A Better Look at Smart Response Technology

Luke Plewa, *Student, Cal Poly San Luis Obispo*

Abstract—Smart Response Technology is a means to take advantage of high speed solid-state disks (SSDs) while avoiding the large, expensive costs of a big solid-state disk. This is done by using the SSD as a cache for a much larger hard disk drive. This paper seeks to discover optimal SSD sizes, write schemes, and cache replacement algorithms given a generic setup, so that it may better apply to a range of implementations. These conclusions can then be used to build a replacement to Intel's well-tested and marketed solution.

Keywords—Computer Architecture, Solid-State Disk, Hard Drive Disk, Cache, Smart Response Technology, Cache Acceleration Software, SSD Caching, DiskSim.

1 Introduction

THIS paper is intended to serve the general public in demonstrating what configurations may be optimal for a SSD as cache scenario. The large majority of published papers reporting on SSD Caching have been done on Intel-specific SSDs and software. Hopefully the information exposed in this paper can help others make responsible decisions for building generic SSDs as a layer between the HDD and cache.

March 21, 2014

2 Background Information

2.1 Smart Response Technology

Smart Response Technology (SRT) was originally called SSD Caching before being introduced by Intel in 2011. It features a SSD as a last-level cache built on top of a HDD as a main disk. [5]

2.2 Cache Acceleration Software

Intel's Cache Acceleration Software (CAS) is a hardware and software solution that uses Smart Response Technology implementations. CAS determines which data is the most important and places it on the smaller SSD rather than the bigger, slower HDD. [5], [6] The original CAS is designed for data centers, but the Cache Acceleration Software for Workstations (CAS-W) was released in late 2013. Data centers is the main focus of CAS, because the cost of having large SSDs to store massive information is less feasible than on a workstation. Regardless, there seem to be applications for the CAS-W as well. This paper hopes to determine optimal substitutes for CAS through cache replacement algorithms, while still using a variation of Smart Response Technology.

3 Competing Technologies

Competing architectures include HDD only, SSD only, and Hybrid drives. In terms of performance, the relationship in terms of slowest to fastest is HDD only, Hybride drives, SRT, and SSD only. [6] Because this relationship is already heavily tested, it is not explored within this paper. The focus is instead on SRT in relation to other SRT implementations.

3.1 HDD Only

The benefits of having a Hard Disk Drive as the main disk without the Solid-State Drive lie in the costs. While this implementation will surely underperform compared to having a Solid-State Drive as a cache, the cost per byte of data is undeniable cheaper. It is also a simpler implementation as there are fewer components.

• *L. Plewa is with the Department of Computer Science, Cal Poly San Luis Obispo, San Luis Obispo, CA, 93405. E-mail: lplewa@calpoly.edu*

Manuscript received March 21, 2014.

3.2 SSD Only

The benefits of having a Solid-State Drive as the main disk without a Hard Disk Drive lie in the performance. It is definitely faster to not have to rely on the communication time between SSD cache and HDD (even in a write-back scheme) as well as any I/O Operations on the HDD. This comes at a higher dollar cost, which is the purpose of SRT as data centers cannot feasibly be built as SSD only given dollar cost restraints.

3.3 Hybrid Drives

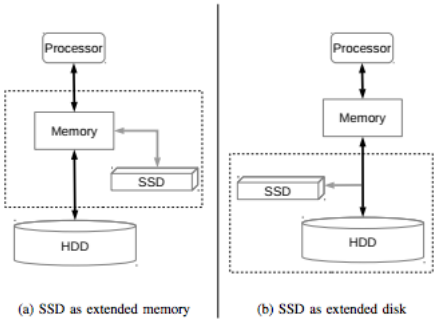

Fig. 1. Drive Configurations [4]

A hybrid drive implementation uses both a SSD and HDD as long term storage. The SSD would statically host the operating system boot block as well as common applications (chosen and specifically installed by the user). The HDD would store not as commonly used programs and data that is infrequently accessed. [4] Fig. 1 demonstrates hybrid drives (b) compared to SRT (a). The benefit compared to Smart Response Technology is the lack of software used to maintain the configuration between the SSD cache and the HDD main disk. This comes at the cost of not being able to dynamically control what data is stored on the SSD for improved operations. So while the design may be more simple, in an ideal Smart Response Technology scenario the SSD would be used more efficiently than the SSD on the Hybrid implementation. This is in terms of saturation of the SSD (programs are user-chosen for hybrid drives and likely unfilled) and choosing what is hosted on the SSD (a smart algorithm may choose programs better than the user). Because Hybrid drives use the SSD as an extension, there is actually more storage available, but less redundancy.

4 Experimental Configuration

4.1 DiskSim

This paper relies on DiskSim v4.0 to provide disk system simulation. DiskSim was first developed at University of Michigan, but was enhanced at Carnegie Mellon University. DiskSim is written in C and requires only a few POSIX interfaces, making it more portable than most simulators. DiskSim's brightest feature for this paper includes the ability to include any device as the cache for another device. It does so using the disksim cachedev structure, which allows additional cache configurations such as size, write scheme, and flush policy. [2] This allows the developer to declare a SSD as a cache for a HDD. Version 4.0 of DiskSim was released in late 2008, but has been updated by third parties since then via patches. Two such patches include Microsoft's SSD Extension and Yonggang Liu's 64 bit patch.

4.2 Microsoft's SSD Extension

Microsoft researchers wrote a patch for DiskSim in 2008, so that they may better understand SSD performance. This research was presented at the 2008 USENIX technical conference. This paper explores the interal organization of SSDs, and their patch reflects upon the various configurations they tested. [1] DiskSim v4.0 does not natively support SSDs, but this patch introduces a model for testing SSDs.

4.3 Yonggang Liu's 64 bit patch

Hosted under github account: https://github.com/myidpt. DiskSim is designed to be run on 32 bit machines only. This patch changes the configuration so that DiskSim may be run on 64 bit machines. These changes include some globals and data type sizes.

5 Experimental Results

5.1 General Statements

The experimental results exhibited in this paper should only be used comparatively to other results internal to this paper. First of all, DiskSim's example I/O requests are simpler than most benchmarks and do not reflect real world scenarios well. [2] Also, in order to gain a general understanding of Smart Response Technology, the SSD and HDD configurations are not specifically accurate to any given device, but are relationally accurate in terms of SSD versus HDD speed ratios. The general rule of thumb used is a 3x performance increase in accessing the SSD compared to the HDD. [3] While some specific devices may be slow or faster than this ratio, a general survey has proved that this is an acceptable performance differential. [3], [1] That said, the configuration used in this paper depends upon an SSD level-one cache in order to avoid any discrepencies caused by using a four level cache structure. This is the configuration that Principled Technologies used /citePrincipled, but this was in relation to a cacheless architecture for HDD and SSD only. This level-one cache implementation is used in relation to other SSD cache configurations.

5.2 SSD Size

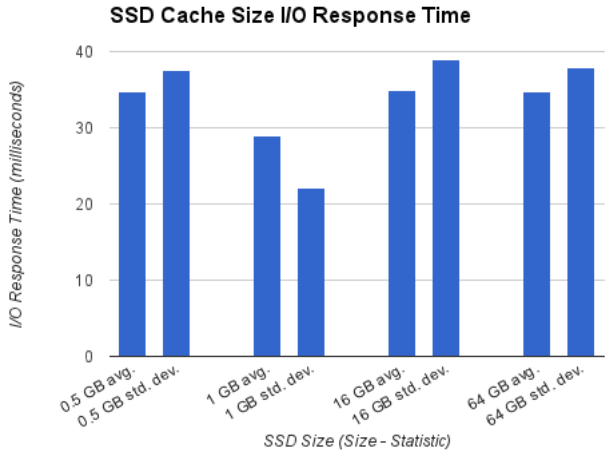

Fig. 2. Varying SSD Cache Sizes

Intel's Cache Acceleration Software is limited to using 64 GB of SSD space as cache. A SSD

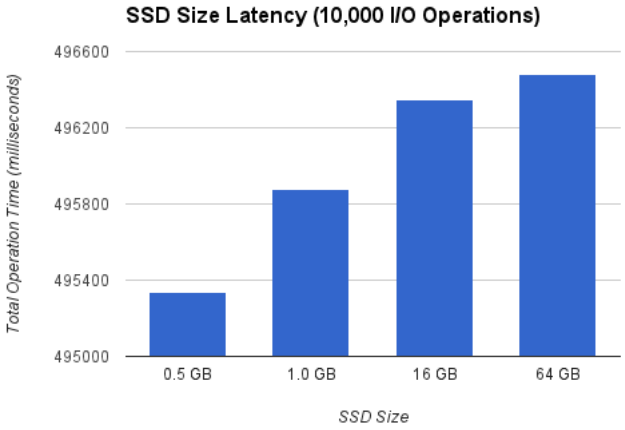

Fig. 3. Varying SSD Sizes

of higher storage is usable, but SRT will only allocate 64 GB and the rest will go unused. This is a weird limitation and unfair specification to put onto a piece of software, so the effects of SSD cache size is pursued.

Fig. 2 displays the comparison between different SSD sizes compared to their average and standard deviations in I/O Response Time. It is interesting to note the trend, as there is an increase from 0.5 GB to 1 GB in performance, but a drop increase to 16 GB and 64 GB. To further understand this, fig. 3 displays the latencies of different SSD sizes. Out in the field, SSD size should not drop the performance of SSD. [3], [1] Unfortunately, the configuration on DiskSim displays a drop in performance with increases in size. Taking this into consideration, and noting the increase in performance from 0.5 GB to 1 GB I/O Response Time, it can be concluded that an increase in SSD cache size improves the I/O Response Time. The 16 GB and 64 GB SSDs are likely not being filled in these tests, as there is little difference between 16 GB and 64 GB.

In terms of SSD size, it is worth noting that the purpose of SRT is to balance the cost of the SSD versus the size of the SSD. In terms of Intel's solution, a 64 GB SSD limitation makes sense in terms of cost. Anything larger and it would make more sense to use the SSD as a main disk. However, it seems unfair to

enforce this limitation on users. In terms of data centers (the focus of CAS), a SSD larger than 64 GB may be useful when handling 10+ TB data sizes. Of course, this is specific to the amount of hot data that would benefit from being on the SSD. Based on these experimental results, large SSD caches should be supported by software wishing to implement SRT in competition with CAS. The only limitation here should be the dollar amount for the SSD.

5.3 Enhanced and Maximized Modes

Fig. 4. Varying Write Policies

Intel's Cache Acceleration Software features two modes: Enhanced and Maximized. Enhanced mode features a write-through write policy for the SSD cache, while the maximized mode features a write-back policy. [5] The write-through policy writes to both SSD and HDD upon updates. The write-back policy will only flush the SSD data onto hard disk when it is demanded. This makes the obvious distinction that using Maximized mode will provide greater performance, but does not benefit from the redundancy and coherence that comes from Enhanced mode.

Fig. 4 displays the different write policies reflected across different SSD sizes in terms of average and standard deviation I/O Response Time. The 0.5 GB SSD features a 4.87

5.4 Cache Replacement Algorithm

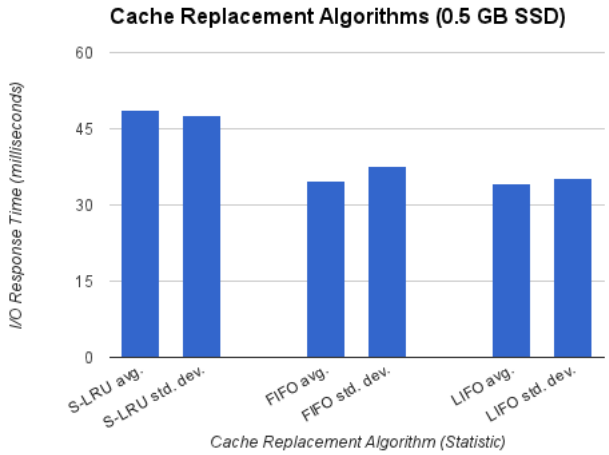

Fig. 5. Varying Cache Replacement Algorithms

Intel's Cache Acceleration Software uses its own enterprise algorithm that is not publicly available. Because of this, general cache replacement algorithms are tested to determine which one would best fit given the unorthodox (extremely large) size of the SSD compared to regular cache implementations.

Fig. 5 displays the three cache replacement algorithms used. While First-In-First-Out (FIFO) and Last-In-First-Out (LIFO) are relatively similar in terms of I/O Response Time, Segmented-Least-Recently-Used (S-LRU) performs significantly worse. This may be due to the increased complexity S-LRU introduces or segmentation size used in this experiment. The protected segment used here is exactly half the size of the SSD size, and the generators used to run processes likely do not access these protected segments often.

6 Conclusion

Determined from the experimental results, it would seem that the best SSD Caching configuration would feature the largest SSD cache available at a good cost which features a write-through policy and a FIFO cache replacement algorithm. Considerations must be taken towards the simulated nature of these conclusions, because these are simulated runs and not realtime user driven studies. An actual user

may draw different results based on their erratic and unpredictable behavior or their static behavior. Regardless, these conclusions shed more light on possible SRT implementations that vary from Intel's solution.

ACKNOWLEDGMENTS

The author would like to thank Dr. John Seng for teaching computer architecture (CSC 520) at Cal Poly San Luis Obispo. The progress made within this paper would not be possible without his infectious passion towards computer architecture. Also thanks to the Computer Science Department at Cal Poly San Luis Obispo, whose facilities and administration made this possible.

REFERENCES

[1] N. Agrawal, V. Prabkharan, T. Wobber, J. Davis, M. Manasse, and R. Panigraphy, *Design Tradeoffs for SSD Performance*, Silicon Valley, CA: Microsoft Research and University of Wisconsin-Madison, 2008.

[2] J. Bucy, J. Schindler, S. Schlosser, and G. Ganger, *The DiskSim Simulation Environment Version 4.0 Reference Manual*, Pittsburgh, PA: Carnegie Mellon University, 2008.

[3] G. Gasior, *OCZ's Vector SSD Reviewed*, The Tech Report, 2012.

[4] S. Huang, Q. Wei, J. Chen, C. Chen, and D. Feng, *Improving Flash-based Disk Cache with Lazy Adaptive Replacement*, Wuhan, China: School of Computer, Huazhong University of Science and Technology, 2013.

[5] Intel Corporation, *Driving Performance: Intel Cache Acceleration Software*, USA: Intel Corporation, 2012.

[6] Principled Technologies, *Intel Cache Acceleration Software with the Intel Solid-State Drive 910 Series*, Durham, NC: Principled Technologies, 2013.

[7] Y. Zhang, J. Curry, F. Roxas, and B. Donie, *Accomplish Optimal I/O Performance on SAS 9.3 with Intel Cache Acceleration Software and Intel DC S3700 Solid State Drive*, Chandler, Arizona: Intel Corporation, 2012.

Luke Plewa Luke Plewa is a student at Cal Poly pursuing a joint BSMS degree for Computer Science. His other works include progress made in OpenGL video games, Android applications, and machine learning.