Improved Read Performance in a Cost-Effective, Fault-Tolerant Parallel Virtual File System (CEFT-PVFS)

Yifeng Zhu*, Hong Jiang*, Xiao Qin*, Dan Feng†, David R. Swanson*
*Department of Computer Science and Engineering
University of Nebraska – Lincoln, NE, U.S.A Email: jiang@cse.unl.edu
†Department of Computer Science and Engineering
Huazhong University of Science and Technology, Wuhan, China Email: dfeng@hust.edu.cn

Abstract

Due to the ever-widening performance gap between processors and disks, I/O operations tend to become the major performance bottleneck of data-intensive applications on modern clusters. If all the existing disks on the nodes of a cluster are connected together to establish high performance parallel storage systems, the cluster's overall performance can be boosted at no additional cost. CEFT-PVFS (a RAID 10 style parallel file system that extends the original PVFS), as one such system, divides the cluster nodes into two groups, stripes the data across one group in a round-robin fashion, and then duplicates the same data to the other group to provide storage service of high performance and high reliability. Previous research has shown that the system reliability is improved by a factor of more than 40 with mirroring while maintaining a comparable write performance. This paper presents another benefit of CEFT-PVFS in which the aggregate peak read performance can be improved by as much as 100% over that of the original PVFS by exploiting the increased parallelism.

Additionally, when the data servers, which typically are also computational nodes in a cluster environment, are loaded in an unbalanced way by applications running in the cluster, the read performance of PVFS will be degraded significantly. On the contrary, in the CEFT-PVFS, a heavily loaded data server can be skipped and all the desired data is read from its mirroring node. Thus the performance will not be affected unless both the server node and its mirroring node are heavily loaded.

1. Introduction

Cluster computing, as a powerful rival of commercial MPPs, has become the fastest growing platforms in parallel computing. A significant number of large-scale scientific applications running on clusters require the input and output of large amounts of data from secondary storage, ranging from mega-bytes to tera-bytes [1, 2]. Therefore, the I/O performance is crucial and can largely determine the overall completion time of these applications. Due to the steadily increasing gap in speed between processors and I/O disks, I/O operations have emerged to be the source of the most severe bottleneck for these applications. One of the most cost-effective approaches to alleviate the I/O bottleneck is to utilize the existing disks (IDE or SCSI) on all cluster nodes to build a parallel file system, which not only provides a multi-terabyte scale storage capacity, but also taps into the aggregate bandwidth of these disks to deliver a high-performance and scalable storage service. PVFS [3] is one example of such file systems and it can achieve multiple GBytes/sec I/O throughputs [4] without any additional cost if the cluster is connected through Myrinet [5] or Gigabit Ethernet. However, like disk arrays [6], without any fault tolerance, these parallel storage systems are too unreliable to be useful since the failure rate of a cluster node, compounded by the failures of cluster hardware components, including CPU, disk, memory and network, and the software components, such as operating system and network drivers, is potentially much higher than that of an individual disk.

To meet the critical demand for reliability, a Cost-Effective, Fault-Tolerant Parallel Virtual File System (CEFT-PVFS) [7], has been designed and implemented that achieves a considerably high throughput. This new system is fundamentally different from PVFS, a RAID-0 style system that does only striping in its current implementation. As a RAID-10 style parallel file system, CEFT-PVFS combines striping with mirroring by first striping among the primary group of storage nodes and then duplicating all the data in the primary group to its backup group to provide fault tolerance. Moreover, CEFT-PVFS changes the naming mechanism from the inode numbers to the MD5 sums [8] and therefore enables backing up the metadata server that holds the most crucial information, which is not possible with the current PVFS design. The above mirroring processes enable CEFT-PVFS to achieve significant improvements in reliability over PVFS with a 50% storage space overhead. In our previous studies on CEFT-PVFS, four different duplication protocols are proposed, striking

Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)
0--7695-1919-9/03 $17.00 © 2003 IEEE

different balances between the write performance and the reliability. Our experiments, conducted on a cluster of 128 nodes (of two processors each), and theoretical reliability analysis based on Markov chain models have shown that, in cluster environments, mirroring can improve the reliability by a factor of over 40 (4000%) while sacrificing the peak write performance by 33-58% when both systems are of identical sizes (i.e., counting the 50% mirroring disks in the mirrored system). In addition, protocols with higher peak write performance are less reliable than those with lower peak write performance, with the latter achieving a higher reliability and availability at the expense of some write bandwidth. A hybrid protocol is then proposed to optimize this tradeoff between the write performance and the reliability.

In this paper, we will address another potential benefit of the mirroring processes on CEFT-PVFS: boosting the read performance. By dividing the I/O load into the primary group and its mirroring group, the potential parallelism of read service is doubled and the read throughput can thus be improved. Further, the existence of mirroring nodes makes it possible to avoid (or skip) a heavily loaded "hot-spot" node, which in the original PVFS can severely degrade the read performance. As shown in Section 5, skipping hot-spot nodes indeed improves read performance significantly.

The rest of this paper is organized as follows. We first discuss the related work in Section 2. Then an overview of CEFT-PVFS is presented in Section 3. Parallel-read schemes for CEFT-PVFS is are developed in Section 4 to improve read performance while Section 5 presents a scheduling scheme that helps avoid severe read performance degradation by skipping hot-spot nodes judiciously. Section 6 concludes the paper with comments on current and future work.

2. Related work

By exploiting parallelism, parallel file systems stripe the data across multiple I/O nodes, keeping the striping details transparent to applications. Amongst many successful parallel I/O systems is Parallel Virtual File System (PVFS) [3], developed at the Clemson University and Argonne National Lab. Like RAID, PVFS partitions the files into equal-sized units, and then distributes them to the disks on multiple cluster nodes in a round-robin fashion. Unlike RAID, PVFS provides a file-level, instead of a block-level interface, and all the data traffic flows in parallel, without going through a centralized component, which can become a performance bottleneck. Experimental measurements show that PVFS provides high performance, even for non-contiguous I/O accesses [9, 10], which may cause significant performance degradation in a conventional storage system. Nevertheless, PVFS in its current form is only a RAID-0 style storage system without any fault-tolerance. Any single server node failure will render the entire data inaccessible. The authors of PVFS shared the same view with us and addressed the importance of and desperate necessity to incorporating fault-tolerance into PVFS [11].

There are several studies related to PVFS. A kernel level caching to improve the I/O performance of concurrently executing processes in PVFS is implemented in [12]. The role of sensitivity of the I/O servers and clients is analyzed in [13], which concludes that when a node serves both as an I/O client and as a data server, the overall I/O performance will be degraded. In [14, 15], a scheduling scheme is introduced so that the service order of different requests on each server is determined by their desired locations in the space of Logical Block Address and disk arm seeking time is reduced accordingly.

In [7], we introduced the design and implementation of CEFT-PVFS, and evaluated the performance and reliability of four mirroring protocols. In [16], we proposed a scheme to optimize the write performance of CEFT-PVFS by adaptively scheduling writes to counter balance the possible disparity in resource availability between two nodes of each mirroring pair and among mirroring pairs within a server group. In this paper, we will address the issue of optimizing read performance in CEFT-PVFS by exploiting the increased read parallelism and redundancy.

3. An Overview of CEFT-PVFS

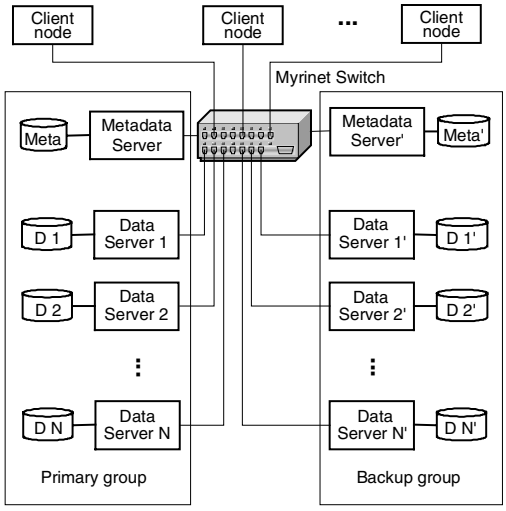

Figure 1. Block Diagram of CEFT-PVFS.

CEFT-PVFS is a RAID-10 style parallel storage system that mirrors the striped data between two logical groups of storage nodes, one primary storage group and one backup storage group with the same number of server nodes, as shown in Figure 1. To improve the response times of write requests, usually half of the server nodes with relatively less workload are assigned to the primary group and the duplication operations can proceed in the background. In each group, there is one metadata server, which records the striping information for each file and/or schedules the I/O requests on the data nodes chosen from each mirroring pair. To access the data in CEFT-PVFS, all clients need to contact the metadata servers first to get the destination data server addresses and the striping information about their desired data. After that all I/O operations will take place between the clients and servers directly in parallel through the network.

For write accesses in CEFT-PVFS, we have designed and implemented four duplication protocols to meet different requirements for reliability and write performance. Duplication can be either synchronous or asynchronous, i.e., the completion of write accesses can be signaled after the data has already taken residence on both groups or only on the primary group. At the same time, duplications can be performed either by the client nodes themselves or by the servers in the primary group. The four protocols are created based on different combinations of these two categories. The experimental measurements and theoretical analysis based on Makov chain models indicate that protocols with higher peak write performance are inferior to those with lower peak write performance in terms of reliability, with the latter achieving a higher reliability at the expense of some write bandwidth.

4. Improving Read Performance

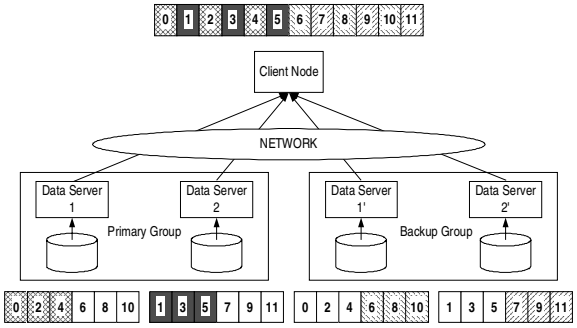

Figure 2. Reading interleaved data from both groups, half from the primary group, and half from the backup group.

Any data stored in CEFT-PVFS will eventually have two copies, one in the primary group and the other in the mirroring group. The storage space overhead for mirroring can be viewed as trading not only for the significantly increased reliability, but also for the increased read parallelism. Instead of reading the whole data from one storage group, the reading operations can divide their load on both storage groups. More specifically, the desired data is split into two halves and the client can simultaneously read interleaved blocks, one half from the primary nodes and the other half from their mirroring nodes. Figure 2 shows an example, in which each storage group is composed of two server nodes and the client node reads the target data from the four servers concurrently.

The performance results presented below are measured on the PrairieFire cluster [17] where CEFT-PVFS has been implemented and installed, at the University of Nebraska - Lincoln. At the time of our experiment, the cluster had 128 computational nodes, each with two AMD Athlon MP 1600 processors, 1 GByte of RAM, a 2-gigabit/s full-duplex Myrinet card, and a 20GB IDE (ATA100) hard drive. The Netperf [18] benchmark reports a TCP bandwidth of 126.5 MBytes/s with 47% CPU utilization. The disk read bandwidth is 26 MBytes/s when reading a large file of 2 GBytes according to the Bonnie benchmark [19].

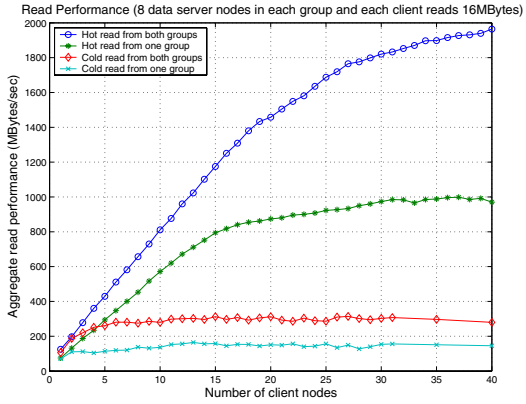

Figure 3. Read performance in the cases of cold read and hot read, as a function of the number of client nodes.

In the modern hierarchical storage architecture, the read performance mainly depends on the data locality of applications and on the cache and prefetch functionalities of storage systems. In this paper, we examine two extreme cases: *hot read* and *cold read*. In the case of hot read, all the data is cached by the memory and thus the number of disk accesses is kept minimal. The hot read

performance is measured by reading the same data repeatedly. In cold read, all the data has to be read from the disks. To clear the cache buffer and guarantee that real disk accesses take place, each data server reads a dummy file of 2 GBytes, twice as much as the total memory size, before each measurement, thus displacing any cached data.

The read performances of CEFT-PVFS are examined with two simple orthogonal micro-benchmarks: 1) all the clients read the same amount of data but the total number of client nodes changes; 2) the total number of client nodes is fixed while the size of the files that each client reads changes. In all experiments, CEFT-PVFS was configured with 18 server nodes, including 8 data servers and 1 metadata server in each group. All the performances reported in this paper are based on the average of 20 measurements. Figure 3 shows the performance of the first benchmark when all servers are lightly loaded by the other applications and each client reads 16 Mbytes data from the servers simultaneously. The aggregate performance is calculated as the ratio between the total size of the data read from all the servers and the average response time of all the clients. The aggregate performance of the hot read reaches its maximum value when all the network bandwidths from these data servers are fully utilized while that of the cold read enters its saturation region quickly disks become saturated. As the measurements indicate, the increased parallelism due to mirroring improves the performance nearly 100% for both the hot read and the cold read.

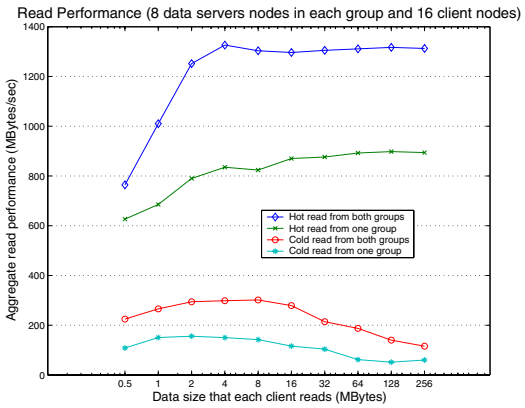

Figure 4. Read performance of cold read and hot read as a function of data size that each client reads.

Figure 4 plots the performances measured by the second benchmark, when there are a total of 16 clients and each of them reads different sizes of data from the servers. In the cold read, the performance begins to drop after an initial rise while this drop is not apparent in the hot read. The performance drop is potentially due to the fact that when the file size is too large, these files may not be stored contiguously on the disks so that more tracks needs to be sought, causing the total disk access time to increase. On the average, our proposed method improves the hot and cold read performance 69% and 91%, respectively.

5. Improving Read Performance in the Presence of Hot-spot Nodes

As an integral part of a cluster, all the CEFT-PVFS server nodes usually also serve as computational nodes. The system resources of these nodes, such as CPU, memory, disk and network, can be heavily stressed by different scientific applications running on these nodes, thus potentially degrading the overall I/O performance. While PVFS cannot avoid this degradation, in the CEFT-PVFS, each piece of a desired data is eventually stored on two different nodes. This redundancy provides an opportunity for the clients to skip the hot-spot node that is heavily loaded (or down due to failure) and read the target data from its mirroring node. More specifically, the server nodes periodically send their load information, including the load of CPU, the average throughput of disks and networks within each period, to the metadata server. The metadata server schedules the I/O requests and informs the clients of reading schemes. Figure 5 shows an example, in which Node 2 is skipped and all the data is read from its mirror Node 2'.

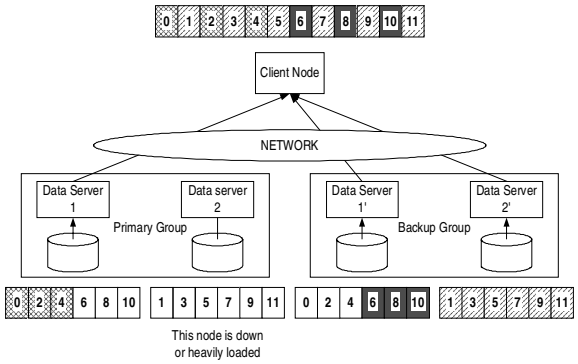

Figure 5. Skipping the heavily loaded data server nodes and reading the data from their mirroring server nodes.

5.1. The improved cold read performance

In the cold read, the data needs to be read from the disks, which generates the largest latency on the critical path of I/O operations, due to the large seek time and the

small read bandwidth of the disks. To compare the performance of skipping the hot-spot nodes, we artificially stress the disk on one server node in the primary group by allocating a memory space of 10 MBytes and repeatedly storing these data synchronously onto the disk. Three different methods are used to read the data: 1) from all servers in the primary group without skipping the busy node; 2) from all servers in both groups without skipping the busy node; 3) from both groups while skipping the busy node. Figure 6 shows the performance curves of those methods measured under the same load pattern, where 16 client nodes read different sizes of data from these servers. When the file size is small, skipping the busy node improves the cold read performance nearly ten times over reading the data from one group or both groups without skipping. As the data size increases, the benefits from skipping decrease since the total data size from the mirroring node of the skipped node increases at a doubled speed, causing the total disk seek time to increase.

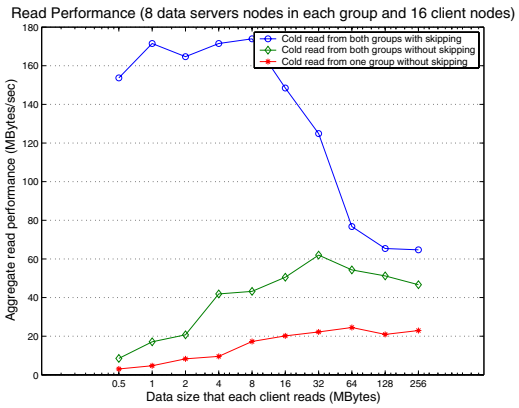

Figure 6. Cold read performance improvement by skipping one server with heavy disk load and reading the data from its mirror.

5.2. The improved hot read performance

Contrary to cold read, hot read can most likely find the data in the cache due to the aggressive design of the Linux operating system, which tends to use all the free memory as the cache buffer for the sake of minimizing disk accesses. This local optimization exploits the data locality exhibited in most application to alleviate the I/O bottleneck. Like PVFS, CEFT-PVFS servers utilize their local file system to store or retrieve all the data and cache the most recently visited data in their memory. As a result, the bottleneck of the peak aggregate performance for hot read is moved from the disk to the network.

Figure 7 plots hot read performance from both groups, under three approaches: 1) without stressing the network; 2) with the network stressed but without skipping; 3) with stressing the network and skipping. The network stressing is artificially added on one server node by repeatedly using a network benchmark, Netperf. In all measurements, each client reads a total of 16Mbytes data. When the total number of client nodes is small, the hot read performance does not show much difference among them since the bottleneck is on the clients' network. As the client number increases, the bottleneck gradually moves from the client side of the network to the server side network. Stressing the network of one server node reduces the peak hot read performance from 2GBytes/s to 1.25GByte/s. By skipping that network stressed node, the hot read performance is improved to 1.53GBytes/s, with an enhancement of 22.4%.

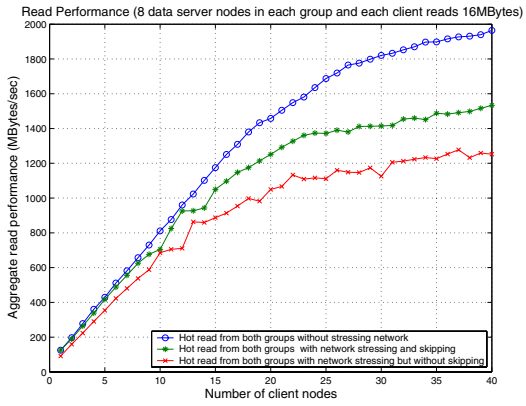

Figure 7. Hot read performance improvement by skipping the server with heavy network load and reading the data from its mirror.

6. Conclusions and future work

To alleviate the I/O bottleneck in cluster computing, PVFS aggregates the bandwidths of all the existing disks on the cluster nodes to provide high performance storage with the help of modern network technologies. CEFT-PVFS, an extension of PVFS, provides redundancy by mirroring all the server nodes to improve the reliability of PVFS while keeping a comparable write performance. In this paper, we proposed to interleave a single read request across both the primary nodes and their mirroring nodes simultaneously. The increased parallelism of I/O operations improves the peak performances of both the cold read and the hot read by as much as 100%. The read performance can be significantly degraded if some disks and/or the network

become heavily loaded, making them hot-spots. PVFS cannot avoid this degradation since there is only one copy of data in the servers. In CEFT-PVFS, on the other hand, there are two copies of data. If the system resources on the home node of one copy are heavily stressed, we can skip this node and read the data from the home node of the other (mirrored) copy. From the simple benchmark used in this paper, we observed that skipping hot-spot nodes can improve the cold read performance by a factor of up to 10, with a minimum improvement of 25%. The peak hot read performance can be improved by 22.4% in our experiments if one hop-spot node with heavy network usage is skipped.

As a possible direction for future work, we will evaluate the read performance of the proposed scheme in CEFT-PVFS in a more comprehensive and realistic benchmark.

7. Acknowledgements

This work was partially supported by an NSF grant (EPS-0091900) and a Nebraska University Foundation grant (26-0511-0019). Work was completed using the Research Computing Facility at University of Nebraska – Lincoln.

8. References

[1] Preliminary survey of I/O intensive applications, http://www.cacr.caltech.edu/SIO/, 1994

[2] J. D. Rosario and A. Choudhary, "High performance I/O for massively parallel computers: Problems and Prospects," *IEEE Computer,* vol. 27, no. 3, pp 59-68, 1994.

[3] P. H. Carns, W. B. Ligon III, R. B. Ross, and R. Thakur, "PVFS: A Parallel File System For Linux Clusters," in *Proceedings of the 4th Annual Linux Showcase and Conference*, Atlanta, GA, October 2000, pp. 317-327.

[4] J. Mache, J. Bower-Cooley, J. Guchereau, P. Thomas, and M. Wilkinson, "How to achieve 1 GByte/sec I/O throughput with commodity IDE disks," *Proceedings of SC2001 - 14th ACM/ IEEE Conference on High-Performance Networking and Computing* (refereed poster exhibit), 2001

[5] Nanette J. Boden, Danny Cohen, Robert E. Felderman, Alan E. Kulawik, Charles L Seitz, Jakov N. Seizovic, and Wen-King Su, "Myrinet: A gigabit-per-second local area network," *IEEE Micro*, 15(1): 29-36, 1995

[6] David A. Patterson, Garth Gibson, and Randy H. Katz, "A case for redundant arrays of inexpensive disks (RAID)," in *Proceedings of the 1988 ACM SIGMOD international conference on Management of data.* 1988, pp. 109-116, ACM Press.

[7] Yifeng Zhu, Hong Jiang, Xiao Qin, Dan Feng, and David R. Swanson, "CEFT-PVFS: A cost-effective, fault-tolerant parallel virtual file system", Technical report TR02-10-03, University of Nebraska – Lincoln, Oct. 2002.

[8] Joseph D. Touch, "Performance analysis of MD5," in *ACM Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, New York, NY, USA, Oct. 1995, pp. 77-86.

[9] Avery Ching, Alok Choudhary, Wei-keng Liao, Robert Ross, and William Gropp, "Noncontiguous I/O through PVFS," *Proceedings of 2002 IEEE International Conference on Cluster Computing*, September, 2002.

[10] R. Ross, D. Nurmi, A. Cheng, and M. Zingale, "A case study in application I/O on Linux clusters," in *Proceedings of SC2001*, Denver, CO, Nov. 2001, pp. 1-17.

[11] W.B Ligon III, "Research Directions in Parallel I/O for Clusters," keynote speech, *Proceedings of 2002 IEEE International Conference on Cluster Computing*, September 2002.

[12] Murali Vilayannur, Mahmut Kandemir, and Anand Sivasubramaniam, "Kernel-level caching for Optimizing I/O by exploiting inter-application data sharing," *Proceedings of 2002 IEEE International Conference on Cluster Computing*, Sept. 2002.

[13] A.W. Apon, P.D. Wolinski, and G.M Amerson, "Sensitivity of cluster file system access to I/O server selection," *Cluster Computing and the Grid 2nd IEEE/ACM International Symposium CCGRID2002*, 2002, pp. 183-192.

[14] R. B. Ross, "Reactive Scheduling for Parallel I/O Systems," Ph.D. dissertation, Clemson University, December 2000.

[15] Ligon, III, W.B., and Ross, R. B., "Server-Side Scheduling in Cluster Parallel I/O Systems," *Calculateurs Parallèles Journal Special Issue on Parallel I/O for Cluster Computing*, accepted for publication October, 2001.

[16] Yifeng Zhu, Hong Jiang, Xiao Qin, Dan Feng, and David R. Swanson, "Scheduling for improved write performance in a fault-tolerant parallel virtual file system," Technical report TR02-10-05, University of Nebraska – Lincoln, Oct. 2002.

[17] Prairiefire Cluster at University of Nebraska - Lincoln, http://rcf.unl.edu, Sept. 2002.

[18] Netperf benchmark, http://www.netperf.org, Oct. 2002.

[19] Bonnie benchmark, http://www.textuality.com, Sept. 2002.