Directions for Shingled-Write and Two-Dimensional Magnetic Recording System Architectures: Synergies with Solid-State Disks

Garth Gibson, Milo Polte,
Carnegie Mellon University

Carnegie Mellon University Parallel Data Lab Technical Report CMU-PDL-09-104
May 2009

Summary: Shingled-writing and two-dimensional magnetic recording, TDMR, will change core characteristics of magnetic disk operation and require systems software be adapted appropriately. Because a band of adjacent tracks overlap one another, they must be written in a specific order. Once overlapped, a track cannot be updated in place, because the tracks overlapping it will be overwritten by the update. If this behavior is exposed to operating systems directly, there will be very low acceptance of these products. However, disk controller software can emulate full compliance with existing interfaces, and may be able to mask almost all performance implications as well.

The number of tracks shingled together is a key parameter of shingled-writing. With a shingle spacing of 10% of the write width, 90% of the final shingle is wasted. If capacity overhead is to be limited to 10%, each band will contain about 100 shingles, as shown in Figure 1. Appending a sector to an incomplete band may have conventional performance, but updating an existing sector could require rewriting as many as 100 adjacent tracks. The system model for shingled-writing will be two operations: "append a sector to a partially written band" and "delete a band and write its first sector".

Reading a sector from a shingle-written surface may have conventional performance, but, in the case of TDMR, 1 or 2 sectors in adjacent tracks on both sides may also need to be read to resolve inter-track interference. Accordingly, reading a single sector might require 3 to 5 rotations, 10 times conventional rotational positioning time.

In summary, sequential reading and writing of 1000s of tracks on a TDMR disk is likely to be similar to today's disks; random reading of smaller amounts of data may have as little as three to five times lower throughput; random rewriting of small amounts of data will probably not be supported; and random appending to previously deleted and partially written bands may be as fast as random writing today, or faster as an append write may not need repositioning. Unfortunately, existing system software almost never reads or writes 100s of tracks at a time, it is just now starting to entertain proposals for a storage "delete" (TRIM) command and small random reading and re-writing operations are key application performance parameters.

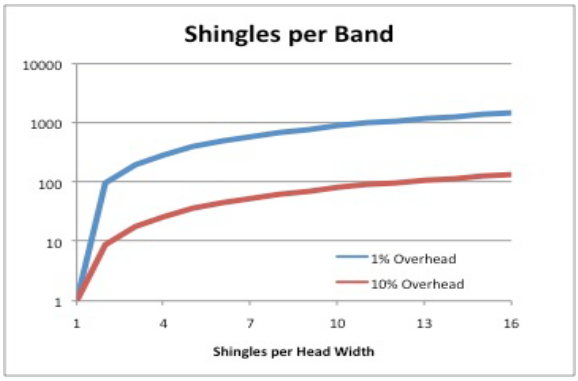


Figure 1: Band size as a function of shingle width and capacity overhead per band.

Log-based System Model: Shingled-write disks may be best understood as a collection of append-only "logs". Software designers have extensive experience with logs. Databases write every change to a "log" on disk to get sequential performance while making the change durable, then more leisurely update all associated read-optimized data structures. File systems technology based on logs was developed in the 1990s because RAID systems execute large writes far more efficiently than small random writes [1, 2]. However even log-based file systems execute random writes in some cases, and generally do not prefetch three or more tracks with each read. Moreover, the vast majority of disk management software is not log-based, and depends heavily on fast small random reads and writes. Effecting a rapid deployment of new technology will depend on achieving as little change as possible in operating systems software.

Synergy with Solid-State Disks: A similar problem faces flash-based solid-state disks (SSDs). SSDs are organized into "pages" that are erased before being written, and erase is relatively slow. While small random reads of SSDs are about 100 times faster than today's magnetic disks, simple SSDs (ie., Memoright, MTron) that read-erase-write each modified page deliver small random write performance no better than magnetic disks, as shown in Figure 2 [3]. More sophisticated SSDs (ie., X25m/e, ioDrive) overcome the read-erase-write problem by dynamically remapping every written sector, logging these potentially disparate sectors onto consecutive physical locations on one SSD page [5, 6], allowing apparently random small writes to be about 100 times faster than magnetic disks [3, 4].

The key point is that by employing similar firmware TDMR disks should be able to offer a fully compliant standard disk interface, supporting small random writes and, by dynamically remapping written sectors to append to the most convenient band of shingles, deliver write performance comparable to today's magnetic disks.

Reading TDMR disks, if multiple adjacent tracks must be read to recover a sector, is more problematic. To mask the large latency of such reads, if possible at all, very effective buffer caching will be needed, and may impact the design of software layers managing these disks [4]. Moreover the address mapping data structures needed for dynamically remapping writes to the end of a log will need to be maintained in a non-volatile random access memory, pushing up the cost of the disk controller, requiring some tracks on the disk to be non-shingled, requiring a complex scheme to find the last written contents of the log on disk.

It is also possible that SSD technology, in addition to being an example of how to overcome the lack of update-in-place writes, will be a good choice for embedded disk cache technology, enabling hybrid solutions that have the cost-effective capacity of shingled-writes and the cost-effective small random read and write performance of sophisticated SSDs.

Closing: Shingled-writing imposes serious change on the order that sectors must be written, but this can be masked with software in the disk controller in much the same manner as SSDs mask the need to erase a block before writing any part of it. The three to five rotations needed to do a small read in TDMR is a more difficult performance problem. It would facilitate rapid deployment much better if TDMR was able to read a random sector in one rotation.

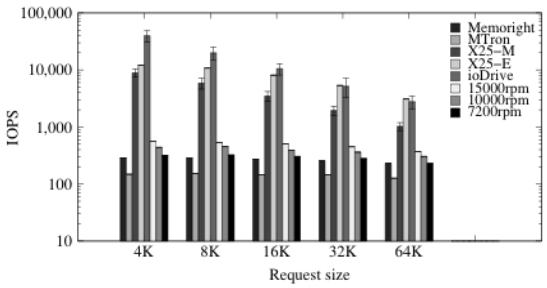


Figure 2: Random small write performance for commercial solid-state and magnetic disks.

Acknowledgements: This work was supported the DOE Petascale Data Storage Institute (PDSI), DE-FC02-06ER25767, and the Institute for Reliable High-Performance Information Technology (IRHPIT), Los Alamos Award 54515-001-07.

References:

[1] Patterson, D., G. Gibson, R. Katz, "A Case for Redundant Arrays of Inexpensive Disks (RAID)," ACM SIGMOD Conf. on Management of Data, 1988.

[2] Rosenblum, M., J. Ousterhout, "The Design and Implementation of a Log-Structured File System," ACM Trans. on Computer Systems, v10, n1, 1992.

[3] Polte, M., J. Simsa, G. Gibson, "Comparing Performance of Solid State Devices & Mechanical Disks," 3rd Petascale Data Storage Workshop (PDSW08), 2008.

[4] Polte, M., J. Simsa, "Enabling Enterprise Solid State Disk Performance," Integrating Solid-state Memory into the Storage Hierarchy (WISH09), 2009.

[5] Gal, E., Toledo, S., "Algorithms and data structures for flash memories," ACM Computing Surveys, v37, n2, June 2005.

[6] Agrawal, N., Prabhakaran, V., Wobber, T., Davis, J. D., Manasse, M., Panigrahy, R., "Design tradeoffs for SSD performance," USENIX 2008 Annual Technical Conference, Boston MA, June 2008.