High Performance & Low Latency in Solid-State Drives Through Redundancy

Dimitris Skourtis, Scott Brandt, Carlos Maltzahn
University of California, Santa Cruz

Baskin Engineering
UC SANTA CRUZ

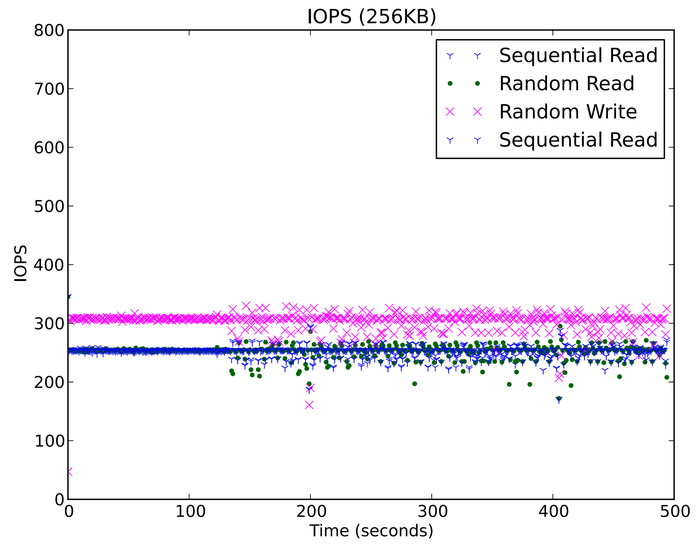
Introduction

- Many applications require
  - Low latency
  - High throughput
  - Consistent performance

- SSDs have fast random access but
  - Performance is workload-dependent
  - Read/write workloads increase latency
    - A read can take more than 100ms
  - Giving priority to reads is not enough

We present a design that physically separates reads from writes and provides high performance, low latency and consistent read performance as well as fault-tolerance.

For certain workloads, SSD performance can be consistent

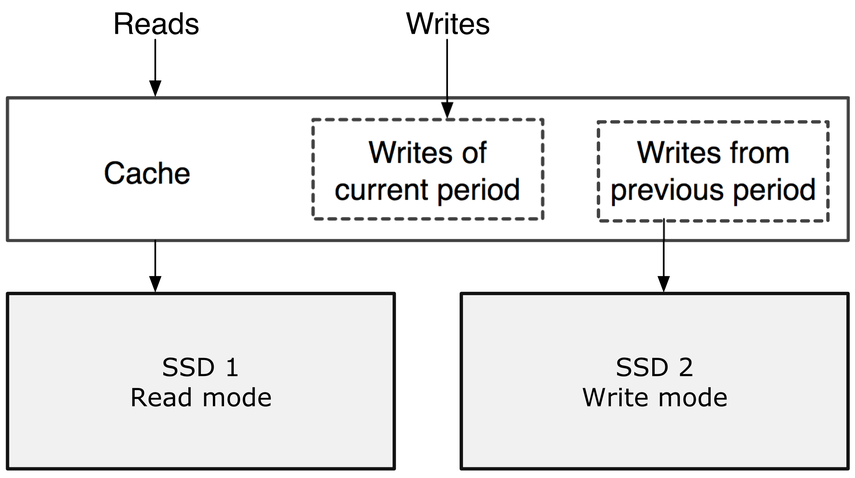


E.g., when random writes happen over just 100MB the write performance remains high and consistent, while the effect of writes on reads is relatively small.

Generally, SSD performance is inconsistent

E.g., when writing over a large range (such as 50% of the drive), the garbage collector cannot keep up, leading to high read latencies and inconsistent performance.

Physically separating reads from writes provides read performance consistency & low latency

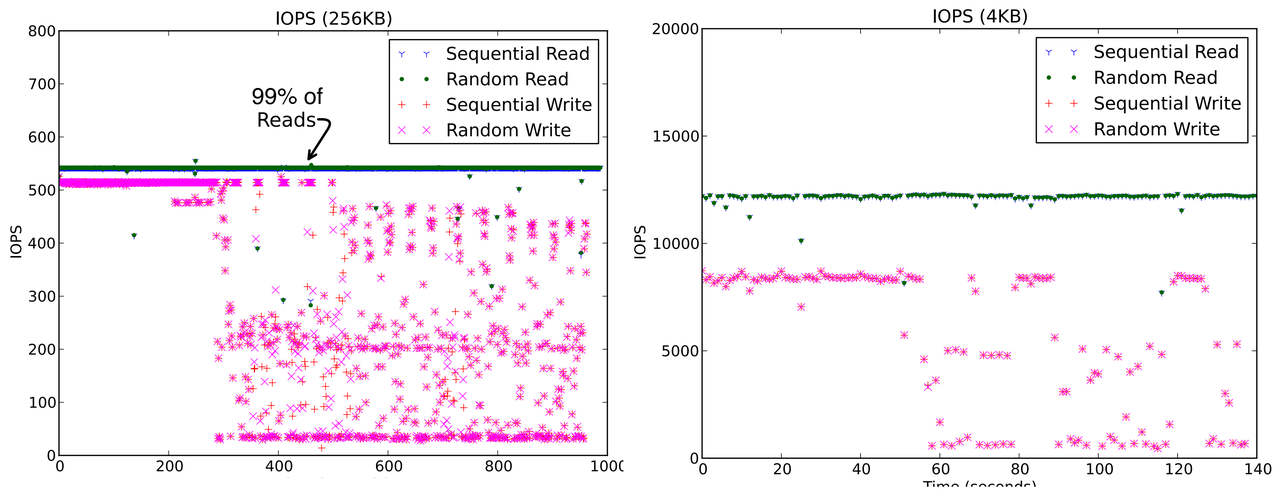


At any given time, each of the two drives is either performing reads or writes. While the first drive is reading, the other drive performs the writes of the previous period - before the drives switched roles.

	Reads	Writes*
Throughput	1x dedicated drive	1/2 dedicated drive
Latency	Minimal	Minimal** (or as before)
Consistency	Nearly perfect	Perfect** (or as before)

*as perceived by the user **if not overloaded

Brief Evaluation



By physically separating reads from writes we achieve high performance, consistency, and minimal latency for both large and small read requests.

Notes

- Reads return the latest data, possibly through cache hits.
- If at any point in time one of the two drives fails, the union of the cache with the other drive contains the latest version of all the data.
- Separating reads from writes allows one to optimize them separately.

Conclusions

Advantages

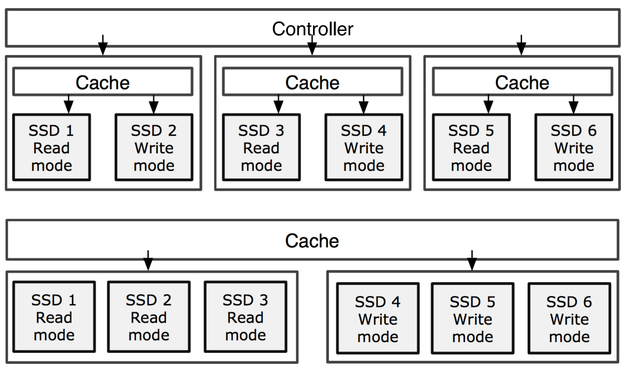
- Minimal read latency
- Consistent read performance
- Fault-tolerance
- Easier to provide high performance guarantees and QoS

Neutral

- Needs a large enough cache
  - Can result in cache hits
- Requires twice the capacity
  - Provides fault-tolerance

Ongoing work

- Evaluation under running applications (not traces or synthetic workloads)
  - E.g., mix of databases & VMs
- Complete integration with QoS
- RAID-10 and RAID-01 setups



- Kernel implementation
- Turn it into a new SSD