Data Management and Layout for Shingled Magnetic Recording

Ahmed Amer[1,2], JoAnne Holliday[1], Darrell D. E. Long[2], Ethan L. Miller[2], Jehan-François Pâris[3], and Thomas Schwarz[4]

[1]Computer Engineering Department, Santa Clara University, Santa Clara, CA 95053 USA
[2]Jack Baskin School of Engineering, University of California at Santa Cruz, Santa Cruz, CA 95064 USA
[3]Department of Computer Science, University of Houston, Houston, TX 77204-3010 USA
[4]Universidad Católica del Uruguay, 11600 Montevideo, Uruguay

Ultimately the performance and success of a shingled write disk (SWD) will be determined by more than the physical hardware realized, but will depend on the data layouts employed, the workloads experienced, and the architecture of the overall system, including the level of interfaces provided by the devices to higher levels of system software. While we discuss several alternative layouts for use with SWD, we also discuss the dramatic implications of observed workloads. Example data access traces demonstrate the surprising stability of written device blocks, with a small fraction requiring multiple updates (the problematic operation for a shingled-write device). Specifically, we discuss how general purpose workloads can show that more than 93% of device blocks can remain unchanged over a day, and that for more specialized workloads less than 0.5% of a shingled-write disk's capacity would be needed to hold randomly updated blocks. We further demonstrate how different approaches to data layout can alternatively improve or reduce the performance of a shingled-write device in comparison to the performance of a traditional non-shingled device.

***Index Terms*—Areal density, data layout, data management, data storage systems, memory, shingled writing, SMR, two-dimensional magnetic recording (TDMR).**

I. Introduction

OVER the past 50 years, disk drive capacity has grown by nearly six orders of magnitude. Thanks to advances in recording technology, manufacturing, and materials, this growth has been advancing at a rapid rate, but is fast approaching the density limit imposed by the super-paramagnetic effect for perpendicular recording. While current drives store 400 GB/in^2, the current limit is estimated to be about 1 Tb/in^2 [22]. While shingled write disks (SWDs) [6], [8], [23] are not the only technology aimed at enabling drives that exceed this limit, it differs from competing approaches by offering an elegant solution that requires the least intrusive departure from current recording materials and manufacturing processes. However, it also introduces interesting new challenges in determining and implementing the most effective data layouts, as it imposes new limitations on our ability to perform random data updates. We discuss the various approaches to data layout that would enable the best use of such devices as replacements to existing drives, and we also discuss how alternative applications and interfaces might offer new opportunities to more effectively integrate such devices.

The elegant solution offered by shingled disks [6], [8], [23] is to use a write head with a stronger, but asymmetric, magnetic field. This approach is made possible by the fact that writes require a much stronger magnetic field than do reads. Shingled writing leverages this property by overlapping the currently written track with the previous track, leaving only a relatively small strip of the previous write track untouched. While this remnant is a fraction of the feasible write size, it is still sufficiently large to be read with current GMR read heads. As a result, shingled writing can place tracks closer together, and data density within a track can also be increased, giving a conservative estimate of density increase of about $2.3\times$ [23]. Achieving continued capacity gains in magnetic hard drives will likely require a combination of SMR and two-dimensional magnetic recording (TDMR) [21] technologies. While an SWD still allows for traditional random access reads, writes must be done sequentially because a single track write destroys the next k tracks, where k is typically 4–8. This radically changes the way in which the system must interact with the SWD. In addition to revised management of the SWD, effective use of nonvolatile RAM (NVRAM) such as flash or newer storage class memories can further overcome any potential architectural limitations of shingled writing. With the recent availability of large-scale solid state disks, magnetic disks may well become relegated to primarily archival storage roles. SWDs would be particularly well suited to the sequential-write, random-read access patterns that are likely in such scenarios. Nonetheless, in order to increase the chances of becoming economically viable and gaining widespread adoption, SWDs must be able to meet the traditional expectations of a random-access persistent storage device.

Disk data density improvements will eventually be limited by the superparamagnetic effect, which creates a tradeoff between the media signal-to-noise ratio, the writeability of the media by a narrow track head, and the thermal stability of the media; Sann *et al.* call this the *media trilemma* [21]. While various approaches to this problem have been proposed; shingled writing offers perhaps the most elegant solution. Rather than radically altering the makeup of the magnetic layer (as is done in bit patterned media recording (BPMR) [18]), or temporarily "softening" the magnetic material through microwaves or lasers (as is done with microve, heat, or thermally assisted magnetic recording, i.e., MAMR [25] or HAMR [3], [11], [13], [20]), shingled writing requires less radical changes to the structure of the underlying media. Shingled writing builds directly upon existing magnetic recording technologies by allowing data in

Manuscript received February 22, 2011; accepted May 09, 2011. Date of current version September 23, 2011. Corresponding author: A. Amer (e-mail: a.amer@acm.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMAG.2011.2157115

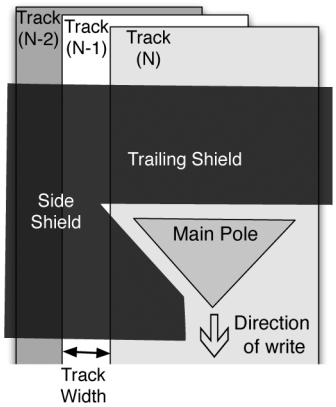

Fig. 1. Corner write head for shingled writes [1].

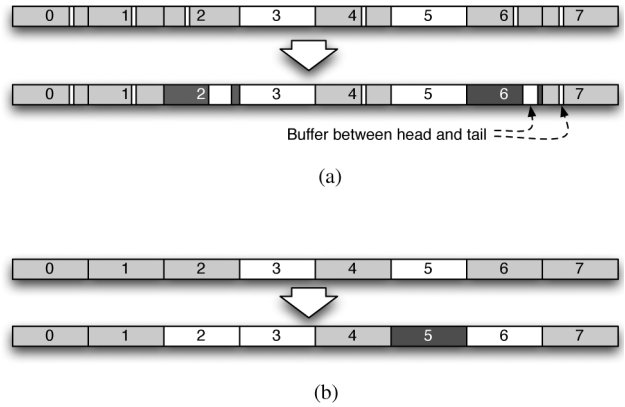

Fig. 2. Two basic layout options for SWDs [1]. (a) Bands filled with circular logs. Bands 2 and 6 are cleaned, moving their free space buffers to the right within each band, and potentially increasing the size of the buffers by not copying "dead" blocks from tail to head. (b) Bands treated as segments. Bands 2 and 6 are cleaned into Band 5, allowing them to be freed.

subsequent, but not prior, tracks to be destroyed during writes. Shingled writing does this by using a write head that generates an asymmetric, wider, and much stronger field that fringes in one lateral direction, but is shielded in the other direction. Fig. 1 shows a larger head writing to track n, as used by Greaves *et al.* in their simulations [7]. Because of the larger pole, the strength of the write field can be increased, allowing the use of a more stable medium. Shingled writing overlaps tracks written sequentially, creating effectively narrower tracks when the once-wider leading track has been partially overwritten. Shingled writing is thereby expected to increase storage densities by a factor of at least 2.5 [23] to 3 [7] times the current superparamagnetic limit of 1 Tb/in^2. This may be further increased by utilizing TDMR [4], [9], [10], [21], which allows the placements of tracks even more closely together thanks to more sophisticated signal processing [10], [24] and write encodings, but at the expense of requiring additional disk rotations or multiple read heads.

In the following, we first present how the technology behind shingled writing changes the functional behavior of a disk drive, and the data layout approaches that can be used to address these changes. We then proceed to describe the various options for integrating such drives into the storage architecture. We further present an initial assessment of typical input/output (I/O) workloads, as ultimately the effectiveness of any layout or hybrid approach is heavily dependent on the nature of such workloads. Of particular interest is our finding that changes to device blocks are very heavily concentrated in hot zones, and that most blocks are written only once over extended periods of time [1].

II. Data Layout for Shingled Write Disks

If an SWD was to write all its tracks in a shingled manner, it would not be possible to overwrite an individual track without affecting subsequent tracks. To limit such effects to a defined region of tracks, an SWD would need to store the bulk of its data in bands, a collection of b contiguous tracks in which data can only be appended. Contrary to traditional wisdom regarding colocation within a cylinder, bands are better constructed from contiguous tracks on the same surface. Nevertheless, the disk can also reserve a substantial amount of storage space for data that needs to be updated in-place. Reserving several gigabytes of such space would consume less than 1% of the total available disk capacity [1]. Such an area could be implemented as either a dedicated set of tracks, or as additional NVRAM memory with which the SWD could be supplemented.

At the end of each band is a buffer of unused tracks, where that buffer is at least the number of tracks that would be overwritten by a shingled write. Fig. 2 shows the two basic layouts that SWDs may use. One option is to keep a circular log within each band and reclaim freed space by moving live data at the tail of the log to the head of the log and then considering the cleansed part free. A second option is to clean bands through compacting one or more complete bands into a smaller number of unused bands, thereby freeing space allocated to written tracks that have since been replaced.

A. Block Layout Options

With the bulk of data in an SWD stored in bands, a simple, elegant solution only writes complete bands that each contain a segment of a log-structured file system (LFS) [19]. This presupposes buffering of data, but would have effective writes. A second possibility only appends to bands. Compared to the previous solution, writes are less efficient (as they would be in the form of smaller write volumes), but both possibilities would utilize the band completely.

A third possibility stores a circular log in each band. Presumably, the size of the band would be at least doubled compared to a design that cleans complete bands atomically. To prevent writes to the head from destroying data in the tail, an additional k track gap (the *intraband gap*) between the head and the tail would typically be necessary. Fig. 3 describes such a layout. To recover freed space, a cleaning operation moves live data from the tail to the head, recovering the freed space by not copying the defunct data to the head.

A final possibility would use flexible band sizes. In the absence of special workloads, neighboring bands could be joined to store large objects more efficiently by using the normally unusable *interband* gap between bands. While we could manage such bands, the deletion of data will eventually necessitate addressing data fragmentation. We consider this strategy unlikely to be suitable for a general purpose SWD, but mention it for completeness.

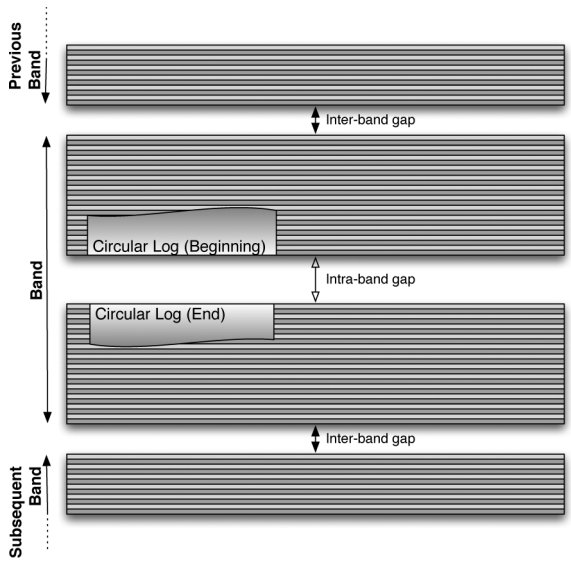

Fig. 3. Layout in a band with circular log [1].

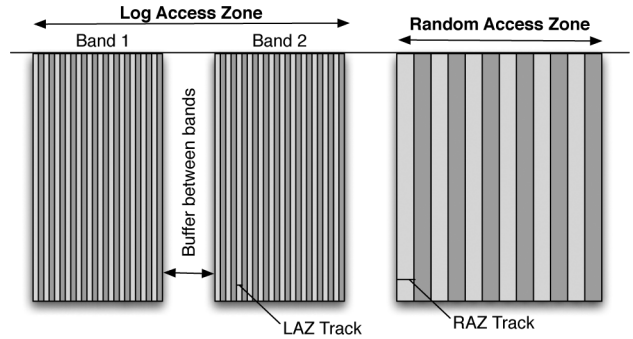

Fig. 4. Example of track layout on a shingled write device with a LAZ and a RAZ. The LAZ consists of two stretches of tracks separated by an interstretch buffer while the RAZ is made up of 12 tracks [1].

While a mainly write-by-append device is likely to use some form of LFS, there is nothing that prevents it from having more than a single log. One reason for the separation is the difference in longevity between metadata and data, as metadata is often more short-lived (i.e., more frequently updated) than data.

Another reason to separate data into different logs is to avoid adversely affecting block read-ahead performance. For example, assigning files to different logs would avoid the intermingling of data blocks from two separate files within the same log (which adversely affects the performance of a read-ahead policy when only one of such files is accessed later). An example where such an ability would be vital is a user who downloads several movies at the same time at much smaller individual download speeds than replay requires. Interspersing all movie objects because they happened to be written as a single stream of blocks to an individual log would result in poor read-ahead of data blocks when individual movies are read for later replay.

A final advantage for managing multiple logs is the ability to designate different data to different logs to aid cleaning and hierarchical management. For example, data that have not been deleted for some time are much more likely to remain undeleted for an extended period in the future. Arranging logs into a hierarchy based on such insight would allow us to migrate increasingly stable subsets of existing logs to more stable logs. This would subsequently result in reduced cleaning overheads as a growing number of logs would remain stable as stable data are migrated to such "lower level" logs. A hierarchy of logs would not set an upper limit on the number of times that a long-living stored object is copied, but it should help to keep this number low, and result in increased write efficiency. It should be noted that a possible drawback of having multiple logs is the need to move to different locations for writes. In contrast, a single log would have much less need to reposition the head for writes. This issue will become more important as writes dominate the workload. In contrast, the rate and frequency of log cleaning and space reclamation is more dependent on the rate of block updates, not just writes. In essence, the rate and need for data relocation and space reclamation will be heavily dependent on the nature of the workload experienced, specifically the updates to previously written blocks.

B. Enabling In-Place Updates Through Data Placement

Uses of an SWD as a primary storage device in a system can potentially benefit greatly from a storage area that allows in-place updates. Continuing with our discussion of the benefits of multiple append-only logs, this can be seen as a region for frequently updated blocks to reside and which is free from the destructive-update nature of a shingled write. We see two mechanisms for providing such a functionality: using a dedicated region of NVRAM with which we would augment the SWD; or using a dedicated region of the disk that consists of solely one-track bands. This random access zone (RAZ) would consist of single tracks, each followed by k empty tracks so that the track can be overwritten without affecting other data on the disk (effectively rendering it a one-track band). Fig. 4 shows a simple layout with two LAZ bands followed by a RAZ of 12 tracks.

While the area we can devote to RAZ is proportionally small, it is quite substantial in absolute capacity as 1% of 1 TB is 10 GB. However, storing all metadata in RAZ is impossible unless files are, by current standards, exceedingly large, or unless individual blocks are large. We can break up the set of tracks making up RAZ in whatever form we want and place the tracks on the disk without any loss of capacity. If we use RAZ to store metadata or directories, we can place it close to the LAZ areas where the files themselves are being stored, in a manner similar to the Berkeley FFS [14] use of cylinder groups.

III. Integration of SWDs Into Systems

We see two basic strategies for using SWD in current computer systems. First, we can mask the operational differences of an SWD. This can be achieved solely by adapting the layout of the device, or through a combination with NVRAM—flash, phase change memory (PCM), or other storage class memories (SCM)—as proposed by Gibson and Polte [6]. The second strategy would be to use a standalone SWD, possibly with relatively little added NVRAM, and with a specialized file system or object store serving as the interface. The former approach will allow the use of an SWD as a drop-in replacement for existing

disks and allow us to offer a standard disk interface, while the latter would allow us to embrace the characteristics of the SWD and more easily mask them or exploit them for specialized applications. In particular, if the latter strategy is used and we opt for an object store interface to the device, this offers distinct advantages in terms of data layout and block management thanks to the added semantic knowledge that can be gleaned and exploited by the device. However, such higher level object-based interfaces also result in more specialized applications and a reduced flexibility when it comes to deployment of SWDs for general purpose usage. That could well be a key factor in determining the success and early widespread adoption of SWDs.

A. SWD With a Block-Device Interface

To increase the likelihood of shingled write recording finding widespread adoption, an SWD needs to be able to function within existing systems without major changes to other system components. Economic considerations make it unlikely that different magnetic recording technologies will be used for specialized components suited only for certain workloads. This means that SWDs would need to function as a "drop-in" replacement for current hard drives.

A principle problem for a general block-based file system using an SWD device is contiguity, as most file systems spend effort on placing related information in contiguous logical block addresses (LBA). In the append-only write mode of SWD, the relocation and remapping of updated blocks would result in such contiguity being thwarted and lead to potential performance degradation. Throughput to a selection of random blocks is about three orders of magnitude slower than access to contiguous blocks in a track. While a heavily edited object might not see such a heavy access degradation, we can certainly imagine a workload such as very large database tables with frequent edits of small records where the performance loss could rapidly become considerable. A similar case might be the interleaving of blocks from different objects created simultaneously and over an extended period of time such as the concurrent downloads of movies or the creation of lengthy log files. This problem could be alleviated by utilizing a large NVRAM write cache, or through intelligent data grouping. While there have been excellent efforts to build viable log-structured solutions to SWD data management [2], these approaches succeed by limiting the scope (and thereby volume) of metadata required to manage such log-structuring. An alternative to limiting the scope of the remapping and log-structuring efforts is to reduce the volume of metadata required to track interblock relationships [5].

B. Benefits of Alternative Interfaces

More freedom in data management is obtained if we dispense with the traditional disk device interface, and instead offer a higher level interface. For example, and object storage interface would be able to better leverage knowledge of block relationships and types. Cleaning would be simplified thanks to an awareness of object deletions, and the selection of blocks to place in valuable RAZ or NVRAM areas of the device would be greatly aided by the ability to readily distinguish metadata and data blocks. It would also be easier to assign blocks to different logs should blocks belonging to different objects need to be written at the same time. This ability becomes particularly useful when read access needs to be optimized, as such an assignment would avoid fragmentation of blocks associated with the same object.

If the goal of 7-TB disks by 2015 is achieved, the usual file system interface for the user might become difficult to use. If SWDs serve several computing systems as secondary storage, an object store would offer a simpler route to a more clearly defined solution to security, sharing, and authentication.

IV. Workload Evaluation

The performance and effectiveness of an SWD, particularly as a replacement for a general purpose disk, will be heavily dependent on the workloads observed by the device. Whether we need to employ larger NVRAM caches, or reserve increased capacity for a RAZ, is dependent on the rate and frequency of updates to previously written blocks. Based on our experiments with recorded disk activity in different settings, we feel that it should most likely be possible to mitigate the append-only nature of shingled writing. To clarify we revisit and summarize our experimental results in evaluating the rate of disk block updates [1]. We focus on the rate at which individual blocks are updated, as when relatively few blocks are updated, this implies a lessened for RAZ and NVRAM space to enable an SWD to be used in place of a conventional disk.

For the workloads we have evaluated, our results indicate that blocks are indeed rarely updated. This in turn suggests that a very limited use of RAZ or NVRAM would make SWD a successful replacement for current disks. The modest 1%–3% capacity overheads of a RAZ we have estimated may indeed be sufficient to mask the majority of block updates. While some workloads would seem perfectly suited to an SWD, particularly workloads with minimal updates to previously written data such as archival workloads, we evaluated workloads typical of general purpose personal usage of a disk, as well as specialized workloads drawn from a system being used to edit video, and a third system dedicated to managing a music library. We present results from all but the last experiment, which was found to have negligible block update events over a period of almost a month. This was not surprising as the update of a block is the rewriting of its contents, and we do not consider the initial writing of data to a block as an update event. This meant that the regular addition of media files to the library did not incur any updates beyond the negligible updates to the library metadata. Our general purpose and video editing workloads showed noticeable block update behavior, but nonetheless this remained restricted to a very small percentage of all blocks and supports the general usefulness of an SWD device.

A. Workload Description

The general-purpose trace sets evaluated were drawn from laptop and personal computers running Mac OSX gathered from November to December 2007, as well as more recent block level traces collected during early 2010 and 2011. Our block-level traces reflect device-level requests, but not requests satisfied by a cache, and are therefore reflective of what would

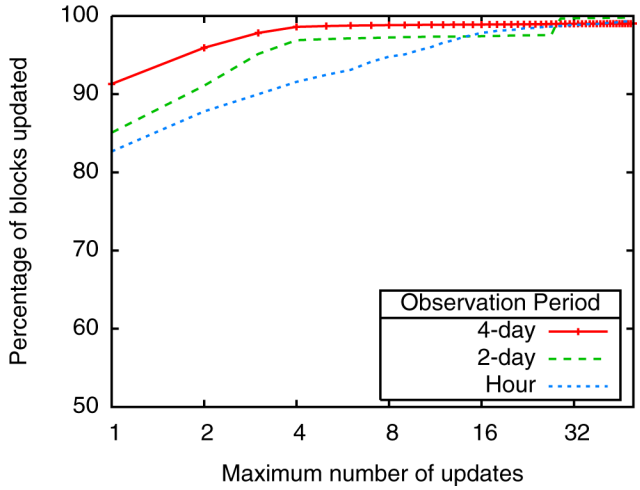

Fig. 5. As the observation period lengthens, we note an increase in the percentage of stable blocks.

be experienced by the individual disk device. The requests are in the form of block reads and writes. They also include requests resulting from paging activity. The "video editing" workload was gathered in January 2010, on a system running Mac OSX 10.5.8, and also using a filesystem formatted for HFS+ with journaling enabled. The workload was gathered over an approximately 3-h period during which two videos were edited and transcoded, each composed of multiple video files edited to form the final videos. The workload also included the transcoding of a 1-h-long video file. In addition to the workloads we have collected (which include another specialized workload from a system serving virtual machine images), we have processed and evaluated block level and web traces collected by other researchers (the NASA web traces, and the "MSR Cambridge traces") [16], [17].

B. Results

The number of times that a given block is updated grows with the observation period. Interestingly enough, we found that a decreasing percentage of written blocks were written multiple times. In other words, we observe a very small percentage of hot blocks being rewritten, whereas a growing percentage of written blocks is written only once. For example, just under 94% of all accessed disk blocks have undergone four or fewer updates. Fig. 5 gives our result. The x-axis gives the maximum number of updates and the y-axis gives the percentage of blocks updated. We give three curves, one for an observation period of an hour, two days, and finally four days. We see that more than 85% of all disk blocks written were never updated within the hour and 93% of all disk blocks written were never updated within a day. This trend continues as the observation period lengthens. This suggests that the reclamation rate of data stored in a LAZ is very low. It also suggests that if data are stored in NVRAM or a RAZ until it reaches a certain age (e.g., an hour or a day), then the vast majority of multiple updates to already written disk blocks are masked.

Fig. 6 shows the impact of differentiating disk blocks that hold filesystem metadata from those that hold user data. Distinguishing disk blocks that were written as metadata from disk blocks that held data resulted in the most notable difference in the percentage of stable disk blocks observed. Specifically, blocks containing metadata are consistently more likely

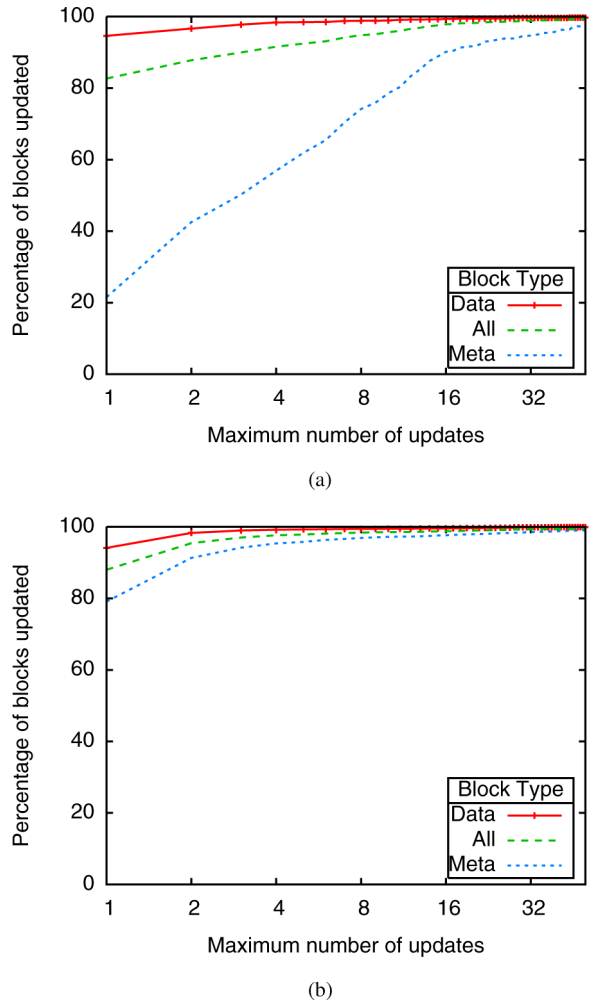

Fig. 6. The impact of block-type separation on update rates, as observed over a period of (a) one hour and (b) one day [1].

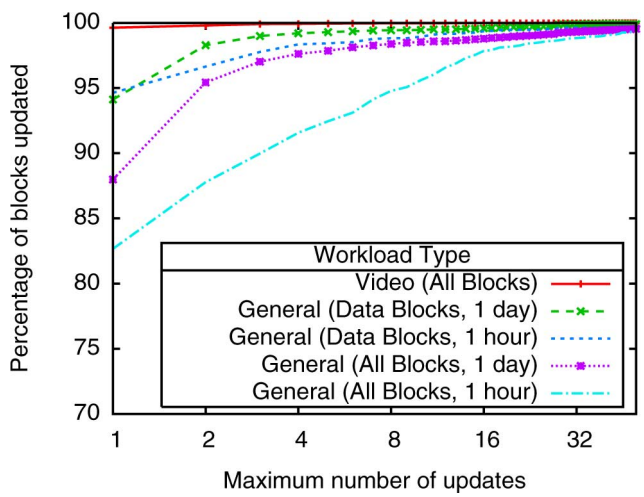

Fig. 7. The impact of favorable workloads. In this instance, we see the video editing workload demonstrating much greater block stability than the general purpose usage workload. While within the general purpose usage workload we continue to observe that eliminating metadata blocks has a greater impact on stability than the length of the observation period.

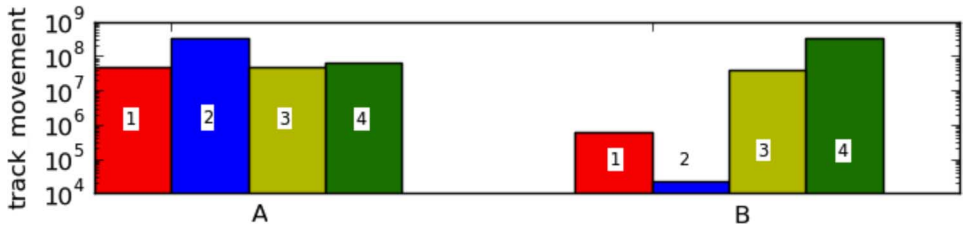

Fig. 8. The widely varying impact of workloads on alternative layout schemes [12].

to endure multiple updates than data blocks. The difference is significantly more pronounced for shorter observation periods, which supports the conclusion that a small amount of NVRAM (or RAZ) would be sufficient to accommodate the majority of updates that occur over shorter time periods. This is particularly true if the blocks selected for caching can be identified as metadata blocks (and therefore more likely to be updated).

Distinguishing metadata and data blocks can be achieved in one of two ways: implementing shingled layout optimizations at the filesystem or object level; or attempting to automatically classify block types based on their behavior. Object and file-level approaches that can separate metadata blocks from data blocks [15] have demonstrated the benefits of such semantic knowledge on improving the utilization of hybrid storage systems. Alternatively block type can be determined through the automated classification of device blocks based on observed update behavior. However the type of block is identified, it is then possible to allocate metadata and data blocks to different regions of the disk, to the appropriate log segment or circular buffer, or to NVRAM storage. Metadata blocks remain more likely to experience multiple and regular updates, whereas data blocks tend to consistently persist as originally written [1].

In Fig. 7, we see the dramatic effect of focusing on a specialized application. The figure shows the percentage of blocks updated for both the general purpose workload and the more specialized video editing workload. The general-purpose workload was not modified to accommodate our SWD model and was drawn from computers acting as their users' primary systems, and yet this workload demonstrates an encouragingly low rate of block updates. The percentage of updated blocks drops as the observation period grows, but the effect of focusing solely on blocks holding user data ("data blocks" in the figure) is even more significant. When we consider the "video" editing workload, we see an even more dramatically reduced percentage of updated blocks. This strongly suggests the additional benefits to be gained by pairing an SWD with a complementary application. While likely suitable for general purpose use, it appears likely that video editing, and media library applications that include large data objects that tend to remain unchanged, are ideally suited for SWDs. In this instance, we see less than 0.4% of blocks experiencing content updates during the entire editing session (which was approximately 3 h of heavy use). As we have described previously, further experiments with an audio library produced results that simply offered no notable block update behavior over the entire trace collection period. This was due to the fact that practically no problematic block updates were observed (as those would imply changing previously written blocks, which is an exceedingly uncommon event for any static media library intended for playback and not destructive editing).

To illustrate the importance of using an appropriate data layout policy, we evaluated multiple layout schemes and found that shingled writes and the accompanying data layout schemes can actually result in performance improvements. In Fig. 8, we compare four layouts (1 through 4). The first is an unmodified layout, as would be used on a traditional disk. The second scheme is an idealized log-structured layout applied to a shingled disk, and requiring relocation of data being written. The third and fourth schemes represent a shingled disk with in-place update of data blocks. This is not possible without relocating any affected adjacent tracks, and in scheme 3, this is done with the aid of an intermediate NV-RAM buffer capable of storing all affected adjacent tracks, while in scheme 4 the NV-RAM buffer is severely limited. The figure represents the sum of intertrack seeks performed to satisfy the workload. The first workload considered $\mathbf{A}$ is a block trace drawn from a web workload, while the workload $\mathbf{B}$ is from a system used for software development. The sum of intertrack seek distances is a measure representative of the mechanical delays incurred by seek operations. While this distance-based metric is not linearly proportional to access latencies, using this logical measure allows us to consistently compare behavior between widely varying workloads without introducing the inaccuracies and inconsistencies inevitable with a time-based metric.

While it might be tempting to consider log-structuring as a better approach to data layout with shingled tracks than deliberate rewriting of adjacent tracks, that is not true for all workloads. While we found it to be a valid observation for most workloads, the few workloads that were dominated by heavy read traffic experienced better performance with the schemes that attempted to keep data in-place (schemes 3 and 4). This was true for workload $\mathbf{A}$ in Fig. 8. It may also seem intuitive a log-structured layout scheme as being a means of overcoming the limitations of a shingled-write scheme, a solution that would likely come at a performance cost, which is not always the case. In Fig. 8, we see that workload $\mathbf{B}$ requires almost $100\times$ less track movements from a log-structured shingled disk (scheme 2) than a traditional disk without any redirection scheme imposed (scheme 1). In short, the success of a layout scheme for shingled-write disks is dependent on the workload, and may result in performance improvements over a nonshingled disk that attempts no remapping of data blocks.

V. Conclusion and Future Work

Our workload analysis reiterated above allows us to reach the following conclusions for the general purpose device workloads considered.

1) Keeping track of updates to blocks allows us to identify hot blocks. The volume of hot blocks is small enough to allow us to efficiently allocate them to a RAZ or NVRAM of minimal capacity relative to the capacity of the overall SWD.
2) A file system or object interface that allows the device to distinguish metadata from user data can gain from the ready and efficient identification of hot blocks. This suggests the benefits of object stores based on SWDs, the automated classification of block types, or the sharing of block-type information with the block device driver.
3) The choice and success of data layout schemes is heavily dependent on the workload observed.

In addition to these conclusions, we have previously shown that opting to track larger block sizes, while slightly detrimental to our ability to distinguish hot blocks, has a negligible effect and yet allows us to reduce metadata overheads by indexing and remapping larger (i.e., fewer) blocks within bands and zones [1]. Our latest workload evaluation results further support our view that the data management and layout challenges posed by shingled writing can likely be surmounted and may be helpful in their own right.

Acknowledgment

The authors would like to thank R. Wood of Hitachi GST for valuable comments and feedback. The work of T. Schwarz was supported by a grant of the Stiles Family Foundation. The work of E. L. Miller and D. D. E. Long was supported by the National Science Foundation under Grants CNS-0917396 (part of the American Recovery and Reinvestment Act of 2009 [Public Law 111-5]) and IIP-0934401, and by the industrial sponsors of the Center for Research in Intelligent Storage (CRIS) and the Storage Systems Research Center (SSRC).

References

[1] A. Amer, D. D. E. Long, E. L. Miller, J.-F. Paris, and T. Schwarz, "Design issues for a shingled write disk system," in *Proc. 26th IEEE Symp. Mass Storage Syst. Technol.*, 2010, DOI: 10.1109/MSST.2010.5496991.

[2] Y. Casutto, M. Sanvido, C. Guyot, D. Hall, and Z. Bandic, "Indirection systems for shingled-recording disk drives," in *Proc. 26th IEEE Symp. Mass Storage Syst. Technol.*, 2010, DOI: 10.1109/MSST.2010.5496971.

[3] W. A. Challenger, C. Peng, A. Itagi, D. Karns, Y. Peng, X. Yang, X. Zhu, N. Gokemeijer, Y. Hsia, G. Yu, R. E. Rottmayer, M. Seigler, and E. C. Gage, "The road to HAMR," in *Proc. Asia-Pacific Magn. Recording Conf.*, 2009, pp. 1–2.

[4] K. S. Chan, J. Miles, E. Hwang, B. V. K. Vijayakumar, J. G. Zhu, W. C. Lin, and R. Negi, "TDMR platform simulations and experiments," *IEEE Trans. Magn.*, vol. 45, no. 10, pp. 3837–3843, Oct. 2009.

[5] D. Essary and A. Amer, "Avoiding state-space explosion of predictive metadata with SESH," in *Proc. Int. Performance Comput. Commun. Conf.*, 2009, pp. 111–120.

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

[7] S. Greaves, Y. Kanai, and H. Muraoka, "Shingled recording for 2–3 Tbit/in^2," *IEEE Trans. Magn.*, vol. 45, no. 10, pp. 3823–3829, Oct. 2009.

[8] P. Kasiraj, R. New, J. de Souza, and M. Williams, "System and method for writing data to dedicated bands of a hard disk drive," U.S. Patent 7490212.

[9] A. R. Krishnan, R. Radhakrishnan, and B. Vasic, "LDPC decoding strategies for two-dimensional magnetic recording," in *Proc. IEEE Global Commun. Conf.*, 2009, DOI: 10.1109/GLOCOM.2009.5425930.

[10] A. R. Krishnan, R. Radhakrishnan, B. Vasic, A. Kavcik, W. Ryan, and F. Erden, "2-D magnetic recording: Read channel modeling and detection," *IEEE Trans. Magn.*, vol. 45, no. 10, pp. 3830–3836, Oct. 2009.

[11] M. Kryder, E. Gage, T. McDaniel, W. Challener, R. Rottmayer, G. Ju, Y.-T. Hsia, and M. Erden, "Heat assisted magnetic recording," *Proc. IEEE*, vol. 96, no. 11, pp. 1810–1835, Nov. 2008.

[12] Q. M. Le, K. SathyanarayanaRaju, A. Amer, and J. Holliday, "Workload impact on shingled write disks," unpublished.

[13] K. Matsumoto, A. Inomata, and S. Hasegawa, "Thermally assisted magnetic recording," *Fujitsu Sci. Tech. J.*, vol. 42, no. 1, pp. 158–167, Jan. 2006.

[14] M. K. McKusick, W. N. Joy, S. J. Leffler, and R. S. Fabry, "A fast file system for UNIX," *ACM Trans. Comput. Syst.*, vol. 2, no. 3, pp. 181–197, 1984.

[15] E. L. Miller, S. A. Brandt, and D. D. E. Long, "HeRMES: High-performance reliable MRAM-enabled storage," in *Proc. IEEE 8th Workshop Hot Topics Oper. Syst.*, Elmau, Germany, May 2001, pp. 83–87.

[16] D. Narayanan, A. Donnelly, and A. Rowstron, "Write off-loading: Practical power management for enterprise storage," in *Proc. 7th USENIX Conf. File Storage Technol.*, 2008, pp. 253–267.

[17] NASA [Online]. Available: http://ita.ee.lbl.gov/html/contrib/nasa-http.html

[18] H. Richter, A. Dobin, O. Heinonen, K. Gao, R. Veerdonk, R. Lynch, J. Xue, D. Weller, P. Asselin, M. Erden, and R. Brockie, "Recording on bit-patterned media at densities of 1 Tb/in^2 and beyond," *IEEE Trans. Magn.*, vol. 42, no. 10, pp. 2255–2260, Oct. 2006.

[19] M. Rosenblum, "The design and implementation of a log-structured file system," Ph.D. dissertation, Univ. California Berkeley, Berkeley, CA, 1992.

[20] R. E. Rottmeyer, S. Batra, D. Buechel, W. A. Challener, J. Hohlfeld, Y. Kubota, L. Li, B. Lu, C. Mihalcea, K. Mountfiled, K. Pelhos, P. Chubing, T. Rausch, M. A. Seigler, D. Weller, and Y. Xiaomin, "Heat-assisted magnetic recording," *IEEE Trans. Magn.*, vol. 42, no. 10, pp. 2417–2421, Oct. 2006.

[21] C. K. Sann, R. Radhakrishnan, K. Eason, R. M. Elidrissi, J. Miles, B. Vasic, and A. R. Krishnan, "Channel models and detectors for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 46, no. 3, pp. 804–811, Mar. 2010.

[22] Y. Shiroishi, K. Fukuda, I. Tagawa, S. Takenoiri, H. Tanaka, and N. Yoshikawa, "Future options for HDD storage," *IEEE Trans. Magn.*, vol. 45, no. 10, pp. 3816–3822, Oct. 2009.

[23] I. Tagawa and M. Williams, "High density data-storage using shingle-write," presented at the IEEE Int. Magn. Conf., May 7, 2009, session FA.

[24] Y. Wu, J. O'Sullivan, N. Singla, and R. Indeck, "Iterative detection and decoding for separable two-dimensional intersymbol interference," *IEEE Trans. Magn.*, vol. 39, no. 4, pp. 2115–2120, Jul. 2003.

[25] J.-G. Zhu, X. Zhu, and Y. Tang, "Microwave assisted magnetic recording," *IEEE Trans. Magn.*, vol. 44, no. 1, pp. 125–131, Jan. 2008.