Carnegie Mellon

Directions for Shingled-Write and TDMR System Architectures: Synergies with Solid-State Disks

Garth Gibson

www.pdl.cmu.edu

May 7, 2009

Short Bio

Co-author, A Case for RAID, 1988

Professor, CS & ECE, CMU, 1991-

Systems Thrust Leader, DSSC, CMU, 1990s

Founder, Parallel Data Lab, CMU, 1993

Founder & CTO, Panasas Inc, 1999

  HPC storage @ Los Alamos, BP, Intel, Boeing, NIH, Ferrari, Citadel

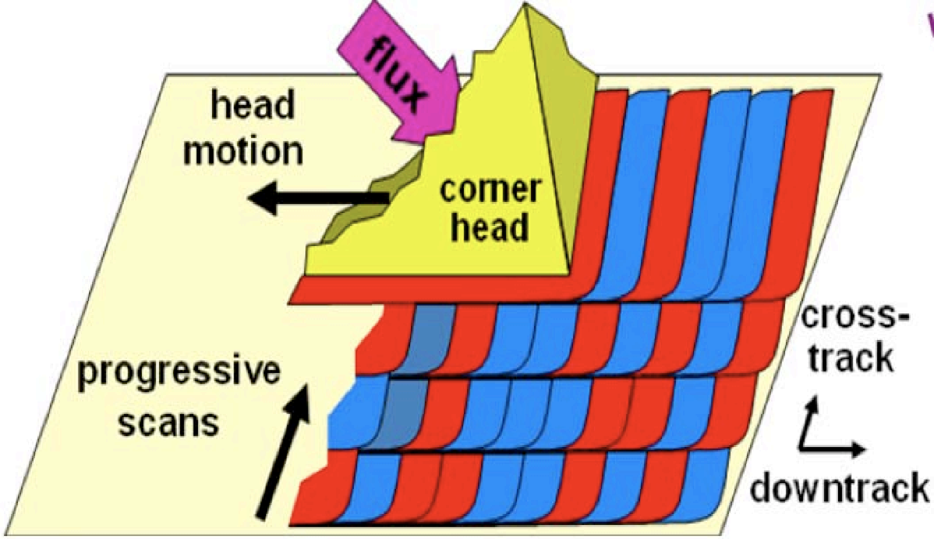
Co-Instigator, SCSI OSD & IETF Parallel NFS stds

Storage Networking Industry Tech Council, 2000s

Steering Cmte, File & Storage Tech (FAST) Conf

PI, DOE Petascale Data Storage Inst., 2006-

Shingled-Writing



Garth's simple world view

- HAMR, BPMR: big changes in fab/assembly
- Shingled-writing does not need big changes

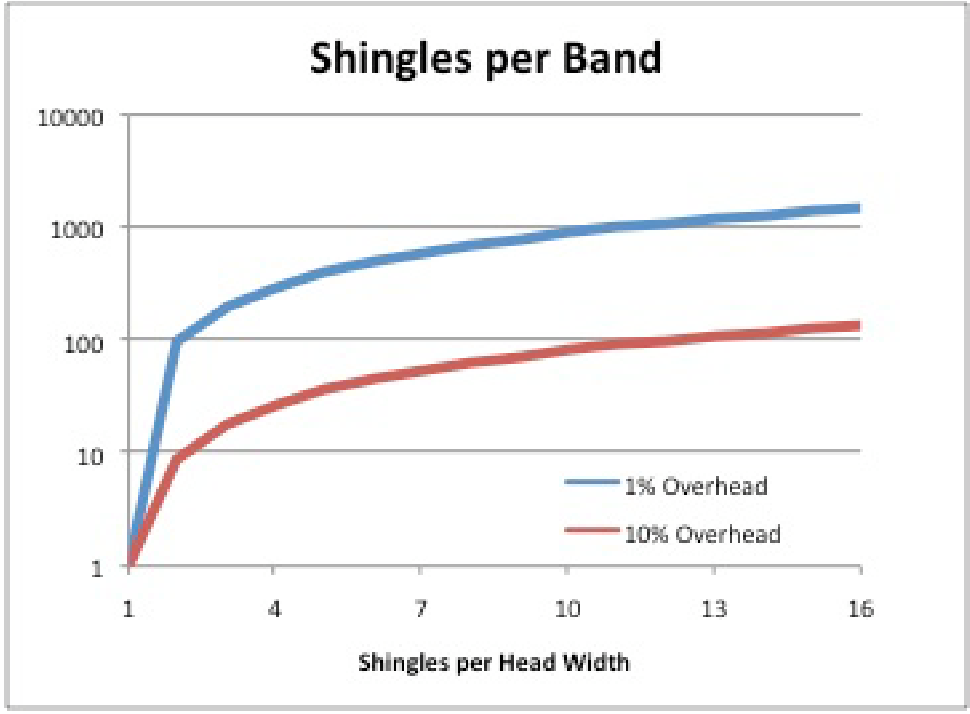
Shingle-writing means

- Partially overwriting tracks, for closer pitch
- Inability to modify one embedded sector without rewriting cross-track neighbors

Loss of Update-in-place

Banding of shingles

- Last track is wider, capacity overhead
- Tracks per band (@ 90% overlap):
- 1% ov => 1000 &
  10% ov => 100



Modifying a random sector in a band of 100 tracks

- Avg. of 50 revs to rewrite overlapped tracks!

Writing System Model

Shingled-write disk is N bands, each of order 1 GB

Append to end of a band has today's performance

Overwriting non-end of band "deletes" rest of band

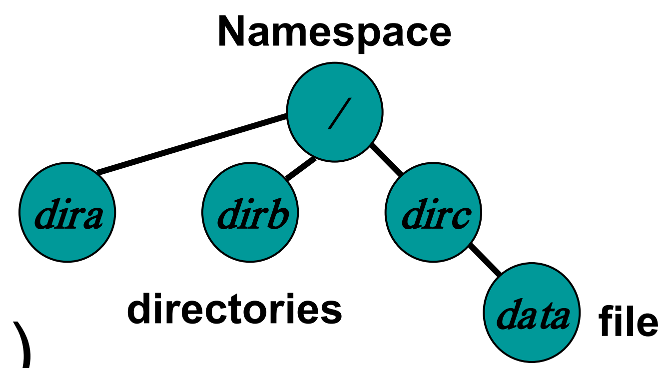
Writing start of band deletes prior content

Performance prohibitive to update-in-place at all

Can systems software cope with this?

No

File Systems 101



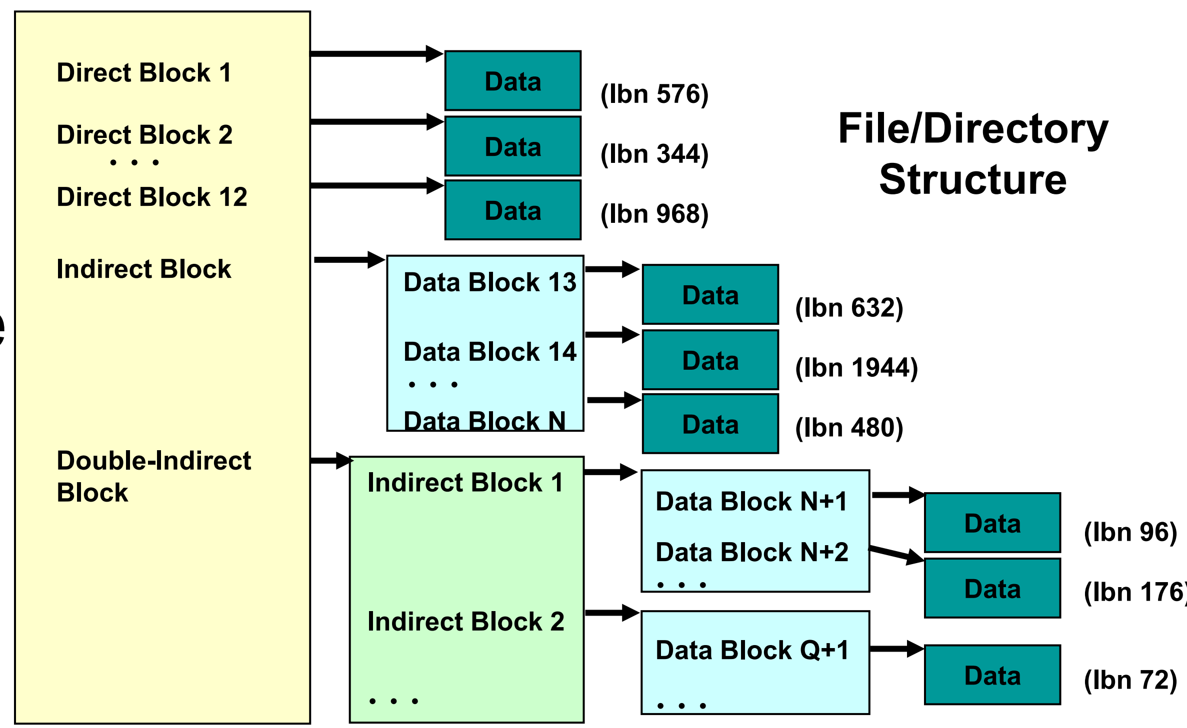
File systems store structured data

Metadata (block lists, attributes, …)
are generally small

Page-at-a-time
from OS

Disk fragments
with delete

Small writes b/c
Metadata!
Hole filling



& Files are Small

CDF of general file size

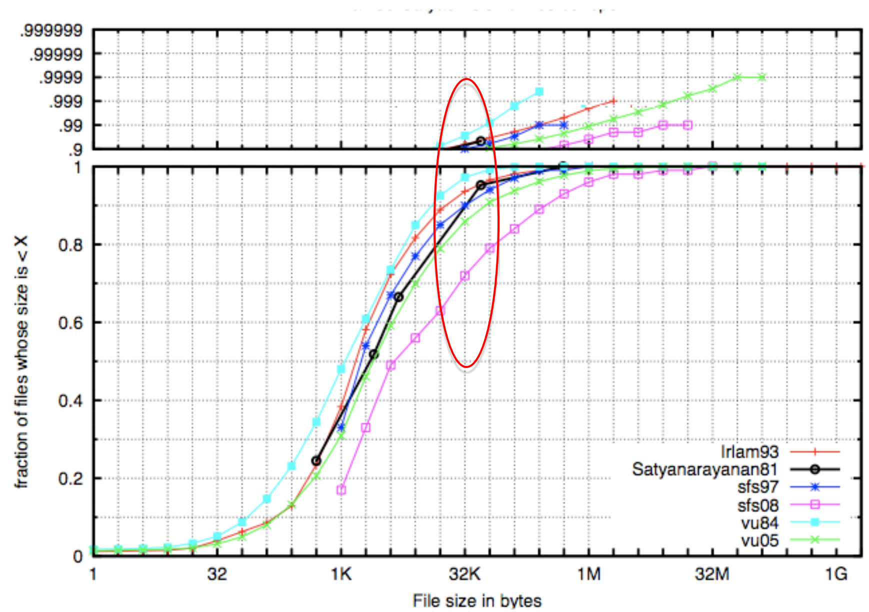
Historically

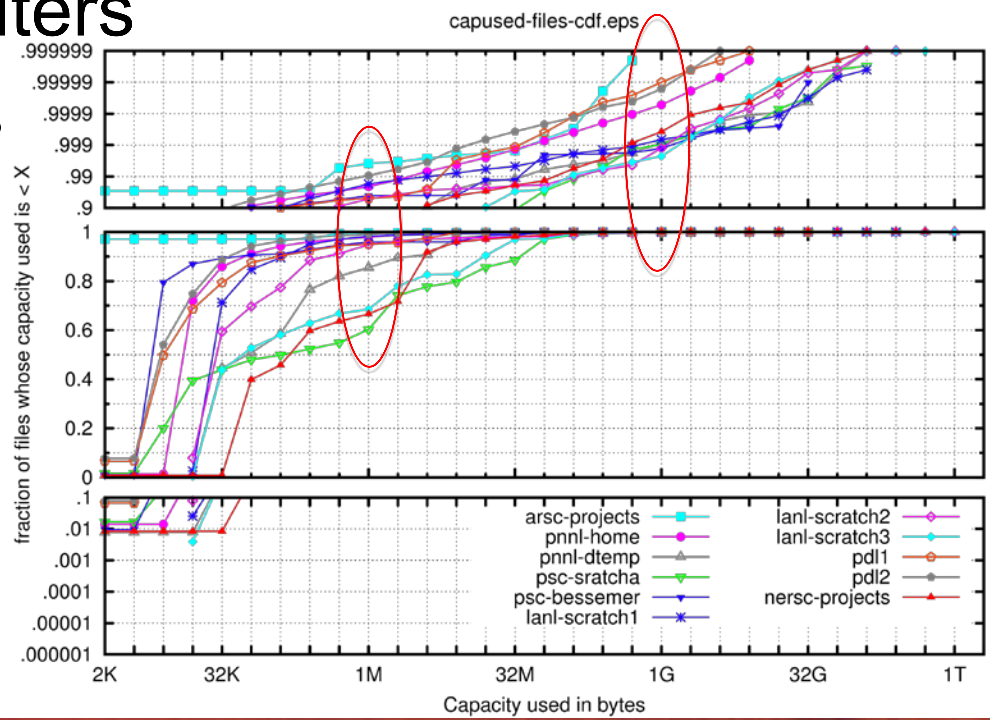
- > 75% < 32KB

Today’s supercomputers

- 60-99% < 1MB
- < 0.1% > 1GB

Most space in large files, but no avoiding the small ones





Carnegie Mellon
Parallel Data Laboratory

System Model for Hard Disks

Hard disk is a memory model: billions of sectors

File system allocation is search for free sectors

- To avoid "losing" space, small holes written

Durability/fault tolerance forces prompt writing

- Metadata is small and often written

Storage performance improvement is always:

- "Make disk writes larger by merging data"

But can't fundamentally avoid small writes

Same Problem for Flash

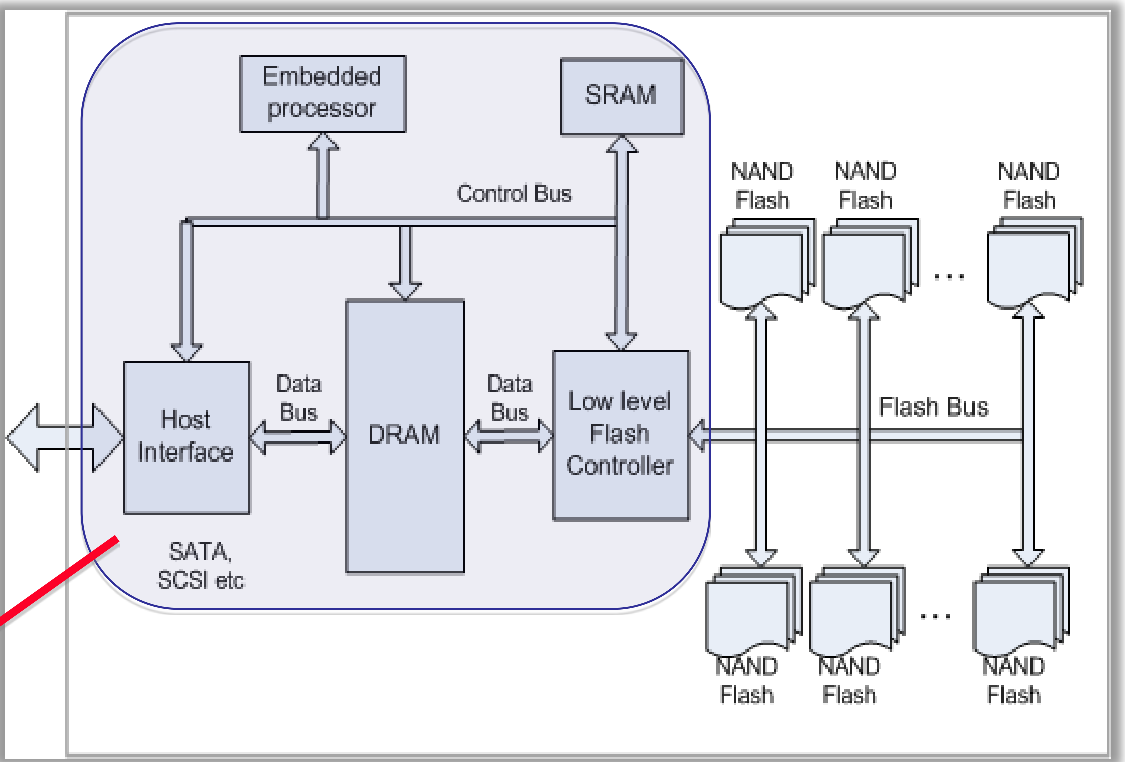
Flash SSD organized as “bands” of “sectors”

Must pre-erase band before programming data

Hide erase in FTL

Simple products
rewrite band
on all writes

Smart products
remap LBN
dynamically



Flash Translation Layer (FTL)

Carnegie Mellon
Parallel Data Laboratory

Shingled-write needs “FTL”

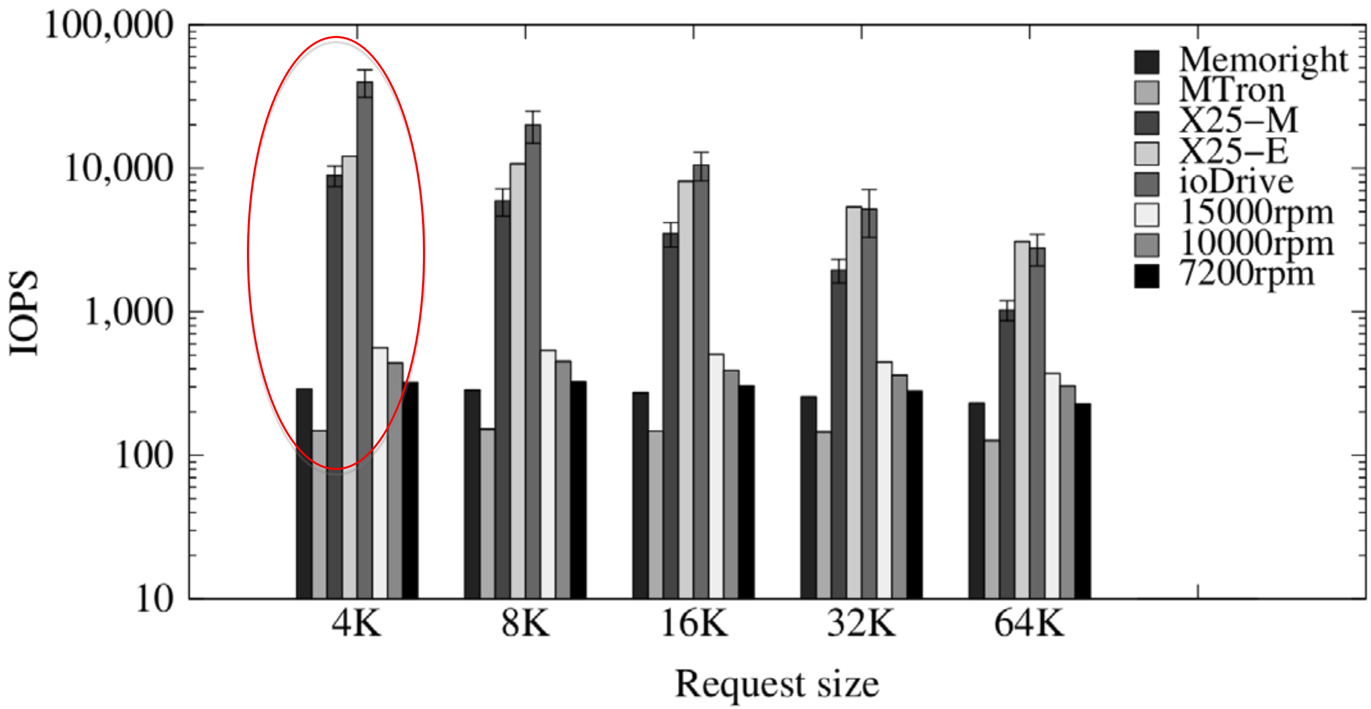
- Use embedded processor to translate full SCSI/ATA command set to “append” & “rewrite”
- Host “overwrite” is append and record new location
  - Prior location is now “wasted space”
    - Overprovision space to absorb waste
  - Background cleaning rewrites live part of bands
    - Same as today’s defrag tools
    - New TRIM command to expose waste
- Not new: 1992 Log-structured file system paper
  - NetApp, Panasas use remapping disk layout

Example: Flash Write Speeds

Measuring today's simple and smart flash SSDs

100x – 1000x more small writes per second

Remapping can rescue Shingled-writing disks!

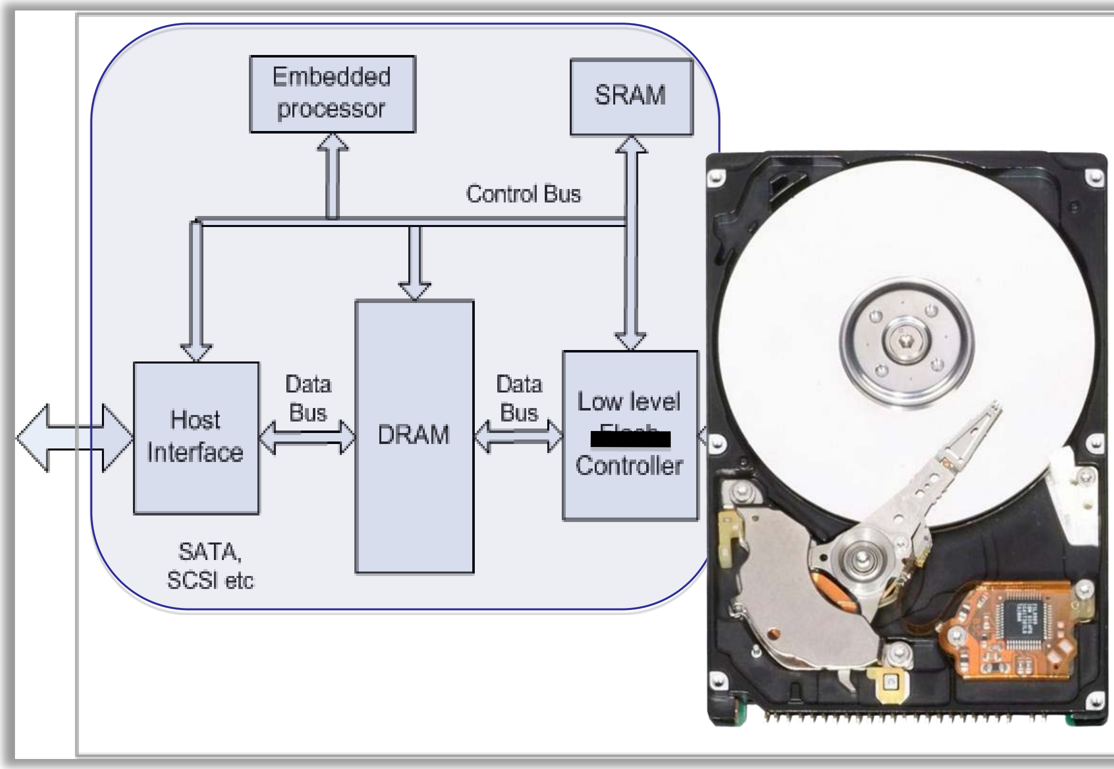


Shingled-write w/ translation

Its just code ☺

Okay, that means a faster CPU and more DRAM

and Complexity!



But you can start with flash translation code

Hire from FusionIO alumni ☺

What About Reading?

Reading a shingle involves signal processing in two dimensions (TD) – down and cross track

One approach to TDMR involves gathering signal from 1-2 adjacent tracks on both sides

Means 3 to 5 revs to read a single sector

- 3x – 5x lower small random read rates

Remapping on write probably doesn't help

- Read traffic depends more on applications than on system software/translation layer

Summary

Shingled-written disk is N bands of sequentially written sectors, each of order GB

Disk can still offer normal commands, write speed using “translation layer” embedded code

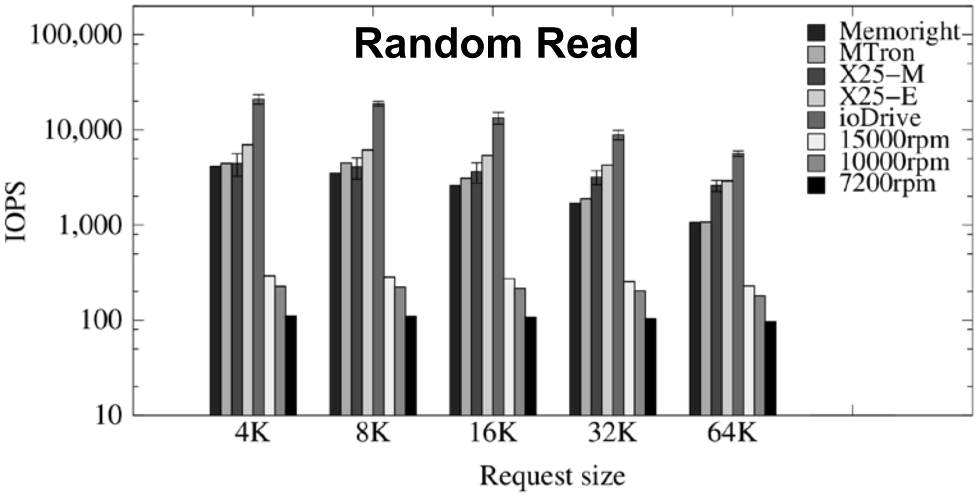
- Take Flash SSD FTL as starting point
- Flash-inspired TRIM command helps

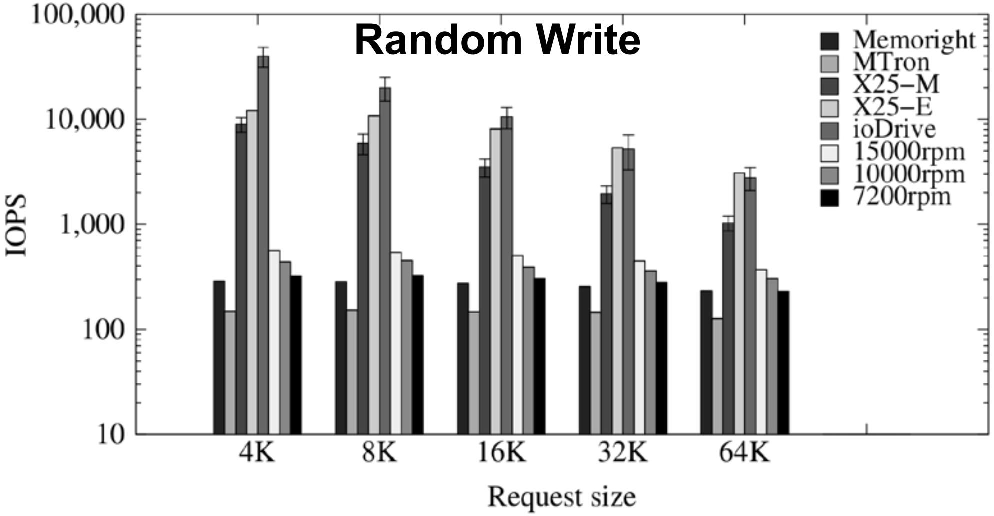
TDMR reading a bigger problem

- 3-5 revs per small read hard to hide
- This could reduce market acceptance

A Little More on SSD & Disks

- SSD performance !!
  - Big impact on systems coming
- Hybrid SSD+Disk
  - Cost of Disk bits
  - Speed of SSD
  - Compelling!
- SSD hybrid could
  - “solve” TDMR speed issues





Carnegie Mellon

www.pdl.cmu.edu

A few references

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

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

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.

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

www.pdl.cmu.edu and www.cs.cmu.edu/~garth