Project 3 Report

Hisen Zhang <zhangz29>
Michael Chen<chenj45>

Memory and Cache

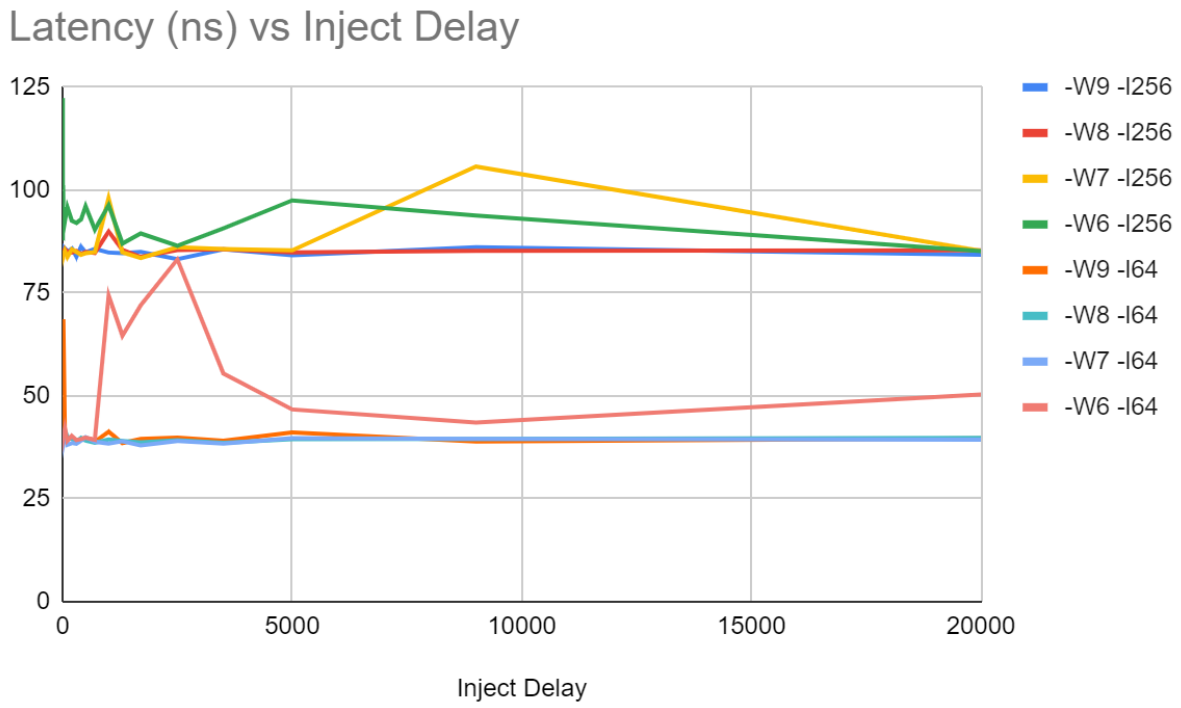
Mode:
-W6 : 0:1 read-Non Temporal Write ratio
-W7 : 2:1 read-Non Temporal Write ratio
-W8 : 1:1 read-Non Temporal Write ratio
-W9 : 3:1 read-Non Temporal Write ratio

Access size:
-I64: 64 Bytes
-I256: 256 Bytes

Latency (ns)

Inject Delay	-W9 -I256	-W8 -I256	-W7 -I256	-W6 -I256	-W9 -I64	-W8 -I64	-W7 -I64	-W6 -I64
0	84.65	85.41	85.25	122.4	38.99	39.43	38.05	39.35
2	85.06	84.91	83.67	87.78	41.92	38.94	38.38	41.81
8	85.04	85.01	85.41	101.26	44.86	39.73	37.88	39.28
15	86.21	84.5	83.9	95.48	68.64	40.12	38.25	39.66
50	85.81	85.62	85.22	92.47	40.18	41.5	39.92	42
100	85.01	85.14	83.76	95.88	39.32	40.11	38.14	39.11
200	85.79	84.81	85.63	92.59	39.83	39.56	38.53	40.3
300	83.8	85.2	85.01	92.05	38.85	38.93	38.39	39.27
400	86.26	84.48	84.31	92.91	39.52	39.83	39.12	39.51
500	84.88	85.01	84.69	96.1	39.82	39.13	39.98	39.85
700	85.73	84.71	84.93	90.42	38.92	38.61	38.87	39.36
1000	84.84	89.92	98.02	96.34	41.3	39.43	38.44	74.37
1300	84.66	85.5	84.89	87.01	38.54	39.06	38.96	64.61
1700	84.97	83.79	83.5	89.48	39.56	38.71	37.99	72.04
2500	83.25	85.43	86.16	86.49	39.85	39.24	39.06	83.14

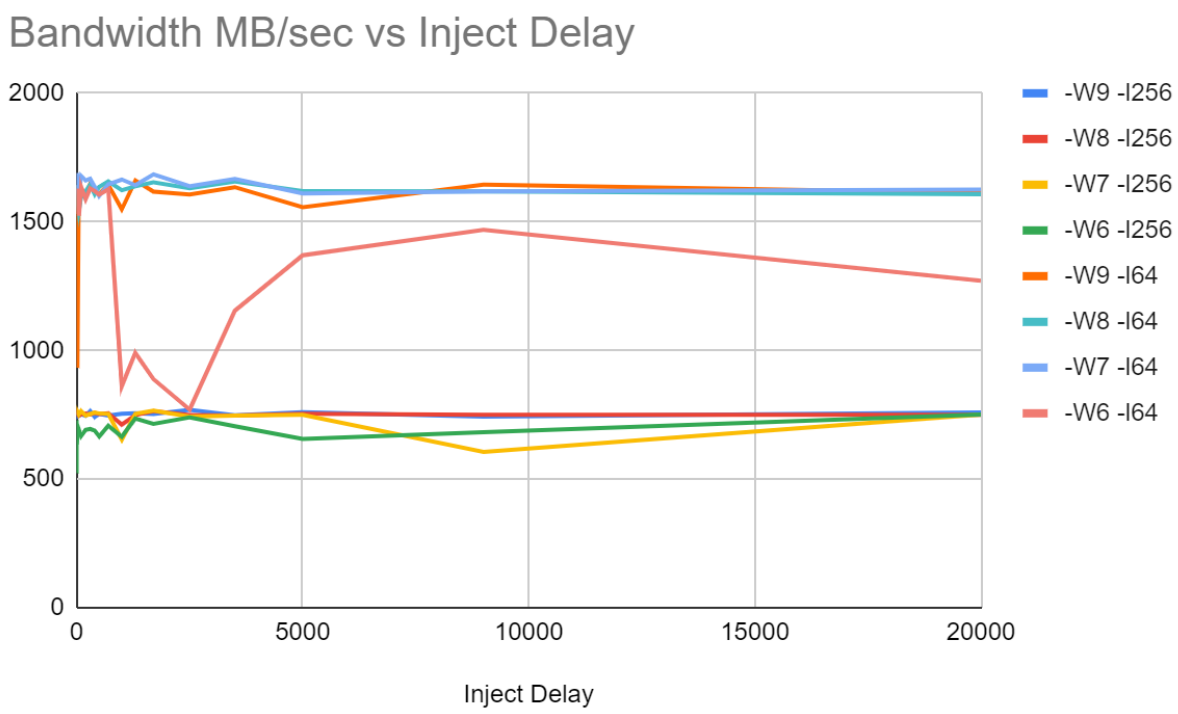
3500	85.61	85.64	85.67	90.7	39.14	38.63	38.4	55.44
5000	84.2	84.86	85.38	97.51	41.1	39.52	39.74	46.71
9000	86.14	85.28	105.73	93.83	38.91	39.55	39.53	43.56
20000	84.32	85.38	85.24	85.18	39.7	39.8	39.38	50.37



Bandwidth MB/sec

Inject Delay	-W9 -I256	-W8 -I256	-W7 -I256	-W6 -I256	-W9 -I64	-W8 -I64	-W7 -I64	-W6 -I64
0	756.1	749.3	750.7	522.9	1641.5	1623.2	1682.1	1626.4
2	752.4	753.7	764.9	729.1	1526.5	1643.5	1667.6	1530.6
8	752.6	752.9	749.3	632	1426.8	1610.8	1689.6	1629.4
15	742.4	757.4	762.9	670.3	932.3	1595.2	1673.4	1613.8
50	745.8	747.5	751	692.1	1592.8	1542	1603	1524
100	752.9	751.7	764.1	667.5	1627.6	1595.5	1677.9	1636.5
200	746	754.6	747.4	691.2	1606.8	1617.9	1661	1588.2
300	763.8	751.2	752.9	695.2	1647.6	1644.2	1667.3	1629.9
400	742	757.6	759.1	688.9	1619.5	1606.9	1636.1	1619.9
500	754	752.8	755.7	666	1607.2	1635.6	1600.9	1606.1

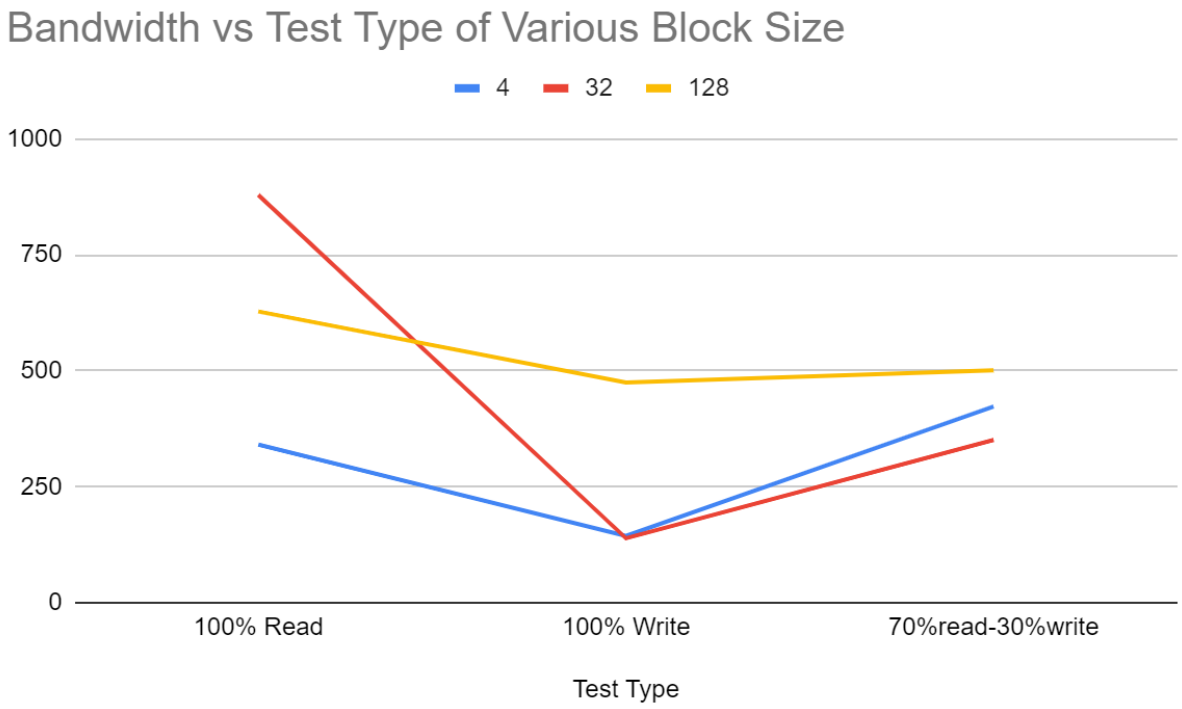
700	746.6	755.5	753.6	707.8	1644.3	1657.5	1646.3	1626.1
1000	754.4	711.7	653	664.3	1549.5	1622.9	1664.9	860.6
1300	756	748.6	753.9	735.6	1660.5	1638.5	1642.6	990.6
1700	753.2	763.8	766.5	715.2	1617.8	1653.4	1684.7	888.4
2500	768.8	749.1	742.8	739.9	1606.2	1631.2	1638.7	769.8
3500	747.6	747.4	747	705.6	1635	1656.7	1666.7	1154.5
5000	760.1	754.2	749.6	656.3	1557.2	1619.5	1610.3	1370.1
9000	743	750.4	605.3	682.1	1644.7	1618.1	1618.8	1469.2
20000	759	749.6	750.8	751.3	1612.1	1607.9	1625.3	1270.7

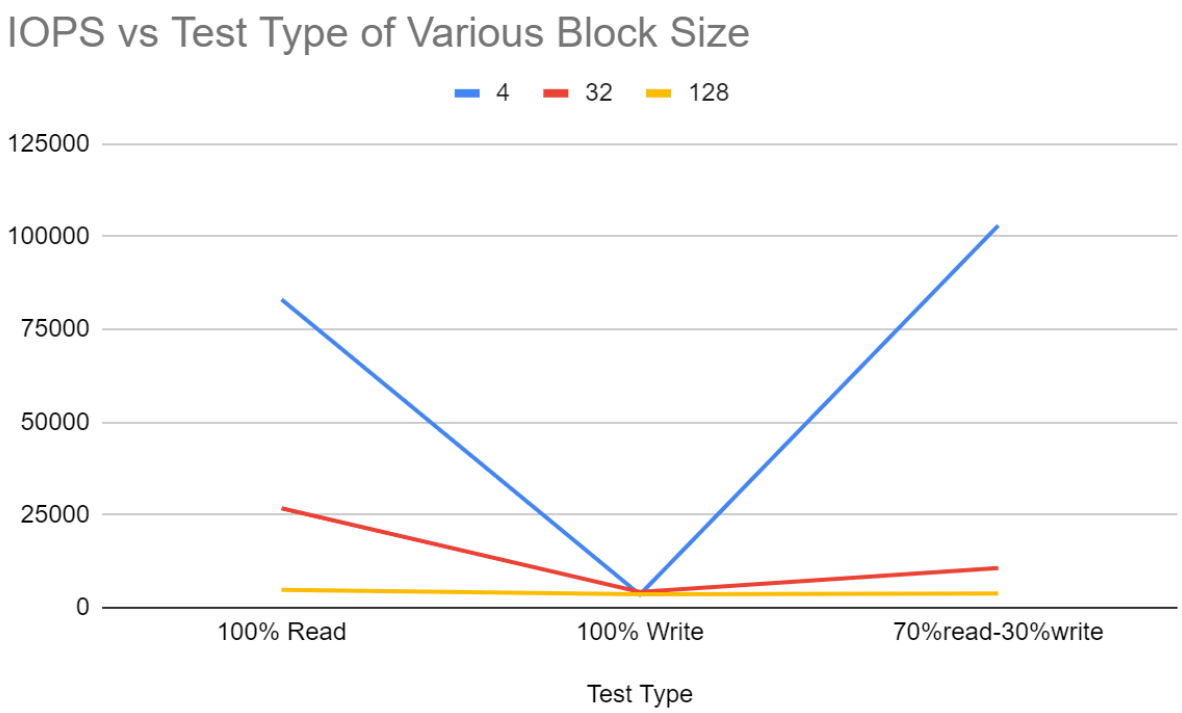


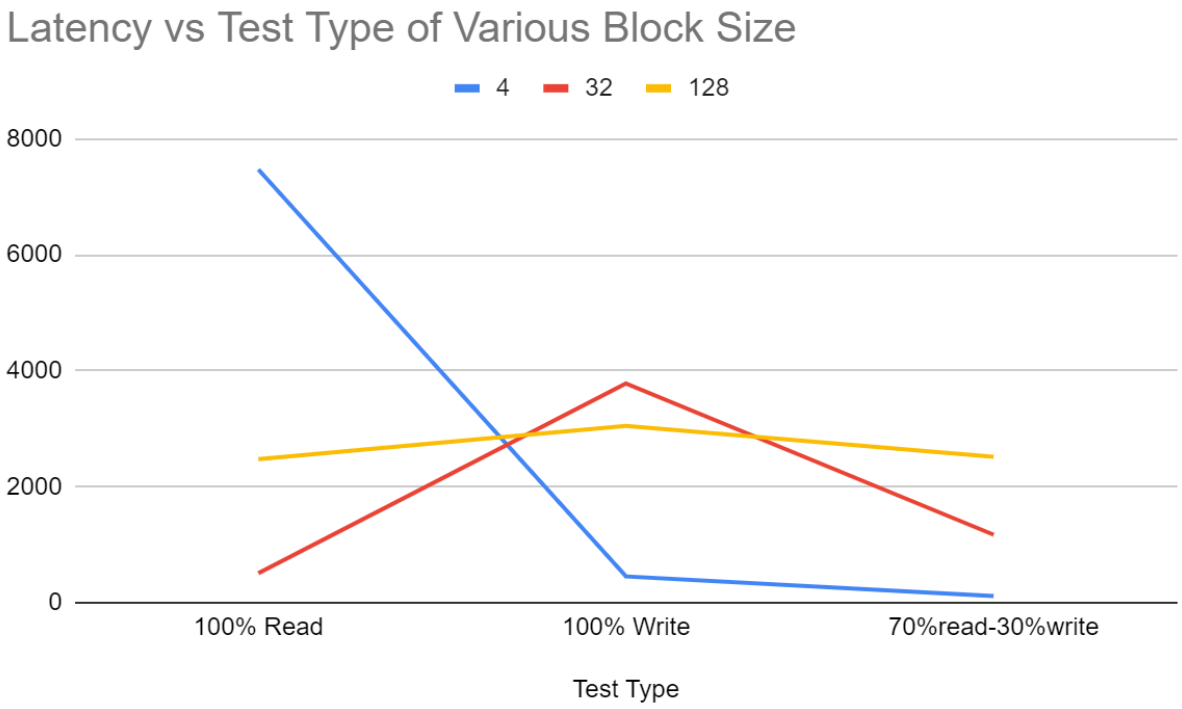
We observed that most test remains flat throughout the inject delay range for both latency and bandwidth. The data for 256-byte access generally takes longer, which makes sense as the size of the cache line is smaller than that (64B). The 64-byte access generally allows higher (about twice) bandwidth and lower (about half) latency, which explains the factor of 4 (256 to 64). The queuing theory tells us at higher resource utilization, the latency will grow drastically longer. However, we did not observe this pattern in the dataset. One possible cause is the small injection delay range specified in the homework PDF. Observing the desired effect on our system may take a much longer delay. The fluctuation at lower injection delay in the “-W6 -I64” test remains unexplained, but an assumption could be that other programs outside of VM had a memory use burst. The abnormal pattern in bandwidth data supports this.

Storage

No.	IO Depth	Test Type	Block Size KB	IOPS	Bandwidth Read/Write	Latency avg (usec)	Bandwidth avg (KiB/s)	CPU usr	CPU sys
1	16	100% Write	4	3520	144	452.27	138629.06	5.43%	42.94%
2	16	100% Write	32	4230	139	3780.7	182231.3	0.69%	24.80%
3	16	100% Write	128	3623	475	3049.82	973785.37	0.93%	15.36%
4	16	70%read-30%write	4	103k	423	115.86	530393.71	8.89%	28.57%
5	16	70%read-30%write	32	10.7k	351	1172.41	819381.89	0.89%	12.56%
6	16	70%read-30%write	128	3820	501	2515.95	889102.5	0.98%	16.96%
7	16	100% Read	4	83.1k	341	7471.88	469837.31	6.50%	23.39%
8	16	100% Read	32	26.8k	879	509.31	1036.44	2.31%	29.04%
9	16	100% Read	128	4788	628	2477.44	905854.81	0.90%	21.81%


Bandwidth vs Test Type of Various Block Size
4
32
128
1000
750
500
250
0
100% Read
100% Write
70%read-30%write
Test Type


IOPS vs Test Type of Various Block Size
4
32
128
125000
100000
75000
50000
25000
0
100% Read
100% Write
70%read-30%write
Test Type



The test suggests that for a block size of 32K and 128K, the highest bandwidth is achieved at 100% read, and the lowest bandwidth presents at 100% write except for 4KB. The 70-30 read-write mix ends up somewhere in the middle. The larger the block size, the higher the bandwidth. This might be the OS and SSD scheduler optimizing the write/read access so even though the block size is larger than 4K, we still get good performance.

IOPS is extremely high for a 4K read. Write is where all the bottlenecks exist. The fact that we are performing test in a VM may negatively impact the result, as VM may not fully take advantage of hardware optimization by writing into a simulated disk. While an enterprise-grade product has 4KB IOPS of 130K and a consumer-grade product can easily go higher than this, the SSD firmware is what makes the difference - enterprise product may tradeoff performance with data integrity so data will be guaranteed stored in the SSD even a power outage happens.

The trend of latency for read and write is totally flipped for increasing block size. The latency for 4K read is high and behaves rather awkwardly compared to that of other block sizes. We assume this is still related to VM virtualization. However, a 70-30 mix actually improves latency. This could be attributed to the SSD scheduler sending data just read while waiting for a page to be programmed in a write command.