Apas: an Application Aware Hybrid Storage System combining SSDs and SWDs

Wenguo Liu, Lingfang Zeng, Dan Feng
School of Computer, Huazhong University of Science and Technology
Wuhan National Laboratory for Optoelectronics
Wuhan, China
Email: liuwenguo_hust@163.com, {lfzeng, dfeng}@hust.edu.cn

***Abstract*—Shingled write disk (SWD) is a new kind of HDD, which increases areal density by adopting SMR technology. In the hybrid storage system consisting of SSDs and SWDs, there are some special characteristics such as large performance gap between SSDs and SWDs. In this paper we explore how to efficiently combine the advantages of SSDs and SWDs together. We propose Apas, an application aware hybrid storage system combining SSDs and SWDs, where SSDs are used as non-volatile cache and SWDs work as lower level storage. In Apas we propose a cache partitioning policy to divide the cache space according to the application's characteristics. To mitigate the write amplification problem in SWDs, we first classify data bands in the SWDs into five types according to the distribution of valid data in the bands, and then design a new garbage collection policy combining the aggressive data cleaning and lazy data cleaning together according to the detected idle time and the five types of bands. Experiments are carried out and the results show that Apas has good performance.**

***Keywords*-SSD cache, SWD, write amplification, data cleaning, band classification.**

I. Introduction

Shingled Magnetic Recording (SMR) is a technology which effectively improves the capacity of hard disk drives with no significant cost impact [1]. In a Shingled write disk (SWD), the behavior of read operations is the same as that in a traditional HDD. However, in spite of improvement in capacity, in-place update is not supported. When updating data in a track in a SWD, the data in subsequent tracks may be damaged and should be read out first, resulting in performance degradation [2]. This phenomenon is called write amplification, which often dramatically restricts random write accesses.

Flash-based SSDs have much better performance than SWDs and have been widely used in storage systems, however, the costs of SSDs are much higher than those of SWDs. To explore how to efficiently combine the advantages of SSDs and SWDs together, and improve the performance of the SWD based hybrid storage system, we propose Apas, an Application aware hybrid storage system combining SSDs and SWDs, where SSDs are used as non-volatile cache and SWDs work as lower level storage. Apas consists of three key components: a cache partitioning policy which divides the cache space according to the application's characteristics, a feedback scheme which collects statistics such as the actually allocated cache space each time period in the storage system, and an idle time aware garbage collection (GC) policy to mitigate the write amplification problem in SWDs during application accesses.

We make the following key contributions in this paper.

- We propose a new cache partitioning scheme which divides the cache space according to the application's characteristics.
- We classify data bands in the SWD into five types according to the distribution of valid data in the bands.
- We design an new garbage collection policy for SWDs according to the detected idle time and the five types of bands.

The rest of the paper is organized as follows: The related work is presented in Section II. We describe the design of Apara in Section III. The experimentation results and analysis are presented in Section IV. And we conclude this paper in Section V.

II. Related Work

Although SMR efficiently improves the areal density of hard disk drives, the problem of write amplification often leads to performance degradation in SWDs. Some solutions to the write amplification problem have been proposed. Cassuto et al. [3] proposed an indirection system to mitigate the random-write access restriction and utilized a storage unit called S-block to achieve better random-write performance. However, this solution didn't efficiently solve the garbage collection problem, and resulted in a lot of data migration under write-intensive workloads, which easily brought severe performance degradation. To improve the performance of SWD-based storage systems, flash-based SSDs and NVRAMs are introduced. Luo et al. [4] proposed a hybrid wave-like shingled recording(HWSR) disk system to improve the performance of a shingled recording disk, in which the memory was used to buffer hot writes and the SSD was used as a read cache. Considering that read and write costs were different in flash memory, Kgil et al. [5] proposed to split the flash memory cache into separate read and write regions. The idea of splitting caching space have been used in many buffer cache management schemes. Kim et al. proposed Unified Buffer Management (UBM), a buffer

978-1-5090-3315-7/16/$31.00 ©2016 IEEE

management scheme that stored sequential and looping references in separate regions in the buffer cache [6]. However, as different workloads often have different characteristics, these solutions failed to efficiently accommodate to various workloads, and often resulted in degradation of the system performance.

III. The Design of Apas

In this section we will firstly give an overview of the system model, and then present the details of Apas.

A. Design Overview

The system model of the hybrid storage system Apas consisting of a SSD cache and several SWDs as shown in Figure 1. The SSD cache is shared by applications, and the cache partitioning scheme divides the SSD cache into read cache and write cache. According to the applications' characteristics, the cache partitioning scheme further divides the read cache and write cache into application-specific read cache and application-specific write cache respectively. The feedback scheme consists of two parts: monitor and adjustor. The monitor is responsible for monitoring the I/O characteristics of each application such as sizes of read and write requests and periodically collecting various statistics from the underlying storage system such as the actually allocated cache space and average latency of each application. The adjustor is responsible for adjusting the sizes of read cache and write cache of each application based on the collected statistics. In each SWD, dynamic mapping is used and bands are composed by one unshingled band and a number of shingled bands, where the unshingled band stores the mapping table and the shingled bands store data. When the SWD is idle or the ratio of the free space of each SWD falls bellow a predefined threshold(such as 20%) , the garbage collection (GC) is triggered. Actually, in this hybrid storage system, we find that there is a lot of idle time under many workloads and the performance of GC could be effectively improved by making full use of the idle time. We divide the whole access of applications into number of time periods, and the length of each time period is 10000s.

B. Cache partitioning policy

The cache partitioning policy, CPP, aims to divide the cache space into write part and read part based on the data volumes of read requests and write requests, which is shown in Algorithm 1.

C. Garbage collection policy

Garbage collection is an important process in SWDs. In Apas, we find that there is a lot of idle time in the SWD under many workloads, which motivates us to combine the aggressive data cleaning and lazy data cleaning together. Actually, according to the distribution of valid data, we find that all the shingled bands except free bands could be categorized into five types in a coarse grain, as shown in Figure 2.

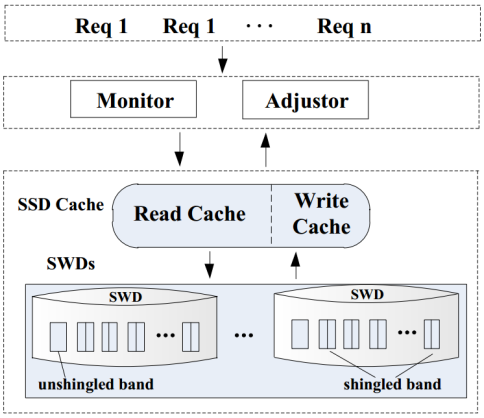


Figure 1. The architecture of Apas

Algorithm 1 CPP:the cache partitioning policy

1: The monitor collects $Data_R$ and $Data_W$ which denotes the data volumes of read requests and write requests respectively each time period.
2: CPP divides the cache space C_0 into read cache C_R and write cache C_W based on the data volumes of its read requests and write requests.
$C_R = C_0 \frac{Data_R}{Data_R+Data_W}$; $C_W = C_0 \frac{Data_W}{Data_R+Data_W}$;

In the band of type A, nearly all the data blocks are invalid, and this kind of band could be reclaimed with very little data migration. In the band of type B, nearly all the data blocks are valid, and this kind of band need not be reclaimed. In the band of type C, valid data blocks are mainly distributed in the back half of the band, and only reclaiming the front half is not enough, because data migration in the back half of the band is always needed. In the band of type D, valid data blocks are mainly distributed in the front half of the band, and it is easy to reclaim the back half of the band. In the band of type E, however, valid blocks are mainly distributed across the whole band, and reclaiming this band would result in a lot of data migration.

According to the detected idle time in SWDs and the categorization of shingled bands, we propose a GC policy shown in Algorithm 2 which implements aggressive data cleaning when idle time is detected and lazy data cleaning when the ratio of free space falls below 20%.

IV. Performance Evaluation

A. Evaluation Setup

We have implemented the Apas in a simulation environment based on Disksim [7] and SSD extension [8]. SWD is emulated based on the disk model of Maxtors Atlas 10K IV disk whose disk size is 146GB. The SSD emulates SLC

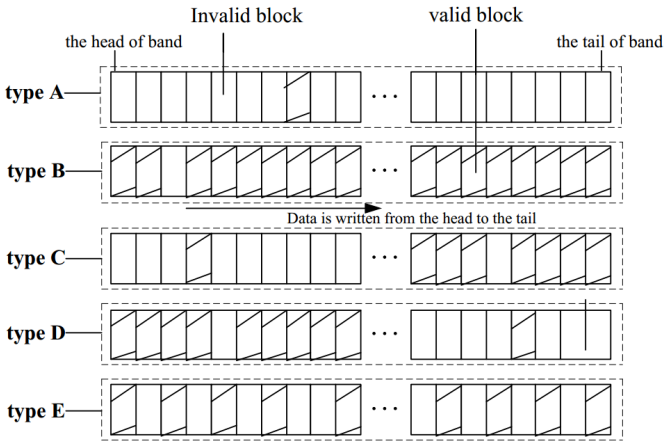


Figure 2. The Five Types of Shingled Bands

Algorithm 2 The GC policy

1: The system scans the shingled bands and categorizes them into five types mentioned in Figure 2.
2: Assign weights 0 to the bands of type B, 1 to the bands of type E, 2 to the bands of type C, 3 to the bands of type D, 4 to the bands of type A respectively. The higher the weight ,the higher priority when implementing the GC policy.
3: When idle time is detected, the aggressive cleaning is triggered.
4: When the ratio of the free space of the SWD fall below 20%, the lazy cleaning is triggered .
5: Bands are reclaimed according to their assigned weights. If there are bands belonging to the same type, bands with fewer valid blocks have the higher priorities.

NAND flash memory chip operations, and it takes 25us to read a page, 200us to write a page, 1.5ms to erase a block.

The kernel management module in the SWD is STL, which consists of two main functions: address translation and garbage collection. Address translation is implemented through maintaining a hash table, which dynamically translates addresses in a log-structured way. Entries compose the hash table and each entry represents a data block. Each entry contains five variables: SMR_LBA, $Target\_LBA$, $hashvalue$, $valid\_state$ and $band\_num$. SMR_LBA represents the original offset of a data block in a SWD. $Target\_LBA$ represents the offset translated by the STL according to SMR_LBA of a data block. $hashvalue$ represents the location of a data block according to its SMR_LBA in the hash table. $valid\_state$ marks the state of the data block. If the data block is valid, the value of $valid\_state$ is 1, otherwise the value is 0. $band\_num$ represents the number of band which the data block belongs to.

When the SWD receives a write request, it first checks if there are corresponding entries in the hash table. If yes, the $valid\_state$ of the $Target\_LBA$ of each entry is set as 0, and new $Target\_LBAs$ are worked out, the $valid\_states$ of which are set as 1; if not, new entries are created according to the request and inserted into the hash table. When the SWD receives a read request, it looks up the hash table and returns corresponding $Target\_LBAs$ according to the request. During the process of data access, the number of invalid data blocks of each band is counted in real time.

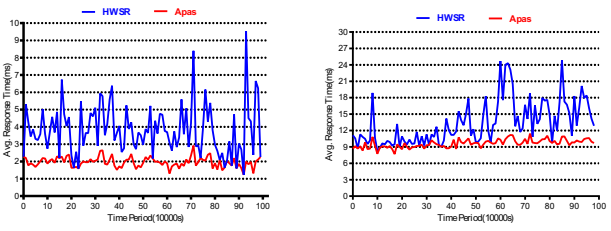
B. Performance Evaluation

In this part the performances of two SWD based hybrid storage systems, Apas and HWSR, are compared. Real traces are used for our evaluation, including Financial and Exchange. The characteristics of the three workloads are listed in Table I.

Table I
CHARACTERISTICS OF TRACES

Trace	Total size	Number of requests	Read ratio
Financial	35.70GB	5.33million	19%
Exchange	152.01GB	25.52million	46%

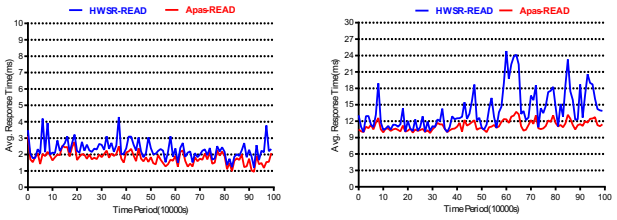
Figure 3 shows the I/O performance of Apas and HWSR in 100 consecutive time periods under workloads of Financial and Exchange. As shown in Figure 3, in most time periods, the average response times of Apas are much better than those of HWSR. One of the reasons is that, in HWSR the sizes of cache spaces allocated to read requests and write requests are fixed, and when too many read requests or write requests arrive in a time period, some requests have to wait a much longer time before to be served. While in Apas, more cache space could be dynamically allocated to read requests or write requests. Another reason is that although HWSR uses a replacement policy and segment-based data layout to reduce write amplification, the replacement policy still brings a lot of data updates and the data layout results in quite a few random access operations. Apas classifies data bands into five types according to the distribution of valid data and reclaims data bands when the SWD is idle, which could effectively mitigate the write amplification problem in the SWD.



(a) Financial (b) Exchange

Figure 3. Average response times of three workloads: Financial and Exchange

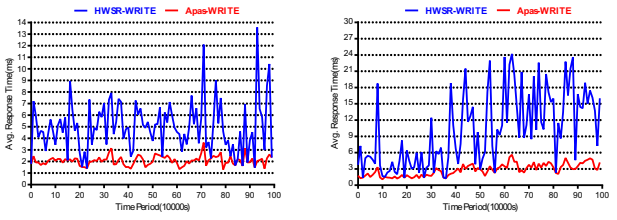
The read performances of Apas and HWSR are compared in Figure 4 under workloads of Financial and Exchange. From Figure 4 (a), we could see that there is not large read performance gap between Apas and HWSR. Apas performs a little better than HWSR. This is because the read ratio of Financial is only 19% and both Apas and HWSR aim to mitigate the write amplification problem. From Figure 4 (b) we could see that Apas performs much better than HWSR in many time periods. This is mainly because that HWSR brings many random reads, resulting in read performance degradation.



(a) Financial (b) Exchange

Figure 4. Average read response times of three workloads: Financial and Exchange

Figure 5 shows the write performances of Apas and HWSR under workloads of Financial and Exchange. From the subfigures we could see that Apas performs much better than HWSR. The main reason is that the replacement policy used in HWSR couldn't effectively reduce the amount of write data destaged to the SWD, and HWSR translates nearly all the sequential write requests to random write requests, which results in poor write performance. Unlike HWSR, Apas reclaims many data bands when the SWD is idle, which effectively migrates the write amplification problem.



(a) Financial (b) Exchange

Figure 5. Average write response times of three workloads: Financial and Exchange

V. Conclusions

In this paper we propose Apas, an Application aware hybrid storage system combining SSDs and SWDs, where SSDs are used as non-volatile cache and SWDs work as lower level storage. Trying to effectively combine the advantages of SSDs and SWDs together, we propose a cache partitioning policy to effectively allocate cache space to applications according to their characteristics. And then, to solve the write amplification problem in SWDs, we design a new garbage collection policy which combines the aggressive data cleaning and lazy data cleaning together according to the detected idle time and the classified five types of bands. Experiments have been implemented and results show that Apas performs better than other SWD-based hybrid storage systems.

Acknowledgment

We are grateful to the anonymous reviewers for their valuable feedback and suggestions.This work was supported by the National High-tech R & D Program of China (863 Program) No. 2015AA016701, No. 2015AA015301; the Fundamental Research Funds for the Central Universities, HUST, under Grant 2015QN072. This work was also supported by Key Laboratory of Information Storage System, Ministry of Education, China.

References

[1] A. Amer, J. Holliday, D. D. E. Long, E. L. Miller, J. -F. Pris, T. Schwarz, Data Management and Layout for Shingled Magnetic Recording. IEEE Transactions on Magnetics, vol. 47, no. 10, pp. 1460-1463, October 2011.

[2] A. Aghayev, P. Desnoyers: Skylight A Window on Shingled Disk Operation. In (FAST 15)Proceedings of the 13th Usenix Conference on File and Storage Technologies (Feb 2015).

[3] Y. Cassuto, A. A. Sanvido, C. Guyot, D. R. Hall and Z. Z. Bandic. Indirection systems for shingled-recording disk drives. In Proceedings of 26th IEEE Conference of Symposium on Mass Storage Systems and Technologies (MSST) (2010), IEEE, pp. 114.

[4] D Luo, J Wan, Y Zhu, et al. Design and implementation of a hybrid shingled write disk system. IEEE Transactions on Parallel and Distributed Systems, 2016, 27(4): 1017-1029.

[5] Kgil T, Roberts D, Mudge T. Improving NAND flash based disk caches. In Proceedings of the 35th International Symposium on Computer Architecture, 2008. ISCA'08.IEEE, 2008: 327-338.

[6] J. M. Kim, J. Choi, J. Kim, S. H. Noh, S. L. Min, Y. Cho, and C. S. Kim, A low-overhead high-performance unified buffer management scheme that exploits sequential and looping references. In Proceedings of the 4th conference on Symposium on Operating System Design and Implementation-Volume 4.USENIX Association, 2000, pp. 99.

[7] J. S. Bucy, J. Schindler, S. W. Schlosser, and G. R. Ganger, The disksim simulation environment version 4.0 reference manual (cmu-pdl-08-101). Parallel Data Laboratory, p. 26,2008.

[8] V. Prabhakaran and T. Wobber, Ssd extension for disksim simulation environment. Microsoft Reseach, 2009.