Pipelining Network Storage I/O*

Lingfang Zeng**, Dan Feng**, and Fang Wang

Key Laboratory of Data Storage System, Ministry of Education
School of Computer, Huazhong University of Science and Technology, Wuhan, China
dfeng@hust.edu.cn, zenglingfang@tom.com

Abstract. In this paper, with introducing pipelining technology, network storage (e.g. network-attached storage, storage area network) and segmenting the reasonable pipelining stage are studied, which may have significant direction for enhancing performance of network storage system. Some experiments about pipelining I/O are implemented in the Heter-RAID. These results show that I/O pipelining scheduling can take advantage of pipelining features to achieve high performance I/O over network storage system.

1 Introduction

In computer architecture, pipelining is an implementation technique whereby multiple instructions are overlapped in execution; it takes advantage of parallelism that exists among the actions needed to execute an instruction.

Reference [1], [2] and [3] discuss the parallel I/O process policy. Reference [2] proposes a strategy for implementing parallel-I/O interfaces portably and efficiently. And they define an abstract-device interface for parallel I/O, called ADIO. Any parallel-I/O API can be implemented on multiple file systems by implementing the API portably on top of ADIO, and implementing only ADIO on different file systems.

Computer storage system has been the bottleneck in all computer systems. Parallel storage is one of the efficient ways to solve this I/O bottleneck problem. In our previous work [4], we provide the key technology to construct a parallel storage system is the implementation of the low I/O level parallel operations. To construct the high performance parallel storage system, the parallelism of storage devices is studied in detail: the research on the parallel operations of multiple storage devices within the multiple strings, the study of multithread I/O scheduling among the storage devices on a string as well as the related characteristic, parallel I/O scheduling and the analysis of disk array system based on the fact to reduce the computational complexity of the code and the cost of the parallel I/O scheduling. Reference [5] [6] [7] analyze some stage in cluster application environment. However, they are all limited in storage device and seldom study the pipelining storage in the whole network environment.

* This paper is supported at Huazhong University of Science and Technology by the National Basic Research Program of China (973 Program) under Grant No. 2004CB318201, National Science Foundation of China No.60273074, No.60303032, Huo Yingdong Education Foundation No.91068.

** Corresponding authors.

2 Network Storage I/O Pipelining

2.1 I/O Pipelining Strategy and Practice in Network Attached Storage (NAS)

Heter-RAID [9] is a network attached storage system which adopts virtual SCSI commands encapsulated by operation type (read/write), start sector, sector number and other information to execute I/O operation. There are multiple virtual SCSI commands in the Heter-RAID command queue within system resources. Pipelining producer/consumer policy divides the I/O cycle of virtual SCSI command into different stages and uses buffer technology to smooth work speed of different function components, the policy overlaps disk I/O and CPU computation to improve system performance.

According to overlap degree of I/O scheduling processes, pipeline operation of network attached storage system can be divided into two methods, one is fixed pipeline scheduling, and the other is flexible pipeline scheduling. Multi processes will execute by fixed scheduling sequence in fixed pipeline scheduling. Otherwise, flexible pipeline scheduling judges the completing sequence and overlaps multi processes freely. In the following Section 3, tests prove that the above two kinds of pipeline scheduling can improve bandwidth utilization when a lot of clients access Heter-RAID.

2.2 I/O Pipelining Strategy in Storage Area Network (SAN)

Figure 1 shows that the object-based storage [8] architecture is built from some components: compute nodes, metadata server (MS), storage nodes, high speed network and high speed networks etc. The MS provides some information necessary to directly access data in those storage nodes. Those compute nodes first contact with the MS and get the information about data. Then those compute nodes send a request to some storage node for wanted data. Obviously, in SAN, those storage nodes, such as object storage nodes, mirroring RAID and archive tape library, which facilitate to copy or move data from one storage node to the other.

Fig. 1. Object storage system I/O pipelining strategy in SAN

3 Experiment Results

Two kinds of experiments are proved in order to show the advantage of I/O pipelining strategy. One is performed in single storage node which is FC RAID configured 0-level. The other is tested in above mentioned NAS environment.

Intel Iometer for windows is adopted as test tool to test sequential access RAID in order to compare the write-only/read-only performance under non-pipelining policy and pipelining policy, respectively. Sequential write-only and sequential read-only are also presented changing with different *Outstanding* I/O (e.g. 1, 2, 4, 8 and 16) and different transfer request size (e.g. 512bytes, 2kbytes, 8kbytes, 32kbytes, 64kbytes and 128kbytes). The number of *Outstanding* I/Os represents IOMeter loads. Tweaking the number of simultaneous *Outstanding* requests affects the aggregate load the tested RAID is under. Test results are shown in Figure 2 and 3.

Fig. 2. Data transfer rate (sequential write-only/read-only) under non-pipelining policy/pipelining policy

Fig. 3. Average I/O response time (sequential write-only/read-only) under non-pipelining and pipelining policy

For the sequential write-only and sequential read-only, the data transfer rate under non-pipelining policy and pipelining policy are shown in Figure 2. For the sequential write-only and the sequential read-only, when the pipelining policy is adopted, the data transfer rate doubles those under non-pipelining policy, except for 512byte transfer data request in Figure 2 (there is not very noticeable) and some *Outstanding* 1 in Figure 2. It is because a depth of 1 is an extremely linear load which, combined with a 100% random test and is not representative of any real kind of access. Also, in Figure 3, the average response time has the similar effect under the circumstances.

Moreover, from Figure 2, the data transfer rate and the average response time all have obviously improved performance when the test changes from *Outstanding* 1 to *Outstanding* 2. But these have not had any effect changing from *Outstanding* 2 to *Outstanding* 16, the other way round, the average response time increasing.

4 Conclusions

The advantage of pipelining over parallelism is specialization (this is to say, different stage finishes different part of a task). The task can be partitioned and assigned to objects according to the specialized stages, and those stages can be kept busy by performing only their stage on repetitive tasks. In this paper, network storage I/O is discussed under pipelining technology. The experiment results show that pipelining solution improves the network storage I/O latency and enhances the storage system throughout.

References

1. J. Hsieh, C. Stanton, R. Ali. Performance evaluation of software RAID vs. hardware RAID for Parallel Virtual File System. Proceedings of Ninth International Conference on Parallel and Distributed Systems, pp:307–313. Dec. 2002.
2. Rajeev Thakur, William Gropp, Ewing Lusk. An Abstract-Device Interface for Implementing Portable Parallel-I/O Interfaces. Proc. of the 6th Symposium on the Frontiers of Massively Parallel Computation, pp: 180-187, October 1996.
3. ROMIO: A High-Performance, Portable MPI-IO Implementation. Website, 2005. <http://www-unix.mcs.anl.gov/romio/>
4. Feng Dan. Research on Parallel Storage System. [Dissertation for the Doctor Degree of Philosophy in Engineering]. Huazhong University of Science and Technology Library. 1997. (In Chinese)
5. A. Biliris, E. Panagos. A high performance configurable storage manager. Proceedings of the Eleventh International Conference on Data Engineering, pp:35–43. March 1995.
6. K. Hwang, Hai Jin, R. Ho, W. Ro. Reliable cluster computing with a new checkpointing RAID-x architecture. Heterogeneous Computing Workshop, pp:171-184. May 2000.
7. Kai Hwang, Hai Jin, Roy Ho. RAID-x: a new distributed disk array for I/O-centric cluster computing. The Ninth International Symposium on High-Performance Distributed Computing, pp:279-286. Aug. 2000.
8. M. Mesnier, G.R. Ganger, and E. Riedel. Object-based storage. Communications Magazine, IEEE, Vol 41, Issue: 8, pp. 84–90, Aug. 2003.
9. Fang Wang, Jiang-Ling Zhang, Dan Feng, Tao Wu, Ke Zhou. Adaptive control in Heter-RAID system. International Conference on Machine Learning and Cybernetics, pp.842-845, 4-5 Nov. 2002.