제 36 회 한국정보처리학회 추계학술발표대회 논문집 제 18 권 제 2 호 (2011. 11)

SATA 하드디스크의 I/O 성능 개선에 관한 연구

압둘 알판*, 김영진**, 권진백*
*선문대학교 컴퓨터공학과
**아주대학교 전자공학부
e-mail : abdul.arfan@gmail.com

A Study of I/O Performance Improvement in SATA Hard Disks

Abdul Arfan*, Young-Jin Kim**, JinBaek Kwon*
*Dept. of Computer Science and Engineering, SunMoon University
**Div. of Electrical and Computer Engineering, Ajou University

요 약

A SATA hard disk has been widely used in recent years and NCQ is one of its crucial features. Despite the development from IDE to SATA disk, there is still much room for improvement for a SATA disk. In addition, until now a hard disk is a black box to us and it is very hard to make research at the level of a disk controller. To enhance the performance of NCQ, we try to do I/O clustering over the requests, which combines multiple sequential requests into a single large one. To evaluate the effect of an I/O clustering mechanism, we created a simple but practical SATA hard disk simulator. Experimental results show that the proposed approach is effective in enhancing the I/O performance of a SATA disk.

1. Introduction

Serial Advanced Technology Attachment (SATA) is a recent technological advancement of the standard Integrated Drive Electronics (IDE) hard drive interface [1]. SATA employs a serial I/O communication bus instead of the parallel I/O bus used in Parallel ATA (or IDE). Most industry experts agree that SATA will replace parallel ATA technology in the dominant desktop, workstation, and servers [2].

There have been some efforts to increase the performance of the SATA disks. Recently, [3] seeks to increase the I/O performance of SATA disks by observing the effect of native command queuing (NCQ) and the disk scheduler in the operating system when they work together because the NCQ and a disk scheduler try to optimize disk I/Os without realizing each other. This situation can result in a strange behavior such as request starvation, which is a problem in OS I/O prioritizing, making a negative effect in some specific workloads.

I/O clustering is a mechanism that will merge the sequential requests into a single long request. I/O clustering is beneficial in enhancing the sequentiality of requests, improving the I/O performance of a disk consequently.

In this paper we propose application of I/O clustering at the level of a disk controller to increase the I/O performance of a SATA disk. Inspired by [4]-[8] we devise a simple but practical hard disk simulator so that we can evaluate our proposed I/O clustering technique.

2. I/O clustering

In [4], we can see the usage of I/O clustering to boost the overall performance as well as to enhance the energy efficiency of a disk-based storage system. I/O clustering will strengthen the sequential property of each request in a request queue within a device driver.

Fig. 1 shows the result of our experiment of measuring the sequentiality of a real disk I/O trace. We can see that even when the sequentiality threshold is 8 logical block address (lba), the percentage of total sequential requests due to I/O clustering still reaches about 50%. This shows that I/O clustering is useful in improving the sequentiality of requests.

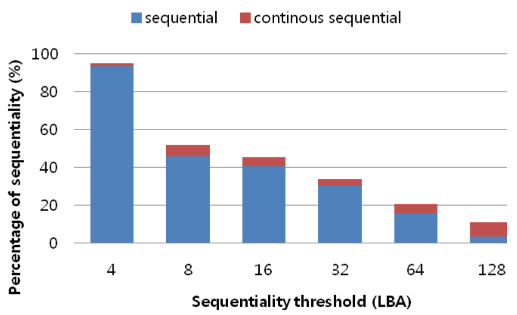

Fig. 1. Percentage sequential requests of a real disk I/O trace.

3. Simulation Architecture

To evaluate our suggestion, we created a simple but practical simulator of a SATA disk. The simulation should be able to read a trace file and also serve the requests. First, each request is inserted to a buffer in a host and if the queue in the disk is not full then we can move the request to the queue.

Fig. 2 shows the overview design of our simulator. The simulator will read a trace file line by line and parse the line,

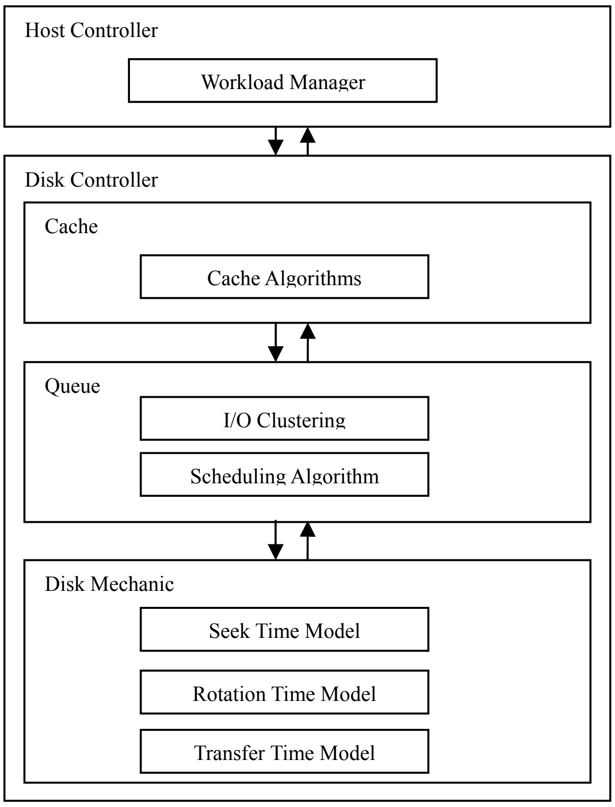

Fig.2. Overall structure of a SATA disk simulator.

reading the time, lba, size information of the request. Then, it will put the request into a buffer. If there is room in the queue, the request in the buffer will be sent to the cache. In this paper, we turn the cache off to check the effect of I/O clustering only.

Inside the queue, multiple sequential requests can be merged into 1 request if its size is more than an I/O clustering threshold. The merging will only occur if the total size of requests is less than a threshold which is called an I/O clustering threshold. This is to prevent the arbitrary requests from being merged indefinitely. In our experiment, we set the default threshold as 32 LBAs.

We have implemented 2 scheduling algorithms: FCFS (first-come first-served) and NCQ. In FCFS, the request is served based on the arrival time. FCFS is very simple and has no chance for a starvation to occur. However, if we introduce I/O clustering, there can be a chance for starvation to occur. This is why we introduce a time out for FCFS.

4. Simulation Results

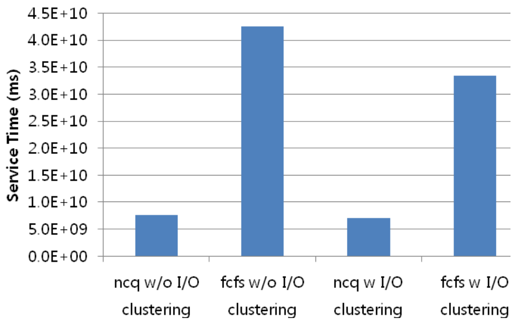

Fig.3. Service times with and without I/O clustering.

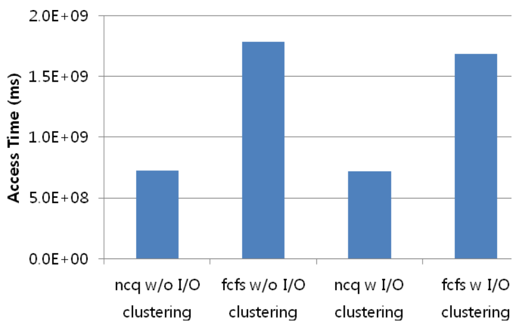

Fig.4. Access times with and without I/O clustering.

Fig. 3 and 4 show the result of I/O clustering. We can see that there are some improvements in the total service time and total access time. We used a trace in [4] and it has some sequential requests. Thus, I/O clustering can enlarge those sequential requests by merging them into a single request. In comparison with FCFS, we notice that NCQ has superior I/O performance irrespective of I/O clustering.

We achieved 8.8% improvement in the service time for NCQ with I/O clustering over NCQ without I/O clustering. For FCFS, we had 21% improvement over the case without I/O clustering. These results come from that I/O clustering has higher possibility to reduce the service time by merging several requests into a larger sequential one. The effect of I/O clustering is shown to be higher in FCFS compared to NCQ because the NCQ has already created a better scheduling for the requests than FCFS in the aspect of the sequentiality.

5. Conclusions

To improve the I/O performance of a SATA disk, we implemented I/O clustering at the level of a disk controller and evaluated it in a realistic SATA disk simulator. With a real disk I/O trace, we found that NCQ with I/O clustering have 8.8% improvement over NCQ without I/O clustering in the aspect of the service time. As future work, we plan to devise a proper cache algorithm for NCQ.

References

[1] Intel Corporation and Seagate Technology, "Serial ATA Native Command Queuing: An Exiting New Performance Feature for Serial ATA", white paper, 2003.
[2] Alliance Systems "Technology Report: Serial ATA" November 2003.
[3] Y.-J. Yu, D.-I. Shin, Hyeonsang Eom, and Heon Young Yeom "NCQ vs. I/O Scheduler: Preventing Unexpected Misbehaviors" ACM Transactions on Storage, Vol. 6, No. 1, Article 2, Publication date: March 2010.
[4] Y.-J. Kim, S.-J. Lee, K. Zhang, and J. Kim "I/O Performance Optimization Techniques for Hybrid Hard Disk-Based Mobile Consumer Devices", IEEE Transactions on Consumer Electronics, vol. 53, issue 4, pp. 1469-1476, Nov. 2007.
[5] D. M. Jacobson and J. Wilkes. "Disk scheduling algorithms based on rotational position", HP Technical report, 1991.
[6] L. Huang and T. Chiueh. "Implementation of a Rotation Latency Sensitive Disk Scheduler", Technical Report ECSL-TR81, SUNY, Stony Brook, March 2000.

[7] J. Zedlewski, S. Sobti, N. Garg, A. Krishnamurthy, R. Wang, "Modeling Hard-Disk Power Consumption", in Proc. of 2nd USENIX Conference on File and Storage Techologies, March 2003.

[8] Lars Reuther, and Martin Pohlack "Rotational-Position-Aware Real-Time Disk Scheduling Using a Dynamic Active Subset (DAS)", in Proc. of the 24th IEEE International Real-Time Systems Symposium, Cancun, Mexico, December 2003.