Design and Development of Enterprise Storage Systems with Hybrid Disks for Future Datacenters

Wang Yonghong Wilson, Lim Chun Teck, Kyawt Kyawt Khaing, and Yong Khai Leong
Data Center Technologies Division, Data Storage Institute, Singapore
Email: Wang_yonghong@dsi.a-star.edu.sg

Abstract — A hybrid disk consists of a small amount of Non-Volatile flash Memory (NVM) as fast storage media to complement the rotational magnetic recording disk, normally considered slow but with large storage capacity. In this paper, we present the design and development work of an enterprise hybrid disk drive storage system for future datacenter. We also design a new type of dynamic cache algorithms that can simplify cache management architecture and enhance operation performance. Evaluation with enterprise storage workloads shows that the average IOPS (Input/Output per second) can be increased up to 300% and can outperform the IOPS of high speed enterprise SAS (Serial Attached SCSI) disk drive for certain workloads. Another advantage of hybrid disk drive is power saving, for which the power consumption can be reduced by more than 50%, compared with the enterprise hard disk drive. We envision that hybrid disks can be a cost-effective replacement of high speed hard disks in enterprise storage systems that empower datacenters with large storage capacity, high performance and low energy consumption.

Index Terms—Datacenter, storage system, hybrid disk drive, non-volatile cache management, interval-tree

I. Introduction

In order to manage massive amount of storage in datacenters with performance demand, today's enterprise storage systems heavily utilize high RPM (Revolutions per Minute) Hard Disk Drives (HDD) as main storage media to provide high IOPS. Typically those are SAS drives with 10,000 or 15,000 RPM spindle speeds. They have average read/write IOPS in the range of 175 to 210 [1]. However enterprise disk manufacturers have hit physical limitation in increasing disk spindle speed beyond 15,000 RPM [2]. High RPM HDDs also consume more power, and large number of such drives can create financial and environmental burden on datacenter operations.

In order to overcome the limitation of HDDs, Solid State Devices (SSD) have been adopted to serve as high performance cache and, together with HDDs, form a hybrid storage system for enterprise applications [3]-[8]. In such storage system, SSDs, with small amount of storage capacity, function as separate storage devices and only keep the data that needs high performance input/output requirement. And HDDs are still used to store large amount of data. However, such kind of hybrid storage systems leaves the control complexity of data movement between SSDs and HDDs to system controller. In this paper, we introduce a new type of device called hybrid disk drive to enterprise storage system and demonstrate that it has promising features to address the problems of high RPM HDDs.

A hybrid disk drive combines existing rotational magnetic media with a small amount of Non-Volatile Memory (NVM) cache consisting of NAND (Negative-AND) flash chips originally developed for consumer devices like laptops and desktops. Hybrid disk was employed to speed up Operating System (OS) boot time and application performance. For example, Microsoft Windows OS developed a feature called ReadyDrive [9] to leverage the flash inside hybrid disk drive to reduce boot time. However, as the frequency and amount of I/O requests is low due to the nature of consumer operation model, the impact of a hybrid disk in such environment is arguably limited. A user may not frequently boot up his computer system to enjoy performance improvement. With the study of data traffic characteristics in the environment of enterprise storage system, we conclude that a hybrid disk is more suitable to work in the system that support larger number of users or applications, which could generate more random and small size I/O requests that can efficiently utilize NVM provided inside a hybrid disk drive.

Our objective of this paper is to position that hybrid disk drive based storage systems can be complementary to existing SSD based storage systems with additional advantages, more importantly, hybrid disk drives can be used to replace high spindle speed HDDs. With hybrid disks in storage systems, the rotational speed of a magnetic disk can be controlled to relatively low (7200 RPM or less) level, while the achievable performance or IOPS can be close to or greater than a high speed Enterprise disk by utilizing embedded NVM cache. This eliminates the need for increasing RPM of the disk. The high power consumption can also be addressed as an additional benefit since a disk with low rotational speeds consumes less power. Even if the lower disk spindle speed and the presence of NVM cache are characteristics of such device, delivering higher performance still requires cache management algorithms catered to the

Manuscript received May 25, 2015; revised December 8, 2015.
The work was supported by the Agency for Science Technology and Research (A*STAR, Singapore) under Grant no: 1121720013.
Corresponding author email: wang_yonghong@dsi.a-star.edu.sg
doi:10.12720/jcm.10.12.1012-1019

need of utilizing the internal NVM cache efficiently. To this end we have developed cache management algorithms to control the placement/eviction of data to/from the internal NVM cache. Through experiments with enterprise workloads, we show that our algorithms can improve average IOPS by around 2 to 3 times compared to a baseline SATA (Serial ATA) disk. For certain workloads it can outperform the IOPS of Enterprise SAS disk. The potential for replacement of SAS disks with hybrid disks is attractive from our evaluation result. The main contributions of our work are summarized as:

1) Design of cache monitoring meta-data structures to track hot data on the hybrid drive;
2) Design of cache manager with placement and eviction algorithm;
3) Development of the Enterprise hybrid storage system to demonstrate the advantages of the hybrid disk cache manager.

The rest of the paper is organized as follows. Section II presents our motivation of applying hybrid disk drives to build new type of storage system to address the problem with the existing cache management solutions. Then in Section III, it presents the design and development of hybrid disk drive cache management algorithm for data placement to and eviction from hybrid disk drive. This is followed by Section IV describing the evaluation results with an enterprise storage system prototype that is integrated with our hybrid disk drive cache manager. The Section V concludes the paper.

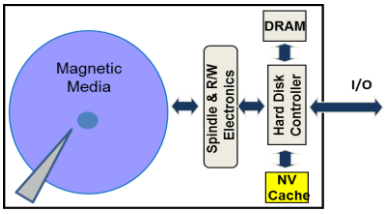


Fig. 1. Schematic of a hybrid disk

II. Hybird Disk Drives for Enterprise Storage

In this section we first present a short background introduction on hybrid drives and existing hybrid storage systems with the integration of HDDs and SSDs. We then describe how the system architecture with hybrid disks differs from exiting hybrid storage system. We also explain why current available cache solutions are not idea for hybrid disk drive based storage system.

A. Background on Hybrid Disk and Hybrid Storage Systems

The schematic of the components inside a typical hybrid disk drive is shown in Fig. 1. The main component is still the rotating disk which serves as the primary storage media and the NVM with NAND flash which acts as a persistent cache media. The DRAM component acts as a small temporary read cache typically with capacities of a few Megabytes. The ATA-8 specification [10] provides commands for controlling data placement/eviction to/from this internal Flash. The data can be served from either the NVM during a cache hit or from the magnetic media during a cache miss. While not shown in the figure, the hybrid drive also maintains a mapping table to map block address on the magnetic media to the corresponding location or block address on the flash for all those cached data.

Currently most of the hard disk drive manufactures have launched hybrid disk drive products in the market. However, there is a limitation of today's available hybrid disks. They mostly adopted self-learning (or black-box) method of applying NVM cache to application data, which are lack of the capability to utilize application-specific or storage system level caching characteristics which motivates us to design and develop cache management algorithm in storage system controller in order to apply hybrid disks for enterprise storage applications.

On the other hand, many existing hybrid storage systems integrate separate solid-state storage, or NAND flash SSDs to improve the system performance. MixedStore [3] is an early proposal to use both HDDs and SSDs to form a storage system. It provides both long term and dynamic resource planning model and mechanism to effectively utilize SSD storage. In order to achieve the design target, the system controller has to manage both HDDs and SSDs separately and schedule data movement explicitly, furthermore MixedStore even provides comprehensive flash cache management functionalities, such as FTL (Flash Translation Layer), and wear-leveling control, thus the operation overhead and complexity are significant. HyStor [4] creates three major components, namely remapper, monitor, and data mover to manage HDDs and SSDs in a hybrid storage system. It maintains a single Logical Block Number (LBN) space that is originally mapped to physical block address in the HDDs. When the data blocks are moved to SSDs, the LBNs are remapped to SSDs which means the associated physical blocks address are mapped in SSDs. HyStor also maintain a block table to monitor the data access cost, which is used to indicate the randomness and access frequency of a "hot" data block. Building such block table requires Operating System (OS) buffer cache based memory page (4KB) structure and has multiple layers of entries. Facebook introduced Flashcache[5] for its hybrid storage system which makes use of SSDs as high performance cache. Its cache is structured as a set of associative hash, where the cache blocks are divided up into a number of fixed size sets (buckets) with linear probing within a set to find blocks. Its replacement policy is either FIFO (First in First Out), or LRU (Least Recently Used) algorithm as used in conventional operating system. FlashTier [6] makes use of SSDs to form a unified address space with HDDs with sparse mapping data structure to map the address in SSDs. System controller does not need to keep the address mapping table, but is still use conventional cache management mechanism, particularly, the per-block

based algorithm, which is similar to Flashcache. In summary, for hybrid storage system with separate SDD devices, the cache management in system controller must maintain block address mapping between flash media and disk. With the use of hybrid disk drive, the need for such a mapping table can be eliminated, as this is offloaded into hybrid disk.

B. Hybrid Disk based Storage Systems

As described above, the existing hybrid storage systems adopt cache management methods used in conventional operating system level, particularly the per-block based address mapping, that create large overhead in terms of system memory consumption and cache lookup performance and efficiency as the number of drives in a RAID (Redundant Array of Inexpensive Disks) increases. Fig. 2 shows comparison of the conventional hybrid storage system consisting of separate SSD and HDD with a hybrid disk drive. In conventional system the storage controller has to maintain the cache mapping table at the individual block level. In addition to this, the controller also has to manage data migration between SSD and HDD during cache placement or eviction.

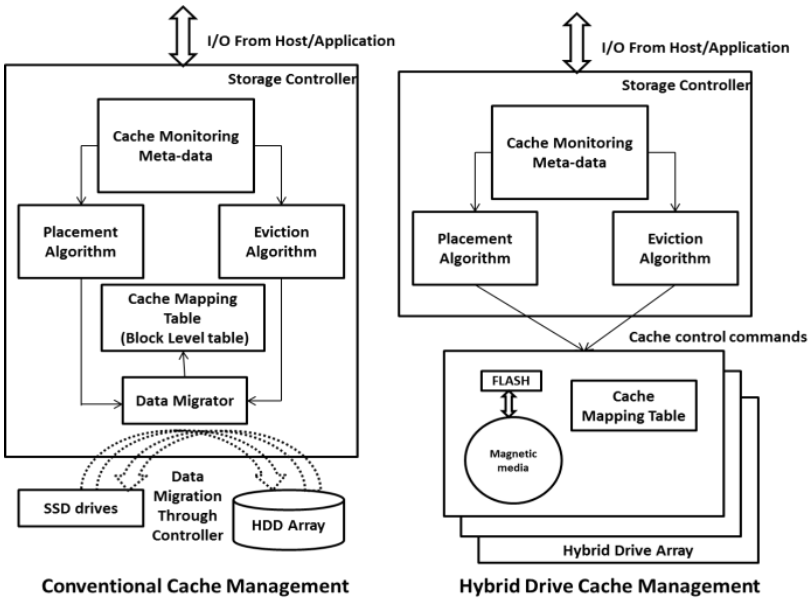


Fig. 2. Conventional SSD caching (left) vs. Hybrid disk drive caching (right)

In hybrid disk drive approach (on the right of Fig. 2), the controller complexity is much reduced as both the cache mapping table and the data migration is handled within the drive. The controller does not need to manage cache mapping tables and cache migration (placement/eviction). Moreover when a large array of disk drives is handled as in a RAID configuration, the overhead of the mapping tables and data migration in conventional approach is increased, whereas for hybrid disk drive cache management, the processing overheads are distributed across all disks. The controller processing is therefore reduced to handle only the primary I/O and the cache control. Another major advantage is that while in the conventional system the placement/eviction of data has to flow through the system controller, the hybrid approach eliminates this unnecessary flow. This reduces the interference of the primary or foreground I/O with the cache data migration.

Though there are many caching algorithms that have been previously developed (e.g. like LRU, ARC, 2Q [11]), there are disadvantages in utilizing existing caching solutions for hybrid drives. Those algorithms are mainly applicable for the caching inside an Operating system buffer cache or in an application, which cover both continuous monitoring of the I/O workload and address mapping in fixed block size of granularity. There are no known methods on determining monitoring granularity (or actual I/O size/regions on disk) of the workload. Conventional system fixes monitoring sizes: E.g. 4KB, 64MB, 1MB. When granularity of the data size is small (e.g. 4KB blocks) the amount of monitoring meta-data is large. More time is spent in updating I/O statistics for each block as well (due to large search space). To address this problem, we developed a cache manager located in storage system controller with cache management algorithm for hot (frequently accessed) data and region dynamic estimation and monitoring.

III. Hybrid Disk Drive Cache Manager and Algorithm

For hybrid disk drive storage system, the cache manager controls the placement and eviction of drive's hot data to/from the internal NAND flash memory of the hybrid disk. For this, it needs to maintain history of I/O access to the individual blocks on disk. The history of data relates to the recency and frequency of access to the individual blocks on disk. We consider this as cache monitoring meta-data. Conventional algorithms only monitor fixed size or granularity of the data on disk, which can vary from a single 4KB block (fine granularity)

to a large number of blocks in range of Megabytes (course granularity). The granularity determines the accuracy of the classification between hot and cold data. The accuracy in turn determines the effectiveness of utilization of the cache space. Fine granularity tracking leads to accurate classification resulting in optimal cache space utilization. But this increases the monitoring overhead or meta-data size. Course granularity results in placement of hot data along with cold data in the cache. We address this monitoring granularity problem by dynamically determining the size based on the I/O workload. To do so, we utilize an Interval-tree data structure that holds the monitoring meta-data in the tree nodes. We keep track of hot data on the cache through LBA ranges or intervals corresponding to the monitoring region. Instead of fixing the granularity of monitoring data, the cache management algorithm adapts data monitoring size to the I/O workload.

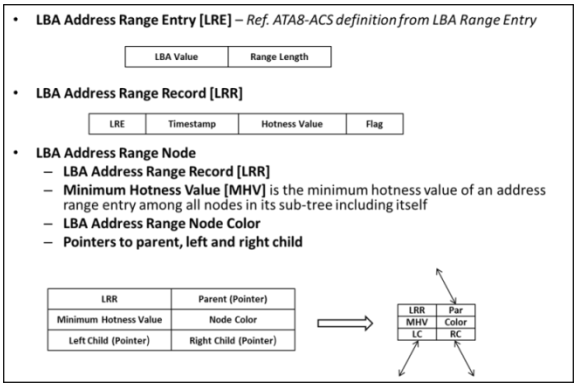


Fig. 3. Structure of LBA address range entry, and node for interval tree

A. I/O Monitoring Data: Interval-Trees

The Interval-tree is basically a Red-Black tree [12] indexed with LBA ranges. LBA ranges correspond to contiguous blocks on disk. Each node in the tree keeps track of a single hot data range on the disk with the corresponding LBA starting address and range size. It also stores the hotness value which is used to determine whether the LBA range corresponding to that node has to be evicted when there is contention for cache space. Hot data are tracked in terms of intervals as the cache control commands work in terms of LBA ranges. Fig. 3 shows the structure of the LBA address range record containing the LBA range entry (LRE) that contains the LBA start address and the range length in terms of disk sectors. This is followed by the time stamp at which this entry was inserted in the tree. The data hotness value (*dhv*) for the LBA range comes next in the record. The hotness value of a specific region on the disk is estimated based on the frequency and recency of the access. This value is maintained in each node of the Interval-tree representing the LBA range on disk. In addition, it also incorporates the randomness of the I/O request based on access pattern to the same disk region. The following equation (1) is used to estimate *dhv*.

$$dhv = n \times r + afv \quad (1)$$

where *n* is the frequency of access to that LBA range and *afv* is the aging factor value incremented each time old nodes are evicted from the Interval-tree. The *afv* captures the recency of the access. Any new node has an *afv* value that is larger than any nodes which already exist in the tree. This favors LBA ranges that are recently added in the cache tree. The value *r* is the randomness in the request to this LBA range, obtained based on exponential moving average of the last LBA distance from the current accessed LBA. It is important to note that, inside a node structure, we also defined a field to store the minimum hot value (mhv). The *mhv* of a particular node holds the minimum *dhv* of all the subtree nodes including this node. Fig. 4 shows the structure of the Interval-tree with the Red-Black nodes containing the caching information like the minimum hotness value (*mhv*). By tracking data on disk in terms of intervals we can reduce significant amount of monitoring meta-data overhead compared to hash based mappings used in traditional SSD based caching solutions. In addition to the reduced monitoring meta-data overhead, by incorporating *mhv* value in each nodes of the tree we can speed up the searching of the cold data for eviction. The O(logN) search time of the Interval-tree (same as a Red black tree also helps in fast updates or book keeping of the I/O statistics counter of each LBA range in the cache based on the incoming workload.

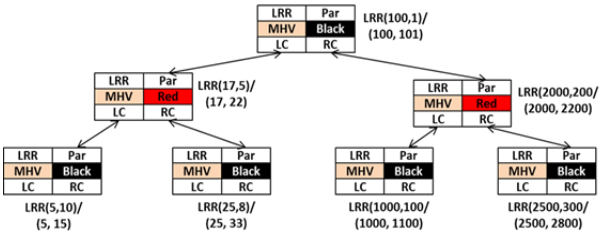


Fig. 4. Interval tree nodes with Hot LBA range tracking

Data: Interval-tree with LBA range address nodes
Result: Cache Node with minimum Hotness Value

```
1  currentNode = rootNode ;
2  while currentNode mhv value not equal to dhv do
3      if mhv of left child equal to mhv of currentNode
       then
4          Assign currentNode left child to
           currentNode;
5      end
6      else
7          Assign currentNode right child to
           currentNode;
8      end
9  end
10 Obtain currentNode LBA Range;
11 Prepare ATA-8 NV-Cache Unpin command for
   Range;
12 Send Unpin Command to the Hybrid Drive;
13 return currentNode
```

Algorithm 1: Interval-tree Search for minimum hotness value.

B. Cache Eviction Algorithm

Equipped with Interval-tree, *dhv*, and *mhv*, the cache manager performs eviction based on a simple tree search

(Algroithm1) to find the cache node with *dhv* that equals the smallest *mhv* value. Since each node stores a *mhv* value which is the minimum of the *dhv* under its sub-tree, the algorithm traverses the tree by looking in the left and right nodes containing the *mhv* equal to the value of current node. Then it branches to the node containing the same *mhv* and recursively repeats the search until it finds the node containing *dhv* that has the same value of *mhv*. This node is then selected for eviction from the hybrid drive's cache. Whenever the hotness value of a node is updated due to a cache hit, the updated *mhv* is propagated from that node all the way up to the root node. This preserves the correctness of the *mhv* value at each node which aids the eviction mechanism.

We have done the simulation with publicly available block traces and the results show that our algorithm can reduce amount of monitoring meta-data up to 64% compared to a simple set associative hash table based mapping used in caching solution found in FlashCache [5]. The reduction is largely due to the monitoring granularity determined by the I/O sizes in the workload. While the methods in FlashCache and other caching system need to map each block (typically the File system block of 4KB) to the corresponding blocks in the cache, the Interval-tree approach stores a large LBA range. This leads to less number of tracking meta-data as an entire range of blocks (several Kilobytes) is monitored. The range maintained in each of the tree nodes is also adjusted based on the access pattern. Therefore we get reduced meta-data utilization without losing the accuracy of classifying hot and cold data. We can also see that the overhead or processing time for cache meta-data updates (like frequency and hotness value) is also reduced by up to 48%. Even though a hash table lookup is faster than an Interval-tree, as the number of cache meta-data nodes are large the hash table leads to collisions. In a chained hashing implementation these collisions end up with a linear search of the meta-data nodes. In contrast, the Interval-tree approach always gives a O(logN) search time to update the nodes since the tree is always balanced.

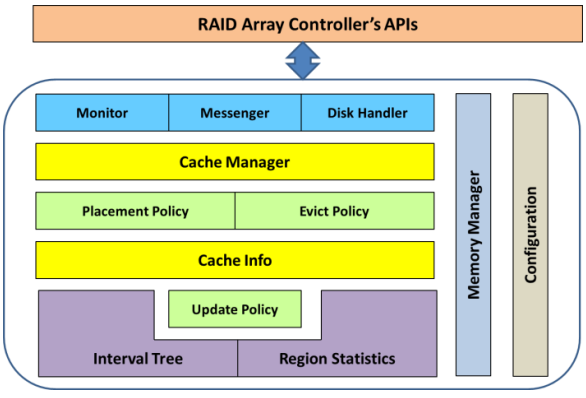


Fig. 5. Hybrid disk drive cache manager block diagram

C. Hybrid Disk Drive Cache Manager

Fig. 5 shows the architecture block diagram of Hybrid disk drive cache manager which controls the placement and eviction of hot data to/from drive's internal NVM storage. *RAID Array Controller APIs* consists of a set of interface functions to provide storage I/O requests information from applications to hybrid disk drive cache manager, and relay cache access request to hybrid disk from hybrid disk drive cache manager. *Cache Manager* is the main module that provides cache management algorithm and related functionalities. *Monitor* is the working layer for cache manager to monitor hybrid disk drive cache status and apply cache management policies. Hybrid disk drive cache manager can also work with RAID controller by exchanging configuration and other information. *Messenger* is the event notification module that gets messages from RAID array controller and passes to cache manager for further process. It provides the necessary information for cache manager to perform proper cache management functionalities. *Disk Handler* is the working layer for cache manager to engage with hybrid drives. *Cache Info* is defined with structures and related functions that support cache meta-data and statistics information update, reservation and operations. *Interval Tree* module is purposely designed and built to serve as cache information repository. It provides detail information of I/O cache status, so that cache manager can quickly decide which data needs to be placed in, and which data needs to be evicted from NVM cache in hybrid disks. Other building blocks are also design and developed to complete hybrid disk drive cache management framework.

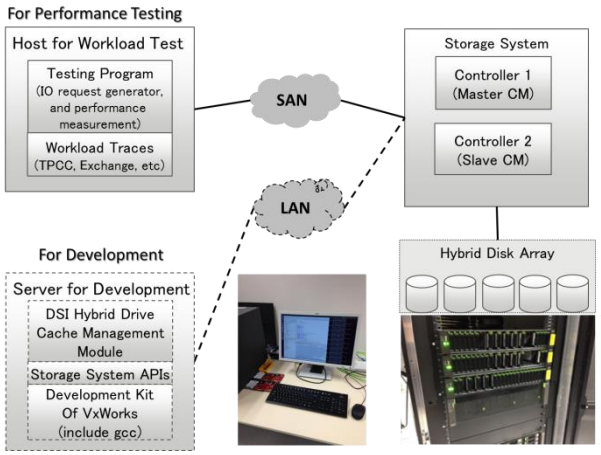


Fig. 6. Hybrid disk drive storage system integration

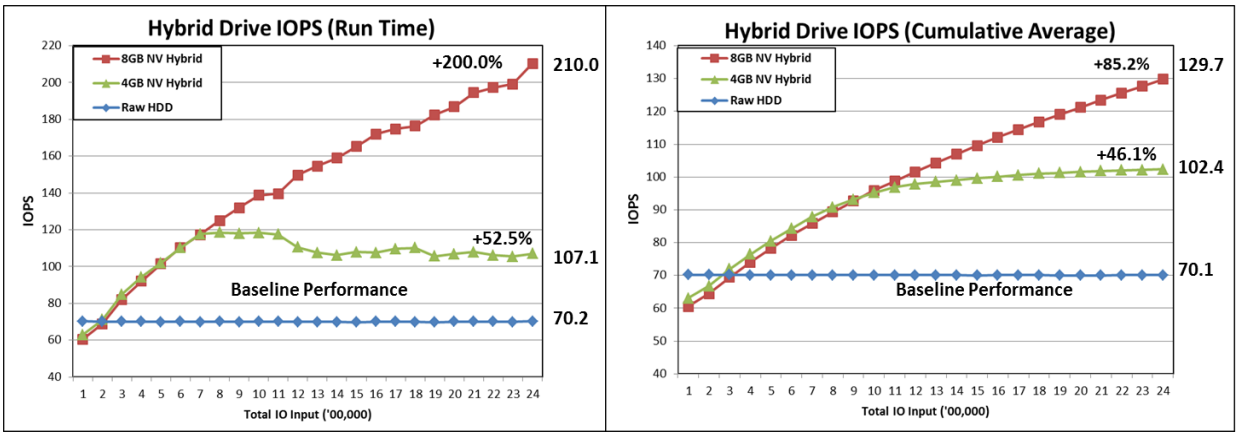
IV. Hybrid Disk Drive Storage System Evaluation

We integrated hybrid disk drive prototypes with our developed cache manager into an enterprise storage system and conducted test to evaluate the impact of hybrid disk drives in terms of IOPS performance and power consumption shown in Fig. 6. The hybrid disk drive prototypes used in this paper are developed and provided by our industry partner. Each drive combines a 5400 RPM Toshiba 2.5 inch HDD and purpose-build PCBA board with NAND NVM flash storage. The prototypes are mainly built to support hybrid functions

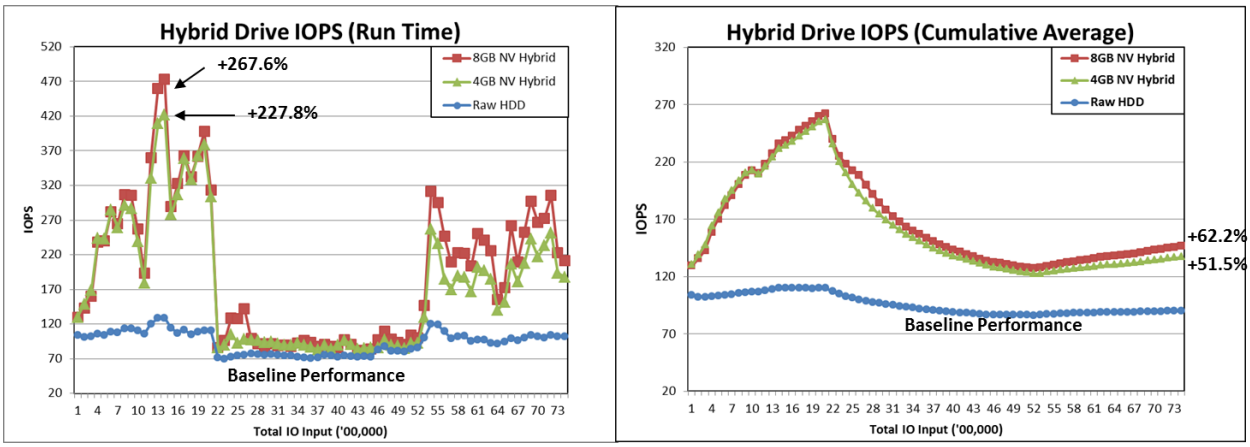
defined in ATA-8 specification [10], thus they still have performance limitations compared with a standard storage device product. In our experiment, hybrid disks are configured in a RAID5 volume. We used Microsoft enterprise workload traces which are published from SNIA block traces repository [13]. The Table I shows a short description of the workload traces used in measurement. The block trace files were replayed through a user space program on the testing host in a single thread and synchronous configuration, which is equivalent to measuring IOPS on a per drive basis. As the total RAID5 volume is 360GB, our measurements are also made for both 4GB and 8GB NVM flash allocation for each hybrid drive in order to conduct test and comparison that is equivalent to allocate NVM flash cache at about 5% and 10% of total storage space. We compare IOPS performance results with the NVM cache allocation (either 4GB/8GB) and baseline configuration (without NVM cache allocation).

TABLE I: WORKLOAD TRACES FOR SYSTEM EVALUATION

Workload Trace	I/O Counts	Trace Types
TPCC	2,458,938	Database transactions
Microsoft Exchange	7,450,837	Email server trace



(a) TPCC Trace



(b) Microsoft Exchange Trace

Fig. 7. Hybrid disk drive IOPS performance improvement in storage system

Fig. 7 depicts the IOPS performance comparison for both TPCC and Microsoft exchange workload trace. The cumulative average measurement is done with the average of IOPS over the total testing period, while the real time measurement is done based on a specific period of time that only contains 100,000 I/O requests during the test. The cumulative average measurement is used to evaluate the overall performance improvement capability over the whole workload test duration. The real time performance IOPS improvement can be considered as the performance improvement capability at any point of time (with the granularity of 100,000 I/O request).

Since TPCC workload trace contains most of the small size (8KB) and randomized database transaction I/O request, we observe that IOPS performance improvement is quite smooth and predictable. As the test proceeds, the improvement is gradually improved. With the allocation of 4GB NVM flash in each hybrid disk, the NVM flash can be fully utilized earlier. After about 750,000 I/O requests, the IOPS performance is stabilized and can reach a saturated range. For such configuration, the cumulative average and real time IOPS performance improvement can be 46.1% and 52.5% respectively. For the configuration with 8GB NVM flash allocation in each hybrid disk, the performance improvement is much more significant as more cache storage is provided. Since the TPCC workload trace only have about 2.4 million I/O request, we could not see the IOPS improvement beyond

that point but already it can deliver 85.2% and 200% more improvement for cumulative average and real time measurement respectively.

The results for exchange trace show similar pattern as TPCC trace with the exception of performance drops in the middle range of test, which rightfully reflects the application data characteristics. As I/O requests on email server demonstrates more fluctuation in terms of temporal and spatial locality, hybrid disk can only deliver corresponding cache hit rate. However, with the high locality in certain range of workload trace, hybrid disk can provide much higher IOPS improvement ratio, which can be as high as 267.6% more for real time measurement with 8GB NVM flash allocation setting. And the IOPS improvement for total cumulative average can be also as high as 62.2% more.

The significant impact of our work demonstrated that with relatively low speed 5400 HDD and a modest cache size of 8GB on hybrid disk, IOPS performance can be up to three time of the result achieved from baseline disk drive. Such IOPS result can exceed that of higher speed 7200 RPM SAS drives, which are originally attached to enterprise storage system. Based on the experimental comparison results, we have done the estimation that projects a future hybrid disk drive IOPS performance and power consumption at 7200 RPM spindle speed. It is promising that with proper cache management algorithm, a 7200 RPM hybrid disk drive could outperform today's 15000 RPM enterprise hard disk drive.

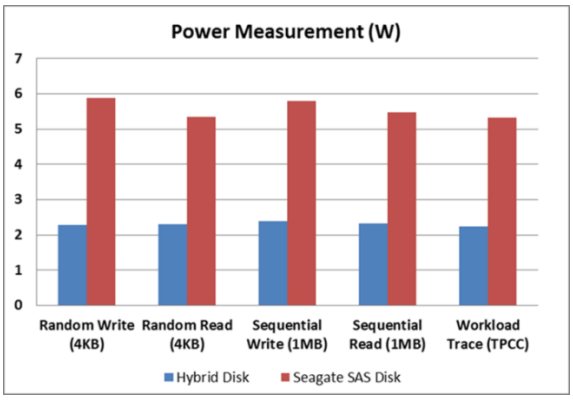


Fig. 8. Hybrid disk power consumption comparison with enterprise SAS disk

We did power consumption measurement under various testing conditions to understand energy saving of hybrid disk drive. Fig. 8 shows the power consumption comparison with Seagate 7200 RPM SAS disk. The hybrid disk prototype works at 5400 RPM. They were both setup in a RAID5 configuration. We measured the power consumption while conducting 4KB random read/write test, 1MB sequential read/write test, and TPCC workload test respectively. For TPCC workload test, we purposely measured hybrid drive power consumption after NVM cache is fully populated, and the there is no significant difference compared with other type of test. The average power consumption for a SAS drive is around 5.33 Watts while a hybrid disk only consumes 2.22 Watts as average. The hybrid disk power consumption is reduced by more than half of the enterprise SAS disk drive.

V. Conclusions

We presented our work of developing hybrid disk drive storage system for enterprise applications, which is, to our knowledge, the first to enable storage system controller initiatively monitor and control the access to NVM in hybrid disks. With such approach, we design and develop a dynamic caching algorithm that simplify but enhance the conventional caching management architecture used in exiting hybrid storage systems consisting of separate SSD and HDD. Significant performance improvement and power saving are observed from the system evaluation. Moreover, such storage system could provide much larger capacity and lower building cost compared with SSD only system. This makes it a promising selection of replacing conventional high RPM enterprise hard disk in future datacenters.

Acknowledgment

The research work was supported by the Agency for Science Technology and Research (A*STAR, Singapore) Grant for the Future Data center Technology thematic research program (Grant no: 1121720013).

References

[1] Weblink. (2015). [Online]. Available: http://en.wikipedia.org/wiki/IOPS

[2] R. Wood, "Future hard disk drive systems," *Journal of Magnetism and Magnetic Materials*, vol. 321, no. 6, pp. 555 – 561, 2009.

[3] Y. Kim, "MixedStore: An enterprise-scale storage system combining solid-state and hard disk drives," Technical Report Pennsylvania State University, Department of Computer Science and Engineering, College of Engineering, 2008.

[4] F. Chen and D. Koufaty, "Hystor: Making the best use of solid state drives in high performance storage systems," in *Proc International Conference on Supercomputing*, Tuscon, Aizona. ACM, 2011.

[5] C. Hollowell, *et al*., "The effect of flashcache and bcache on I/O performance," *Journal of Physics: Conference Series*, vol. 513, no. 6, 2014.

[6] M. Saxena, M. M. Swift, and Y. Zhang, "Flashtier: A lightweight, consistent and durable storage cache," in *Proc. 7th ACM European Conference on Computer Systems*, New York, NY, USA: ACM, 2012, pp. 267–280.

[7] X. Liu and S. Kenneth, "Hybrid storage management for database systems," *Proc. of the VLDB Endowment*, vol. 6, no. 8, pp. 541-552, 2013.

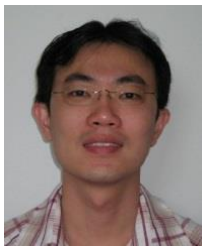
[8] T. Luo, R. Lee, M. Mesnier, F. Chen, and X. D. Zhang, "hStorage-DB: Heterogeneity-aware data management to exploit the full capability of hybrid storage systems," *Proc. of the VLDB Endowment*, vol. 5, no. 10, pp. 1076-1087, 2012.

[9] Weblink. [Online]. Available: http://technet.microsoft.com/en-us/magazine/2007.03.vistakernel.aspx.

[10] "AT Attachment 8 - ATA/ATAPI Command Set (ATA8-ACS)," Information Technology Industry Council, Approved September 29, 2009, ANSI, Inc.

[11] A. Floratou, N. Megiddo, N. Potti, F. Özcan, U. Kale, and J. Schmitz-Hermes, "Adaptive caching algorithms for big data systems," IBM Research Report, 2015.
[12] P. C. Huang, Y. H. Chang, and T. W. Kuo, "Joint management of RAM and flash memory with access pattern considerations," in *Proc. 49th IEEE Design Automation Conference*, 2012.
[13] SNIA Block I/O Traces. [Online]. Available: http://iotta.snia.org/traces/130

Wang Yonghong Wilson received his Ph.D. degree in Computer Engineering from the National University of Singapore in 2005. He is currently working in Datacenter Technologies Division, Data Storage Institute, the Agency for Science, Technology and Research (A*STAR), Singapore, as a research scientist for storage system design and development. His research interests include storage networking, on-demand storage resource provisioning, and hybrid disk drive storage system development for future datacneters.



Chun Teck Lim received his B.A.Sc (Computer Engineering) degree from Nanyang Technological University, Singapore in 1999. He is currently a senior research engineer in Data Storage Institute, the Agency for Science, Technology and Research (A*STAR), Singapore. His research includes storage systems and software defined storage with interest in embedded systems, operating systems and real time systems.

Kyawt Kyawt Khaing received her M.Eng(EcE) degree in Electrical & Computer Engineering from the National University of Singapore in 2005. She is currently working in Datacenter Technologies Division, Data Storage Institute, the Agency for Science, Technology and Research (A*STAR), Singapore, as a research engineer for storage system design and development. Her research interests include hybrid disk drive storage system, large scale storage system and cluster file systems.

Yong Khai Leong received his Engineering degree from the National University of Singapore and holds a postgraduate degree in Communication Software and Networks. He is a Division Manager of the Data Storage Institute, the Agency for Science, Technology and Research (A*STAR), Singapore. He has more than 10 years of research experience in network storage systems, networking and software designs with many of these years in leading positions. He currently leads a team of research scientists and engineers in developing data systems and storage technologies for next generation data centers.