sodacon
— Global 2021 — July 13-14

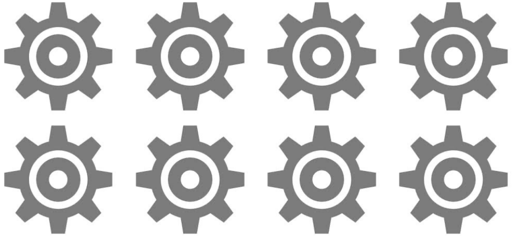
Innovating IT Infrastructure with Computational Storage – Can you believe (or achieve) the hype?

Tong Zhang, Chief Scientist @ ScaleFlux

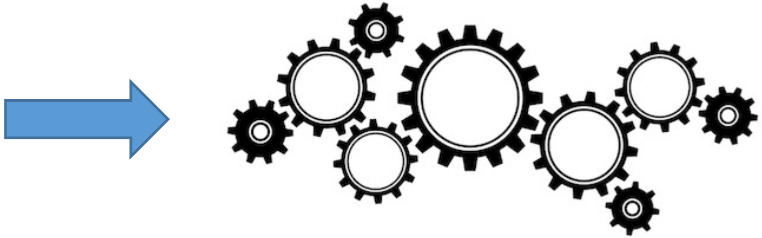
soda foundation

#sodacon2021

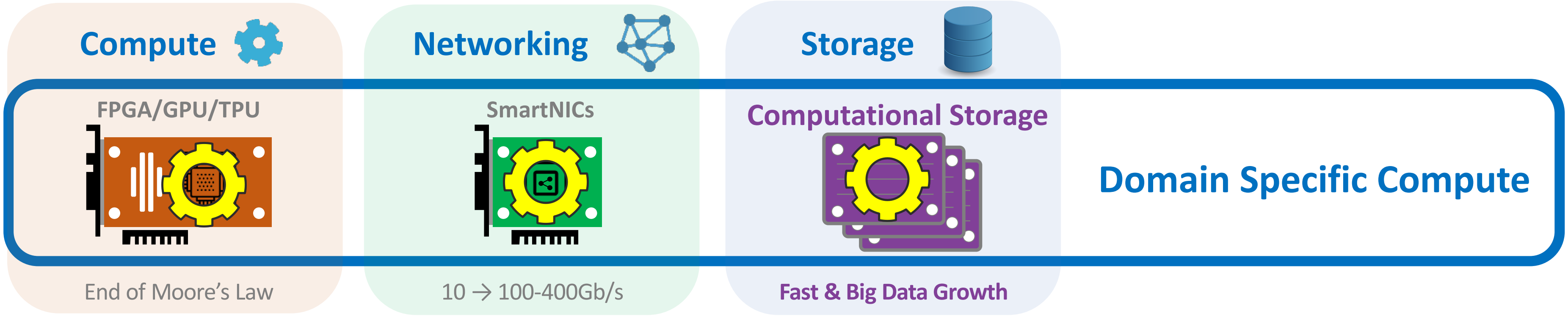
The Rise of Computational Storage



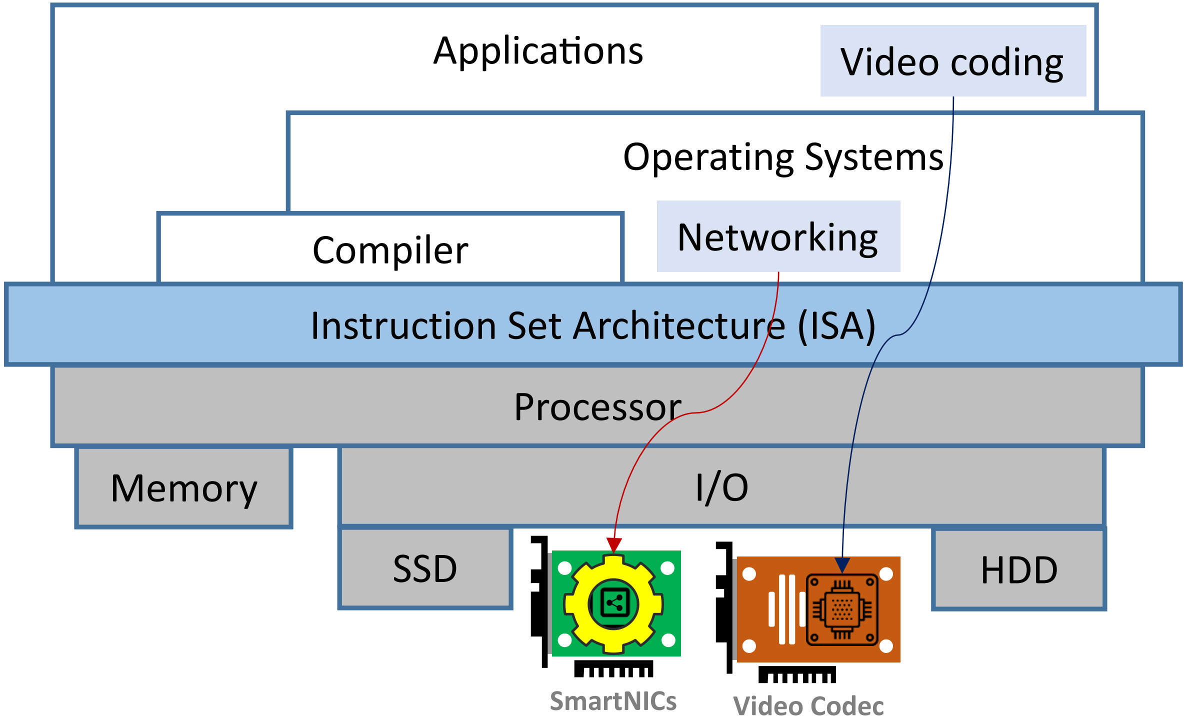
Homogeneous Computing



Heterogenous Computing



Beauty of Abstraction



Break the principle of abstraction

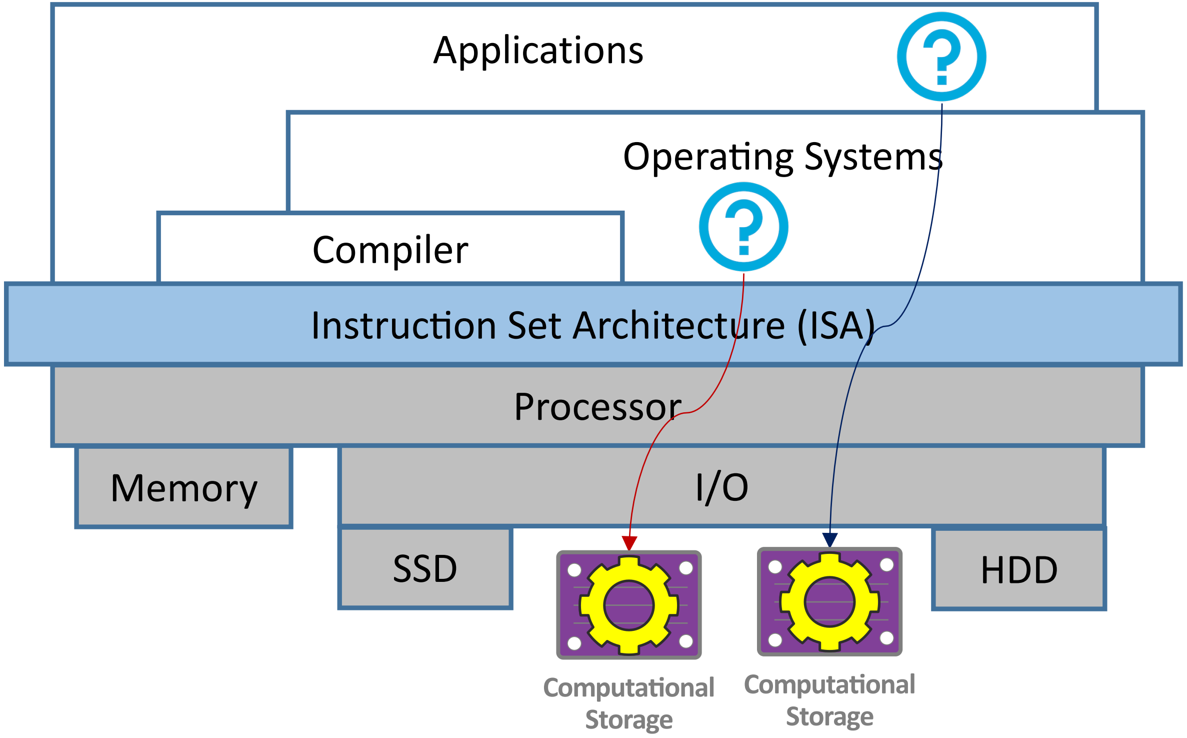
x **Cost**: Modify one (or more) layers

x **Cost**: Enhance cross-layer interface

x **Cost**: Vendor lock-in

✓ **Benefit**: Improve the system performance/efficiency

Beauty of Abstraction



Break the principle of abstraction

x **Cost**: Modify one (or more) layers

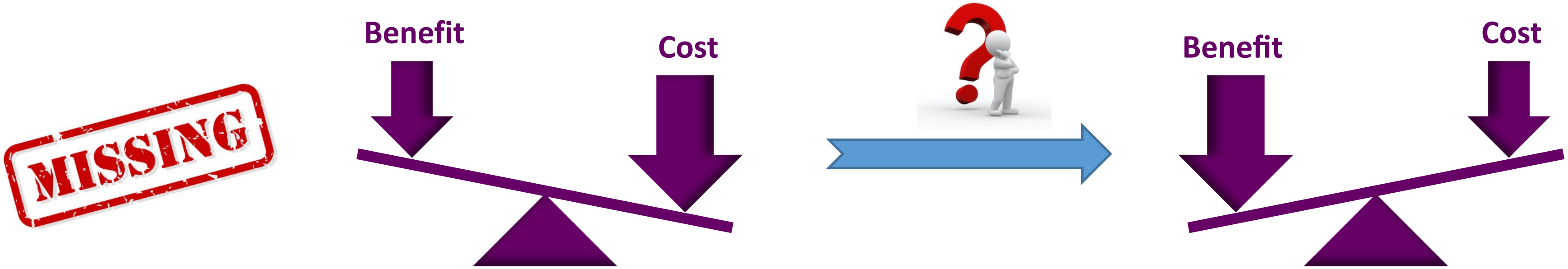
x **Cost**: Enhance cross-layer interface

x **Cost**: Vendor lock-in

✓ **Benefit**: Improve the system performance/efficiency

Computational Storage

- Academia started to explore this simple concept since 20 years ago
  - Coined many terms: “Intelligent RAM”, “Active Disk”, “Intelligent SSD”, …
  - Not surprisingly, very impressive performance gains were demonstrated **on papers** for applications such as linear algebra, multimedia/graph processing, database, …

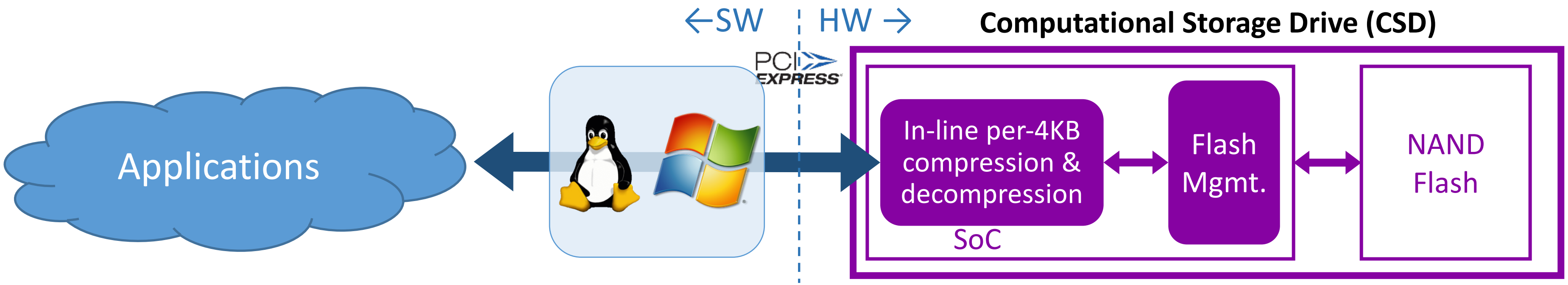


- x **Cost**: Modify one (or more) layers
- x **Cost**: Enhance cross-layer interface
- x **Cost**: Vendor lock-in

- ✓ **Benefit**: Improve the system performance/efficiency

Computational Storage

- A perfect low-hanging fruit: In-storage **transparent** compression

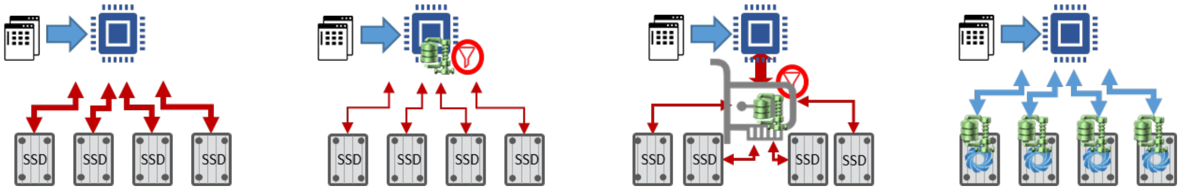


- x ~~**Cost**: Modify one (or more) layers~~
- x ~~**Cost**: Enhance cross layer interface~~
- x ~~**Cost**: Vendor lock in~~

VS

✓ **Benefit**: Improve the system performance/efficiency

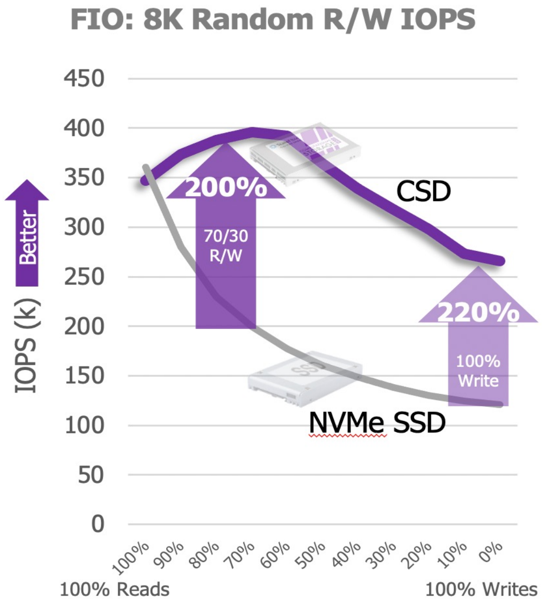
Comparing Compression Options



	No Compression	Host-Based	Offload Card	CSD
No CPU Overhead	✓	✗	✓	✓
Reduced $/User GB	✗	✓	✓	✓
Performance scales with capacity	✓	✗	✗	✓
Transparent App Integration	-	✗	✗	✓
Zero App Latency	✓	✗	✗	✓
No incremental power usage	✓	✗	✗	✓
No incremental physical footprint	✓	✓	✗	✓

Scalable CSD-based compression reduces Cost/GB without choking the CPU

Seamless Deployment



- Zero changes to application source code
- >2x storage cost reduction at zero host CPU usage
- Representative use cases: Relational database (e.g., MySQL, Postgre, Oracle), key-value store (e.g., Aerospike), data analytics (e.g., ClickHouse, Apache Kudu)

MySQL

- >50% storage cost saving
- 50% TPS improvement
- 38% latency reduction

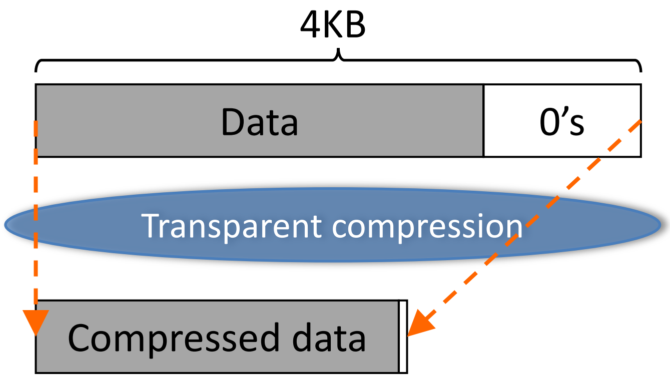
PostgreSQL

- >50% storage cost saving
- 35% TPS improvement
- 20% latency reduction

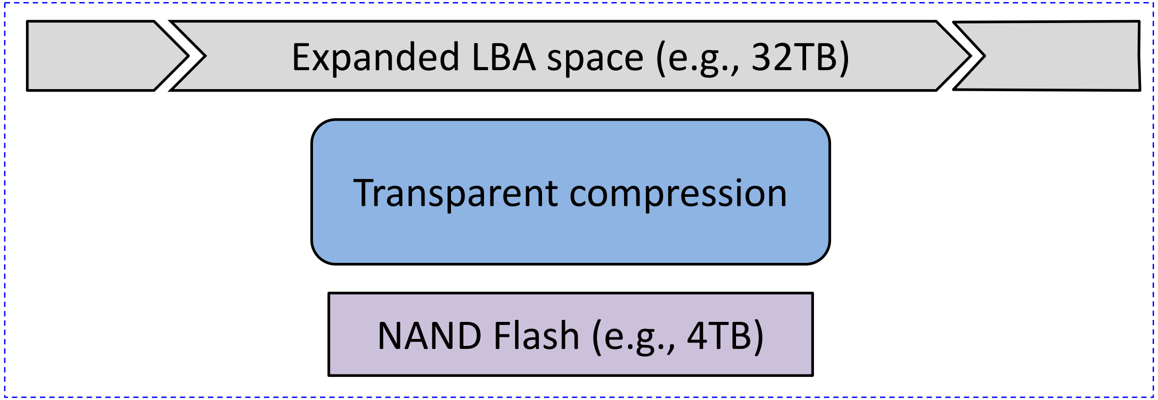
AEROSPIKE

- >50% storage cost saving
- 50% TPS improvement
- 98% tail latency reduction

System-level Innovations Enabled by Transparent Compression



Unnecessary to completely fill each 4KB sector with user data

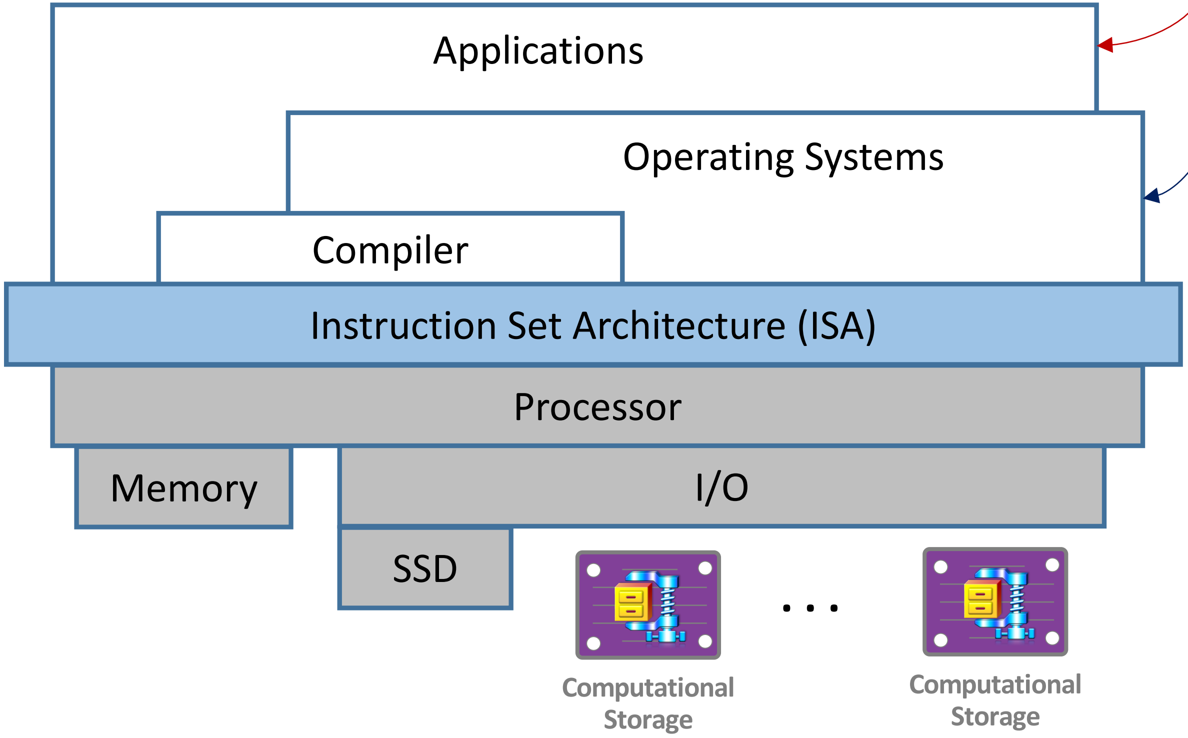


Unnecessary to use all the LBAs

OS/Applications can employ ***sparse*** data structure without sacrificing storage cost

Sparse data structure ➔ cost/performance benefits

Beauty of Abstraction



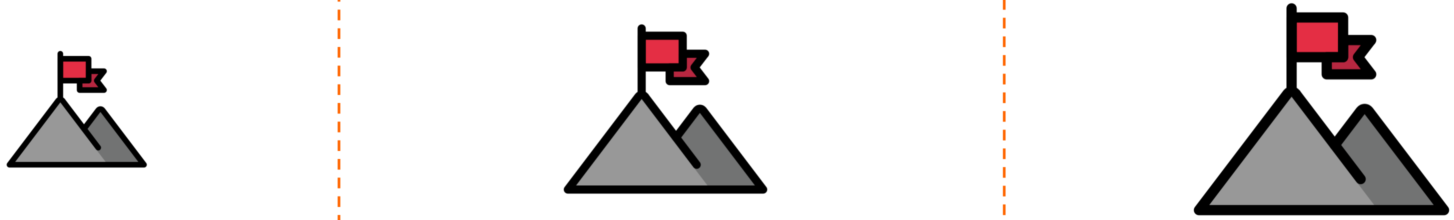
Data management with sparse data structure

- x **Cost**: Modify one (or more) layers
- ~~x **Cost**: Enhance cross-layer interface~~
- ~~x **Cost**: Vendor lock-in~~
- ✓ **Benefit**: Improve the system performance/efficiency

System-level Innovations Enabled by Transparent Compression

Stage 1	Stage 2	Stage 3
Zero App modification	Modest App modification	Ground-up App development

Evolutionary Path

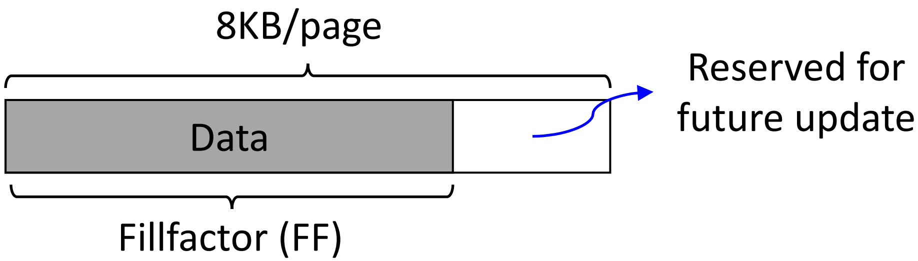


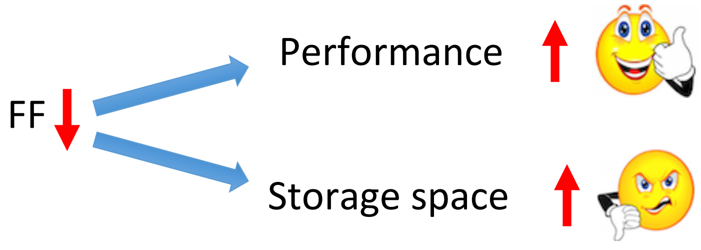
System design/optimization via sparse data structure

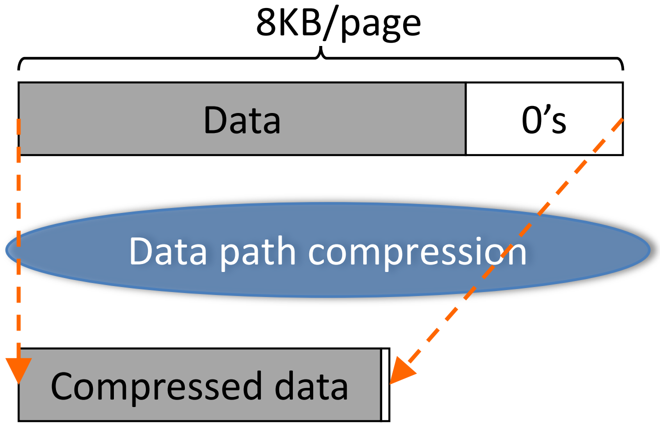
CSD CSD CSD CSD CSD . . . CSD

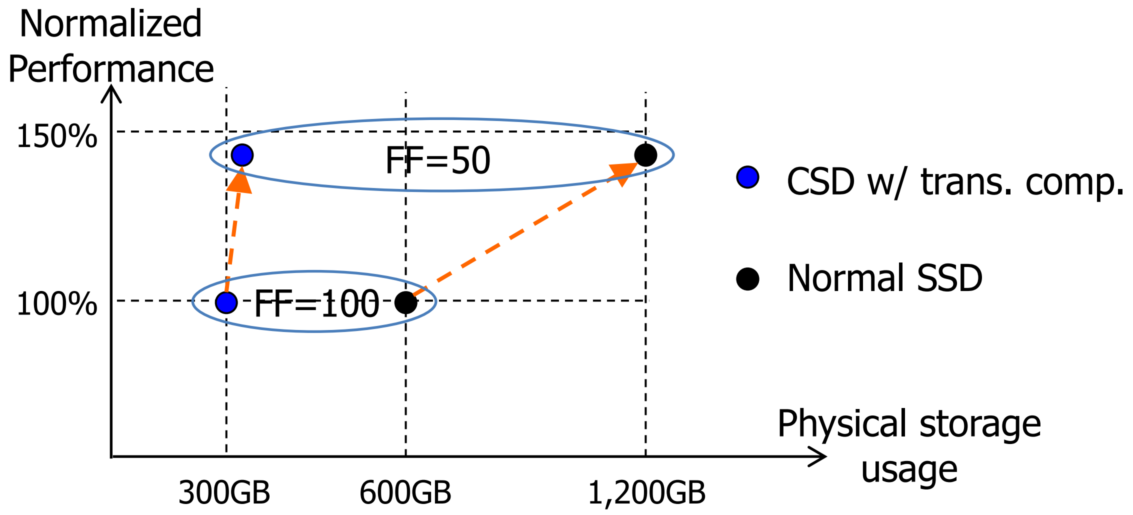
Sparse data structure ➔ cost/performance benefits

Stage-1 Example: PostgreSQL

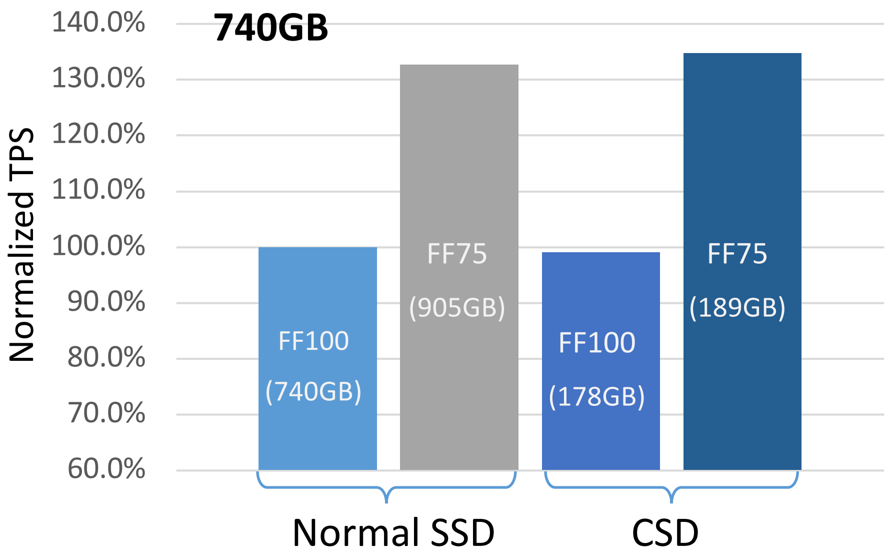


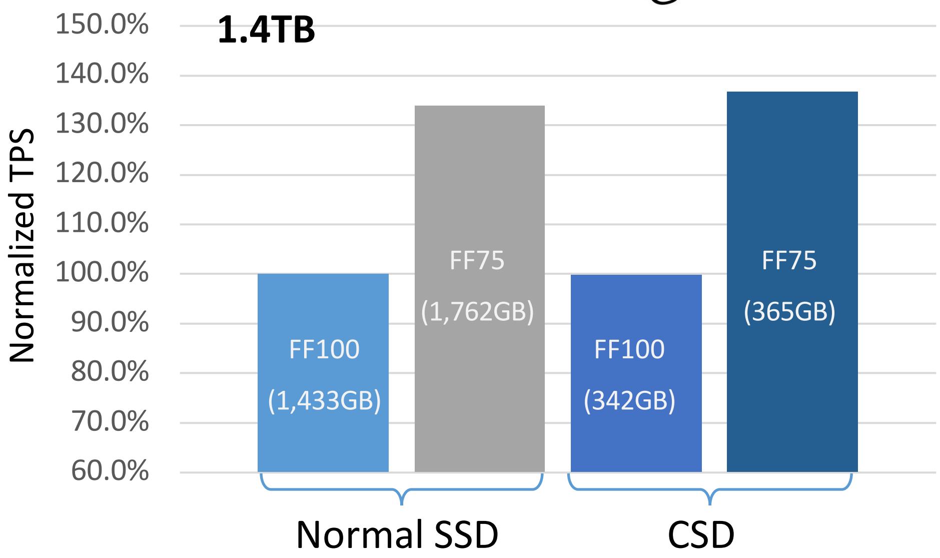






Stage-1 Example: Experiments (Sysbench-TPCC)





Fillfactor	Logical size (GB)	Drive	Physical size (GB)	Comp Ratio
100	740	Normal SSD	740	1.00
		CSD	**178**	4.12
75	905	Normal SSD	905	1.00
		CSD	**189**	4.75

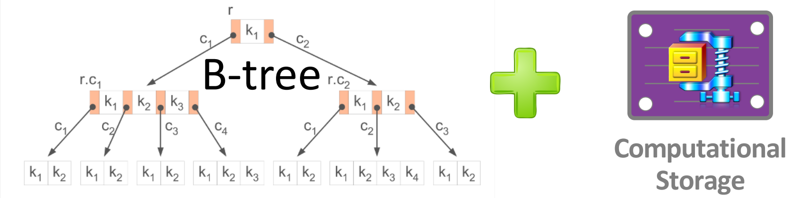
Fillfactor	Logical size (GB)	Drive	Physical size (GB)	Comp Ratio
100	1,433	Normal SSD	1,433	1.00
		CSD	**342**	4.19
75	1,762	Normal SSD	1,762	1.00
		CSD	**365**	4.82

Stage-2 Example: Reduce B-tree Write Amplification

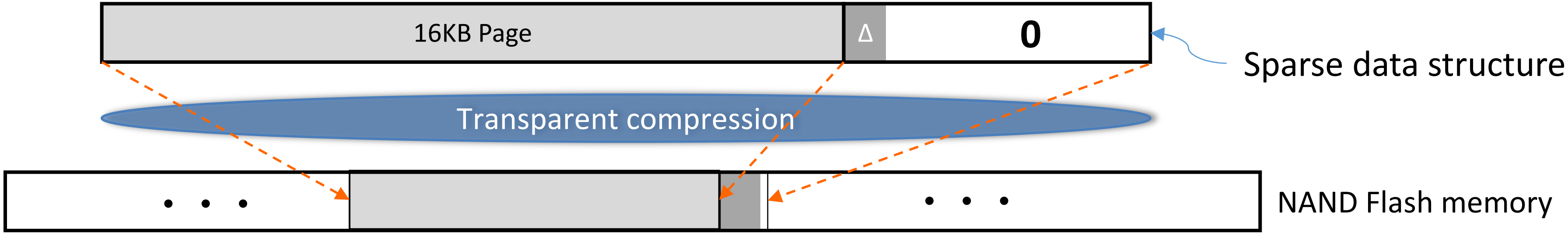
- Log-structured merge tree (LSM-tree)
  - Significant recent interest: RocksDB, Cassandra, HBase, ScyllaDB, …

Advantages of LSM-tree

- ~~Less storage space usage~~
- ~~Much smaller write amplification~~

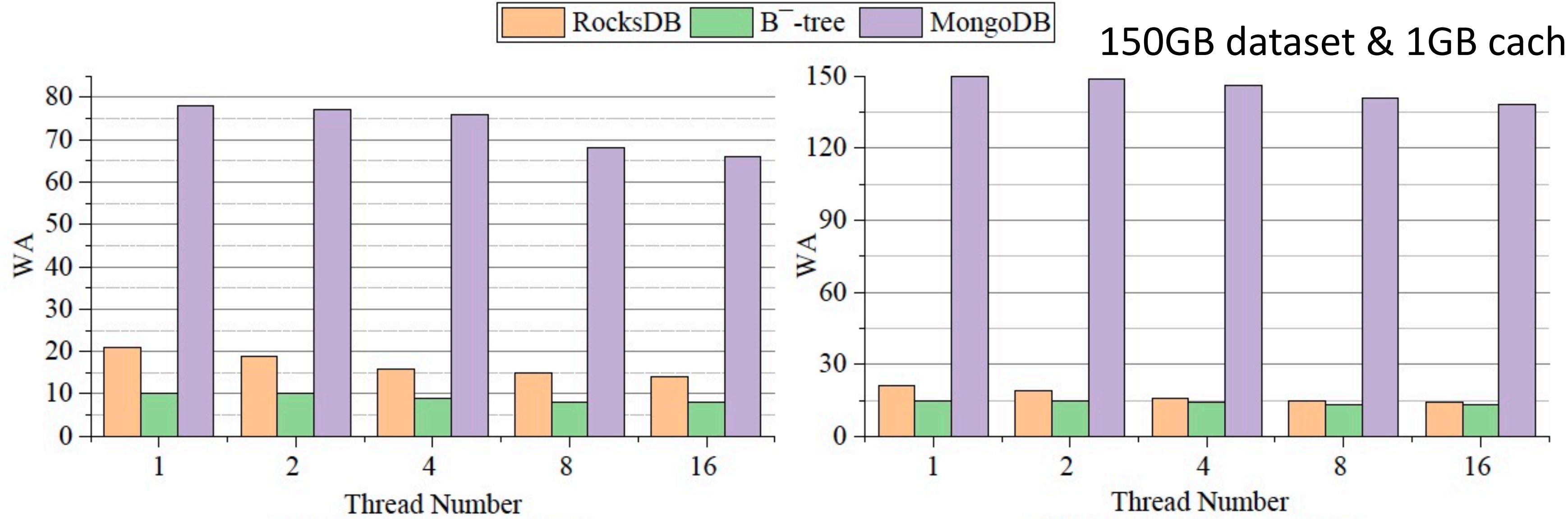


- Localized page modification logging
  - Very simple idea: Log page modification instead of re-writing the entire page every time



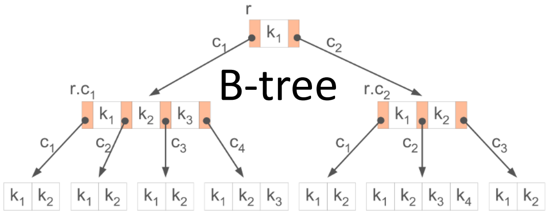
Stage-2 Example: Reduce B-tree Write Amplification

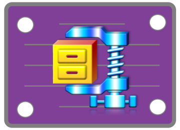
150GB dataset & 1GB cache



(a) 128B record, 8KB page

(b) 128B record, 16KB page



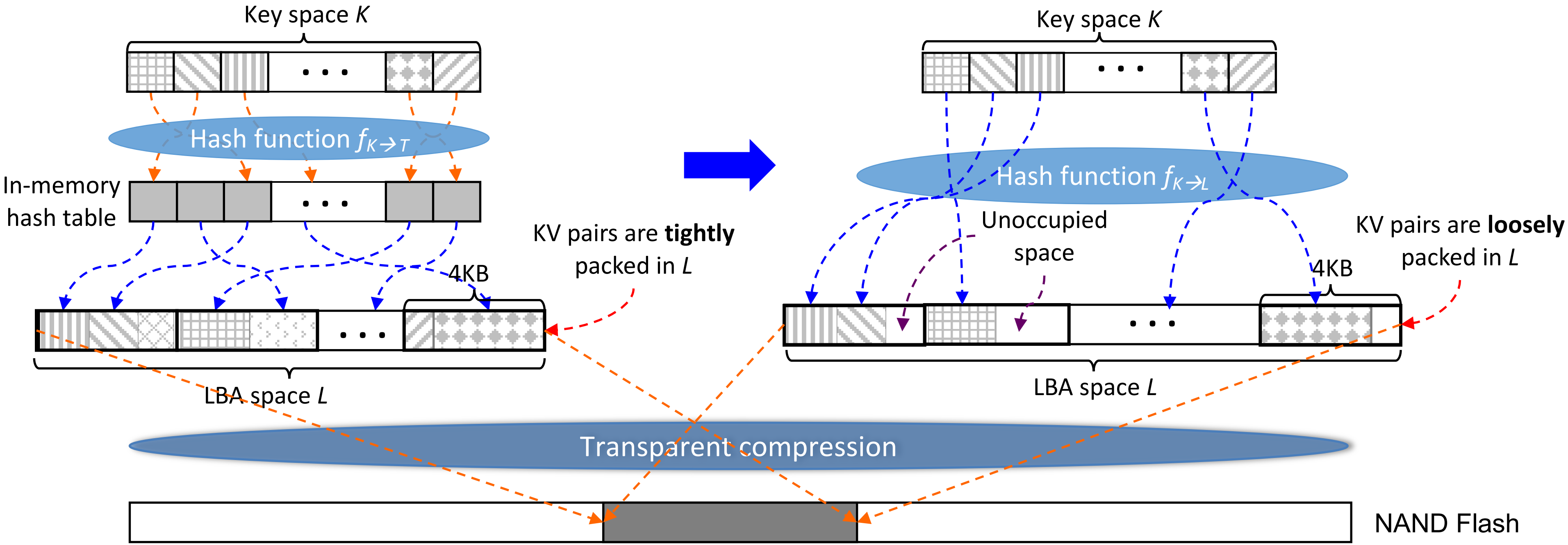


Computational Storage

Make LSM-tree (e.g., RocksDB) **much less** appealing

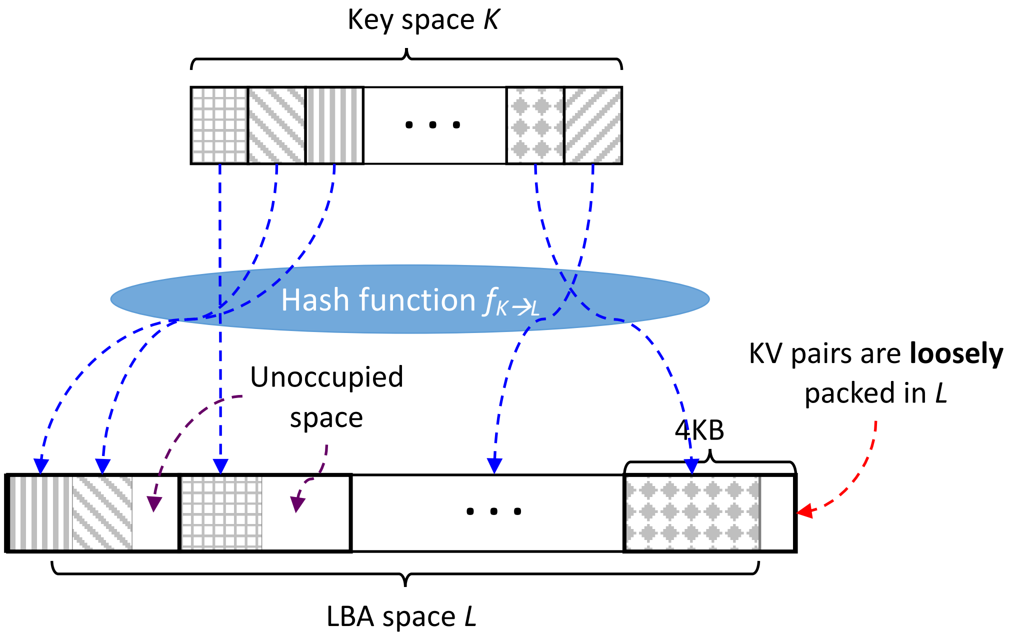
Stage-3 Example: Table-less Hash-based KV Store

- Very simple idea
  - Hash *key space* directly onto *logical storage space* ➔ eliminate the in-memory hash table
  - Transparent compression eliminates the "unoccupied space" from physical storage space



Stage-3 Example: Table-less Hash-based KV Store

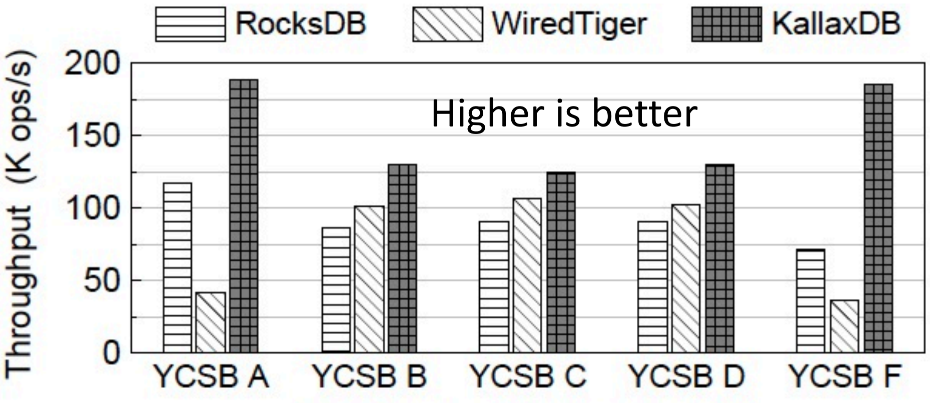
- ❑ Eliminate in-memory hash table
  - ✓ Very small memory footprint
  - ✓ High operational parallelism
  - ✓ Short data access data path
  - ✓ Very simple code base



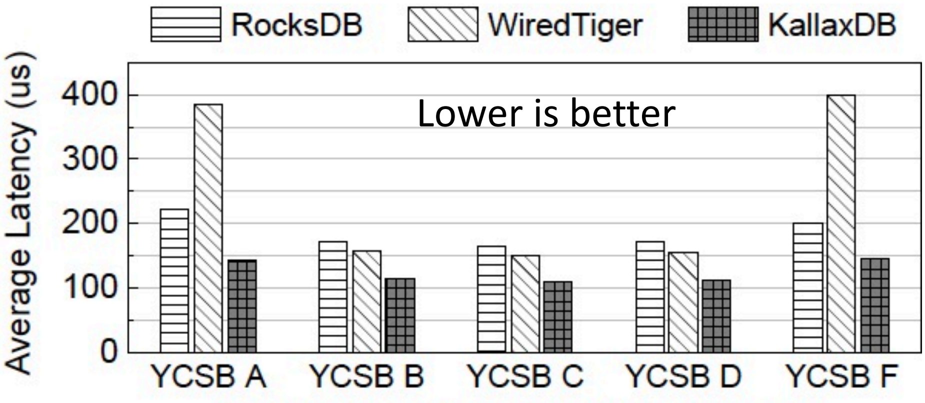
- ❑ Under-utilize logical storage space
  - ✓ Obviate frequent background operations (e.g., GC and compaction)

➡ High performance, low memory cost, and low CPU usage

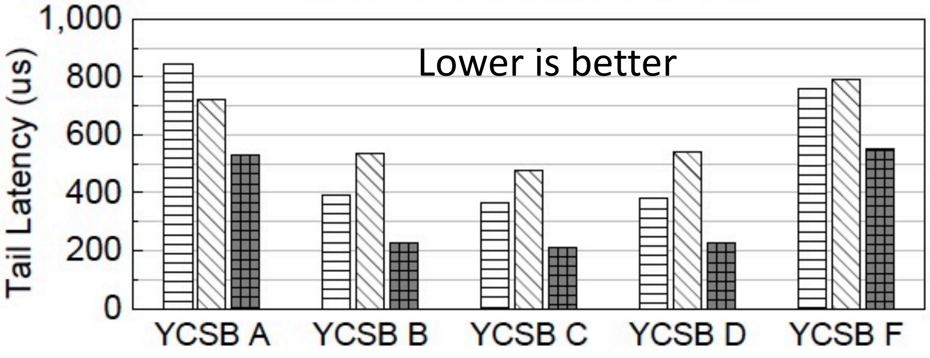
Stage-3 Example: Results (400-byte KV, 400GB dataset)



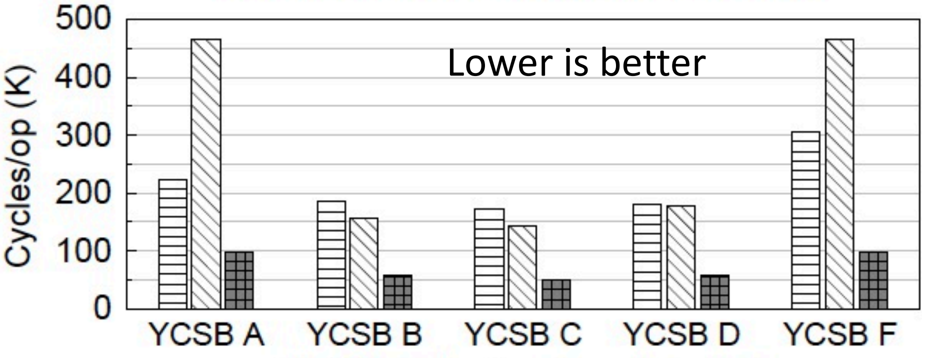
(a) Throughput (higher is better)



(b) Average Read Latency (lower is better)



(c) 99-percentile Read Latency (lower is better)



(d) CPU Cycles/op (lower is better)

Conclusion

- Yes, computational storage indeed has a real potential to innovate IT infrastructure
- The best starting point: Computational storage drive with transparent compression

Stage 1	Stage 2	Stage 3
Zero App modification	Modest App modification	Ground-up App development

Exciting IT Infrastructure Innovation Potential Enabled by Transparent Compression

CASE STUDY	CASE STUDY	CASE STUDY
PostgreSQL	Modify B-tree to reduce write amplification	A table-less hash-based key-value store

- Beyond performance gain, two additional HUGE benefits to end customers:
  - ✓ Avoid modifying interface across applications↔OS/filesystem↔driver ↔HW
  - ✓ Avoid computational storage drive vendor lock-in