Novel Address Mappings for Shingled Write Disks

Weiping He and David H.C. Du

Department of Computer Science, University of Minnesota, Twin Cities

{weihe,du}@cs.umn.edu

Abstract

Shingled Write Disks (SWDs) increase the storage density by writing data in overlapping tracks. Consequently, data cannot be updated freely in place without overwriting the valid data in subsequent tracks if any. A write operation therefore may incur several extra read and write operations, which creates a write amplification problem. In this paper, we propose several novel static *Logical Block Address* (LBA) to *Physical Block Address* (PBA) mapping schemes for in-place update SWDs which significantly reduce the write amplification. The experiments with four traces demonstrate that our scheme can provide comparable performance to that of regular *Hard Disk Drives* (HDDs) when the SWD space usage is no more than 75%.

1 Introduction

Traditional hard disk drives are reaching the areal data density limit [12]. To overcome this limit, many new recording technologies have been investigated, among which Shingled Magnetic Recording (SMR) [11, 7] is a promising technology because it does not require significant changes to either magnetic recording or manufacturing process. It increases the storage density by recording data in overlapping tracks. Consequently data has to be written sequentially onto the tracks in order not to destroy the valid data on the subsequent tracks. Alternatively, we have to safely read the impacted valid data in the subsequent tracks out first before writing/updating[1] to the current track and then write those impacted valid data back afterwards [4, 9]. In this way, extra read and write operations are incurred as an extra cost, which is known as the write amplification problem. However, random read operations will not be affected in SWDs.

In order for the shingled write disks to be adopted in the existing storage systems without significant performance degradation, it is necessary to mitigate or circumvent this write amplification problem. Two major types of shingled write disks are therefore being proposed, one is the in-place update SWD and the other is the out-of-place update SWD. Both types of SWDs use a small portion (1% to 3%) of the total shingled space as a random access zone for efficient metadata access [9, 4, 10].

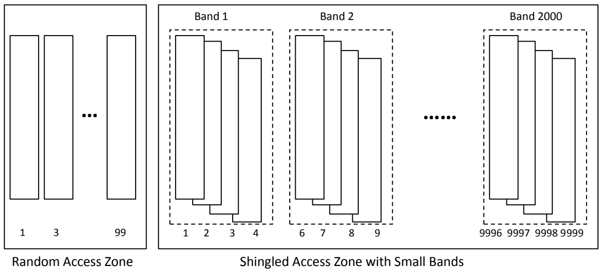

Figure 1: Physical Space Layout for In-place Update SWD

Out-of-place update SWDs divide the bulk shingled access zone into one E-region and several I-regions [5, 10]. The number of I-regions can vary based on different indirection system designs. E-region will be used as a circular buffer space to caching/reorganize data. All in-coming writes will first go to the E-region. Data in the E-region will then be destaged to the I-regions when needed. Writes to both E-region and I-regions have to be done sequentially. Out-of-place update SWDs can circumvent the write amplification problem by writing to new block locations on updates and invalidating the original blocks. On the other hand, both E-region and I-regions will perform *Garbage Collection* (GC) operations to reclaim the invalid blocks as the data in them ages and becomes too fragmented. Besides, a LBA-to-PBA mapping table has to be maintained to keep track of data movement and data accessing. Several approaches [5, 10, 8] have been done to minimize the overhead of GC operations and maintaining mapping tables.

On the other hand, in-place update SWDs require no GC operations and complicated mapping tables. The main body of an in-place update SWD consists of many small bands, each of which contains several tracks as shown in Figure 1. A safety gap sits between neighbor bands, the width of which depends on the write head width. It is important to choose the right band size because of the tradeoff between space gain and performance which will be further discussed in Section 3. Generally, if there are too many tracks in one band, more space gain will be obtained but on the other hand more write amplification overhead will be caused.

In this paper, we show that write amplification over-

[1] An update request is essentially a write request that modifies existing data blocks.

head of in-place update SWDs can be greatly reduced with novel static LBA-to-PBA mappings and these are simple mappings without incurring large overheads. Experiments with four traces demonstrate our proposed schemes can provide comparable performance to that of regular HDDs when space usage is no more than 75%.

The remainder of the paper is organized as follows. Section 2 describes some related work and Section 3 shows the motivations for this work. Novel data mapping schemes with performance predictions are discussed in Section 4. Experiments and result discussions are presented in Section 5. Finally conclusion is made in Section 6.

2 Related Work

Several work have been done for out-of-place update SWDs. For example, Cassuto *et al.* proposed two indirection systems in [5]. Both systems use two types of data regions, one for caching incoming write requests and the other for permanent data storage. GC operations are used in both systems, which has been improved in [10] by identifying hot (frequently updated) data and cold data. Hall *et al.* proposed a background GC algorithm [8] to refresh the tracks in the I-region while data is continuously written into the E-region buffer.

The closest work to ours is the shingled file system [9], which is a host-managed design for in-place update SWDs. The shingled file system directly works on SWD PBAs. The SWD main space is organized into small bands of size 64 MB. Files will be written sequentially from head to tail in a selected band. When a file is updated, impacted data in the subsequent tracks will be first read out to a block cache and written back to the original locations afterwards. However this work did not address the write amplification problem. Another drawback is that popular file systems (like EXT4 and NTFS) as well as other data management software have to be modified in order to use these SWDs. Our work improves the write amplification problem with novel address mapping schemes that make SWDs support general file systems in a drop-in manner.

3 Motivation

In this section, we discuss two factors that motivate our work, one is the intrinsic tradeoff in in-place update SWDs and the other is the conventional static address mapping used in regular HDDs.

3.1 Space Gain Tradeoff

Figure 1 shows the physical layout of an in-place update SWD. It uses a write head width of 2 tracks. There are k = 100 physical tracks in the random access zone, half of which are effectively used to construct the random access zone. There are also 10000 physical tracks in the shingled access zone which form m = 2000 bands with band size of 4 tracks. Totally 2000 tracks are used as safety gaps to separate the bands. The space efficiency is therefore 0.8 = 4m/(4m+m). As the write head width is 2 tracks, the actual space gain is 1.6 = 0.8*2. Although the outer tracks are bigger than inner tracks in a real disk drive, we assume a fixed track size of 100 blocks or sectors for simplicity in this example.

More generally, assume band size is N tracks and write head width is W tracks, then the *Space Gain* (SG) and the expected *Write Amplification Ratio* (WAR) for a single update request to a full band can be calculated according to Equation (1) and (2). Other discussions on areal density increase factor can also be found in [4, 6]. The WAR for a single update request is defined as the total number of requests associated with an amplified update request. Ratio 1 means no amplification is incurred. The equations clearly indicate that the bigger the band size is, the bigger space gain is but the larger write amplification overhead is created at meantime. We assume in this paper that the band width is 4 tracks and the write head width is 2 tracks to balance this tradeoff. Other configurations, such as band width of 5 tracks with write head width of 3 tracks, can also be used as long as the tradeoff is balanced and manufacturing process allows.

$$SG = W\frac{N}{N+W-1} \tag{1}$$

$$WAR = \frac{1}{N}\sum_{i=0}^{N-1}(1+2i) = N \tag{2}$$

3.2 LBA-to-PBA mapping

Different from [9], the LBA-to-PBA mapping function is built into the in-place update SWDs in our design. As a result, sector-based file systems such as EXT4 and NTFS can be built on top of these SWDs nearly without any change. Write amplification management is transparent to the file systems.

Following conventional static address mapping used in HDD for in-place update SWD and using Figure 1 for illustration, the conventional mapping scheme will sequentially map LBAs [1-100] to physical track 1, LBAs [101-200] to physical track 2 and so on. Physical track 5 is a safety gap so it is skipped. LBAs [401-500] will then be mapped to physical track 6.

This mapping scheme is noted as “1234” in this paper as tracks are utilized in a left-to-right order. This scheme works fine for workloads with a small update percentage such as those “write once read multiple times” workloads or backup workloads. However, it will be expensive to make data updates because of significant write amplification overhead. As a result, better mapping schemes should be proposed for in-place update SWDs.

4 Novel Static Address Mapping Schemes

In this section, we describe several new address mapping schemes for in-place update SWDs and analyze their per-

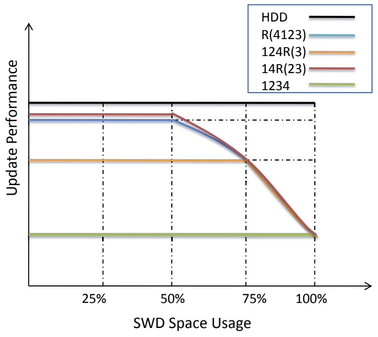

Figure 2: Update Operation Performance Prediction

formance for update operations. The comparison result is shown in Figure 2 and is validated later by experiments in Section 5.

4.1 General Principles

A band can be used more efficiently if we change the order of utilizing the tracks. Take one band as an example, the overall performance will be improved if the tracks are utilized in the order of “4123”. In other words, the 4th track will be first used, followed by the 1st track, the 2nd track, and finally the 3rd track. By doing so, when the space utilization of this band is less than 25%, and if all data is made to be present only in the last track then all the data can be updated freely. When the space utilization is less than 50%, let data appear only in the first track and last track. The two tracks (2nd and 3rd track) between them will work as a safety gap, therefore allowing both first track and last track to be updated without incurring any extra cost. When space utilization is no more than 75%, with same principle, the 2nd track and last track are free to be updated. Only updates to the first track will incur 1 extra read and 1 extra write. However, when the space utilization becomes close to 100%, then the overhead becomes similar to the “1234” allocation. This observation triggers us to propose space allocation schemes that take SWD space utilization into consideration since the space in the SWD will be used (or allocated) gradually.

The story is similar for the entire SWD. The general principle is the third tracks of all bands should be delayed for use until the SWD is 75% full. Although several static LBA-to-PBA mapping schemes can be proposed using this principle, we will only present three representative new address mapping schemes which are indicated respectively by “R(4123)”, “124R(3)” and “14R(23)”.

4.2 Mapping Scheme “R(4123)”

Mapping scheme “R(4123)” maps LBAs to the tracks of all bands in a Round-Robin fashion. It maps the first 25% LBAs to the 4th tracks across all bands. Similarly, the second 25% LBAs are mapped to 1st tracks across all bands. The rest LBAs are mapped in the same Round-Robin manner to 2nd and 3rd tracks. Symbol “R” therefore means Round-Robin as a naming convention.

This mapping scheme makes sure no write amplification will be incurred when SWD usage is no more than 50%. When SWD usage becomes close to 75%, only 1 extra read and 1 extra write request will be incurred if an update request is made to the 1st tracks. However, SWD performance drops quickly when it is almost full.

4.3 Mapping Scheme “124R(3)”

Mapping scheme “124R(3)” is an alternate option, which maps the first 75% LBAs to the 1st, 2nd and 4th tracks in an ordered sequential manner but maps the rest 25% LBAs to the 3rd tracks in a Round-Robin fashion, as the name suggests.

This scheme preserves better LBAs spatial locality than scheme “R(4123)” so less seek overhead can be expected. However update requests may incur write amplification even when SWD usage is less than 50%. This actually indicates a tradeoff between amplification overhead and seek overhead.

4.4 Mapping Scheme “14R(23)”

This mapping scheme maps the first 50% LBAs to the 1st and 4th tracks in an ordered sequential manner and maps the next 25% LBAs to the 2nd tracks in a Round-Robin fashion. The last 25% LBAs will finally be mapped to the 3rd tracks in a Round-Robin fashion.

In terms of update performance, this scheme generally follows the prediction curve of “R(4123)”in Figure 2 but may perform slightly better when SWD usage is less than 50% because of a little better LBAs locality. The actual performance, however, also depends on the LBAs distribution in a given workload.

4.5 Performance Prediction for Updates

Assuming all factors are the same but the LBA-to-PBA mapping scheme difference, Figure 2 roughly predicts the average update performance for all the mapping schemes as the SWD space grows. This prediction will be validated later in our experiments.

5 Experimental Evaluations

The overall performance of SWDs with these new allocation schemes are evaluated with several realistic traces and then compared to that of a SWD with the conventional scheme and a regular HDD.

5.1 Enhanced DiskSim

We emulate the in-place update SWD with an enhanced DiskSim. We enhance the DiskSim with two components: one is the *address mapper* component and the other is the *write amplifier* component. The address mapper translates a given LBA into a PBA according to a specified static mapping scheme and the write amplifier converts a write/update request into a set of read and write requests if write amplification is incurred. Whether

Trace	Inter-Arrival Time	Average Seek Distance	MAX LBA	MAX Request Size	Write Ratio
web_0	297.9411468	6245717.249	71116454	3200	0.70123
hp_c2247	14.19225897	273730.0428	2049836	134	0.488449
Financial2_0	0.06453672	591141.4663	2676179	3072	0.096978
SYN	50.00721344	0	2399999	8	1

Table 1: Trace Statistics

a write/update request will be amplified depends on the LBA and the current SWD usage.

We are simulating a SWD based on the parameters of an hp_c3323a disk drive in the DiskSim package. The SWD contains 3000 physical cylinders, each of which consists of 1000 blocks. Band size is 4 and write head width is 2. No obvious performance difference is observed when we configure to use 1 or 2 disk surfaces. The results we show below represents a single surface.

5.2 Traces

Four traces are used in our experiments, including one MSR trace (*web_0*)[2], one HP trace (*hp_c2247*)[1], one Financial trace (volume 0 of *Financial2*)[3] and a synthetic trace (*SYN*). The characteristics of these traces, including inter-arrival time (IAT) and write ratio, are shown in table 1. Since write amplification is essentially caused by update operations, these traces are picked according the write/update operation ratio[2]. For example, *web_0* is an update intensive workload, *hp_c2247* is a moderate update workload, *Financial2* (read intensive) and *SYN* (cold sequential write) are light update workloads.

SYN is used to mimic a backup workload which continuously writes data to an empty SWD until the space is fully used. Therefore this is a cold sequential write workload with no update. Its average request size is 8 blocks and the inter-arrival time follows a normal distribution of which mean is 50 ms with standard deviation 10 ms.

5.3 Experiment Design

As Figure 2 indicates, update operation performance changes as the SWD space usage grows. We therefore choose 25%, 50%, 75% and 100% as the sampling points to make performance comparisons. The synthetic trace only requires 75% and 100% as sampling points because none of the allocation scheme incurs write amplification overhead before 75% usage. We run 70 experiments in total, using different mapping schemes, different SWD space usages and different workloads combinations.

We run *web_0*, *Financial2* and *hp_c2247* with our enhanced DiskSim 4 times and each time we pre-fill the SWD with data to a particular usage (i.e., 25%, 50%, 75% and 100%). This will logically convert all writes in the workloads into updates. These traces have to be adapted before input to the enhanced DiskSim. For example, LBAs larger than the specified SWD usage have to scale down with modulus operations. Besides, request arrival rate has to be scaled down in order not to saturate the emulated SWD because of two reasons. First, the traces we use represent workloads to the storage arrays with multiple HDDs which have much better performance than a single SWD. Second, write amplification in a SWD incurs extra read and write operations, which results in a much bursty workload to the SWD. Therefore, in our experiments, we increase the inter-arrival time by 200 times for web_0, similarly, 5000 times for *Finanical2* [3] and 5 times for *hp_2247*.

We run *SYN* twice. The enhanced DiskSim runs the *SYN* workload and writes data into an empty SWD until the SWD is 75% full in the first experiments. Data is continuously written into an empty SWD until space is 100% full in the second experiments. This is done to emulate a backup workload or cold sequential write workload.

5.4 Result Discussions

In this section, we make performance comparisons using average response time and average write amplification ratio.

5.4.1 When SWD Space Usage Is Less than 75%

The average response time for the four traces are shown in Figure 3 (a) (b) (c) (d) respectively. It can be observed that SWDs using the three new mapping schemes constantly outperform the SWD using "1234" scheme. The performance difference is especially significant for moderate update and update intensive workloads such as *hp_c2247* and *web_0*. Besides, SWDs using new mapping schemes can provide a similar performance to that of a regular HDD. Furthermore, "R(4123)" and "14R(23)" constantly outperform "124R(3)" for traces with updates when SWD space usage is no more than 50%, which indicates that write amplification overhead has more performance impact than seek overhead in our experiments. This is because write amplification incurs extra operations to the SWD, which increases the number of outstanding requests and consequently cause longer queuing time for other requests. In other words, the workload becomes more bursty because of the extra requests.

The performance difference can be well explained with the average write amplification ratio graphs as shown in Figure 3 (e) (f) (g) (h). For example, due to the nature of "1234" scheme, its average WARs always

[2] We logically convert writes into updates as shown in Section 5.3

[3] *Financial2* has a huge variance in inter-arrival times. The workload is quite bursty from time to time but mean IAT is bigger than expected.

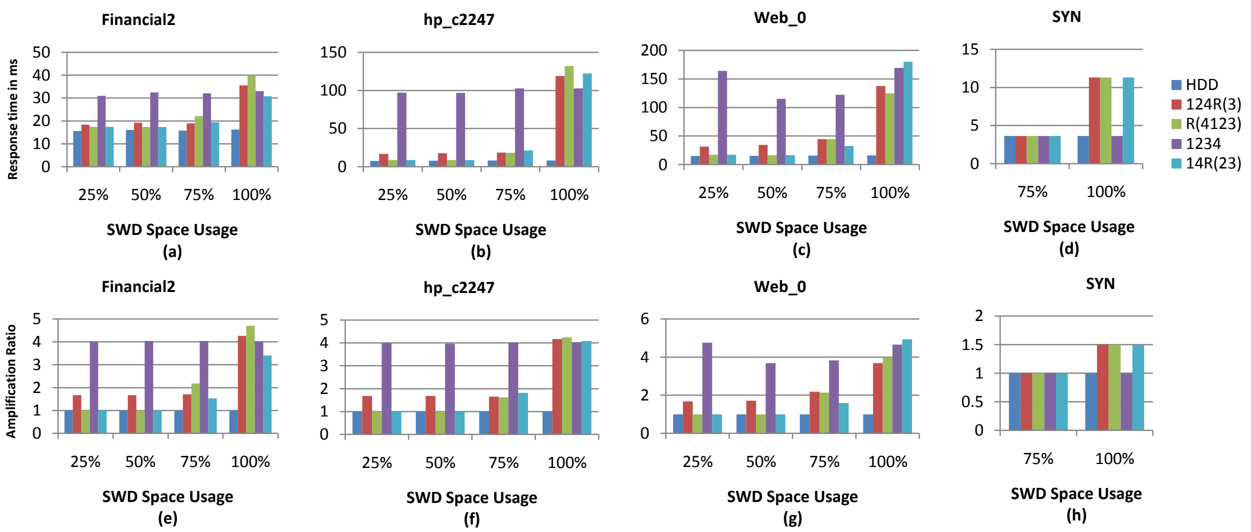

Figure 3: Average response time and write amplification comparison for four traces under different SWD space usages

stay around 4 regardless of the SWD space usages and traces. Note that *SYN* is a special case because it is a cold sequential write trace and it contains no update request at all. Similarly, the average WARs for “124R(3)” always stay around 1.67 when SWD space usage is no more than 75%, the average WARs for “R(1234)” and “14R(23)” stay at 1 when SWD space usage is no more than 50% and become around 1.67 when SWD space grows to 75%. These observations are consistent with our theoretical performance analysis and prediction previously. A bigger average WAR simply means a more bursty workload is resulted.

SYN is used to show that for backup-like workloads, the “1234” scheme does outperform the new mapping schemes when SWD space usage is over 75% full, although nearly the same performance is achieved when usage is lower than 75%. This may indicate a possible future direction of adaptive mapping schemes for multiple workloads and multiple volumes case.

5.4.2 When SWD Space Usage Is Close to 100%

All SWDs produce an average WAR of 4 when SWD space usage is close to 100% regardless of the mapping scheme used. Therefore the performance drops quickly and significantly bigger response time can be observed. This implies that when space usage is over 75%, defragmentation should be performed to make more room in the 3rd tracks, which will practically make SWDs maintain good performance.

Another observation is that when SWD space usage is close to 100%, every mapping scheme including “1234” has a chance to win because the actual performance depends on the LBAs distribution in the trace. For example, “R(1234)” works best for *web_0* but performs worst for *Financial2*. The reason is that more updates happen to take place in the 3rd and 4th tracks in *web_0* but more go to the 1st and 2nd tracks in trace *Financial2*.

6 Conclusions

In this paper, we have presented several new address mapping schemes for in-place update SWDs. By appropriately changing the order of space allocation, the new mapping schemes can improve the write amplification overhead significantly. Our experiments with four traces demonstrate that new mapping schemes provide comparable performance to that of regular HDDs when SWD space usage is less than 75%.

References

[1] DiskSim. `http://www.pdl.cmu.edu/DiskSim/`.

[2] MSR Cambridge Block I /O Traces. `http://iotta.snia.org/traces/list/BlockIO`.

[3] University of Massachusetts Amhesrst Storage Traces. `http://traces.cs.umass.edu/index.php/Storage/Storage`.

[4] A. Amer, D. D. Long, E. L. Miller, J.-F. Paris, and S. Schwarz. Design issues for a shingled write disk system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–12. IEEE, 2010.

[5] Y. Cassuto, M. A. Sanvido, C. Guyot, D. R. Hall, and Z. Z. Bandic. Indirection systems for shingled-recording disk drives. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–14. IEEE, 2010.

[6] G. Gibson and G. Ganger. Principles of operation for shingled disk devices. Technical report, Tech. Rep. CMU-PDL-11-107, Carnegie Mellon University, 2011.

[7] G. Gibson and M. Polte. Directions for shingled-write and twodimensional magnetic recording system architectures: Synergies with solid-state disks. *Parallel Data Lab, Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-PDL-09-014*, 2009.

[8] D. Hall, J. H. Marcos, and J. D. Coker. Data handling algorithms for autonomous shingled magnetic recording hdds. *Magnetics, IEEE Transactions on*, 48(5):1777–1781, 2012.

[9] D. Le Moal, Z. Bandic, and C. Guyot. Shingled file system host-side management of shingled magnetic recording disks. In *Consumer Electronics (ICCE), 2012 IEEE International Conference on*, pages 425–426. IEEE, 2012.

[10] C.-I. Lin, D. Park, W. He, and D. H. Du. H-swd: Incorporating hot data identification into shingled write disks. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2012 IEEE 20th International Symposium on*, pages 321–330. IEEE, 2012.

[11] I. Tagawa and M. Williams. High density data-storage using shingled-write. *Proceedings of the IEEE International Magnetics Conference (INTERMAG)*, 2009.

[12] R. Wood. The feasibility of magnetic recording at 1 terabit per square inch. *Magnetics, IEEE Transactions on*, 36(1):36–42, 2000.