Comparison of leading parallel NAS file systems on commodity hardware

Richard Hedges, Keith Fitzgerald, Mark Gary, D. Marc Stearman
Lawrence Livermore National Laboratory

Introduction

We are planning the environment for Sequoia, a 20 PetaFlop/s IBM system with an I/O target of 512 GB/s, and a stretch goal of 1TB/s. In considering file system options, we wanted to compare aspects of the two leading solutions with identical hardware. The intention was to get an apples-to-apples comparison between Lustre and GPFS, to the extent possible.

Test Configuration

The physical storage is a single DDN9550 system capable of delivering 2.4 GB/s with 4MB I/O requests or 2.0 GB/s with 1MB requests. There are 48 tiers of 250GB SATA disks in 24 8+2 RAID3 luns per controller. On top of the storage hardware there are four OSS nodes which are Dell R610 systems with dual socket, quad core Intel Nehalem E5530 processors at 2.4 GHz. Each OSS node has a 10 Gb/s Ethernet interface to the storage network and an SDR InfiniBand (IB) connection to the DDN 9550. There is a pair of MDS (failover) servers (Dell R610) attached to a 16 bay SAS/SATA JBOD enclosure with sixteen 15K RPM SAS drives.

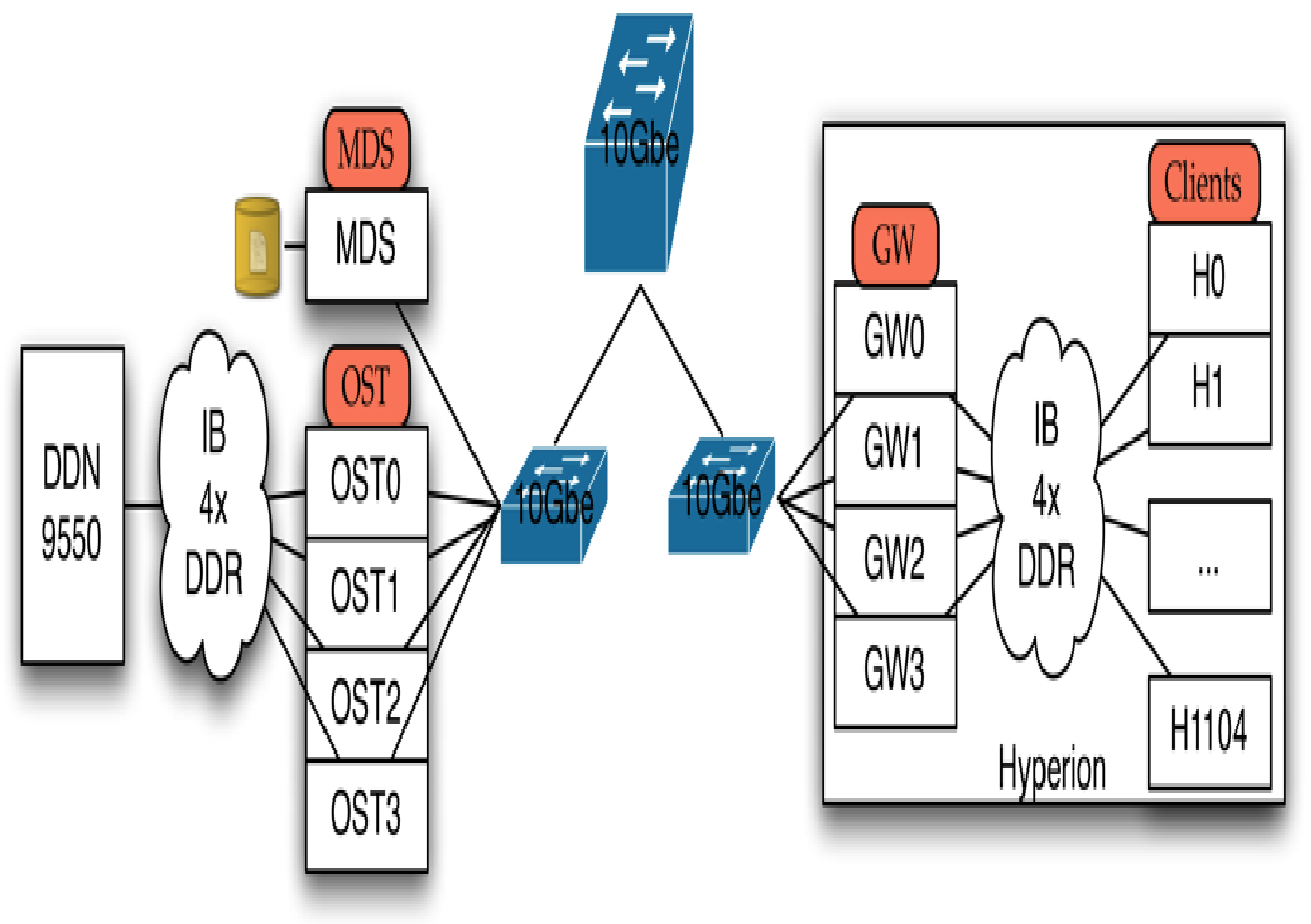

Figure 1. Block diagram of hardware for the Lustre test configuration. For GPFS, metadata was placed on a dedicated disk, and data across the remaining disks: the block diagram is analogous with "metadata" substituting for MDS and "data" for the OSTs

Testing strategy

We configured Lustre for the testing with the patch stack that we were running on our production systems at the time. For GPFS, expertise was provided by IBM to configure and tune GPFS version 3.2

Testing was done using LLNL developed test codes IOR, and mdtest. Parameters and test conditions were chosen as appropriate to our workload and previous history of testing production systems. We ran tests on clean and "aged" filesystems, but found minimal differences for clean vs. aged.

Results

Here we present some selected results. There are tests beyond these, so please inquire if there are tests of interest to you that we have not presented.

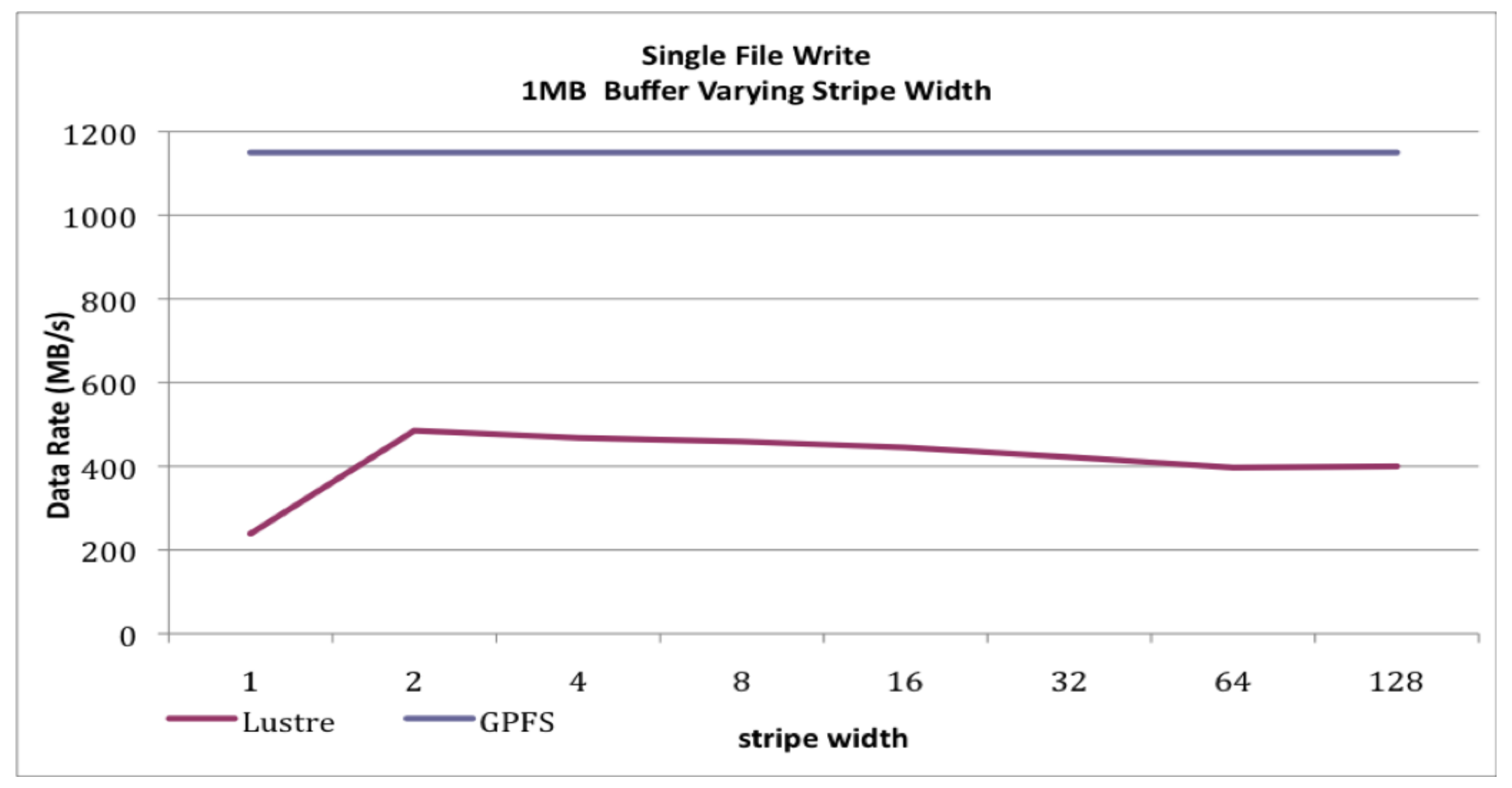

Figure 2. Write throughput for a single process. GPFS has the capability to stripe file adaptively over all servers. Also it appears that the GPFS client is able to use multiple cores in the client node.

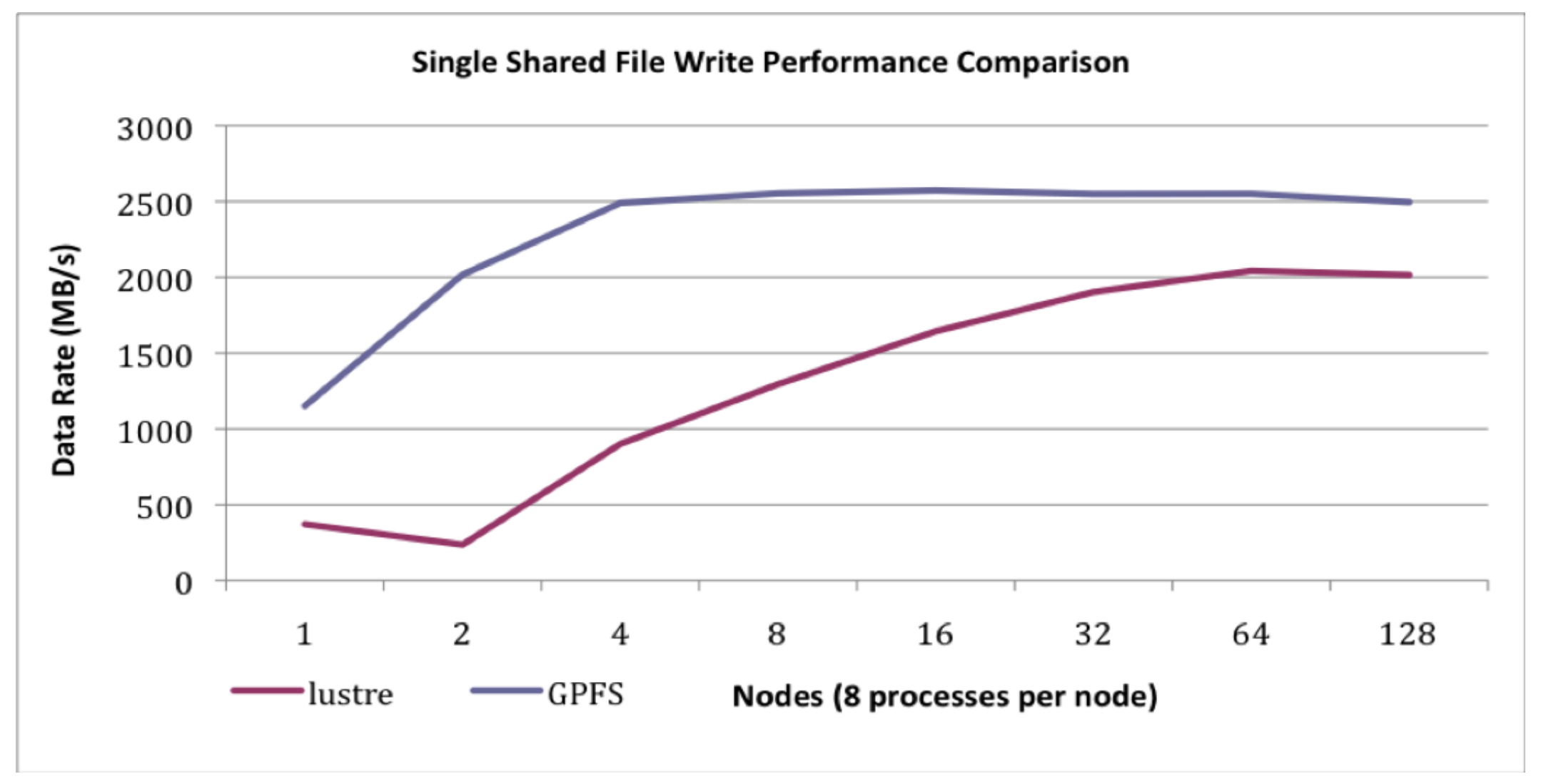

Figure 3. Write performance for the shared file case. GPFS has been tuned for this case through the years. GPFS striping and use of multiple cores can explain the difference.

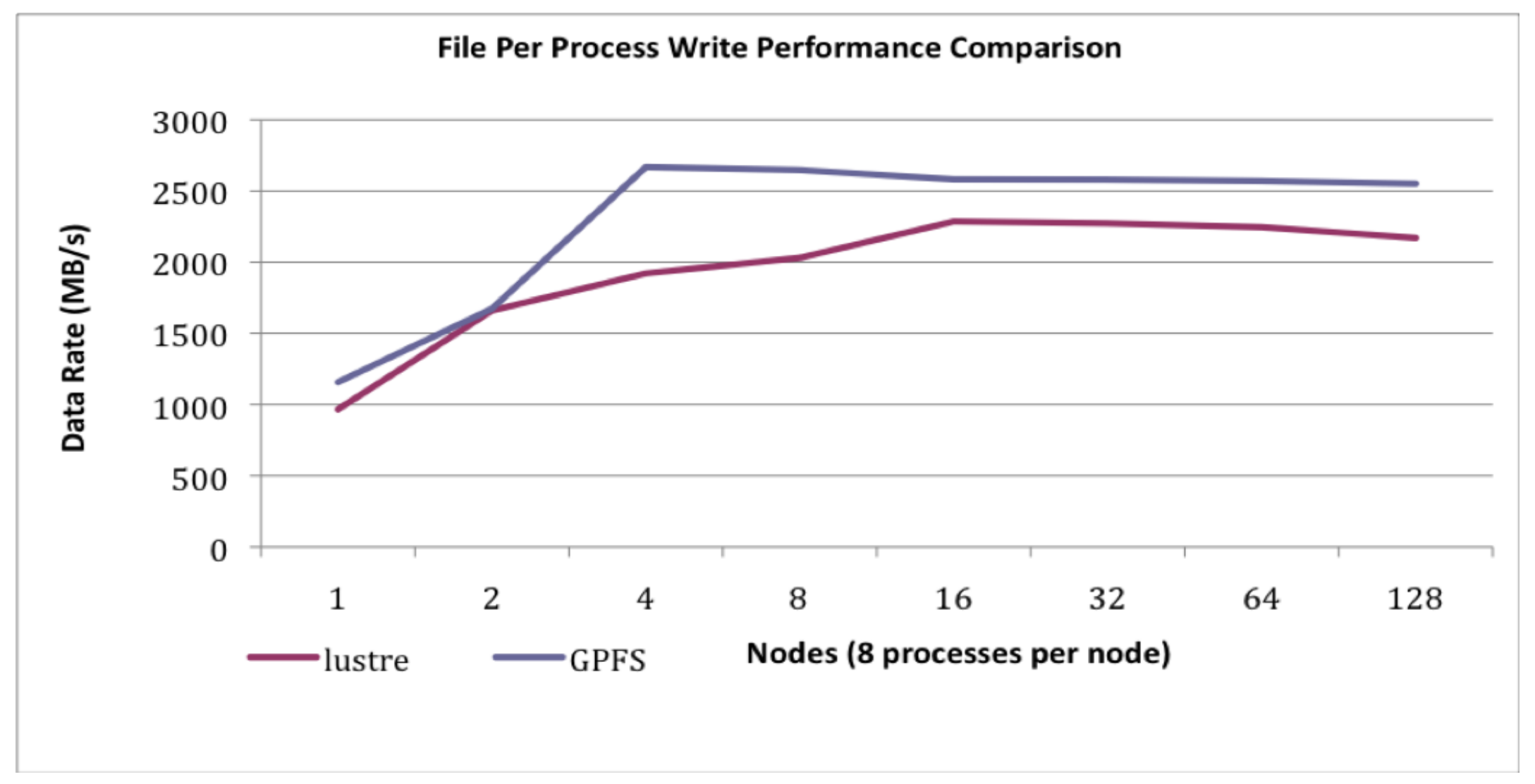

Figure 4. Write throughput for the file per process case. GPFS rises faster initially, presumable due to the striping of files across all servers. Asymptotically, Lustre performance is less likely due to the overhead of checksumming.

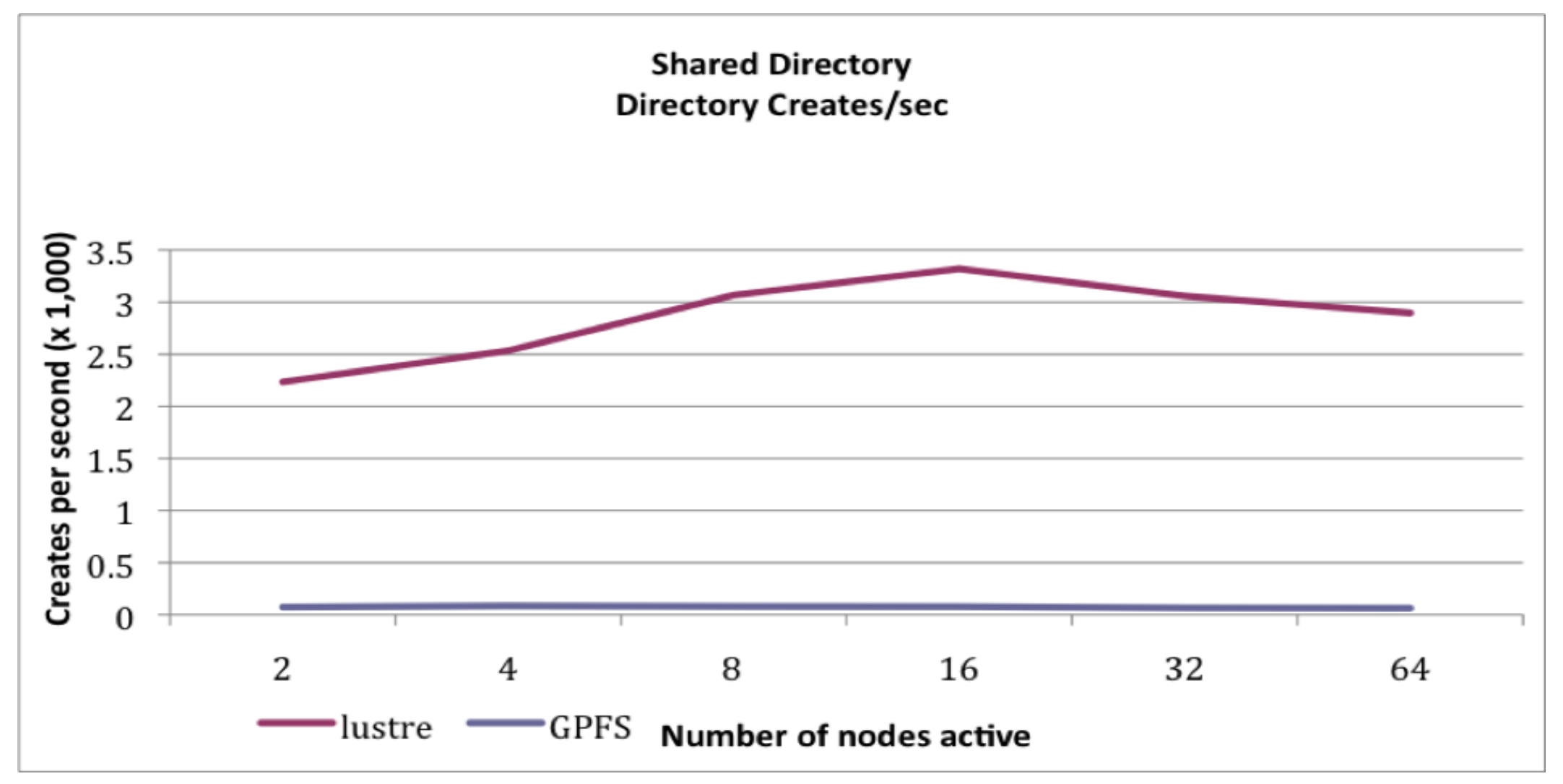

Figure 5. Directory creates are much slower on GPFS in particular for a shared directory. File and directory removals follow a similar pattern.

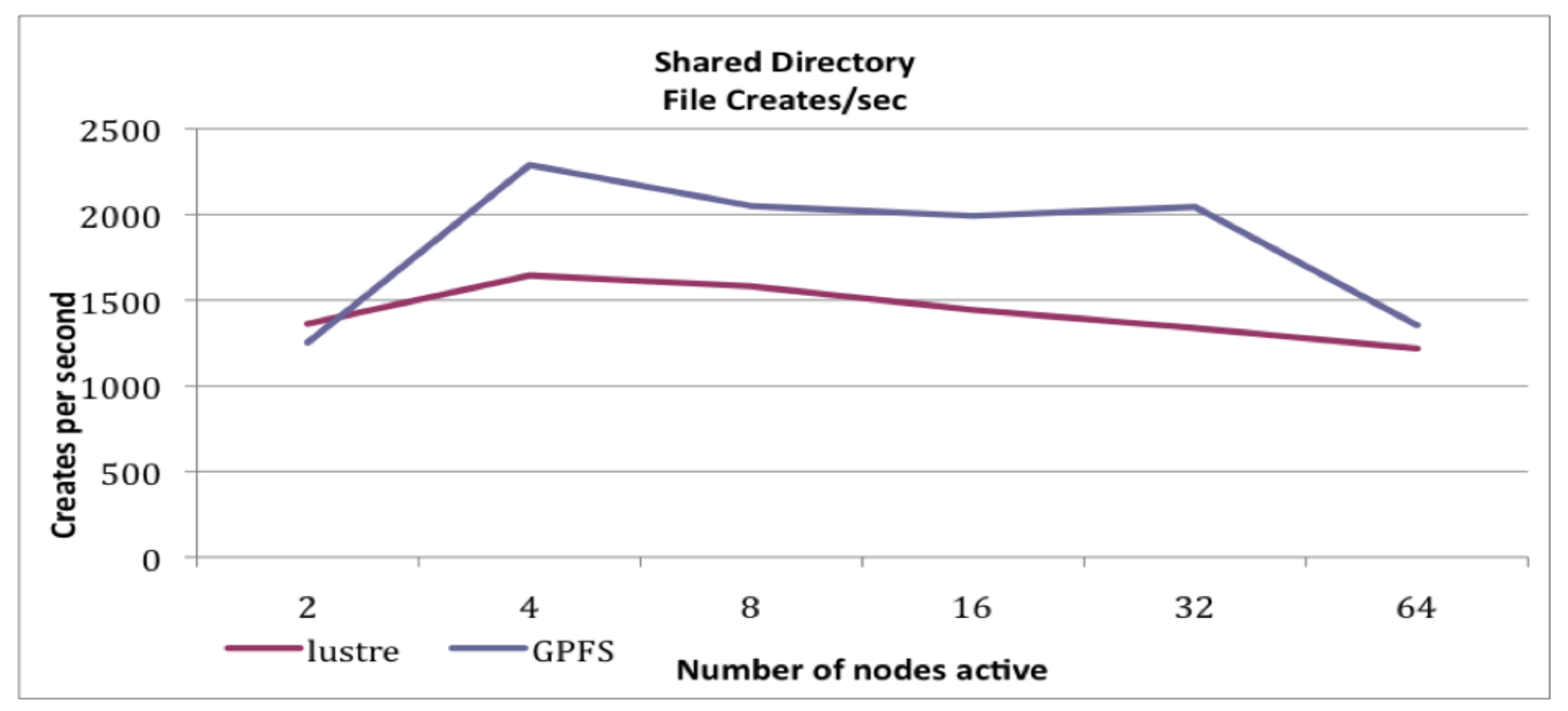

Figure 6. File creation in a shared directory was a sore point for GPFS, but was sufficiently important that a parallel algorithm was implemented.

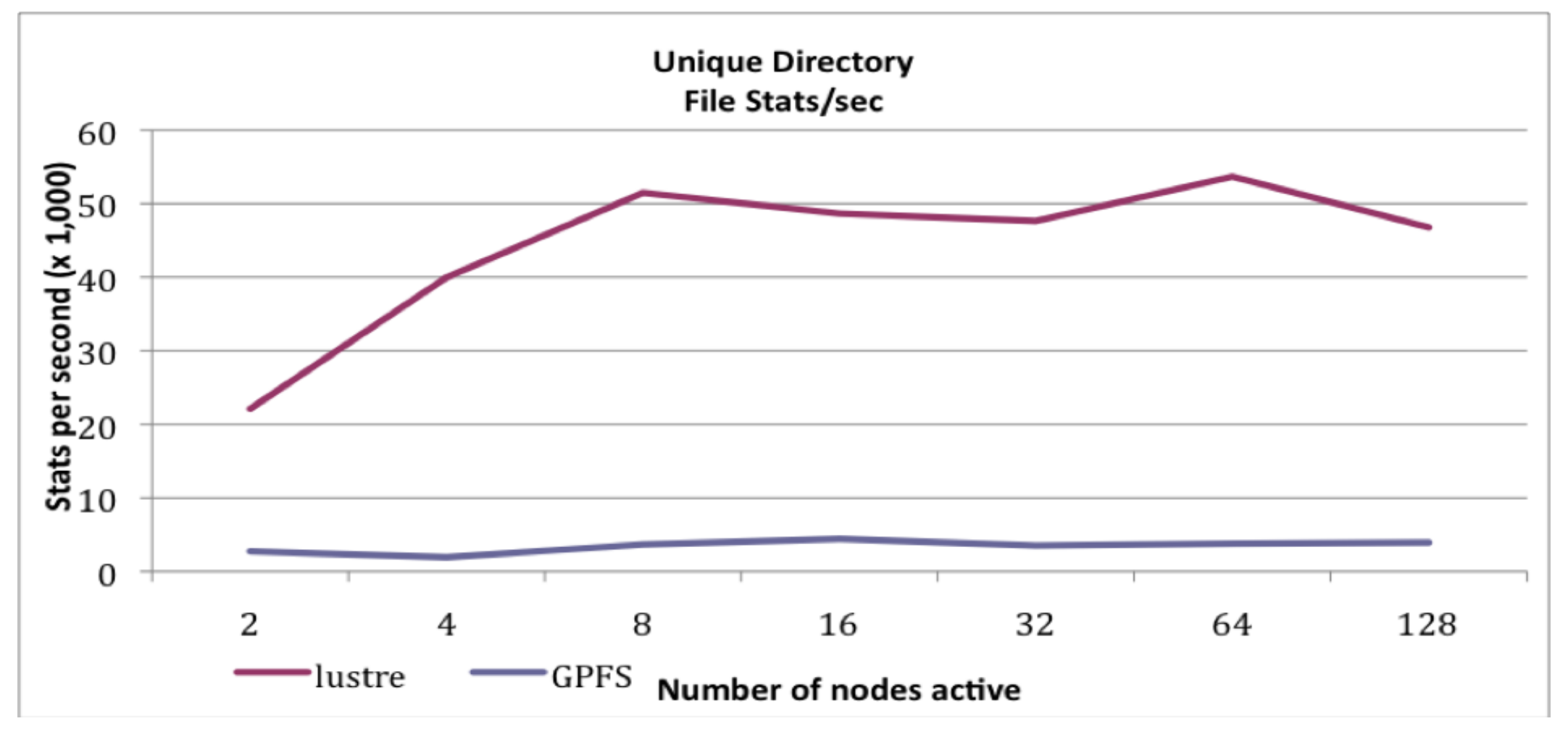

Figure 7. For GPFS, a stat initiated by a node other than the node which created the file forces a flush to disk.

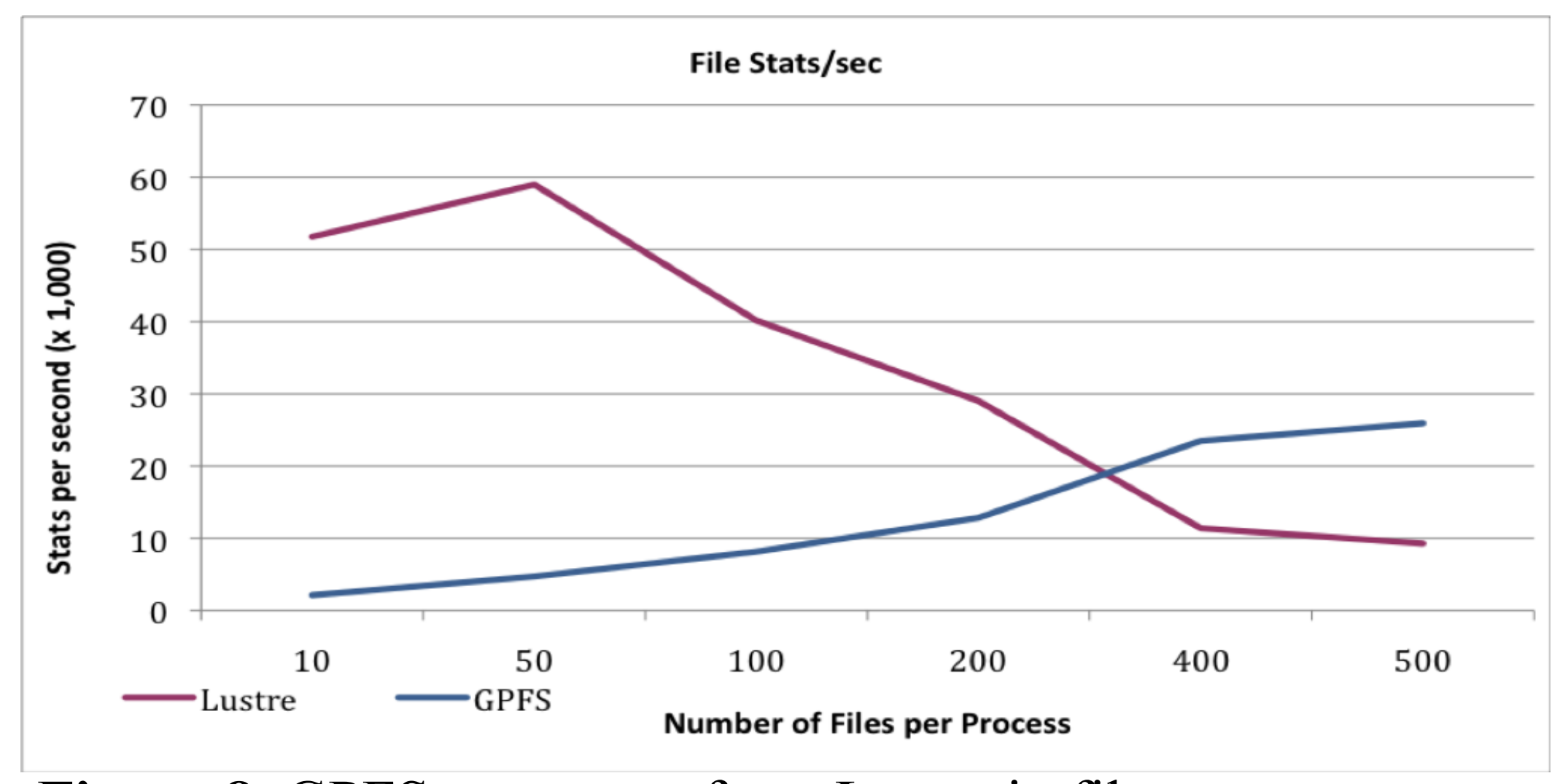

Figure 8. GPFS can outperform Lustre in file stats per second when there are more files in each unique (to the processor) directory.

Conclusions

1.Both file systems are able to drive hardware at high rates.
2.GPFS has the advantage for throughput tests due to larger blocksize, and since Lustre has the additional overhead of internal checksumming.
3.Lustre is generally significant faster for the metadata operations that are most important in our workload: operations where data cannot be locally cached on the client.
4.GPFS is better able to spread data over multiple servers and to use multiple cores in the client.
5.Where metadata can be cached on the client, GPFS will have an advantage.
6.For stating a large number of files in a shared directory GPFS can beat Lustre, with the advantage increasing as the number of files increase.
7.File creations in a shared directory have been special optimized algorithm for GPFS with "fine grained directory locks".
8.We now run GPFS on a BGL system.

References

1 Frank B. Schmuck , Roger L. Haskin, GPFS: A Shared-Disk File System for Large Computing Clusters, Proceedings of the Conference on File and Storage Technologies, p.231-244, January 28-30, 2002
2 "Lustre File System: High-Performance Storage Architecture and Scalable Cluster File System", white paper available at https://www.sun.com/offers/details/LustreFileSystem.xml
3 David Nagle , Denis Serenyi , Abbie Matthews, The Panasas ActiveScale Storage Cluster: Delivering Scalable High Bandwidth Storage, Proceedings of the 2004 ACM/IEEE conference on Supercomputing, p.53, November 06-12, 2004
4 IOR (http://sourceforge.net/projects/ior-sio)
5 mdtest[(http://sourceforge.net/projects/mdtest/)

For further information

Please contact *hedges1@llnl.gov*. More information about Livermore Computing and some related projects can be obtained at https://www.llnl.gov/computation/home/index.php

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.