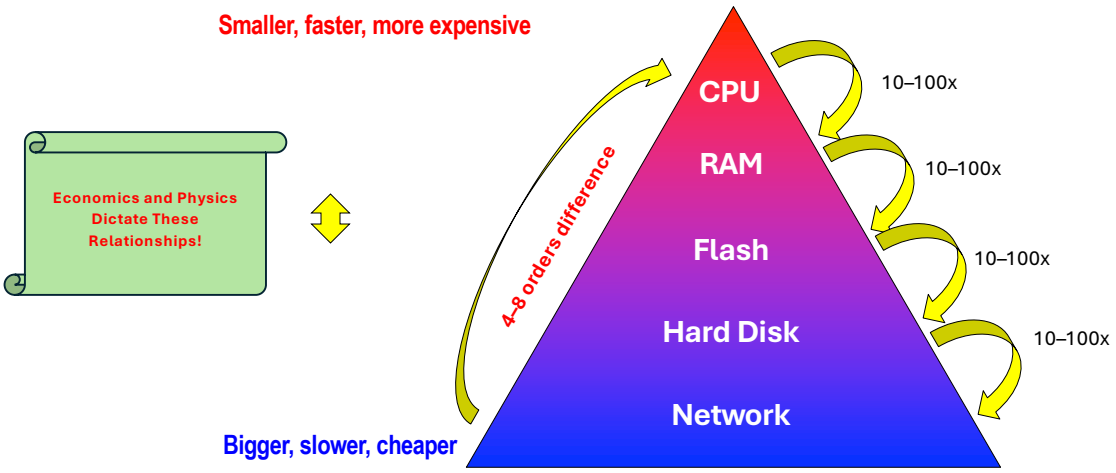
Everything You Always Wanted to Know about Storage Analysis

(But Were Afraid to Ask ;)

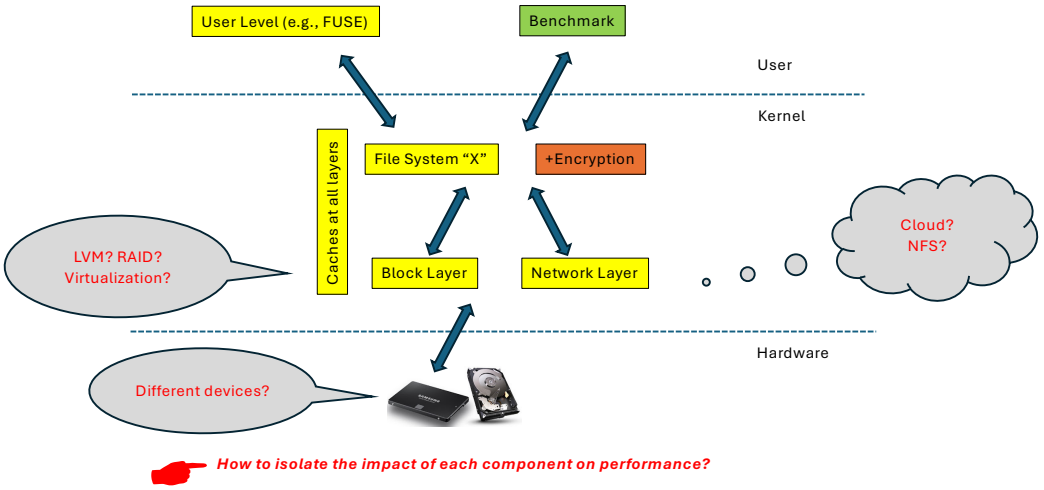
Erez Zadok
Stony Brook University


Storage is Complex,
Data is Everywhere,
Life is Good.

The Storage Hierarchy Pyramid



Deep Storage Stack, Data Everywhere

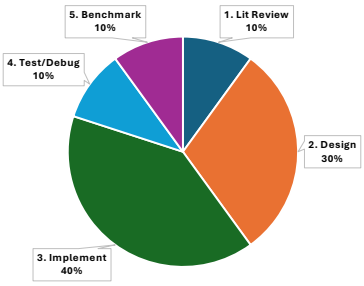


How to isolate the impact of each component on performance?

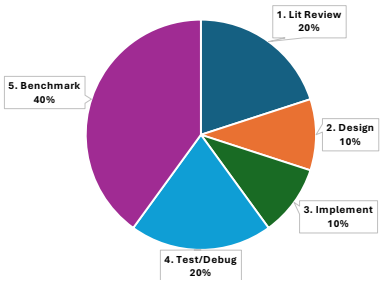
Stony Brook University

Storage Benchmarking is Time Consuming

Typical Process



Better (More Realistic) Process



Benchmarking reveals:

- Bugs (crash)
- Performance bottlenecks
- Implementation flaws
- Design flaws

Other:

- Hardening
- Regressions
- Code/Data Release

Dirty secret: missing related work, can get you rejected, esp. related work of PC member. I look at abstract -> Intro -> Skim eval (hop through figs+captions).

Then you need to benchmark…

What Tools to Benchmark?

A. Micro-Benchmarks? (e.g., FIO, dd, many more)
   i. Useful to test specific features (e.g., added encryption)
   ii. Can evaluate worst-case behavior, isolate features/components
   iii. But, not representative of "real world" workloads

B. Macro-benchmarks? (e.g., Filebench, RocksDB/LevelDB, many more)
   i. Useful to evaluate more realistic workloads
   ii. But, still synthetically generated

C. Trace Replay? (e.g., SNIA IOTTA trace repository)
   i. Considered most realistic, based on actual system traces
   ii. But: traces can be stale, take long time to replay (e.g., clock-time, AFAP?)

☛ ***Answer: "D" – All of the above, but justify***

Don't just pick what other published papers did: it can get stale. Find what benchmarks fit your system.

How Many Parameters to Vary?

Type	Values	Number of Tests
Benchmark	Micro, Macro, Traces	3
Workloads	Sequential, random, mixed	3
Workload/Cache size	Small, medium/default, large	3
No. of Systems	At least two CPUs	2
No. of Threads	1, 4, 16	3
Custom (hash, cipher, key sizes)	5 (?)	5
No. of runs per experiment	5	5
Total No. of Experiments		**4,050**
Total time to benchmark	**If 1 min per experiment**	**2.8 days**
Total time to benchmark	**If 2 min per experiment**	**5.6 days**
Total time to benchmark	**If 15 min per experiment**	**42 days**

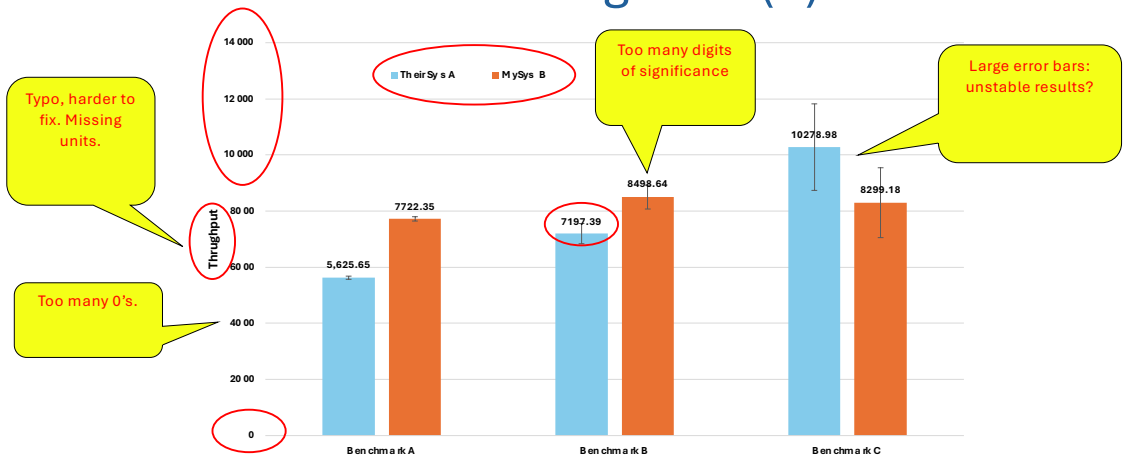
☛ ***These are conservative values!***


The Dark, Often Ignored
Art of Presentation

Presenting Data (1)

Stony Brook University

Presenting Data (2)

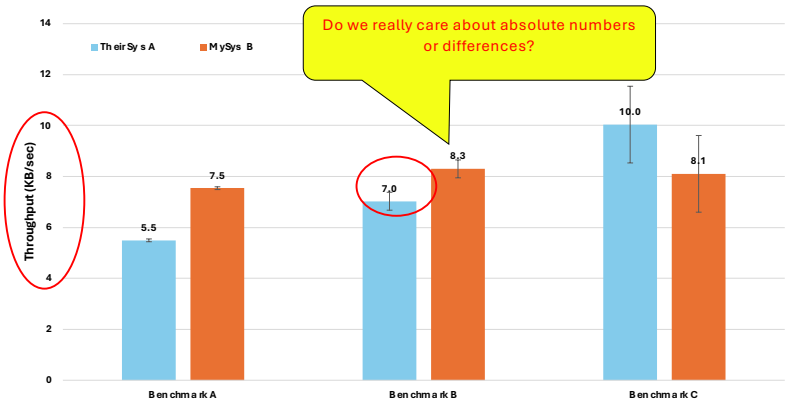


Stony Brook University

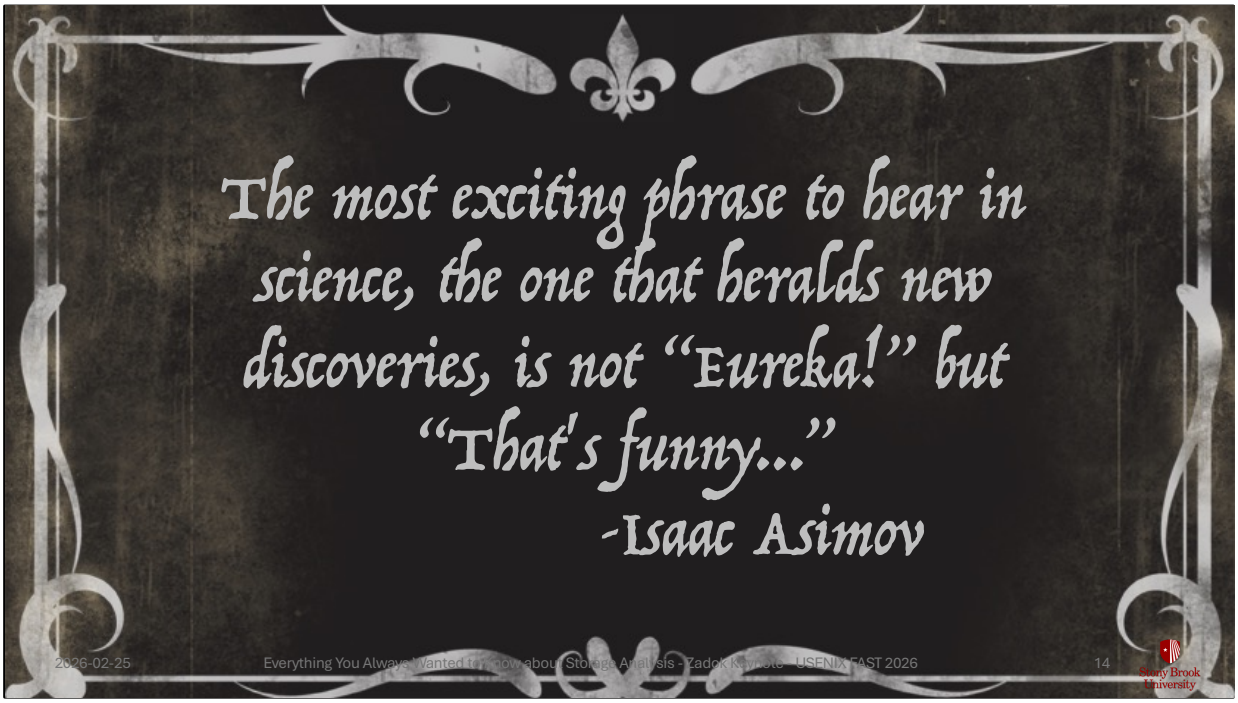
Presenting Data (3)

Stony Brook University

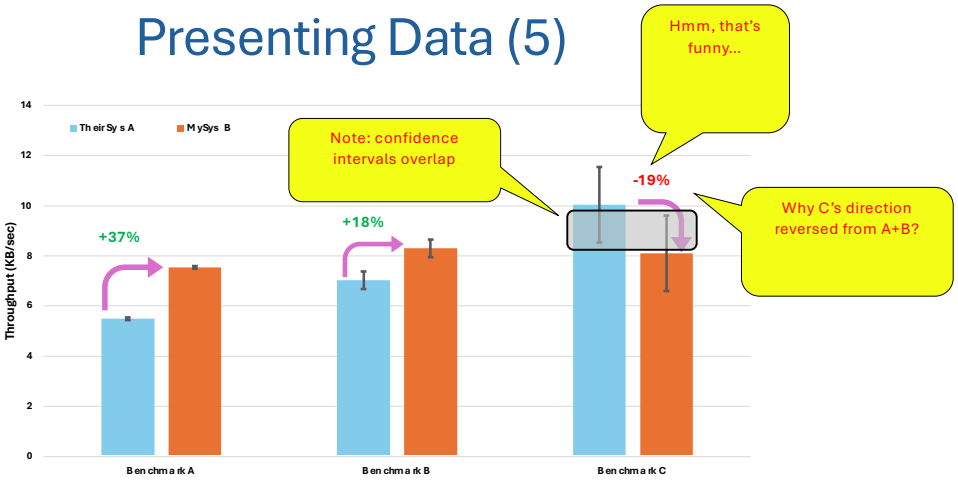
Presenting Data (4)



Stony Brook University

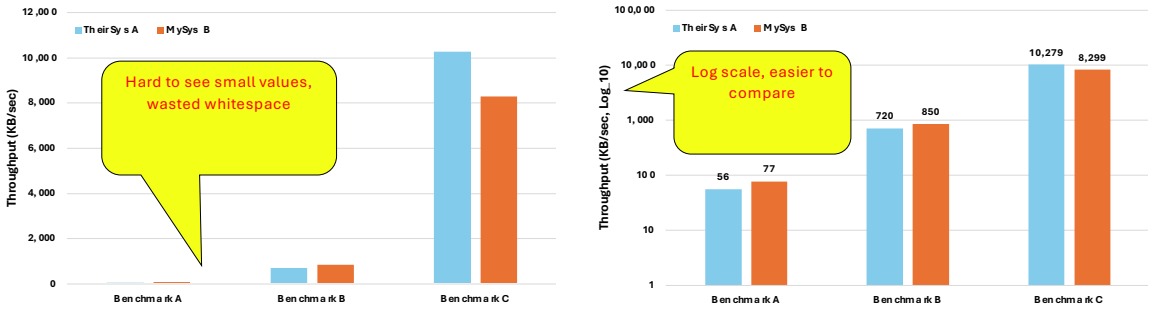

The most exciting phrase to hear in science, the one that heralds new discoveries, is not "Eureka!" but "That's funny..."
-Isaac Asimov

Presenting Data (5)



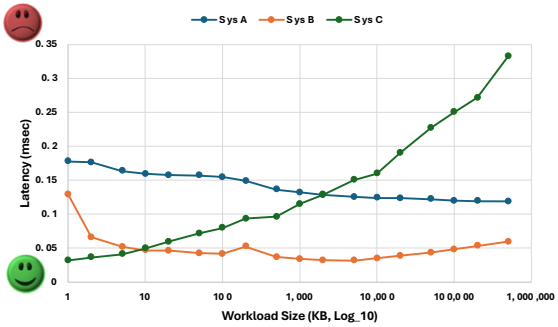
When confidence intervals overlap, it means that the true mean may be anywhere in that range, with some probability. These two "means" may be statistically insignificant! Consider using magnitudes (e.g., 1.37x) instead of percentages.

Presentation Scales



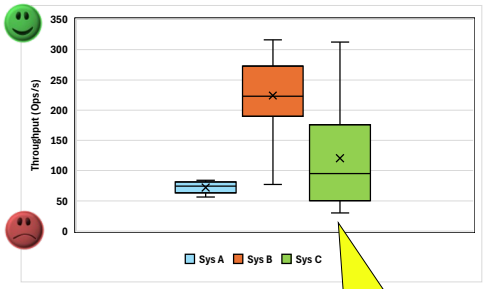
Highlight that you're using log scale in caption.

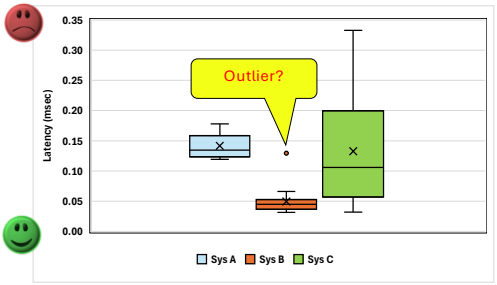
Directionality, Trends, Cross-Overs



Important: depending on which system you use/purchase, and the workload size, you'd get VERY different results!

Who Needs Bar Plots?

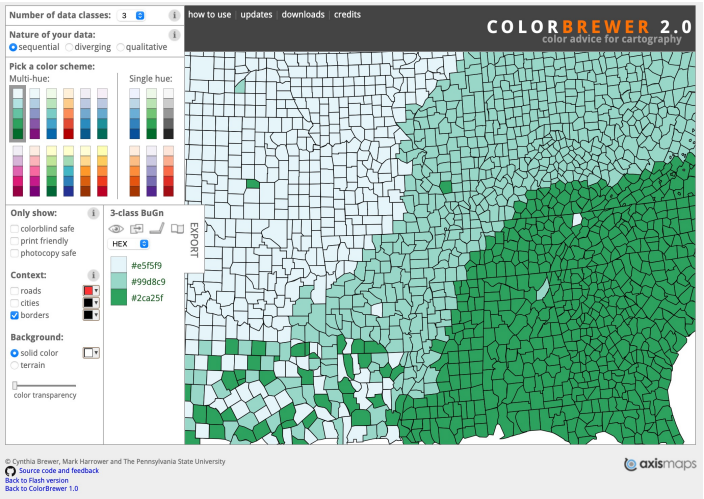




Boxplots: four quartiles, mean (X), median (lines)

Stony Brook University

Choosing Colors



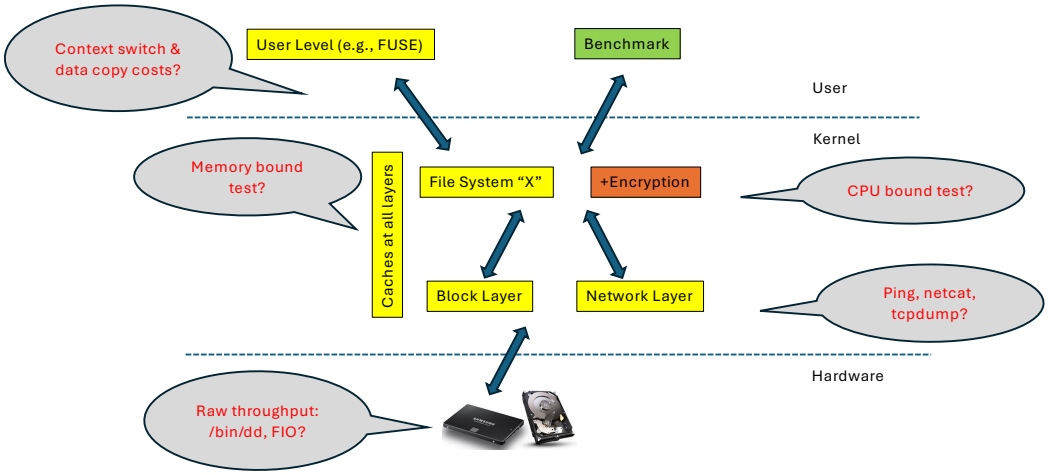
Optimize for

- No. of distinct colors
- Color, B&W, Grayscale
- Print vs. display
- Color blindness
- And more

- https://colorbrewer2.org
- https://projects.susielu.com/viz-palette


Analyze This!

Isolating Component Impact



Stony Brook University

Finer-Grained Impact Isolation

- Suppose I want to find out bottlenecks inside my code?
- How to isolate the impact of specific functions deep inside your code:
  - EBPF, tracepoints, blktrace, tcpdump, etc.
    - Lots of useful info
    - Concerns: overheads, interference, and lost events
  - Do we really need to capture all events?
    - Just calculate averages – not enough information
  - How to capture more information with minimal overhead?

Collecting Histograms Efficiently

Wrap each function with code such as:

1. Start = TSC() // sample the Time Stamp Counter
2. Run function f()
3. End = TSC()
4. Diff = End – Start
5. Bucket=0; while (Diff > 0) {Diff>>=1; Bucket++;} // calc bucket no.
6. Histogram[Bucket]++; // record counts per bucket
7. // Offline: sample Histogram[] vector periodically or at end of experiment

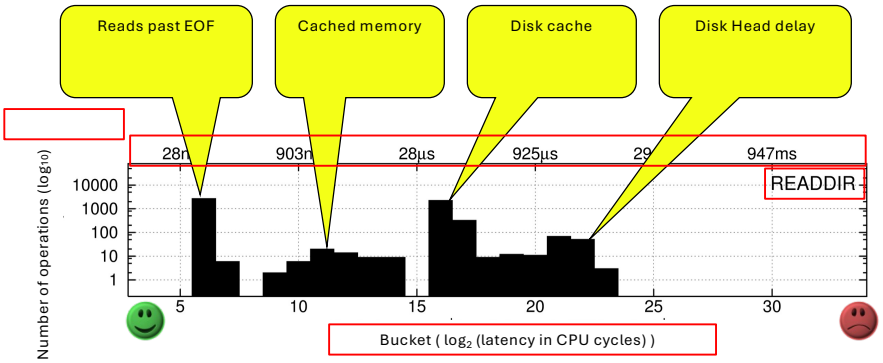
Creates Log_2 buckets

Negligible memory use, CPU overheads (c. 2006) ~4%, mainly TSC()

[OSProf, OSDI 2006]

TSC: Time Stamp Counter, CPU Ticks; Invariant TSC doesn't change by CPUfreq/DVFS (Dynamic Voltage and Frequency Scaling)

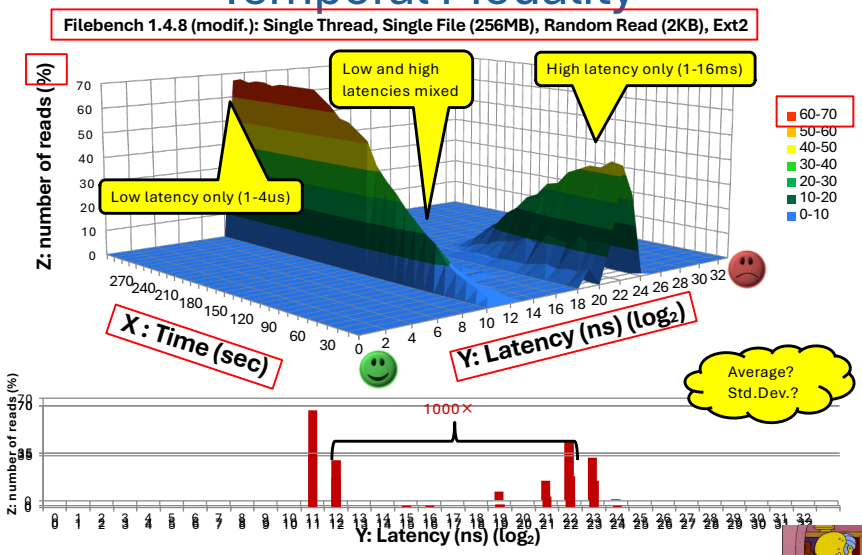
Multi-Modal Behavior



Linux 2.6.11, Ext3 on HDD, `grep -r` on source tree

Dirty secret: depending on your benchmark, you could get very good/bad results!

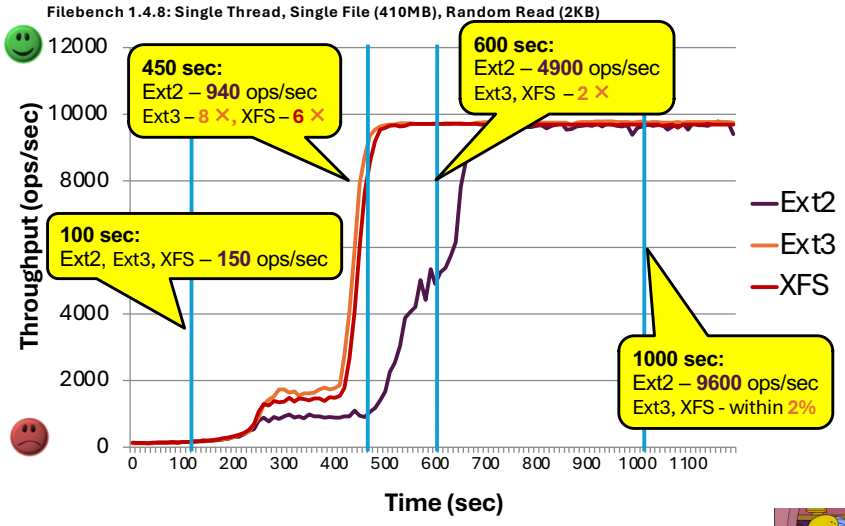
Temporal Modality



[HotOS 2011]

Dirty secret: you can get bad/good results depending how long you run, or results that have very large standard deviations (“unstable”)

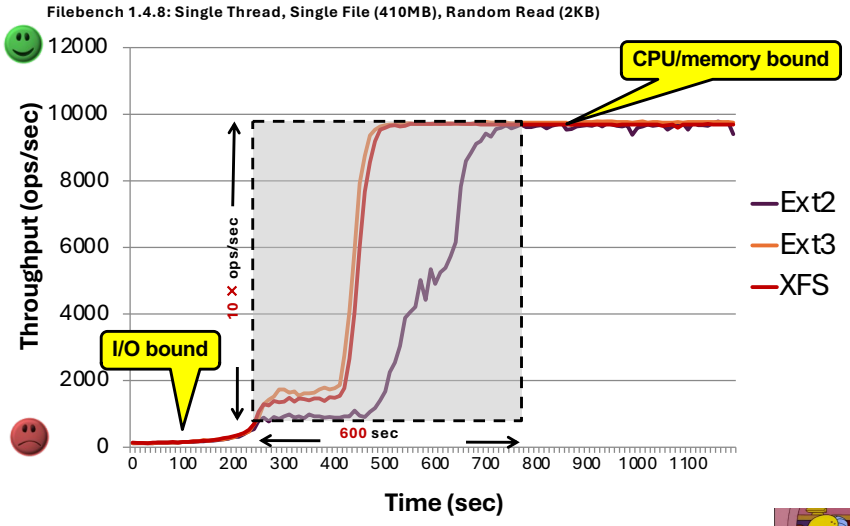
Using Different File Systems



Dirty secret: the file system (or storage system) you run may affect your results.

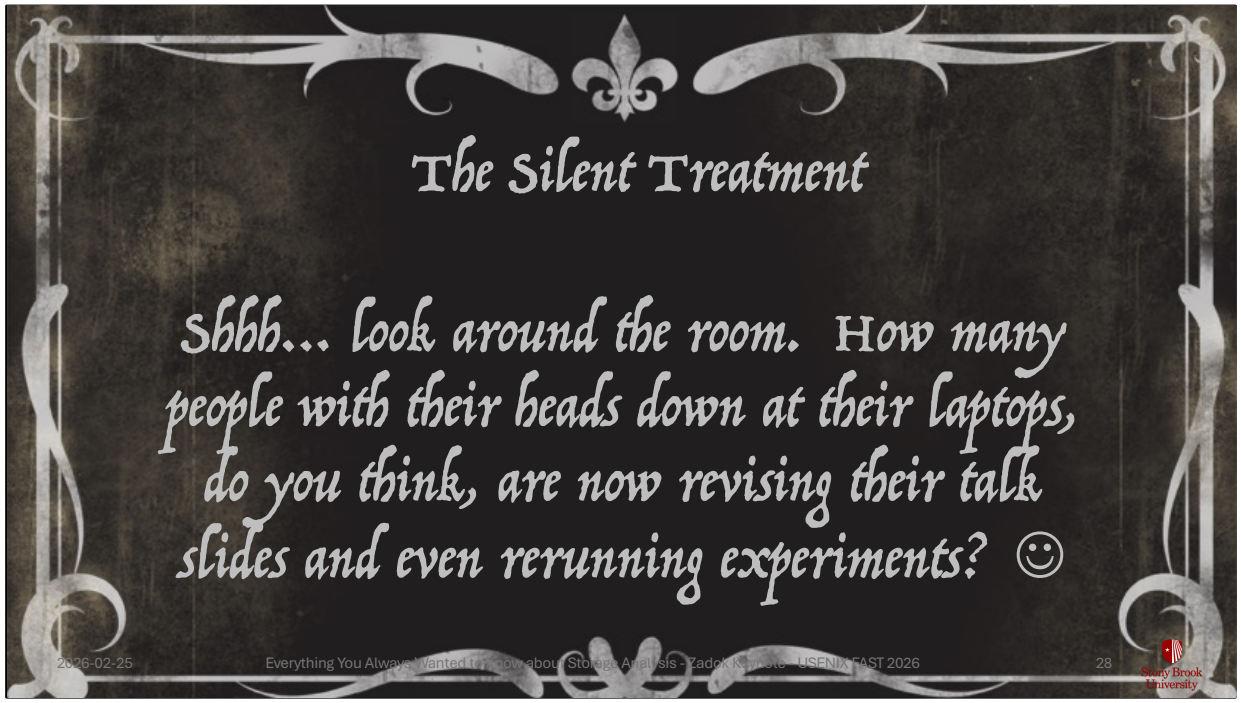
If you add encryption to a f/s, you may show “low overhead” even if your implementation is terrible, b/c you’re I/O, CPU/Memory bound!

Using Different File Systems



Dirty secret: the file system (or storage system) you run may affect your results.

If you add encryption to a f/s, you may show “low overhead” even if your implementation is terrible, b/c you’re I/O, CPU/Memory bound!


The Silent Treatment
Shhh… look around the room. How many people with their heads down at their laptops, do you think, are now revising their talk slides and even rerunning experiments? ☺


The Long Goodbye

Data is Everywhere!

Storage “research” is *intra*-disciplinary:

- Formal methods, runtime verification
  - Finding bugs and performance anomalies, efficient tracing
- Cryptography and security
  - Long term archival security
- Natural Language Processing
  - Analyzing storage RFCs, finding ambiguities, translating to models/code
- Visual Analytics
  - Analyzing massive multi-dimensional data

Dirty little secret: that’s how you get more papers and grants, collaborate!

Quest for More Data

- Data is critical
- Lack of current, large data sets/traces
- SNIA IOTTA has many
- FSL Dedup data set
  - Collected over six years
  - 5TB compressed
  - Mentioned or cited 250+ times in papers
  - Subsets downloaded 36,000+ times

Use Zenodo to create citable DOI for data released

Dirty secret: use Zenodo to cite data.

Courtesy Gemini

Zadok's Law (2026)

"Computer systems,

when scaled,

become a storage problem."

Shameless Plug for Journals

- Two of top-10 most downloaded ACM TOS papers were mine:
  - 9-year study of file system benchmarking [2008]
  - FUSE performance and optimization [2019]
- No upper page limit
- Good for expanded conference versions
- Excellent for survey papers
  - E.g., ACM TOS "Past, Present, and Future of Storage Systems"
    - Special issue papers on DNA, Silica/Glass, Holographic, SSD, SMR, NVM, Lustre, Tape

Remember to look for and cite journal versions

Stony Brook University

Dirty secret: no one will cite past papers! Ideas, storage for AI, AI for storage, KV caches? CXL?

Storage Research is Hard, and…

"Kernel hacking is hard!

Really hard.

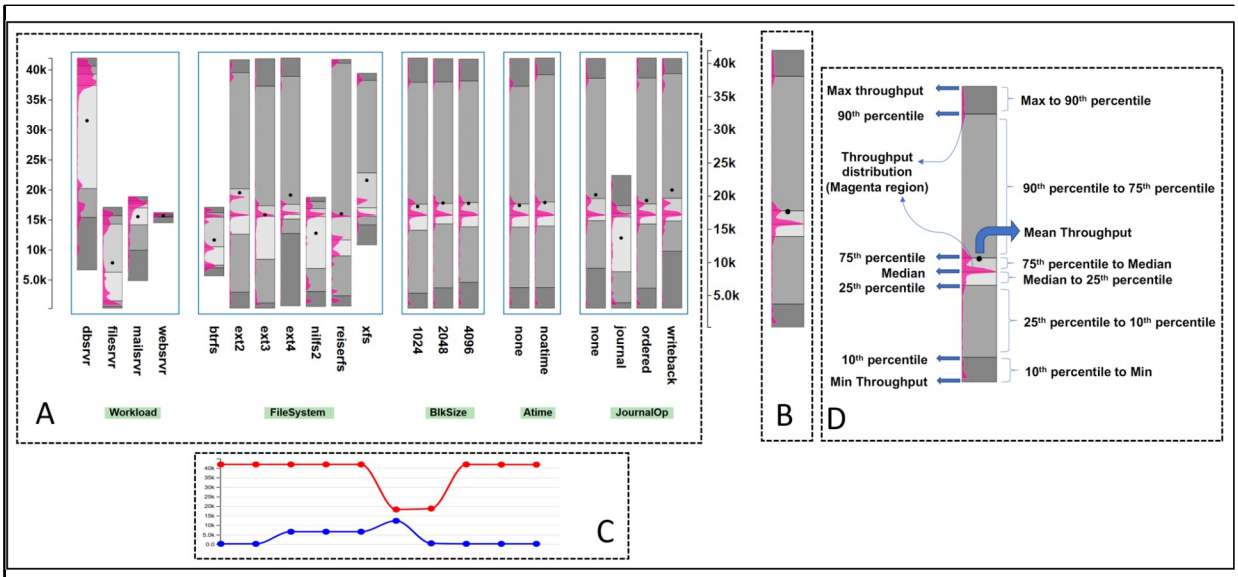
But once it works, man.

It's better than sex.

Everything You Always Wanted to Know about Storage Analysis

(But Were Afraid to Ask ;)

Erez Zadok
Stony Brook University

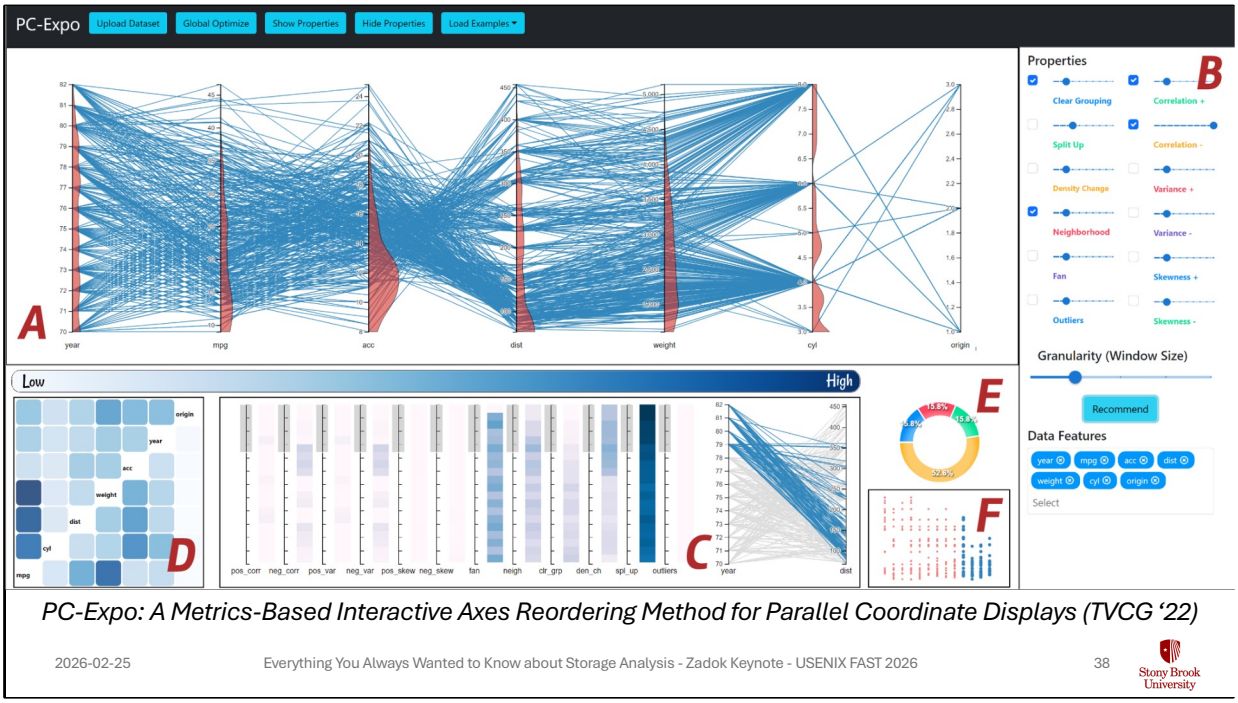


ICE: Interactive Configuration Explorer for High Dimensional Categorical Parameter Spaces (VAST '19)

VAST 2019, HotStorage 2019

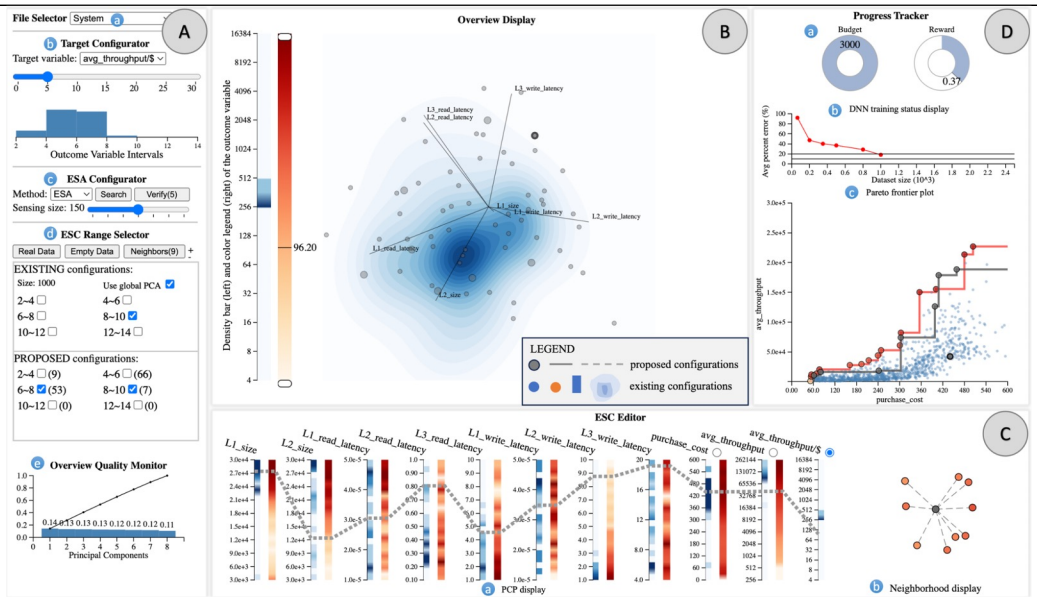
Stony Brook University

Added "Violin plot": pink histogram, vertical, overlayed on special kind of boxplot.



PC-Expo: A Metrics-Based Interactive Axes Reordering Method for Parallel Coordinate Displays (TVCG '22)

Exploring multi-dimensional data more efficiently using parallel coordinates.



Into the Void: Mapping the Unseen Gaps in High Dimensional Data (TVCG '25)

Using ML and Generic Algorithms to find more valuable unexplored regions of the configuration space, where better optimal configurations of your storage system may be.