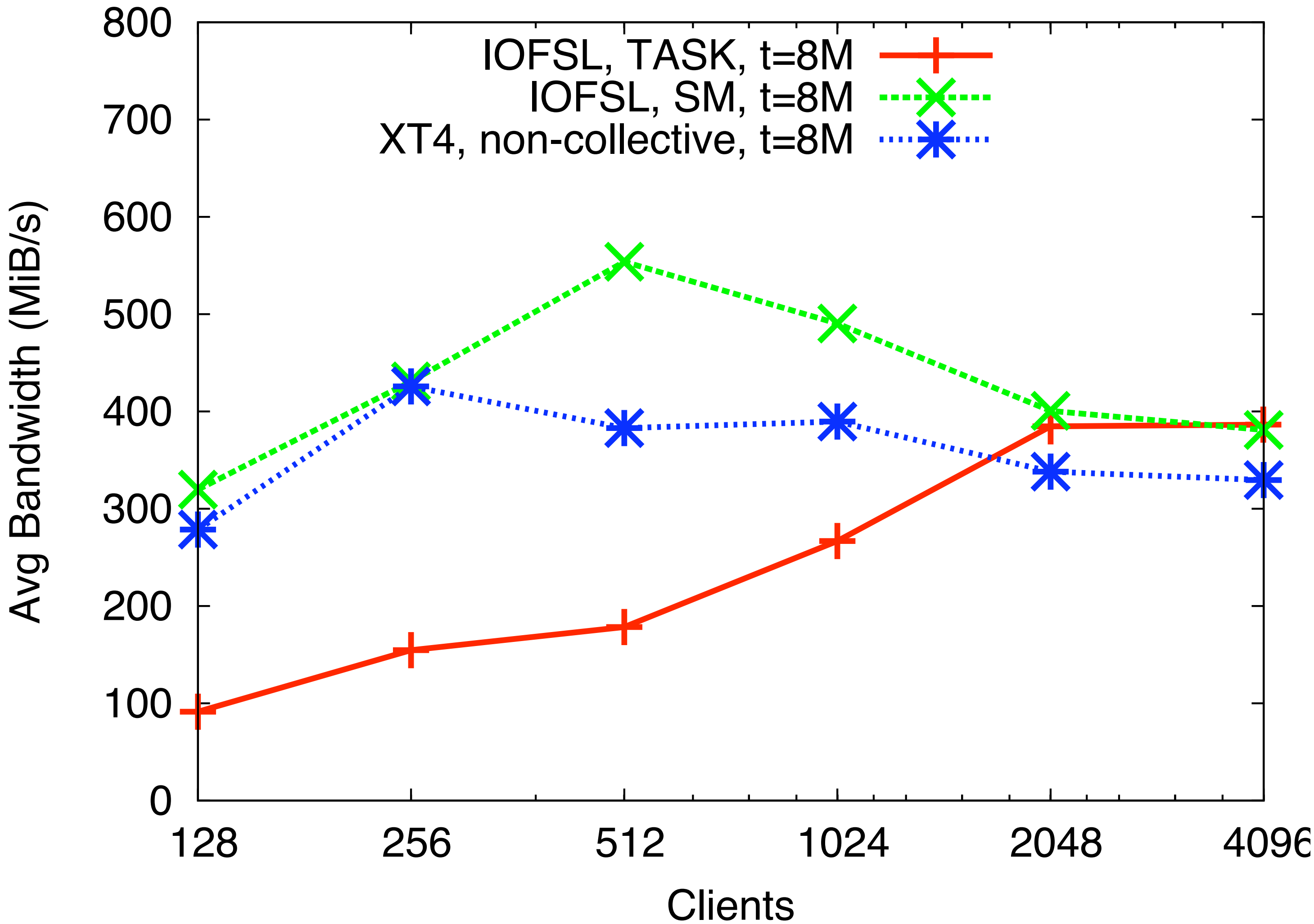
IOFSL Optimization #3: Request Processing and Event Mode

- Multi-Threaded Task Mode
  - New thread for executing each IO request
  - Simple implementation
  - Thread contention and scalability issues
- State Machine Mode
  - Use a fixed number of threads from a thread pool to execute IO requests
  - Divide IO requests into smaller units of work
  - Thread pools schedules IO requests to run non-blocking units of work (data manipulation, pipeline calculations, request merging)
  - Yield execution of IO requests on blocking resource accesses (network communication, timer events, memory allocations)

IOFSL Request Processing and Event Mode: Argonne's IBM Blue Gene/P Results



IOFSL Request Processing and Event Mode: Oak Ridge's Cray XT4 Results



Current and Future Work

- Scaling and tuning of IOFSL on IBM BG/P and Cray XT systems
- Collaborative caching layer between IOFSL servers
- Security infrastructure
- Integrating IOFSL with end-to-end I/O tracing and visualization tools for the NSF HECURA IOVIS / Jupiter project

Project Participants and Support

- **Argonne National Laboratory:** Rob Ross, Pete Beckman, Kamil Iskra, Dries Kimpe, Jason Cope
- **Los Alamos National Laboratory:** James Nunez, John Bent, Gary Grider, Sean Blanchard, Latchesar Ionkov, Hugh Greenberg
- **Oak Ridge National Laboratory:** Steve Poole, Terry Jones
- **Sandia National Laboratories:** Lee Ward
- **University of Tokyo:** Kazuki Ohta, Yutaka Ishikawa

- The IOFSL project is supported by the DOE Office of Science and NNSA.

IOFSL Software Access, Documentation, and Links

- IOFSL Project Website: http://www.iofsl.org
- IOFSL Wiki and Developers Website: http://trac.mcs.anl.gov/projects/iofsl/wiki
- Access to IOFSL Public git Repository:

  git clone http://www.mcs.anl.gov/research/projects/iofsl/git iofsl
- Recent publications
  - K. Ohta, D. Kimpe, J. Cope, K. Iskra, R. Ross, and Y. Ishikawa, "Optimization Techniques at the I/O Forwarding Layer," IEEE Cluster 2010 (to appear).
  - D. Kimpe, J. Cope, K. Iskra, and R. Ross. "Grids and HPC: Not as Different as you might think," Para2010 mini-symposium on Real-time access and Processing of Large Data Sets, April 2010.
  -

Questions?

Jason Cope

copej@mcs.anl.gov