Lab 4: Benchmark Testing

Table 1: Measured Throughput

Workload	Concurrency	Record Size	FileIO Throughput (MB/s)	IOZone Throughput (MB/s)
RS/RR/WS/WR	1	64KB	1070/919/439/560	249/185/126/424
RS/RR/WS/WR	2	64KB	1669/1200/1157/789	221/194/956/806
RS/RR/WS/WR	4	64KB	3881/2595/1957/1726	341/303/334/107
RS/RR/WS/WR	8	64KB	5194/9456/3739/2458	389/427/125/544
RS/RR/WS/WR	1	1MB	1168/1044/612/488	412/338/954/367
RS/RR/WS/WR	2	1MB	1910/1858/1367/871	361/356/104/1008
RS/RR/WS/WR	4	1MB	2992/3126/1917/2102	364/364/723/719
RS/RR/WS/WR	8	1MB	5827/9418/3734/3723	358/348/222/761
RS/RR/WS/WR	1	16MB	1323/1681/602/570	352/388/223/90
RS/RR/WS/WR	2	16MB	2445/3665/795/1072	377/345/760/130
RS/RR/WS/WR	4	16MB	3794/6737/2282/1486	342/358/746/116
RS/RR/WS/WR	8	16MB	8457/10058/3180/3425	317/334/524/719

Table 2: Measured Latency

Workload	Concurrency	Record Size	FileIO IOPS (Thousand Ops/s)	IOZone IOPS(Thousand Ops/s)
RR/WR	1	4KB	162/61	10/11
RR/WR	2	4KB	203/38	15/13
RR/WR	4	4KB	207/35	35/22
RR/WR	8	4KB	156/25	22/33

Findings

It is important to note that the FileIO was tested once (but I viewed multiple times to confirm trends), and IOZone was averaged. All in all, I believe that the data was of very poor quality. Even a renown program like IOZone had some very odd outliers, so the data is a bit hard to analyze. However, their are a few trends that were noticed:

- Every FileIO test had the greatest throughput with 8 processes, but also the worst latency.
- IOZone had a clear trend of more operations as processes increased, opposite to FileIO. However, FileIO's latency seems very weird and could have problems.
- Record Size seemed to have little to no impact on high throughput, but ⅝ worst throughput rows were running with 64kb, while ⅝ best throughput ran 1mb.
- In general, FileIO had similar random vs sequential read throughput, and overall greater sequential write than random write.
- IOZone random read throughput was almost identical to sequential read throughput, while sequential vs random write throughput was very odd and fluctuated a lot.

From this we gained a few insights. First, data with read/write fluctuates quite a bit, and even well established programs can be wacky (or virtual space makes it weird). Second, more processes leads to more throughput and better latency. Despite FileIO latency (likely error), more processes outperformed less in almost all cases. This makes sense because multiple processes can utilize multiple threads, and have other processes do things during down time in one process. Third, record size seems to have little impact, but gets bad if the record size is set substantially low. This also makes sense, as the lower record size is, the more the record has to be reset. Fourth, sequential read/write is faster than random read/write. Lastly, this is logical because in random read/write, you have to move to the start of the new data every time, vs simply continual reading from their sequential counterparts.