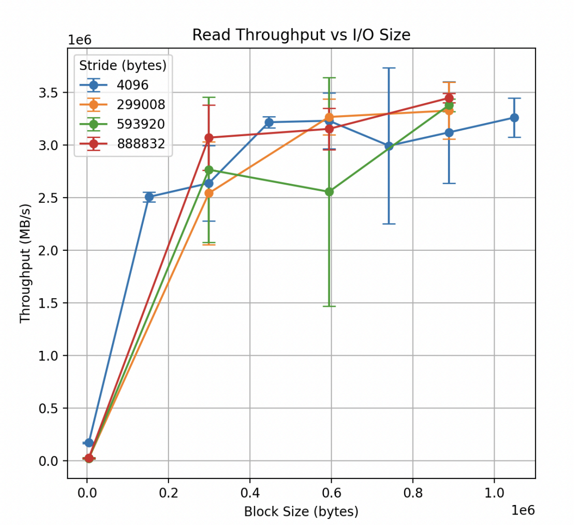
Disk Performance Analysis (MacBook Air and Lenovo ThinkPad)

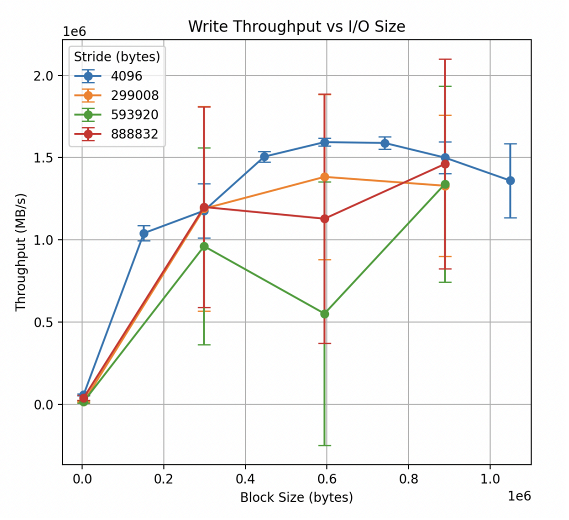
MacBook Air

Read Throughput vs. I/O Size:



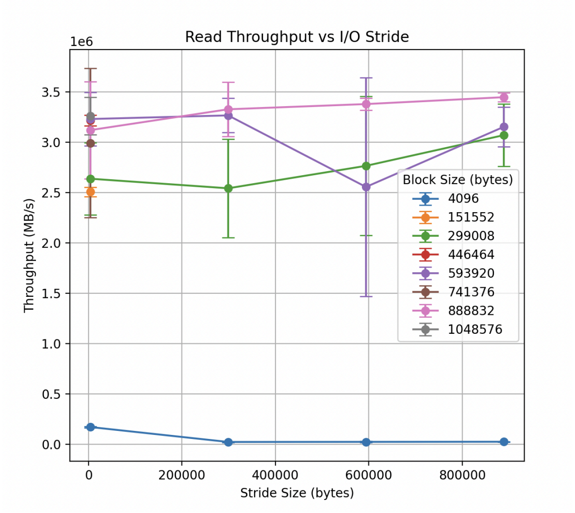
- Read throughput increases significantly with larger block sizes, peaking between 3.3-3.7 GB/s for both deterministic and random reads.
- Small block sizes (e.g., 4096 bytes) exhibit the lowest throughput (140-180 MB/s), likely due to high I/O overhead.
- Beyond 0.7(1e6) bytes, the performance plateaus, suggesting it to be an optimal block size range for this system.

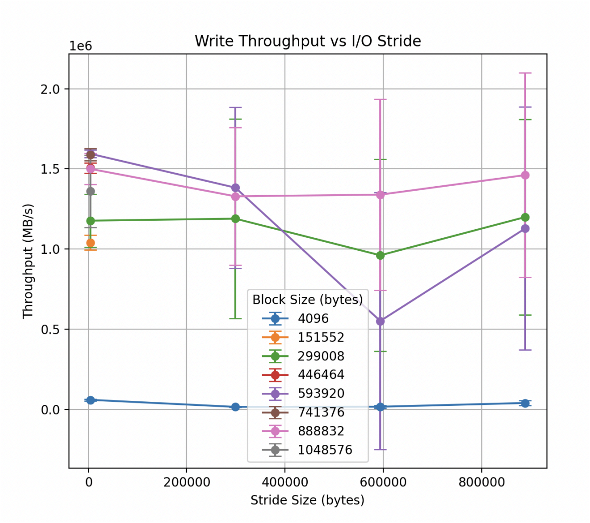
Write Throughput vs. I/O Size:



- Write throughput follows a similar trend as reads, peaking between 1.4-1.6 GB/s for deterministic writes and stabilizing around 1.05 GB/s.
- Small block sizes (4096 bytes) yield poor write performance (60-70 MB/s), reflecting the inefficiency of small transfers.

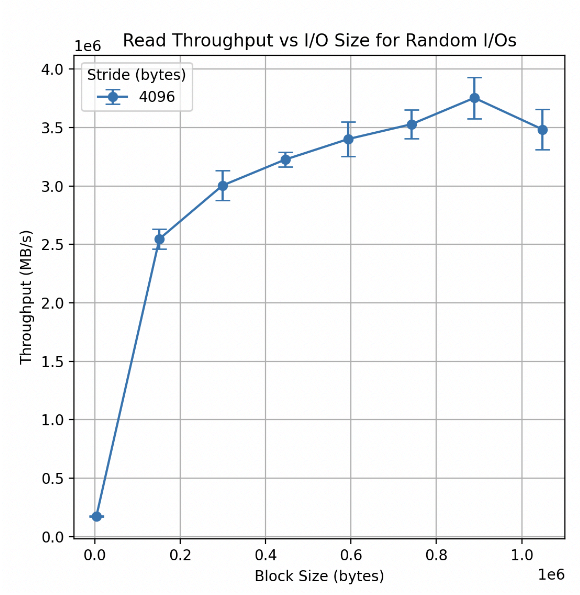
I/O Stride Experiment (Read and Write):

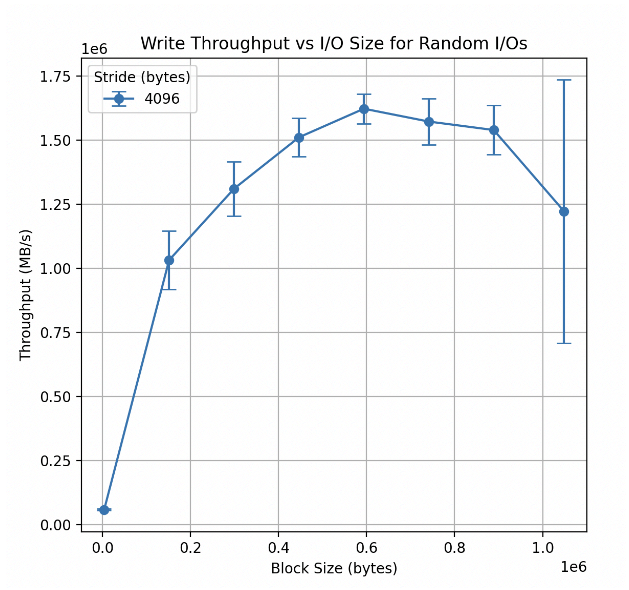




- Small stride sizes (e.g., 4096 bytes) show consistently higher throughput for reads and writes across block sizes.
- Larger strides introduce variability, with some configurations maintaining high throughput and others showing drops. This aligns with SSD behaviors, as they aren't constrained by any physical alignment requirements. This eliminates sector alignment overhead and explains why high strides don't significantly degrade throughput (in most cases).

Random I/O Performance:

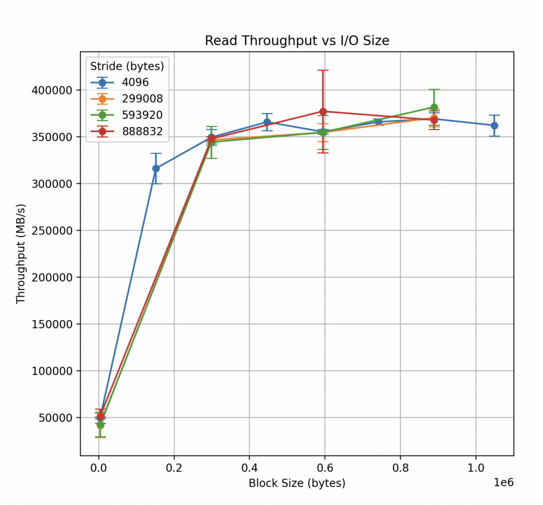




- Random reads achieve slightly lower peak throughput than deterministic reads, demonstrating the additional overhead of random access patterns.
- Random writes are comparable to deterministic writes, indicating robust storage controller optimization for write operations.

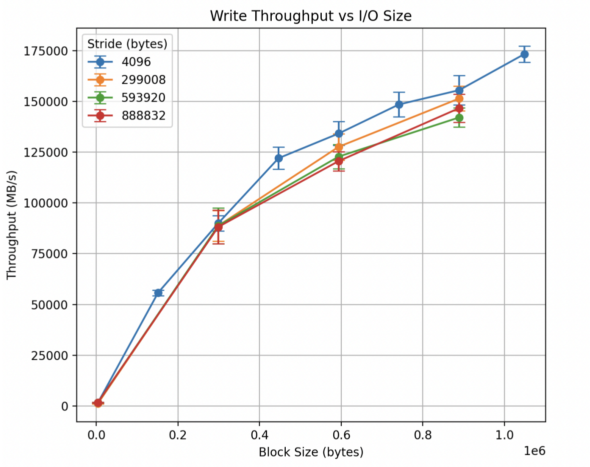
<u>Lenovo ThinkPad</u>

Read Throughput vs. I/O Size:



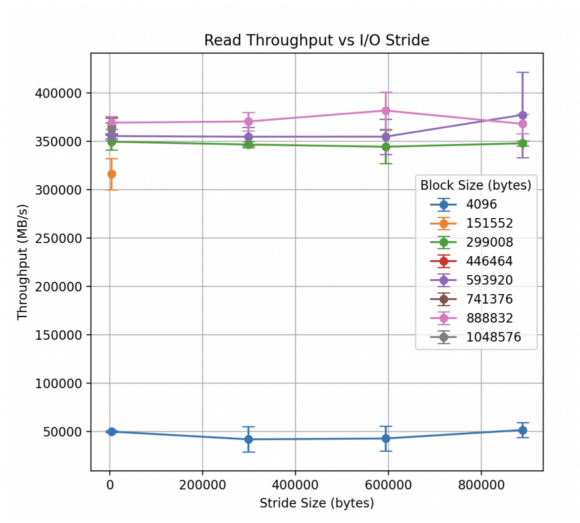
- Read throughput increases sharply with block size, peaking around 350,000-370,000 MB/s.
- Smaller block sizes (4096 bytes) achieve very little throughput, whereas performance plateaus beyond 741,376 bytes, displaying diminishing returns.

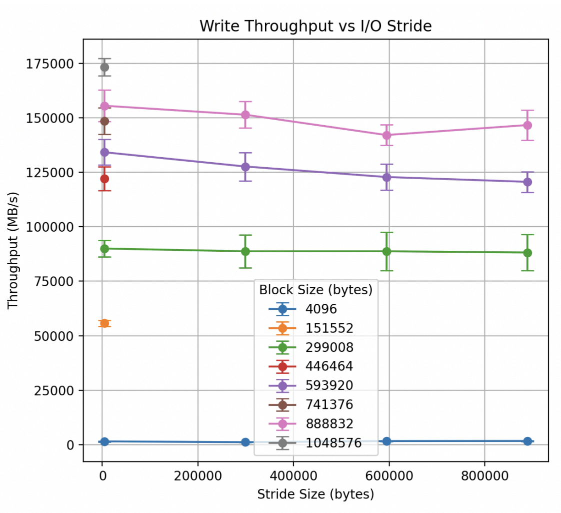
Write Throughput vs. I/O Size:



- Write throughput scales similarly, peaking over 1 GB/s block size with around 170,000 MB/s throughput. This means smaller block sizes perform poorly (1200-2200 MB/s) while performance improves as block size increases.

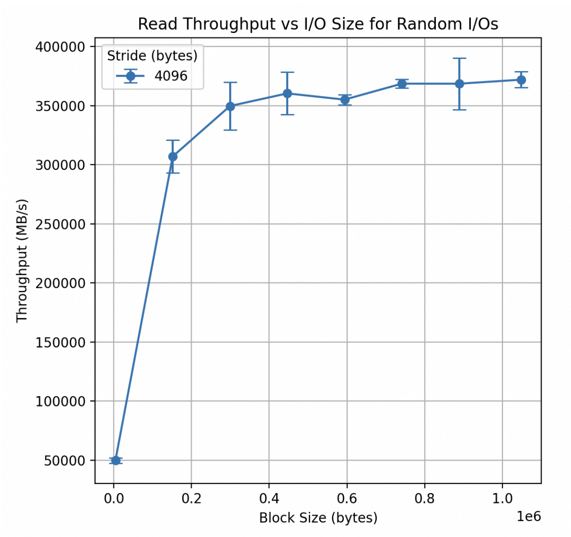
I/O Stride Experiment (Read and Write):

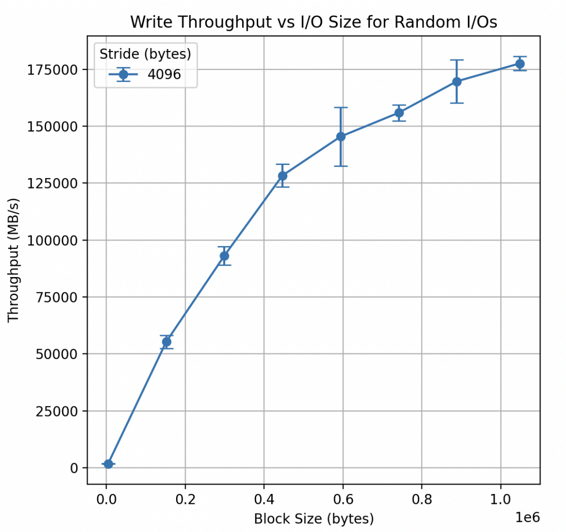




- Small strides in read throughput (e.g., 4096 bytes) result in consistent performance across block sizes.
- The minimal degradation seen along stride sizes is consistent with SSD architecture. SSDs handle high strides well due to the lack of mechanical components, allowing these optimizations to maintain throughput.

Random I/O Performance:





- Random reads and writes exhibit similar scaling trends to deterministic operations, but with slightly reduced peak throughput.
- Random writes improve dramatically with larger block sizes, peaking at 160,000-180,000 MB/s.

Key Insights from Experiments

From the graphs and datasets, it is evident that file I/O performance varies significantly based on block size and operation type (read vs. write), with some limited variation seen with stride size. Larger block sizes typically improve throughput by reducing overhead from frequent I/O operations. This trend is more pronounced on the MacBook Air, where throughput increases sharply with larger block sizes, particularly for deterministic reads and writes. While the Lenovo ThinkPad also benefits from larger block sizes, its throughput improvement plateaus earlier, indicating either hardware or file system constraints.

The effects of access patterns were not felt as heavily in this experiment as would be expected if tests were run using an HDD. Because both systems were equipped with SSDs, digital writes are able to avoid the hit to performance that we suspect would have occurred in an HDD due to the spinning and aligning of a physical disk. Thus, in all, I/O performance was largely independent of the stride size and sequential/random access pattern, although there was still a small degree of variability in the stride size performance and a slight hit to performance in the random I/O case. This possibly reflects some limited optimization for random I/O but still nothing substantial.

Read and write performances lightly differ between devices. The MacBook Air does show robust support for both read and write operations, suggesting well-optimized SSD technology. However, reads consistently demonstrate higher throughput than writes, with read speeds approximately 2-2.3 times faster than write speeds across different block sizes and access patterns. On the Lenovo ThinkPad, we see excellent scalability with block size, with slightly lower random I/O performance than sequential. As block size increases, read throughput scales more effectively than write throughput. Peak throughput for both read and write operations occurs with block sizes between 600 KB and 900 KB.

Tailoring file I/O operations to hardware and file system characteristics is very important. Using larger block sizes and sequential patterns maximizes performance, while random workloads and high stride sizes need a bit more optimization but still do not heavily impact performance in the two systems tested. However, we suspect that optimizing block sizes for sequential operations in systems equipped with older bulk storage devices allows applications on those systems to better align with their file system's capabilities, significantly enhancing throughput. Effective caching and buffering can also reduce the overhead of small writes, improving performance. Mitigating the impact of random access through intelligent caching and prefetching ensures consistent performance across varied workloads and device types. Furthermore, to confirm these speculations, we would ideally extend the experiments to include varied workloads (e.g., mixed reads and writes) and additional hardware (e.g., HDDs) to gain a more comprehensive understanding of the differences between SSD and HDD performance, as opposed to comparisons between SSDs alone. Overall, these insights provide a foundation for designing efficient applications and file systems, enhancing overall system effectiveness.