A Case Study: Performance Evaluation of a DRAM-Based Solid State Disk

Hitoshi Oi
The University of Aizu,
Aizu-Wakamatsu, Japan
hitoshi at u-aizu.ac.jp

Abstract

The speed gap between processors and hard disk drives (HDDs) has been spreading at a faster speed than the gap between processors and main memory devices. The most adopted methods for reducing this speed gap are caching and widening the data path (such as striping a file into multiple disks). Solid state disks (SSDs) are data storage devices that look like (ordinary) magnetic hard disk drives to the programs but are actually made of the same semiconductor devices and therefore are much faster than HDDs. In this paper, we present a case study of evaluating a DRAM-based SSD with OLTP workload which has a high bandwidth requirement and is write-intensive. We also discuss the suitable ways of using the SSD by taking its advantages and disadvantages.

1 Introduction

A solid state disk (or drive, SSD) is a storage device that looks like and is accessed in exactly the same way as a traditional (magnetic) hard disk drive (HDD) but is actually made of semiconductor memory devices. There are two types of SSDs: those made of flash memory devices and the others made of DRAM devices. The former type is non-volatile and is used, for example, as a replacement of a hard disk drive in a laptop computer or a portable music player [1]. On the other hand, the latter type has a low latency and a high bandwidth and is used for the systems that are critical to the performance of storage devices.

An issue that made DRAM-based high performance SSDs impractical was the higher price-per-bit of DRAM modules than that of magnetic disks. However, recent semiconductor technologies have pushed the price of DRAMs down and pulled their density up significantly and have made DRAM-based SSDs possible [2].

In this paper, we present our initial evaluation of a DRAM-based SSD under the on-line transaction processing (OLTP) workload. In the OLTP workload, there are frequent updates and deletions of database tables. Therefore, simply caching the database by the main memory of the server does not significantly improve the performance. In the next section, the experimental environment is described. In Section 3, the results of experiments are evaluated. We then discuss the suitable usage of SSDs by taking the advantages and drawbacks of the SSDs into consideration and conclude the paper.

2 Experimental Environment

Table 1 shows the software and hardware components used in the performance evaluation. For the SSD, we use a 16GB GigaExpress from FujiXerox [2]. It is connected to the server via PCI Express. For the comparison purposes, we also use four SCSI HDDs (15000rpm) configured to a RAID-0 (striped) drive by a hardware RAID controller.

Software	
Operating System	CentOS 4.4
Kernel	2.6.9
Benchmark	OSDL DBT-2 (V 0.40)
DBMS	PostgreSQL (V 8.2.4)
Hardware	
CPU	Xeon 3GHz $\times$ 2
Memory	2GB
HDDs	SCSI 15Krpm $\times$ 4
SSD	GigaExpress (16GB)

Table 1. Evaluation Environment

As the benchmark program, we use the OSDL DBT-2, which is an OLTP workload based on the TPC-C benchmark specification [4]. It models the activity of wholesale supplier and is update intensive. For the rest of this paper, we will use the terminologies of the TPC-C to describe and analyze the evaluation results. However, it is important to notice the differences between TPC-C and DBT-2. While DBT-2 is implemented based on the specification of the TPC-C, its purpose is to analyze and evaluate the behavior of hardware and software components under the OLTP workload. In contrast, TPC-C compares the performance of commercial products and TPC rigorously inspects the published results.

The primary scale factor of the TPC-C is W, which is the number of warehouses in the simulated wholesale supplier. The sizes of database tables and the number of clients connected to the database server are proportional to this figure. There are five types of transactions performed on the database and its performance metric is tpmC, the number of "New Order" transactions processed by the system in a minute. It is update-intensive workload and changes made on the database tables must be recorded in a special file, called a transaction log. Therefore, it has a high proportion of write access to the disk as we will see later. These database updates must be committed, that is, must actually be written to the disk rather than modified in the write buffer. As a result, TPC-C, (and DBT-2) requires a high throughput performance for the disk subsystem. For each W and disk system, we run DBT-2 for 10 minutes after the system has reached the stable state. All the figures presented below are averaged over this 10-minute period.

3 Performance Evaluation

In this section, we present and analyze the experimental results. First, let us take a look at primary performance metrics in the TPC-C specification, which is the number of New Order transactions processed per minute (NOTPM) [1] by the system (Figure 1). For the both HDD and SSD cases, NOTPM grow linearly up to around $W = 80$. For the HDD case, however, NOTPM does not increase beyond $W > 80$. On the other hand, for the SSD case, it further grows linearly up to $W = 132$, which is the maximum W bound by the capacity (16GB) but not by the performance.

The response times for the New Order transactions are shown in Figure 2. These response times are measured at 90th percentile (i. e. 90% of transactions are finished within these response times). Please note that in our measurements, the response time for the Deliver transactions grew faster than that for the New Order transaction and broke the response time conditions defined in the TPC-C specification (less than 5 seconds for transactions other than Stock Level). However, the fraction of Deliver transactions is much smaller than that of New Order transactions (4% versus 44%). Thus, we use the response time of New Order transaction for analysis as it should better reflect the system behavior.

For a smaller W, the response time in the HDD case is acceptable, however, it increases constantly up to around

[1]We avoid the use of previously mentioned term tpmC since it is an official performance metrics defined in the TPC-C specification

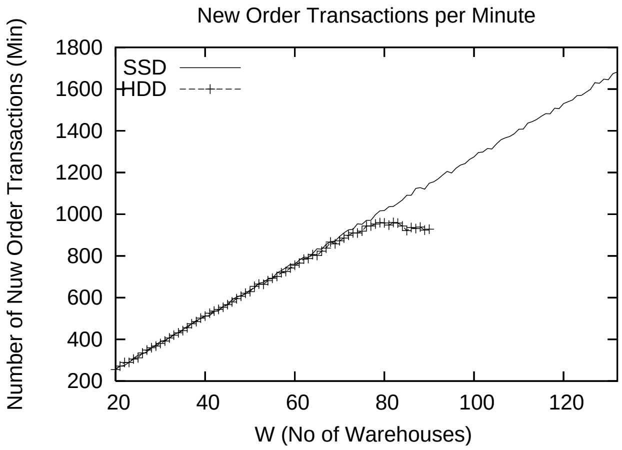

Figure 1. Number of New Order Transactions Per Minute

$W = 70$ and then draws a steep upward curve beyond that point. On the contrary, the response time in the SSD case is so small that we can hardly see it in the figure. It starts with 0.015 second at $W = 20$ and increases only up to 0.026 second at $W = 132$.

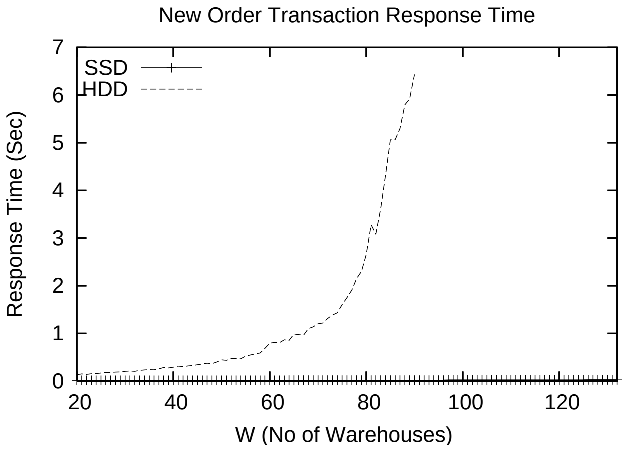

Figure 2. New Order Transaction Response Time (90th Percentile)

Figure 3 shows the I/O transfer rates for the HDD and SSD cases. The curves for the I/O rate in the HDD case show a similar trend as that of NOTPM in Figure 1 as both curves shows saturation at around $W = 80$. As explained earlier, DBT-2 is an update-intensive workload and we understand this characteristic from the higher fractions of write data transfer rates in the figure. Although it is not clearly visible in the figure, however, the mixture of read and write data rates changes as W is increased. For $W = 20$, the fractions of read data transfer are 4 % (HDD) and 6% (SSD). These numbers increase up to 20% for SSD at $W = 132$ (for the HDD case, it was 15% at $W = 90$ which is the largest W we measured). We may interpret this result as follows. For a larger W, the number of updates which must actually be written to the disks is increased. The number of updates is proportional to the number of transactions which in tern is proportional to W as seen in Figure 1. We also have a larger number of read accesses for a larger W. However, the larger W further means that a larger fraction of database tables do not fit in the main memory used as the disk cache. Therefore, in the case of read access, increases in both the number of transactions and the disk cache misses may have caused the super-linear increase of the read I/O transfer rate.

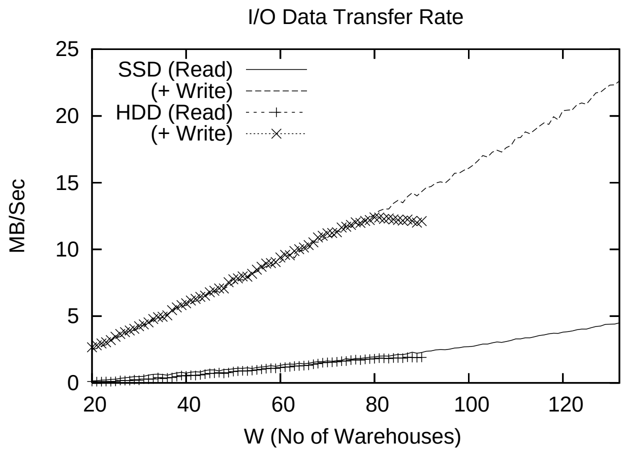

Figure 3. I/O Data Transfer Rates

Finally, the breakdowns of CPU time for the HDD and the SSD cases are shown in Figures 4 and 5, respectively. User, Sys and Wait in the figures stand for the fractions of CPU time spent in the user mode, the kernel mode and the I/O wait respectively. In the case of HDD, about 5% of CPU time is spent for I/O wait at $W = 20$ and reaches 45% for $W = 80$. At this point, only 14% of CPU time is spent for the useful tasks. The execution of DBT-2 is much more efficient in the SSD case. Up to around $W = 60$, the fraction of I/O wait is nearly zero and even for the largest $W = 132$ it is less than 2%. It should also be mentioned that, for the same W, the fractions of user and system modes are slightly higher in the HDD than in the SSD. While further investigation is required to understand the details of this behavior, a possible explanation is as follows. In the case of HDD, there are more pending disk access requests in the system. These pending requests cause more context switchings and synchronization operations which result in higher (non-I/O wait) CPU time fractions.

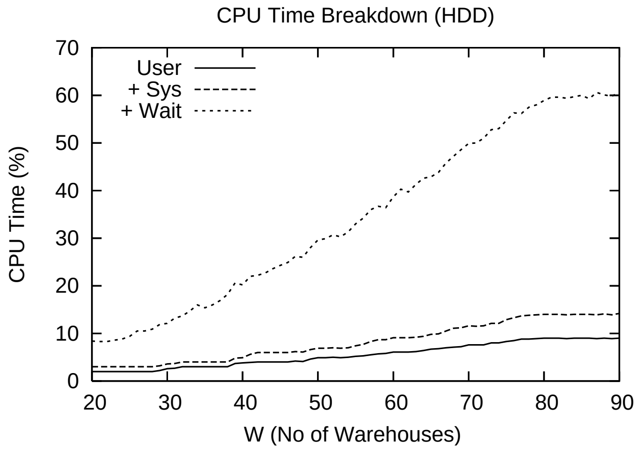

Figure 4. CPU Time Breakdown (HDD)

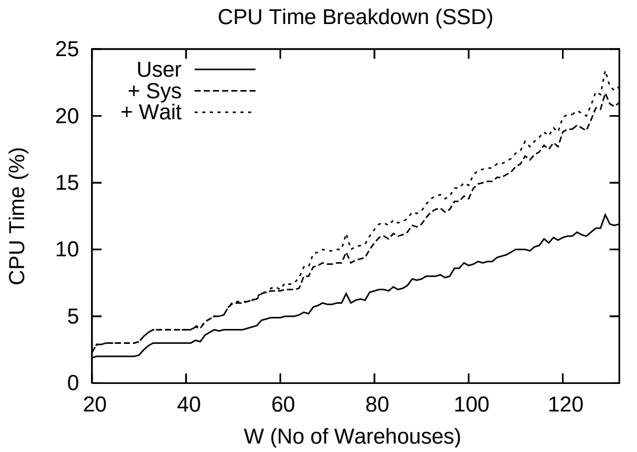

Figure 5. CPU Time Breakdown (SSD)

4 Discussions

In terms of performance, both latency and bandwidth, DRAM-based SSDs are superior than traditional magnetic disks. As we have seen in the previous section, the throughput of the SSD used in this paper did not saturate even for $W = 132$ which was the largest DBT-2 scale factor that came from its capacity limit. This high throughput resulted in almost no CPU time spent in the I/O wait mode. However, this high performance comes with a very high price-per-bit figure: roughly speaking, we have to pay about two orders of magnitude of money for the same amount of storage capacity [2].

For reliability, DRAM-based SSDs have two conflicting aspects. It is volatile by nature and the loss of power means the loss of information. However, the risk of power failure can be reduced by backing up the power source such as UPS. Since SSDs are built in a separate cabinet from host computers, unlike the disk cache using the main memory of the host, power failure in host computers does not mean the loss of data in SSDs.

In addition, unlike magnetic disks, SSDs have no mechanical components. Therefore, it is expected that the probability of mechanical breakdown is quite low.

Another issue is the capacity of SSDs. As we saw in the previous section, the SSD reached at its capacity limit before its I/O throughput saturated under the OLTP workload. A larger SSD is possible in theory but not practical due to its high cost in the current and near future technologies.

With the strengths and weaknesses of the SSDs described above, there are several cases where the use of SSDs are suitable. Figure 4 indicates that much higher amount of CPU time is wasted for I/O wait than useful task when HDDs are used for a high I/O throughput workload. However, the abundant bandwidth of SSDs places nearly zero pressure on the CPU due to the pending I/O requests even when a high I/O throughput is required. From this observation, it is expected that a mixture of CPU-intensive and I/O-intensive applications can better utilize the CPU time of a time-shared server with the help of SSDs.

In the same line of thinking, system-level virtualization software, such as Xen or VMware, may benefit from the SSDs. Currently, a virtualized system is typically equipped with a large number of HDDs and a small number of HDDs are assigned to each virtual machine (VM). Each VM tries to reduce the intervention of the virtual machine monitor (VMM) by directly dispatching the I/O requests within the VM to the assigned HDDs. In the case of SSDs, a single device may be powerful enough to handle the I/O requests from all VMs in the system. In such a case, however, both hardware and software enhancements will be desired to provide the features specific to the virtualized systems (such as address translation services) [5].

While the reliability of SSDs can be enhanced by backing up the power source as mentioned above, there may be a situation where backing up the file system is also required. In this case, we may run an incremental backup program in parallel to other applications. In addition, we may compress the resulting backup files utilizing the spare CPU power. With this scheme, it is expected that the bandwidth requirement for the backup devices would be lowered and we may use inexpensive HDDs for storing backup files. Finally, more traditional usage models, such as a scratch disk for temporary files or the secondary disk cache (the main memory in the host being the primary cache), are also possible.

[2]We also have to mention that high performance storage devices are inherently expensive and SSDs are not the only cases. For example, a SCSI disk costs around 20 times more than an SATA disk for the same capacity at the time of writing

5 Conclusions

In this paper, we have presented an initial performance evaluation of the SSD under the OLTP workload. The SSD used in this work significantly outperformed the conventional magnetic disk drive consisting of four SCSI drives configured to a RAID-0 drive. However, we could not fully utilize the high performance of the SSD due to its capacity limit. While the OLTP is a representative workload for a server computer, there are many other types of workloads that have different I/O throughput and capacity requirements and we need to evaluate SSDs with such workloads.

Acknowledgment

This work is supported in part by the University of Aizu Competitive Research Funding scheme.

References

[1] George Lawton, "Improved flash memory grows in popularity", IEEE Computer, Vol. 39, Issue 1, pp16–18, 2006.

[2] "Fuji Xerox Commercializes Solid-State Disk Device Incorporating Optical Technology", http://www.fujixerox.co.jp/eng/headline/2006/0627_gigaexp.html

[3] "OSDL Database Test 2", http://old.linux-foundation.org/lab_activities/kernel_testing/osdl_database_test_suite/osdl_dbt-2/ , The Linux Foundation.

[4] "TPC BENCHMARK^{TM} C, Standard Specification, Revision 5.9", Transaction Processing Performance Council, June 2007

[5] "I/O Virtualization", http://www.pcisig.com/specifications/iov/ , PCI-SIG, 2007.