8-2001

Massive Data Processing on the Acxiom Cluster Testbed

Amy Apon
Clemson University, aapon@clemson.edu

Pawel Wolinski
University of Arkansas - Main Campus

Dennis Reed
University of Arkansas - Main Campus

Greg Amerson
University of Arkansas - Main Campus

Prathima Gorjala
University of Arkansas - Main Campus

Massive Data Processing on the Acxiom Cluster Testbed

Amy Apon, Pawel Wolinski, Dennis Reed
Greg Amerson, Prathima Gorjala
University of Arkansas

Commercial Applications of
High Performance Computing

The Acxiom Cluster Testbed

- A high performance cluster and cluster computing research project at the University of Arkansas
- Primary goal of the research is to investigate cluster computing hardware and middleware architectures for use in massive data processing (search, sort, match, ...)

The University of Arkansas
Fayetteville, AR

Outline of talk

- Motivation and application description
- Experimental study setup
  - Two cluster platforms
  - System and application software
  - Two file systems
  - Four workloads
- Results
- Future work

Sponsors

1) National Science Foundation
2) Arkansas Science and Technology Authority (ASTA)
3) Acxiom Corporation
   - Billion dollar corporation, based in Little Rock, Arkansas
   - Provides generous support to universities in Arkansas
   - Provides products and services for information integration

Application Characteristics

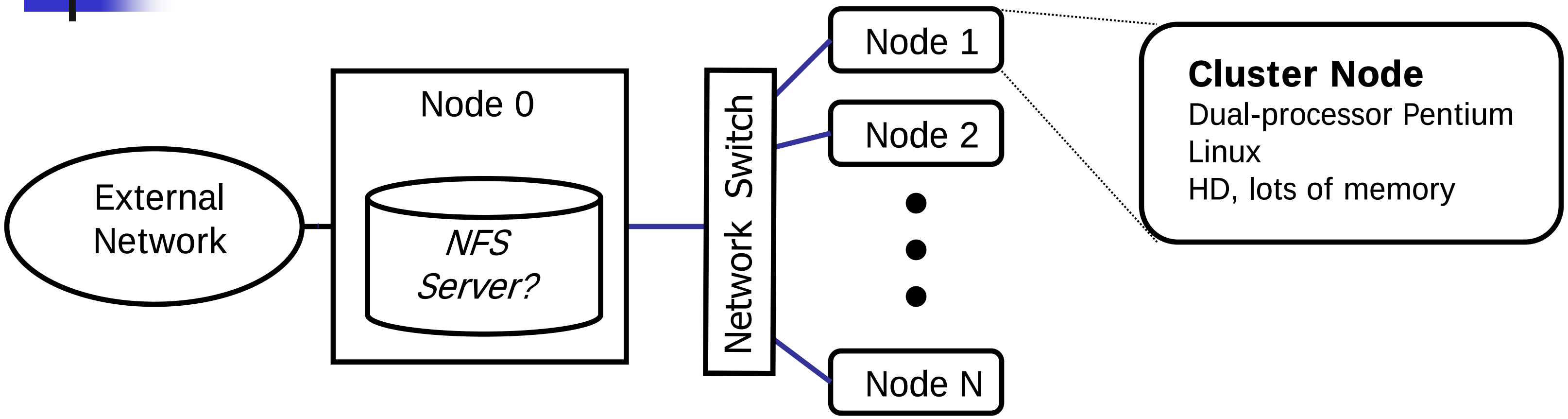
- The files are REALLY BIG. (>>100 GB)
  - Never underestimate the bandwidth of a Sentra carrying a hard drive and grad student across campus
- File access may be sequential through all or portions of a file
  - E.g., stepping through a list of all addresses in a very large customer file

Application Characteristics

- Or, file access may be “random”
  - E.g., reading the record of a particular customer number

➢ File cache may be ineffective for these types of workloads

Typical Cluster Architecture



- Network File System server?
  - Easy to configure
  - But, may not have good performance for Acxiom workload – even with a fast network and disks

Two Cluster Platforms

- The Eagle Cluster
  - Four single processor Pentium II, 450MHz computers and four Pentium III, 500MHz computers, Fast Ethernet (12.5MBps)
  - Node 0, NFS server, IDE HD tput≅19MBps
  - Nodes 1, 2, 3 IDE HD tput≅13MBps
  - Nodes 4, 5, 6 SCSI HD tput≅18MBps
  - Node 7 IDE HD tput≅18MBps

Two Cluster Platforms

- The ACT Cluster
  - Seven dual-processor Pentium III 1GHz computers
  - Dual EIDE disk RAID 0 subsystem in all nodes, tput≅60MBps
  - Both Fast Ethernet (12.5MBps raw bw) and Myrinet (250MBps raw bw) switches, both full duplex

System and Application Software

- RedHat Linux version 7.1, kernel version 2.4 on both clusters
- MPI version 1.2.1 for spawning processes in parallel

```
For each node
   Open file
   Barrier synchronize
   Start timer
   Read/write my portion
   Barrier synchronize
   End timer
   Report bytes processed
```

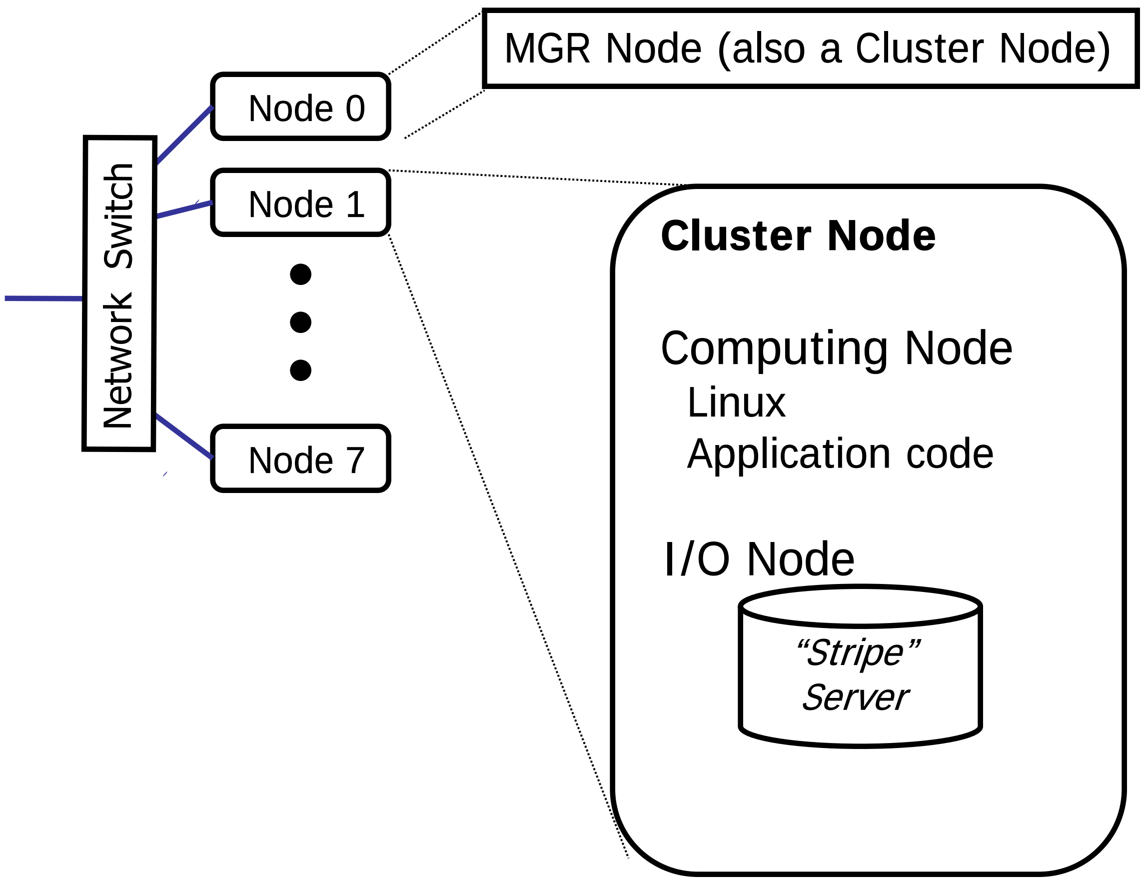
Two File Systems

- NFS, Version 3
- Distributed file view to clients, but uses a central server for files
- Sophisticated client-side cache, block size of 32KB
- Uses the Linux buffer cache on the server side

Two File Systems

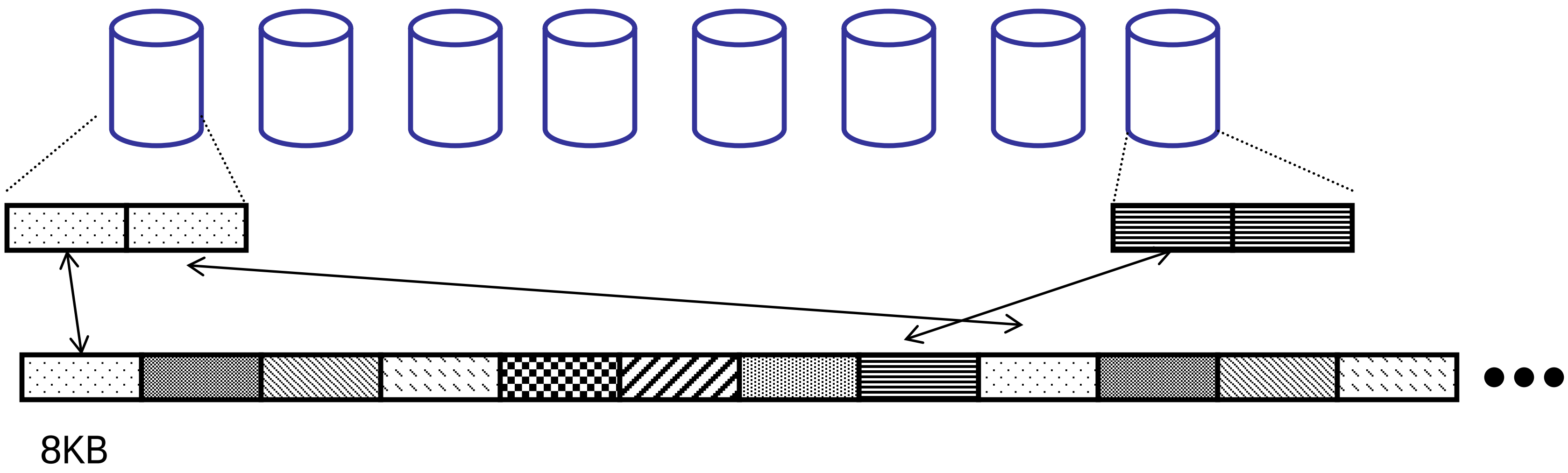
- Parallel Virtual File System (PVFS), kernel version 0.9.2
- Also uses the Linux buffer cache on the server side
- No client cache

PVFS Setup


MGR Node (also a Cluster Node)
Node 0
Node 1
Node 7
Network Switch
Cluster Node
Computing Node
Linux
Application code
I/O Node
"Stripe" Server

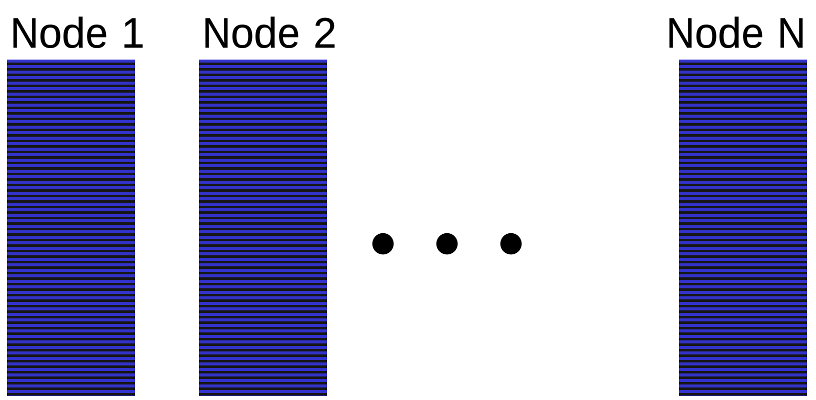
File Striping on PVFS

- A very large file is striped across 7 or 8 nodes, with stripe size of 8KB, fixed record length of 839 bytes



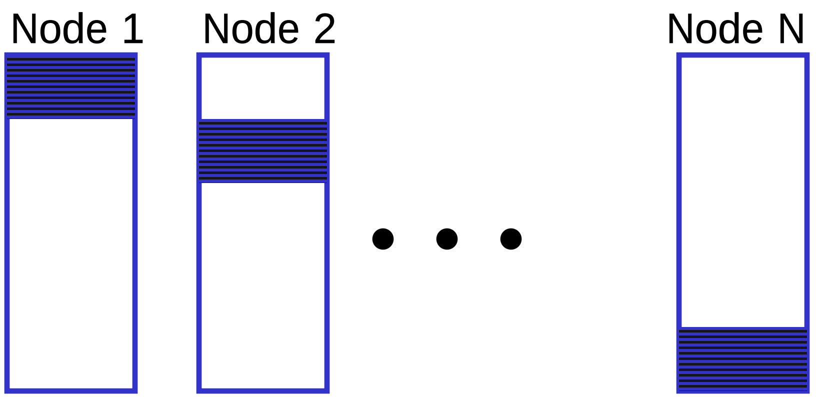
Experimental Workload One

- Local Whole File (LWF)
  - N processes run on N nodes. Each process reads the entire file to memory



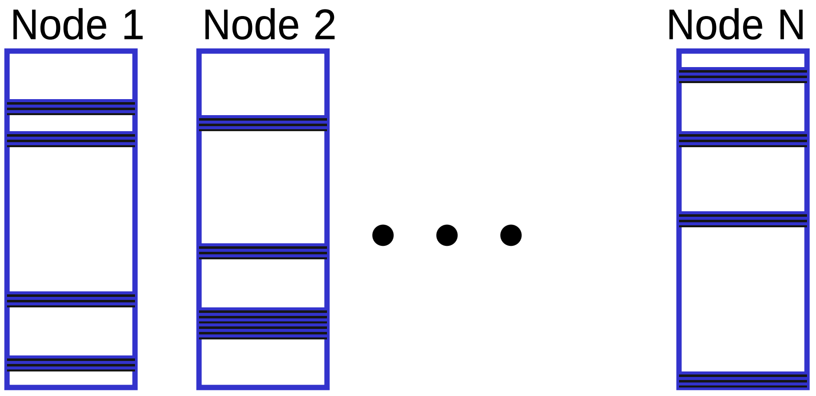
Experimental Workload Two

- Global Whole File (GWF)
  - N processes run on N nodes. Each process reads an equal-sized disjoint portion of the file. From a global perspective the entire file is read.



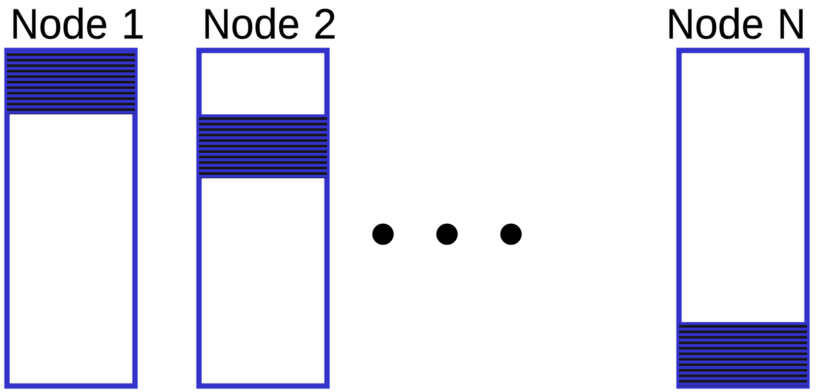
Experimental Workload Three

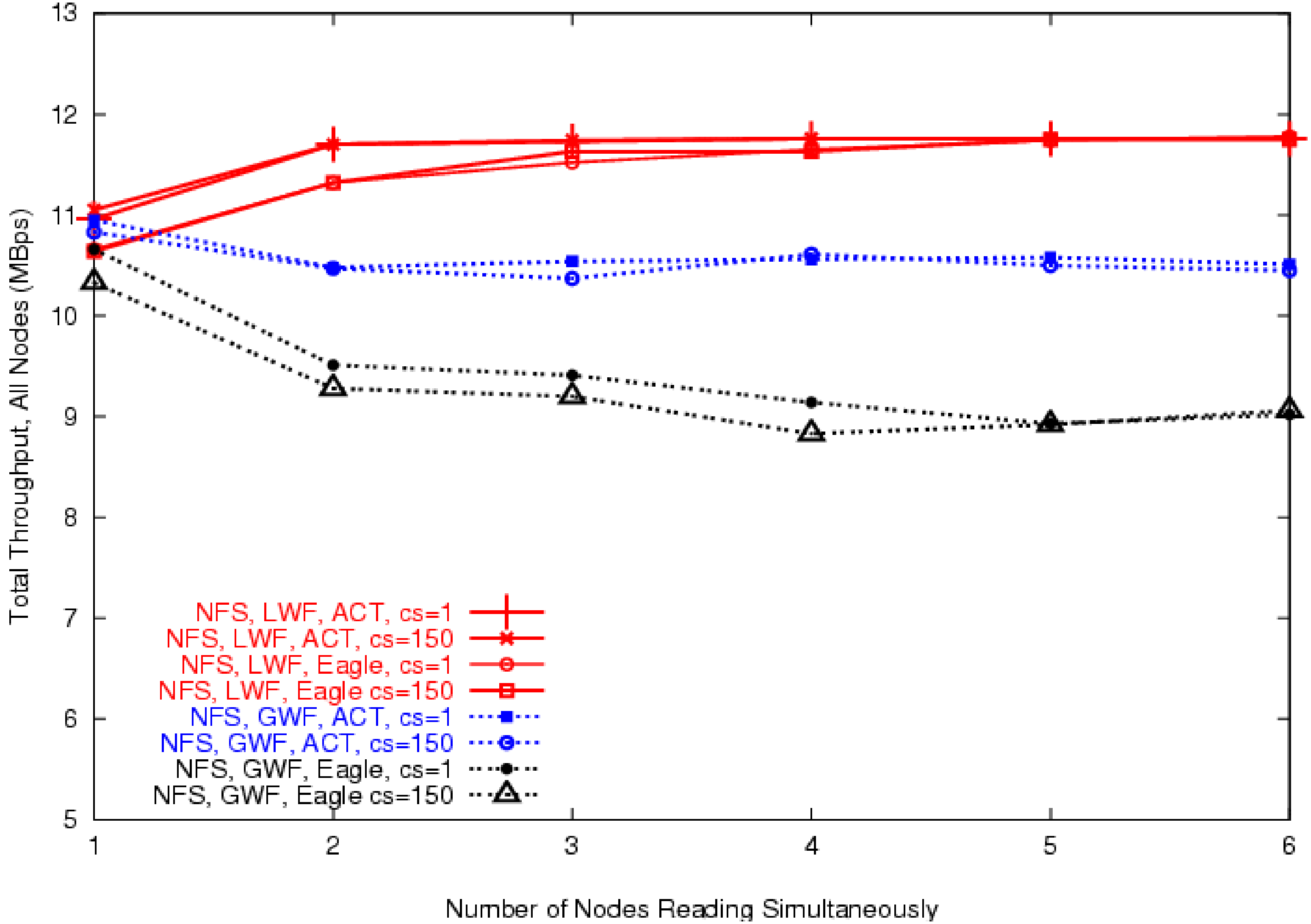
- Random (RND)
  - N processes run on N nodes. Each process reads an equal number of records from the file from random starting locations



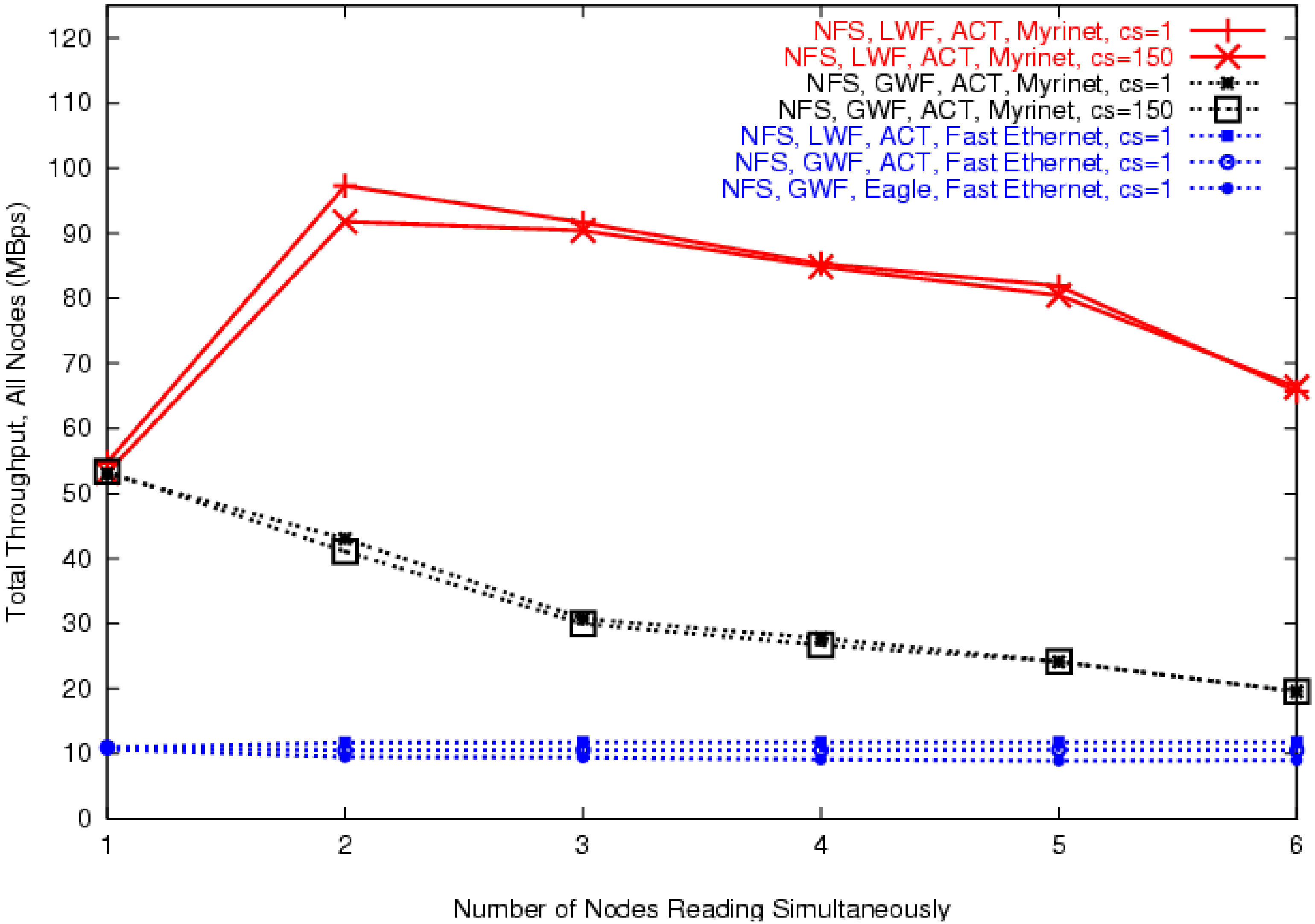
Experimental Workload Four

- Global Whole File Write (GWFW)
  - N processes run on N nodes. Each process writes an equal-sized disjoint portion of the file. From a global perspective the entire file is written. No locking is used since the writes are disjoint.

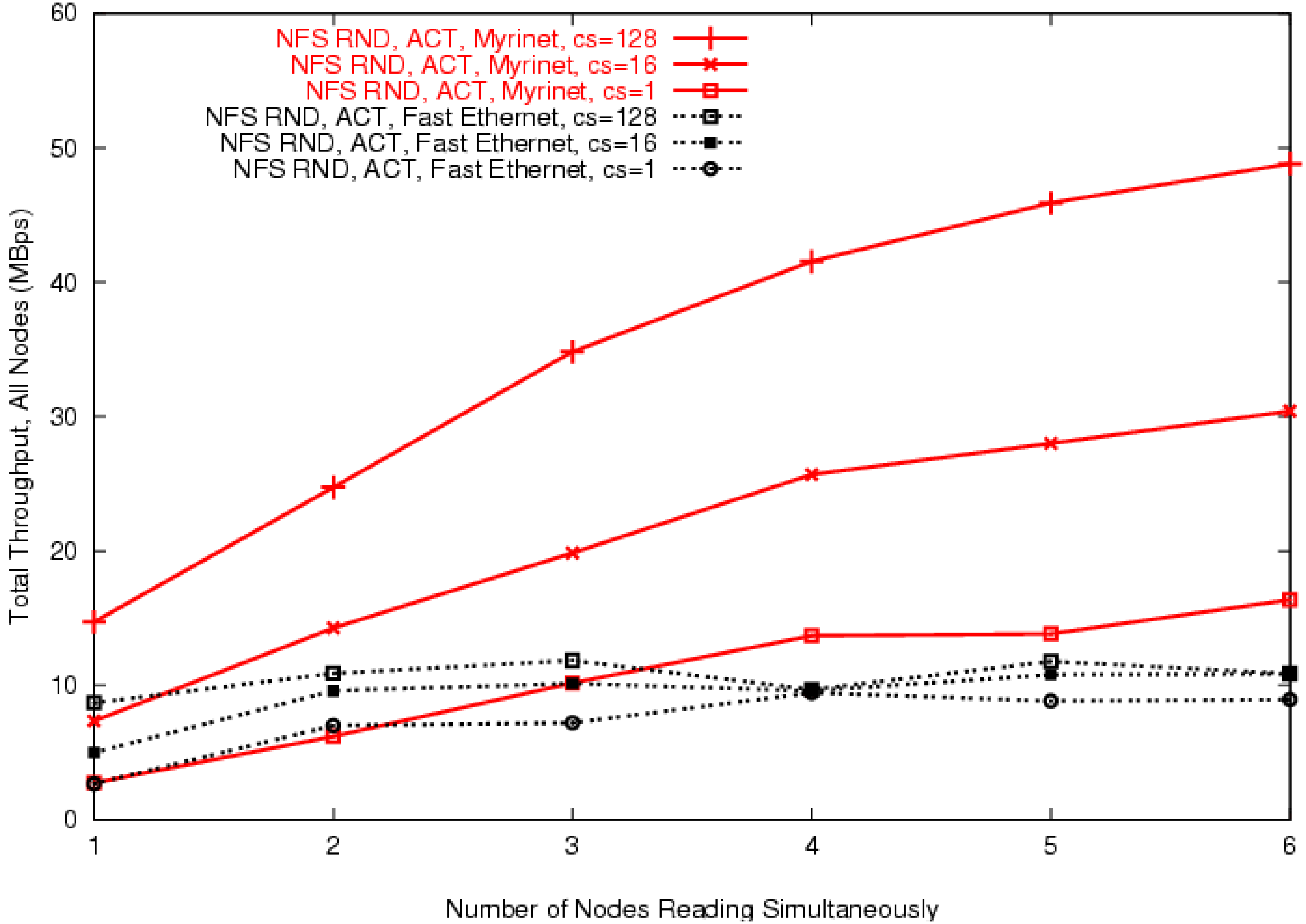




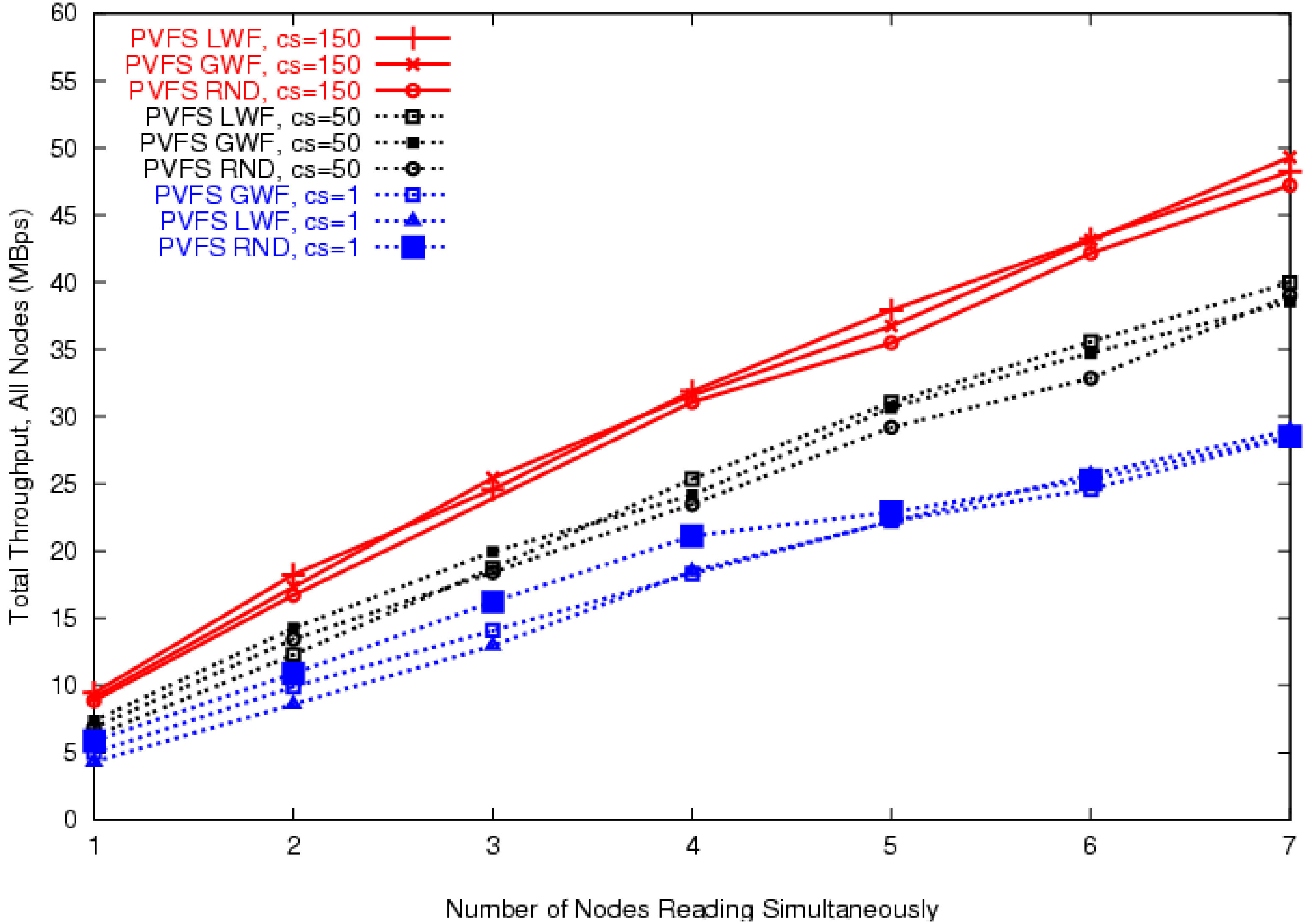
NFS Read, LWF, GWF, ACT with Fast Ethernet and Eagle Total Throughput across all Nodes, varying Chunksize



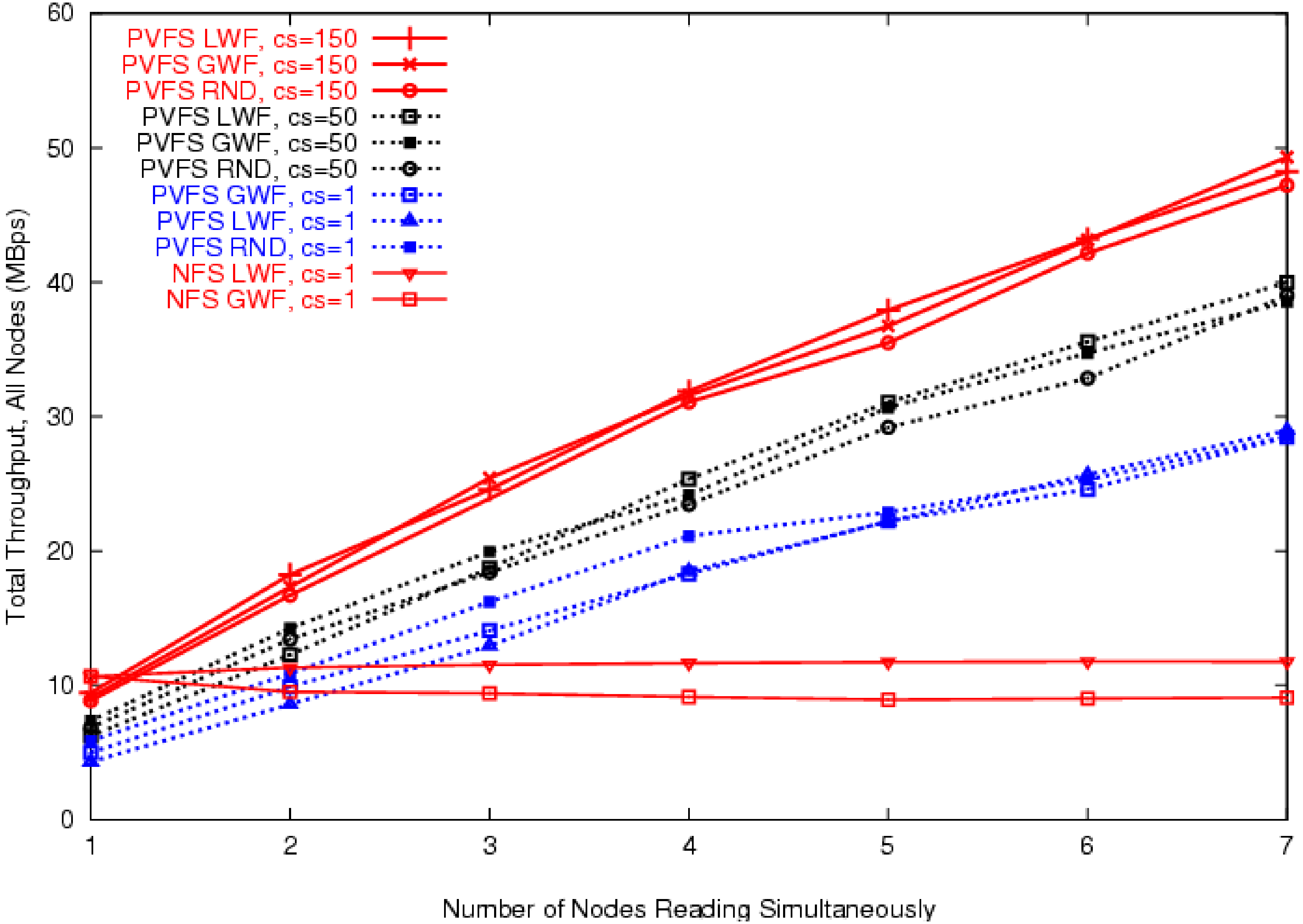
NFS Read, LWF, GWF, ACT with Myrinet, Fast Ethernet, and Eagle Total Throughput across all Nodes, varying Chunksize



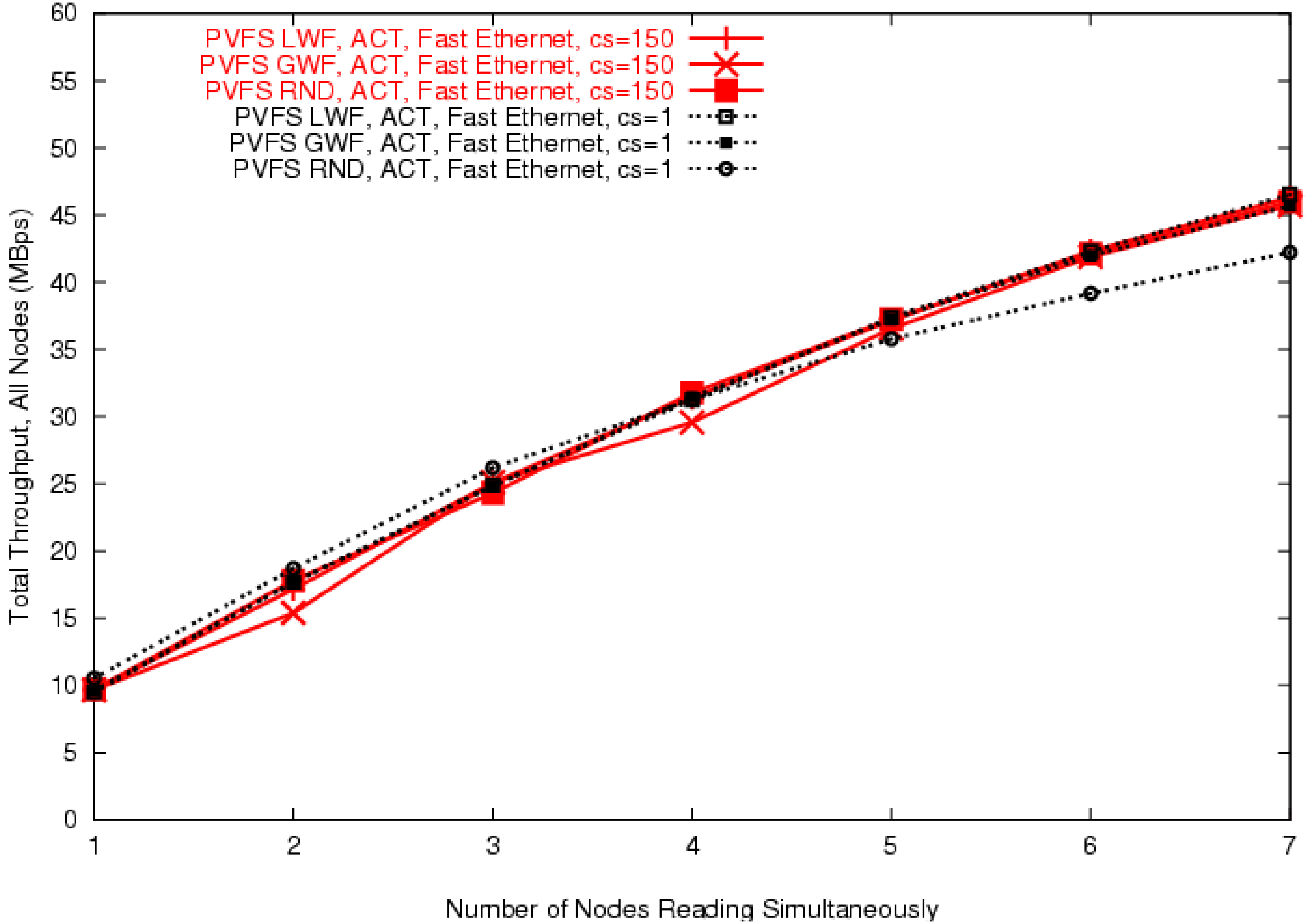
NFS Random Read, ACT with Myrinet and ACT with Fast Ethernet
Total Throughput across all Nodes, varying Chunksize



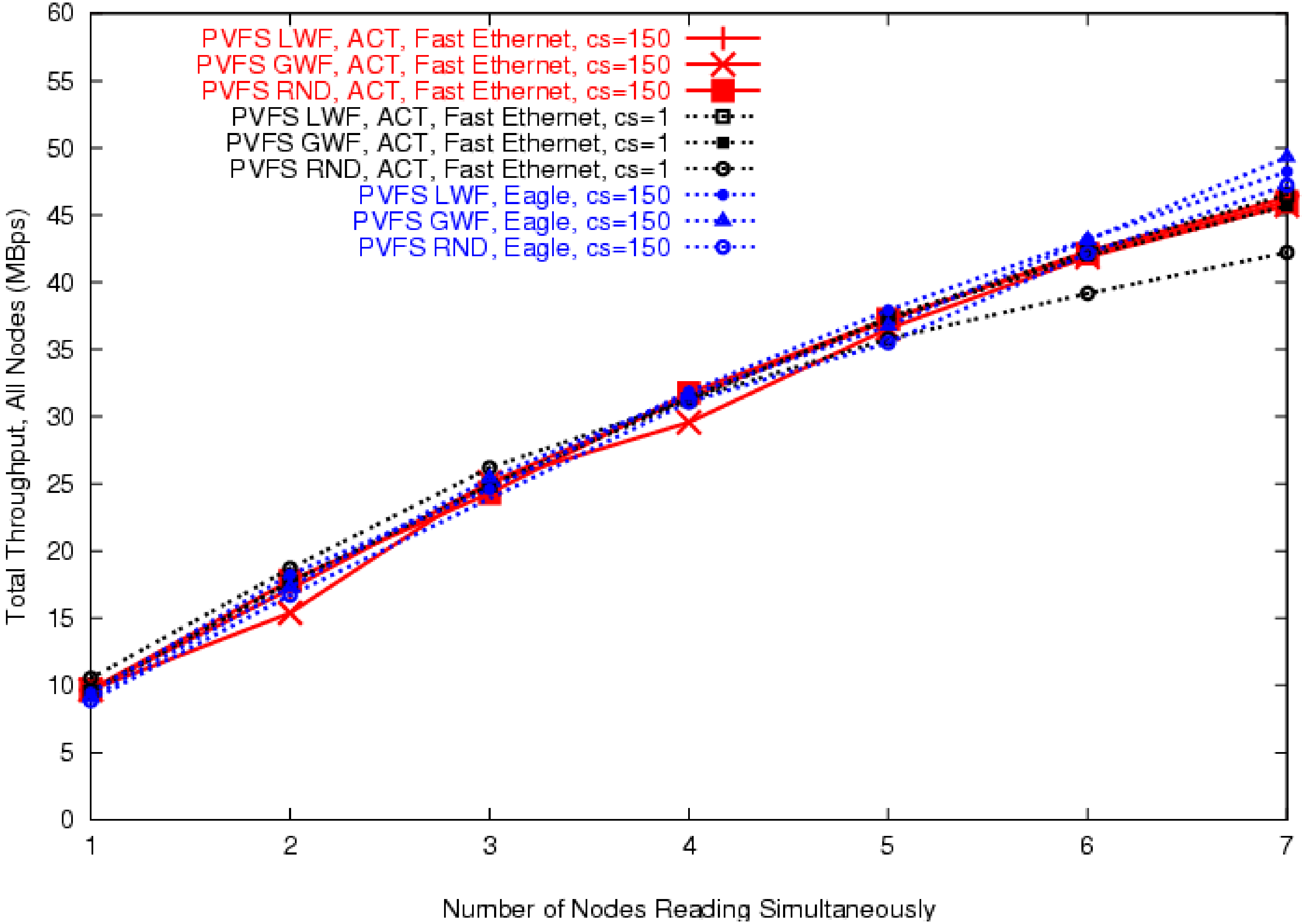
PVFS Read, All Workloads, Eagle
Total Throughput across all Nodes, varying Chunksize



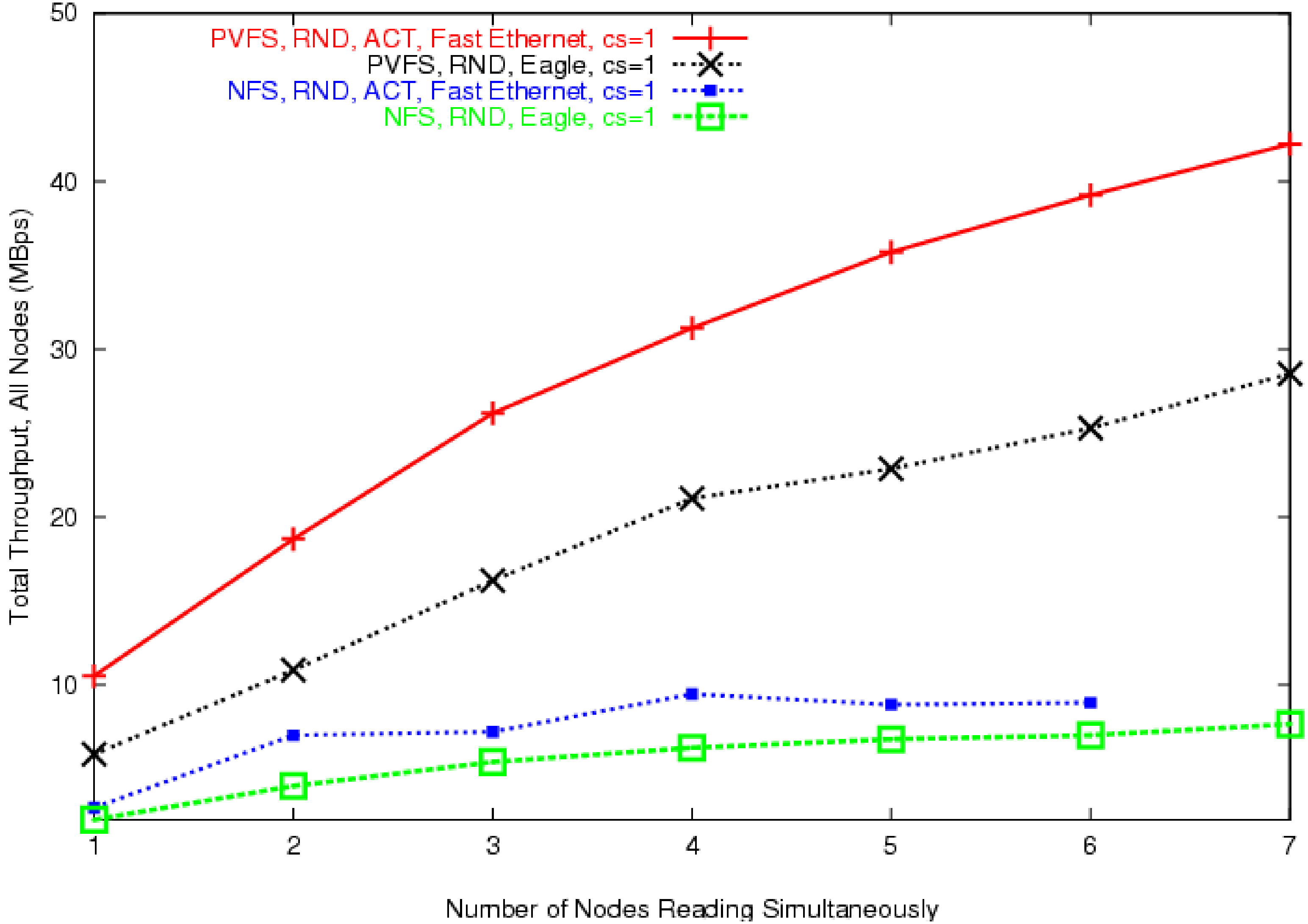
PVFS Read, All Workloads, shown with NFS read, GWF, LWF, Eagle Total Throughput across all Nodes, varying Chunksize



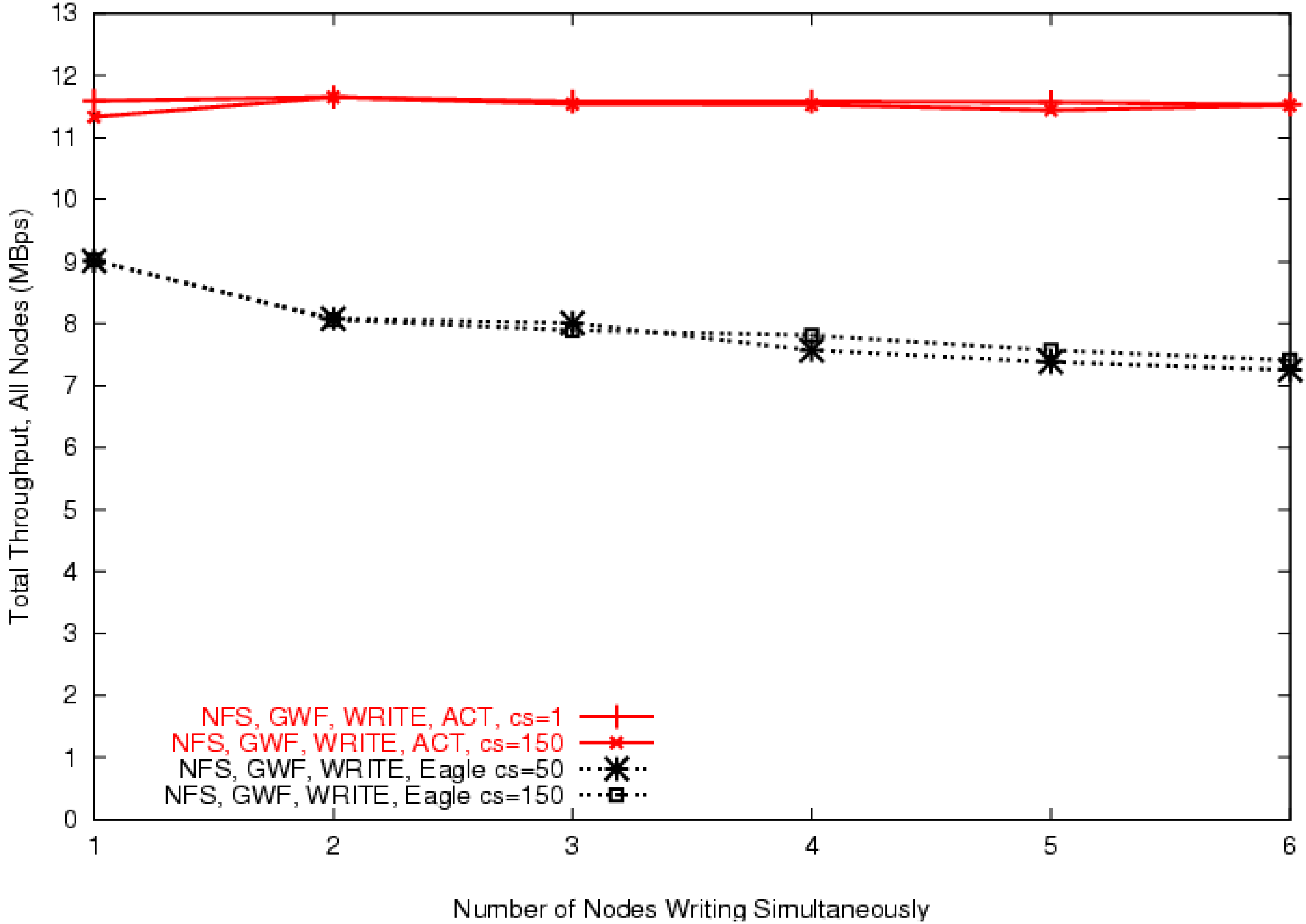
PVFS Read, Act with Fast Ethernet, All Workloads
Total Throughput across all Nodes, varying Chunksize



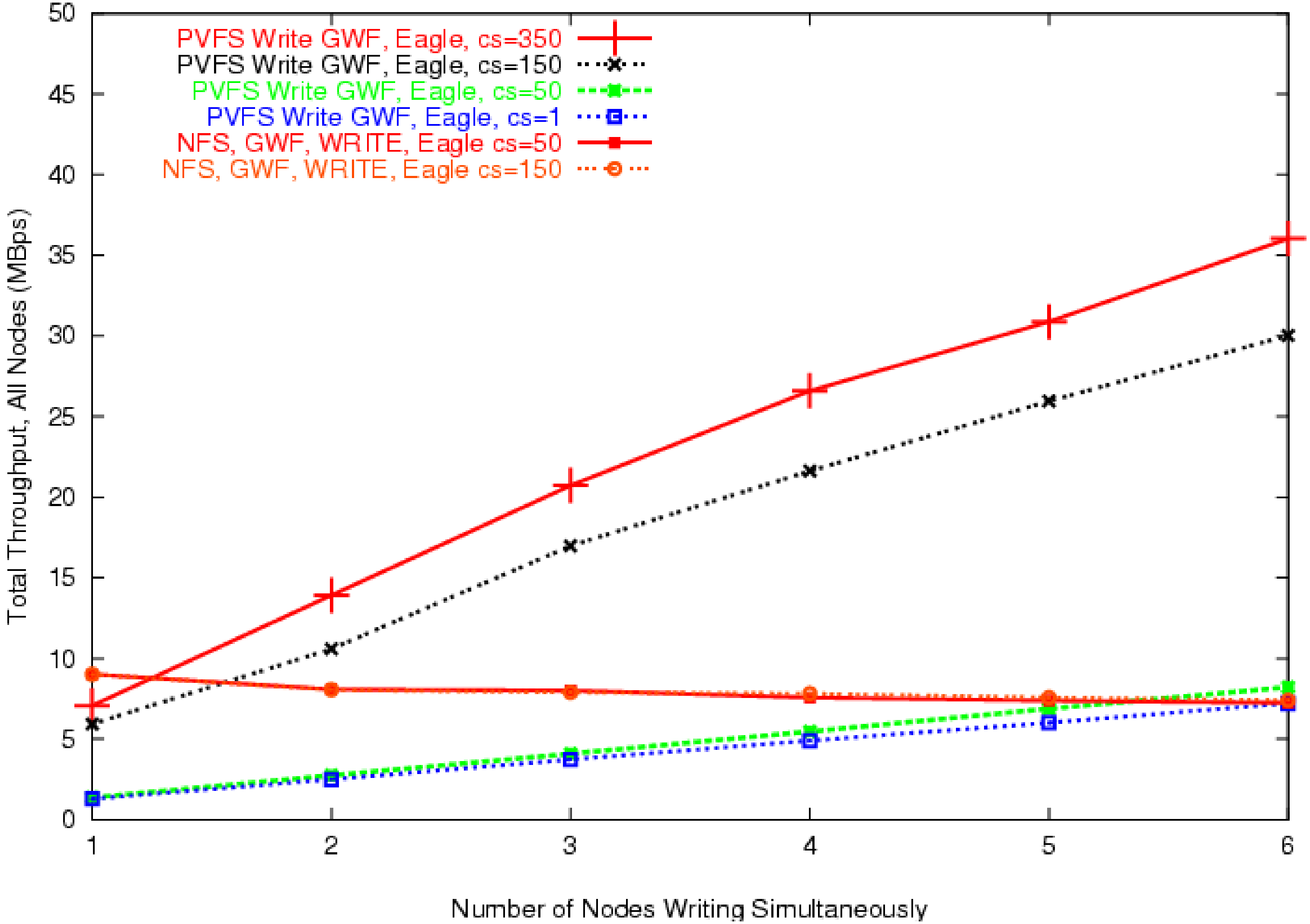
PVFS Read, Act with Fast Ethernet, All Workloads, shown with Eagle cs=150
Total Throughput across all Nodes, varying Chunksize



RND Read, CS=1, PVFS versus NFS, ACT with Fast Ethernet and Eagle
Total Throughput across all Nodes



NFS Write, GWF, Eagle and ACT Fast Ethernet
Total Throughput across all Nodes, varying Chunksize



PVFS Write, Eagle, (with NFS Write Eagle)
Total Throughput across all Nodes, varying Chunksize

Conclusions

- File system performance is limited by disk throughput as well as network throughput, and depends on workload
- NFS overall throughput degrades with more parallel (different data) access
  - Probably due to contention at the disks
  - Even more dramatically with our faster hardware!

Conclusions

- For our system where disk speed is close to network speed, PVFS read performance is best when the access is spread across many servers
  - Small stripes seem to be good in this case
- For our system where the disks are much faster than the network, PVFS read performance does not depend on access size

Conclusions

- PVFS write performance is dependent on access size for all platforms tested
- Myrinet is not even close to being saturated with these workloads and hardware

Future Work

- Read and write performance with Myrinet
- Sensitivity studies of how PVFS stripe size affects parallel file system performance
- Development of a lightweight locking mechanism for PVFS
  - PVFS currently does not support concurrent writes
- Exploration of fast, fault-tolerant techniques for metadata storage