Team Members:

1. **Ayelet Bendor**
2. **Varun Gunda**
3. **Chandu Chinta**

The document elaborate on the benchmark design, benchmark results in separate tables and graphs to highlight the findings.

We worked in collaboration, each member of the group worked on the different tasks of this assignment.

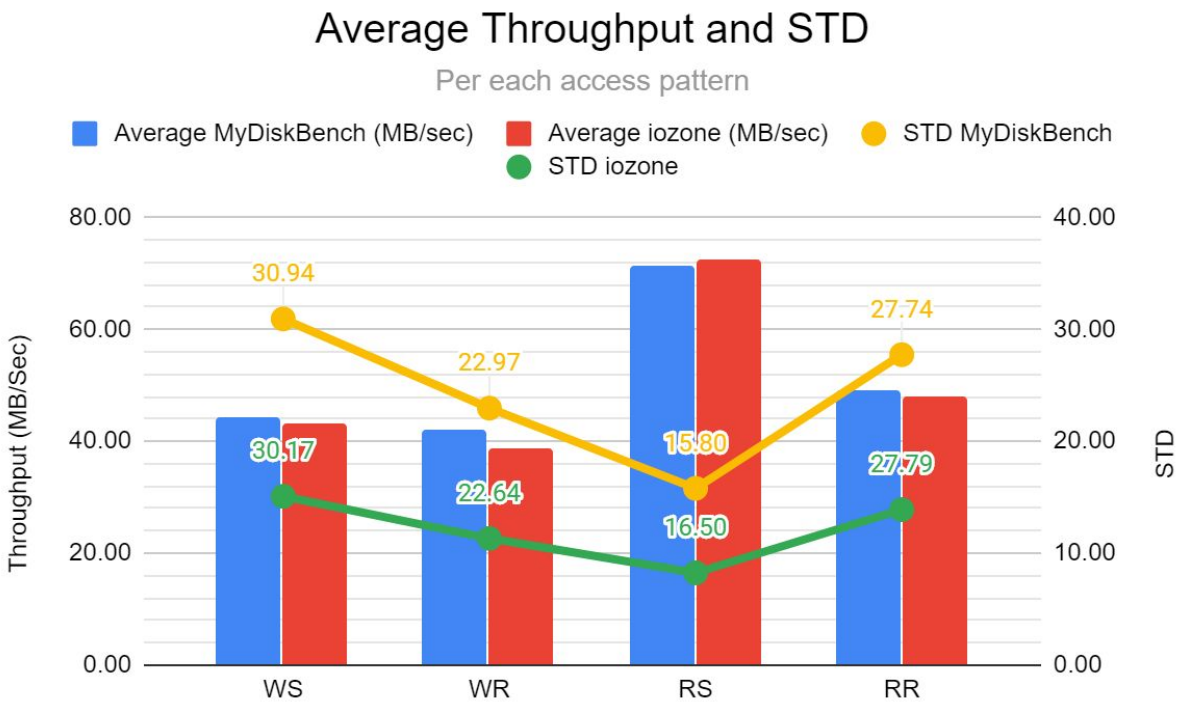
Benchmark Design:

The benchmark is designed considering the following factors: Usability, reliability.
It takes in the parameters of record size, file size, list of files, number of threads and test type (sequential read/write, random read/write), performs the test and gives the throughput in MBPS and IOPS.
We used C++ for this assignment. The assignment can be done using C but considering the extra features that C++ gives over C, we decided to use C++. Since we need to deal with multiple threads, we used thread library in C++.
The code is c style (structural oriented). There is a function for reading and another for writing. The main function parses the arguments, creates threads (and also creates file for write tests), and waits for the threads to return. In the functions used by threads for reading and writing tests, the time taken is measured and is printed to the stdout. Since this code involves multithreading, in order to eliminate any type of dependency between threads, we created another file **MyDiskBenchWrapper.py** which takes the output from the c++ executable, parses it and presents it in human readable format. The user only interacts with **MyDiskBenchWrapper.py** and it calls c++ executable, parses the output and presents it to the user. Although there are multiple ways to access files, we sticked to linux system calls of open, read and write. Since we are dealing with hard disk directly, the linux system calls will be faster that other libraries since all other libraries add overhead on the top of these system calls.

Possible Improvements:

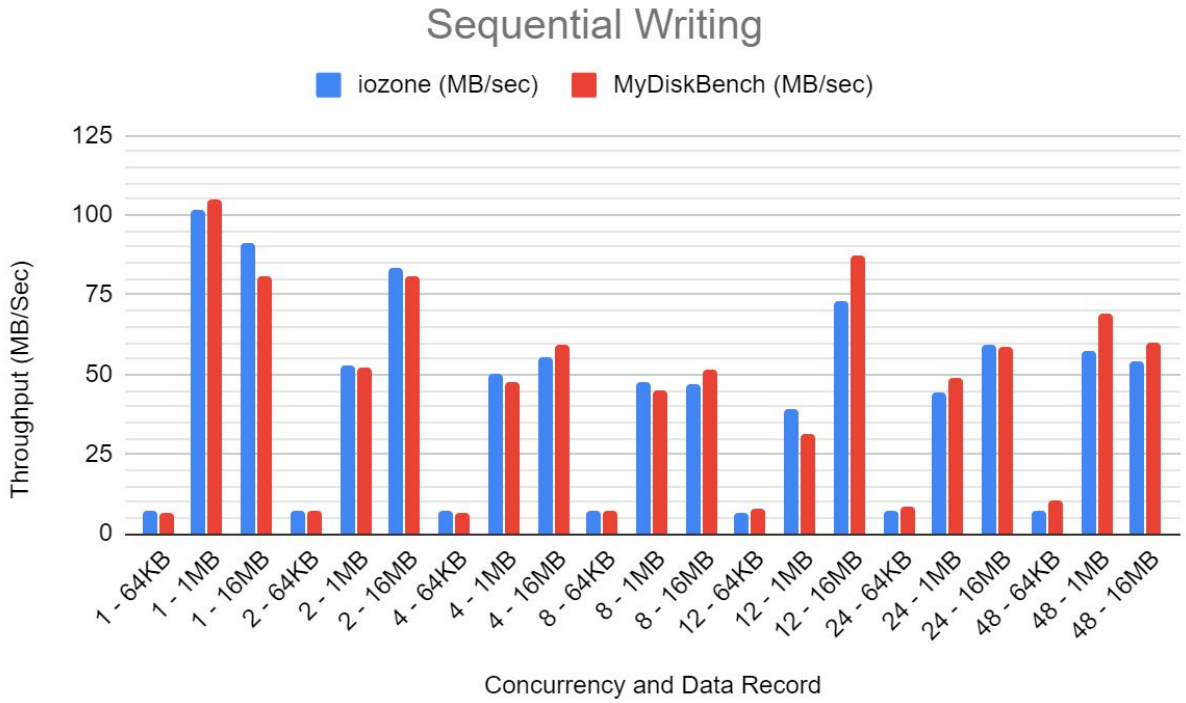
The current implementation of MyDiskBench does not have as many features as IOZone does. Some improvements include: Having only one file do everything so that the wrapper is not required, give the output for every thread like IOZone instead of the average, creating histogram plots. Since these are not under the purview of the assignment, these make good useful extensions of this program.

Tables are appended to the end of this sheet.



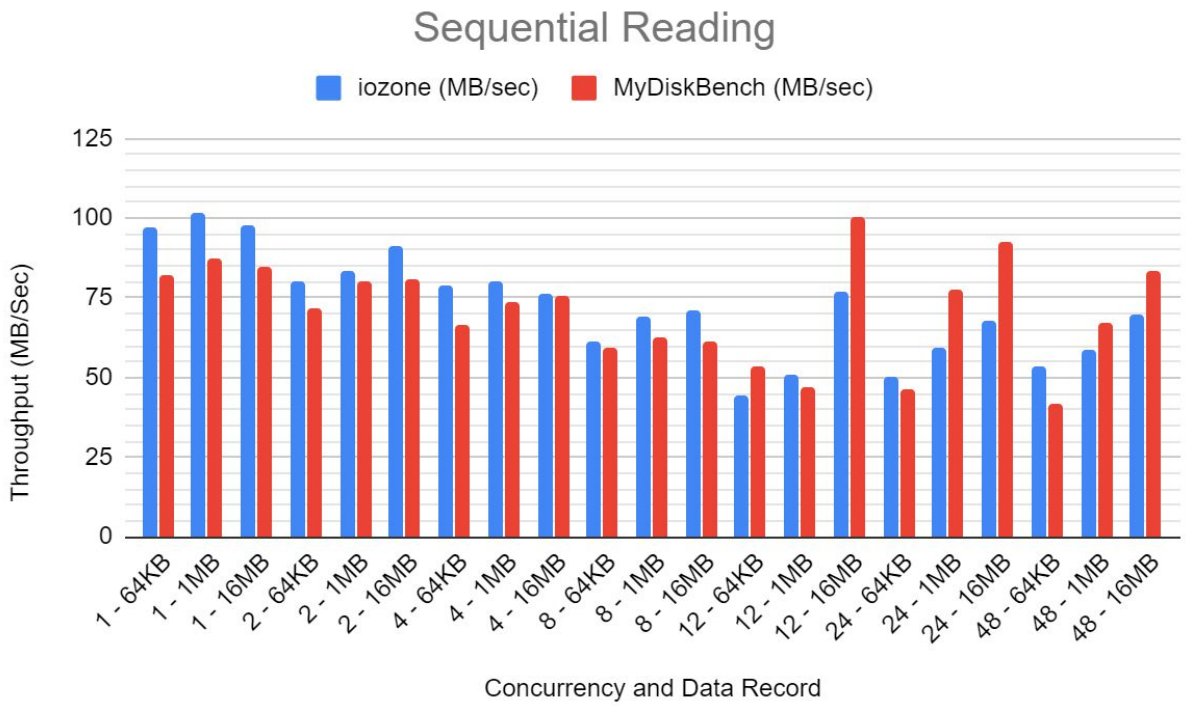
This graph highlights the average between writing sequentially to randomly and reading sequentially to randomly, and more importantly we calculated the STD per each access pattern so we can indicate the range of Throughput values.
For example, by looking at WS and WR we notice that their average is close, while STD indicates that the range of values between min to max value is larger. It can be seen that while both averages are fairly close, WS is more “spread out” on the graph and WR is closer together. This type of information helps to analyze the graphs below.

Reading from disk is faster than writing, the throughput overall values are higher than writing and the range between the max and min value is smaller.
There could be a number of reasons for the variation between the two operations, the main reason is when concurrently writing to a file.
POSIX file system avoids race conditions between threads when more than one thread wishes to write to a shared file by implementing a lock mechanism, which causes overhead for threads in the queue. On the other hand, when a thread tries to access a shared file to read only, it can be done in parallel with other threads.



The graph above represents the throughput of sequential writing to the disk of iozone and in a column next to it, MyDiskBench throughput.
While the disk manufacturer estimates the theoretical throughput as 115 MB/sec, we can see that the throughput's values vary, as a result of writing sequentially to disk in different record sets and different number of threads. The closest value to the theoretical throughput (and also the highest throughput) we receive for 1 thread with a record size of 1MB.
We can also notice a trend on the graph, the throughput of 16MB and 1MB decreases as we add more threads, it peaks at 16MB with 12 threads.

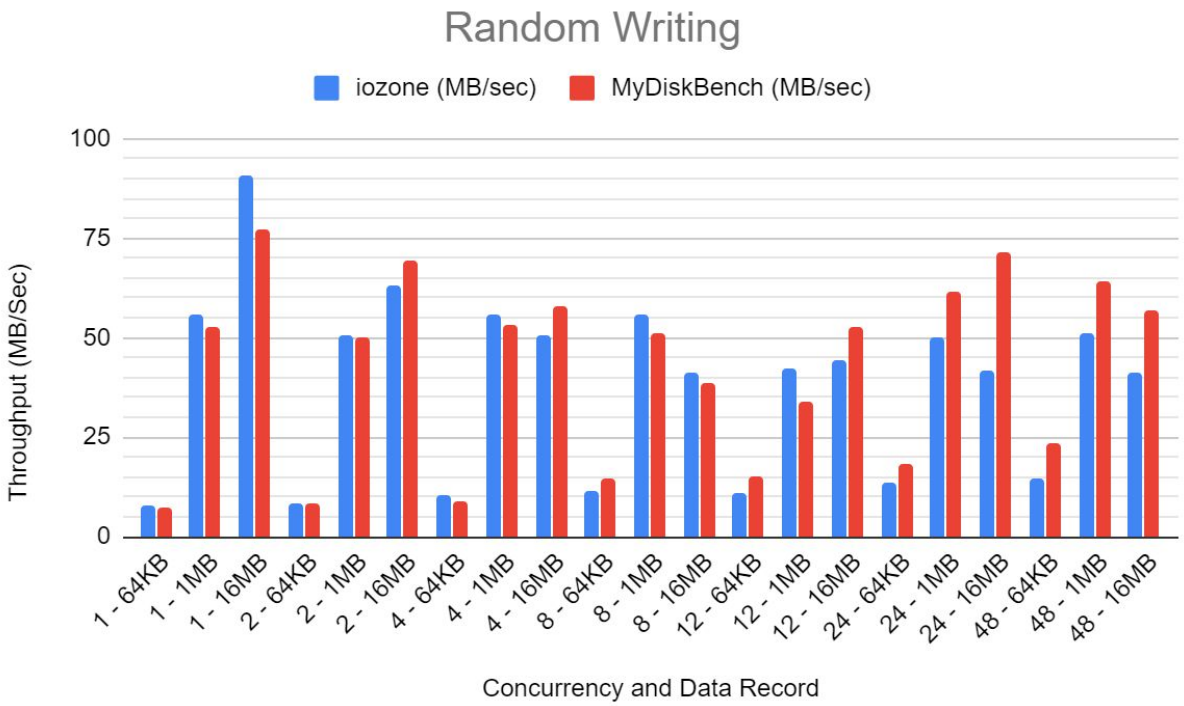
Another noticeable trend on the graph is when writing in blocks of 64 KB the throughput is very low. When writing a big data set in significantly small records, there's OS overhead (trapping and returning from a larger system calls) and hardware overhead (moving more frequently the head to write).



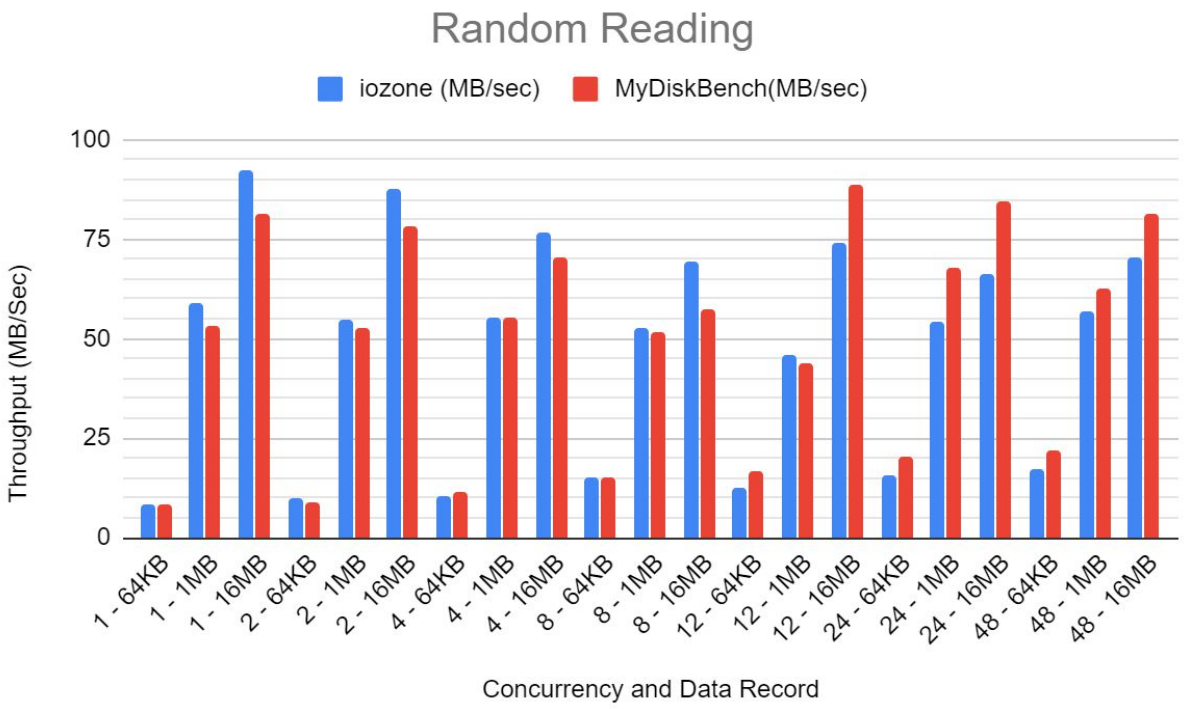
The graph above represents the throughput of sequential writing to the disk of iozone and in a column next to it, MyDiskBench throughput.

Reading operation throughput is also affected by the reading record size, however, the deviation between the three record sizes is not far off from one another, 64KB throughput value is by little smaller than the other two record sizes. We also notice the same tread as the above graph, the throughput decreases up until running with 12 treads. One of the possible explanations is that threads share the same ALU, when it gets full threads will take turns. On the opposite side, the smaller the data is, it can fit into the cache.

Random i/o access pattern throughput generally is lower than sequential i/o access. With each writing/reading to the disk, the disk performs a seek operation to reach the requested data block. The smaller the data record is, the more seek operation the disk will perform and will add additional time to it's latency.
The following two graphs below showcase the throughput of random writing and reading to the disk of iozone and in a column next to it, MyDiskBench throughput.



Writing randomly to the disk has the highest throughput when writing in blocks of 16MB with one thread, then as we add more threads we see a trend fluctuation of the throughput, and the lowest throughput is around 8 and 12 threads.



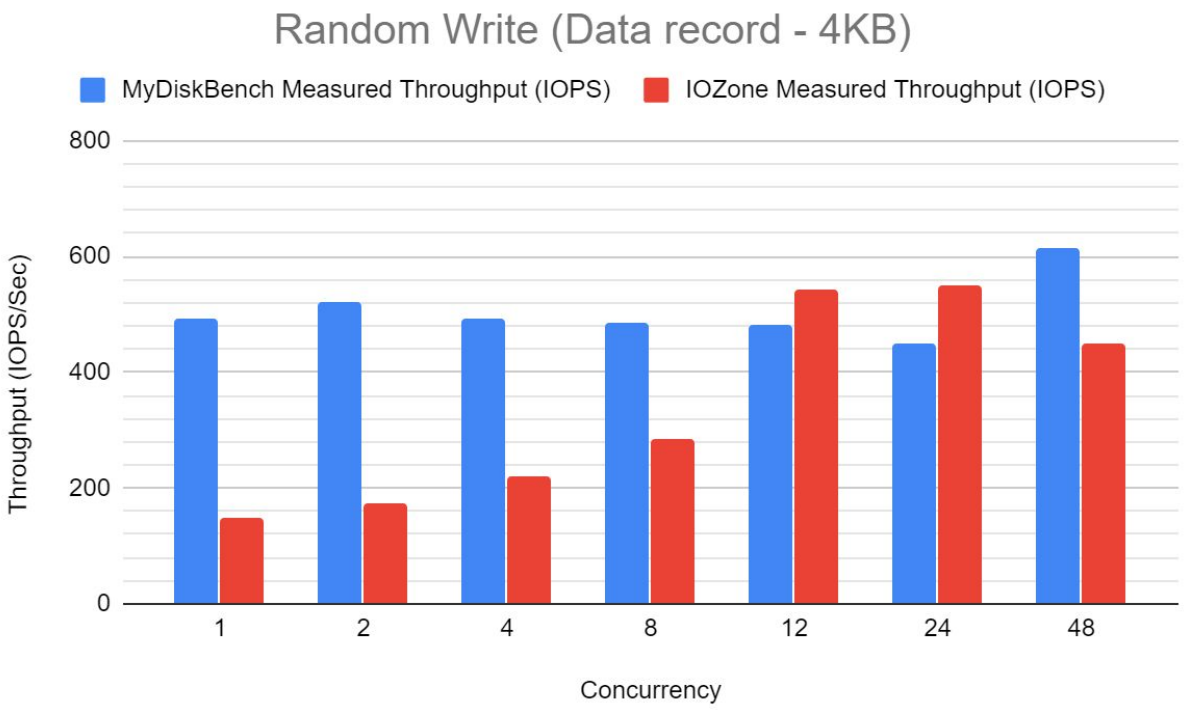
Similarly to sequential read, random read average throughput is higher than writing randomly. In random access we notice that the

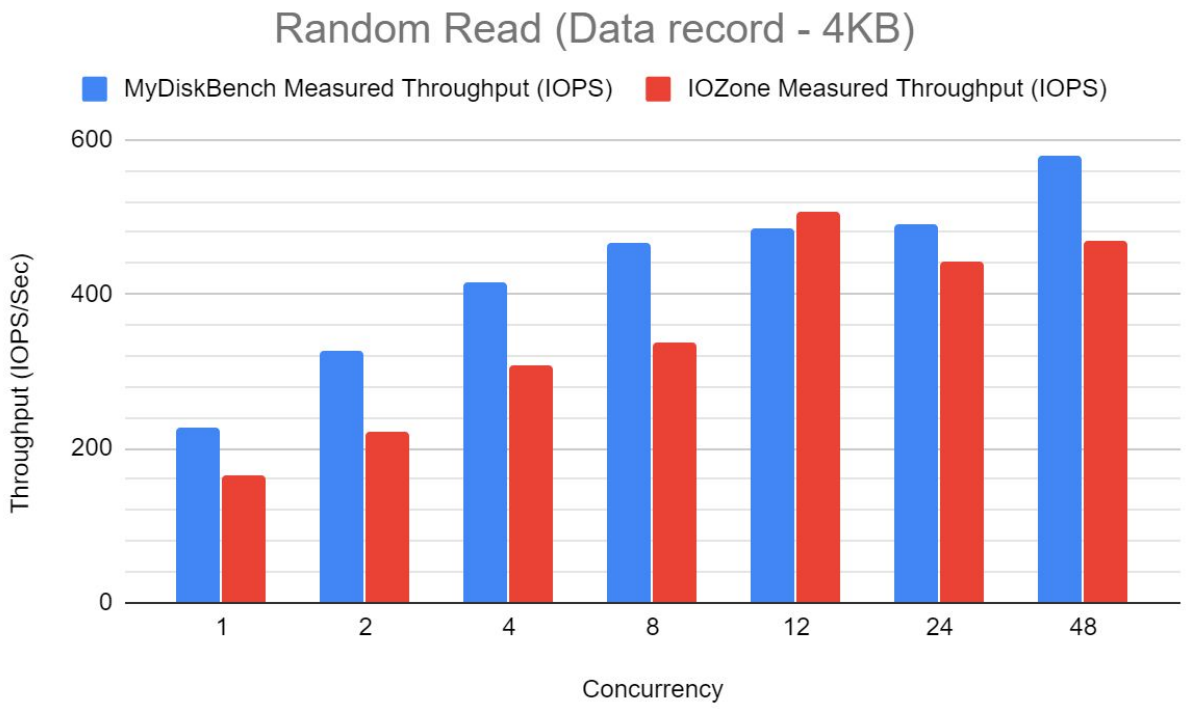
We also notice that writing/reading in blocks of 64KB throughput is slowly increasing as we add more threads. Previously when we wrote sequentionaly, 64KB's throughput variant was small regardless of the number of threads.

For smaller data records the throughput is higher when using multithreading. Random access to disk is using a seek function to find the stored data on the disk. While sequential access looks for the data ones, random access.

The two graphs below showcase the IOPS for random access to disk and record size of 4KB. The average IOPS by the manufacturer is 82.23 IOPS/sec . We can see that we get a close output when the concurrency is one, as we add more threads, we increase the throughput. We've seen a similar pattern for random access with small data records (64KB).

Average IOPS = 1/(average latency in ms + average seek time in ms) = 1/(0.00416 + (0.0075 + 0.0085)/2) = 82.23
Where average latency = 4.16 ms
Average seek time = (average read time + average write time)/2 = (7.5+8.5)/2 ms




Random Read (Data record - 4KB)
MyDiskBench Measured Throughput (IOPS)
IOZone Measured Throughput (IOPS)
600
400
200
0
Throughput (IOPS/Sec)
1
2
4
8
12
24
48
Concurrency

TABLES:

Table 1a:

Workload	Concurrency	Record size	MyDiskBench Measured Throughput (MB/sec)	Iozone Measured throughput MB/Sec	Theoretical Throughput (MS/sec)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
WS	1	64KB	78.1071227	6.88884	115	67.91923713	5.990295652
WS	1	1MB	104.9675052	101.54698	115	91.27609146	88.30172174
WS	1	16MB	80.59057783	91.22655	115	70.07876333	79.32743478
WS	2	64KB	7.084745036	7.43352	115	6.160647857	6.463930435
WS	2	1MB	52.19483353	53.08798	115	45.38681176	46.16346087
WS	2	16MB	80.99599766	83.65636	115	70.43130231	72.74466087
WS	4	64KB	6.845327028	7.05844	115	5.952458285	6.137773913
WS	4	1MB	47.91898696	50.02704	115	41.66868432	43.50177391
WS	4	16MB	59.62848658	55.14828	115	51.85085789	47.95502609
WS	8	64KB	7.151637054	7.11496	115	6.21881483	6.186921739
WS	8	1MB	44.81930031	47.58264	115	38.97330462	41.3762087
WS	8	16MB	51.38086063	46.81464	115	44.67900924	40.70838261
WS	12	64KB	7.536787842	6.831	115	6.553728558	5.94
WS	12	1MB	31.49519114	38.91072	115	27.38712273	33.8354087
WS	12	16MB	87.17193472	72.75312	115	75.80168237	63.26358261
WS	24	64KB	8.830532255	7.42296	115	7.6787237	6.454747826
WS	24	64KB	48.87781501	44.66592	115	42.50244783	38.83993043
WS	24	1MB	58.98753427	59.2008	115	51.29350807	51.47895652
WS	48	16MB	10.47216796	7.46688	115	9.106233012	6.49293913
WS	48	64KB	69.35742781	57.144	115	60.3108068	49.69043478
WS	48	1MB	59.75863115	53.82864	115	51.96402709	46.80751304

Table 1b:

Workload	Concurrency	Record size	MyDiskBench Measured Throughput (MB/sec)	Iozone Measured throughput MB/Sec	Theoretical Throughput (MS/sec)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
RS	1	64KB	82.18496432	96.86175	115	71.46518637	84.2276087
RS	1	1MB	87.57600896	102.00809	115	76.15305127	88.70268696
RS	1	16MB	84.82157649	97.63425	115	73.7578926	84.89934783
RS	2	64KB	71.47443951	80.48238	115	62.15168653	69.98467826
RS	2	1MB	80.28790742	83.528	115	69.81557167	72.63304348
RS	2	16MB	81.0857894	91.20068	115	70.50938209	79.30493913
RS	4	64KB	66.76446618	79.15296	115	58.05605755	68.82866087
RS	4	1MB	73.91100365	79.93828	115	64.27043795	69.51154783
RS	4	16MB	75.76149925	76.59848	115	65.87956456	66.60737391
RS	8	64KB	5.564370639	61.19264	115	4.838583165	5.32109913
RS	8	1MB	62.4622573	69.14248	115	54.31500635	60.12389565
RS	8	16MB	60.9977662	71.26824	115	53.04153582	61.97238261
RS	12	64KB	53.47677364	44.36496	115	46.50154229	38.57822609
RS	12	1MB	47.15916679	50.69784	115	41.00797112	44.08507826
RS	12	16MB	100.0948164	77.01012	115	87.03897076	66.96532174
RS	24	64KB	46.1367251	50.0904	115	40.11889139	43.55686957
RS	24	1MB	77.50353837	59.154	115	67.39438119	51.43826087
RS	24	16MB	92.81415416	68.08392	115	80.70796014	59.2034087
RS	48	64KB	42.06047014	53.44224	115	36.57432186	46.47151304
RS	48	1MB	67.45584738	57.144	115	60.3108068	49.69043478
RS	48	16MB	83.34893413	53.82864	115	51.96402709	46.80751304

Table 1c:

Workload	Concurrency	Record size	MyDiskBench Measured Throughput (MB/sec)	Iozone Measured throughput MB/Sec	Theoretical Throughput (MS/sec)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
WR	1	64KB	7.316302924	7.65302	115	6.362002542	6.6548
WR	1	1MB	52.74326802	55.71789	115	45.86371132	48.45033913
WR	1	16MB	77.4215011	90.54823	115	67.32304443	78.7375913
WR	2	64KB	8.180024444	8.21322	115	7.113064734	7.141930435
WR	2	1MB	49.9409877	50.78588	115	43.42694583	44.16163478
WR	2	16MB	69.26828608	63.3854	115	60.23329225	55.11773913
WR	4	64KB	9.08566612	10.40672	115	7.900579234	9.049321739
WR	4	1MB	52.9741027	55.98288	115	46.06443713	48.68076522
WR	4	16MB	57.65311293	50.37276	115	50.13314168	43.8024
WR	8	64KB	14.50815375	11.32816	115	12.61578587	9.850573913
WR	8	1MB	50.96885624	56.01928	115	44.32074455	48.71241739
WR	8	16MB	38.59825026	41.39192	115	33.56369588	35.99297391
WR	12	64KB	15.41043063	10.953	115	13.40037446	9.524347826
WR	12	1MB	34.09174141	42.204	115	29.64499253	36.69913043
WR	12	16MB	52.67760687	44.18004	115	45.80661467	38.41742609
WR	24	64KB	18.45031748	13.51632	115	16.04375433	11.75332174
WR	24	1MB	61.33279028	50.28672	115	53.33286112	43.72758261
WR	24	16MB	71.41761169	41.58168	115	62.10227104	36.15798261
WR	48	64KB	23.47390935	14.6184	115	20.41209508	12.71165217
WR	48	1MB	64.03201601	51.384	115	55.68001392	44.68173913
WR	48	16MB	56.98861897	41.27472	115	49.55532085	35.89106087

Table 1d:

Workload	Concurrency	Record size	MyDiskBench Measured Throughput (MB/sec)	Iozone Measured throughput MB/Sec	Theoretical Throughput (MS/sec)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
RR	1	64KB	8.178391157	8.49395	115	7.111644484	7.386043478
RR	1	1MB	53.21941053	58.91894	115	46.27774829	51.23386087
RR	1	16MB	81.48584343	92.49944	115	70.85725515	80.43429565
RR	2	64KB	9.109785368	9.71184	115	7.921552494	8.445078261
RR	2	1MB	52.88081676	54.68926	115	45.98331892	47.55587826
RR	2	16MB	78.1071227	87.39658	115	67.91923713	75.99702609
RR	4	64KB	11.40481987	10.61212	115	9.917234672	9.227930435
RR	4	1MB	55.18371218	55.45	115	47.98583668	48.2173913
RR	4	16MB	70.63676561	76.48644	115	61.42327444	66.50994783
RR	8	64KB	15.03746149	15.1468	115	13.07605347	13.17113043
RR	8	1MB	51.6717632	52.84056	115	44.931968	45.94831304
RR	8	16MB	57.18913183	69.54912	115	49.72967985	60.47749565
RR	12	64KB	16.61566221	12.34764	115	14.44840192	10.73707826
RR	12	1MB	43.85345068	45.84384	115	38.13343537	39.8642087
RR	12	16MB	88.73483536	73.91256	115	77.1607264	64.2717913
RR	24	64KB	20.16911329	15.56232	115	17.53835938	13.53245217
RR	24	1MB	67.70740351	54.46368	115	58.87600306	47.35972174
RR	24	16MB	84.28261013	66.462	115	73.2892262	57.79304348
RR	48	64KB	21.79145324	17.48304	115	18.94908977	15.20264348
RR	48	1MB	62.79588883	56.79792	115	54.60512072	49.38949565
RR	48	16MB	81.30468614	70.43808	115	70.69972707	61.25050435

Table 2a:

Workload	Concurrency	Record Size	MyDiskBench Measured Throughput (IOPS)	IOZone Measured Throughput (IOPS)	Theoretical Throughput (ops/sec)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
WR	1	4KB	491.7946038	148.69	82.23	598.0719978	180.8220844
WR	2	4KB	522.6451699	174.58	82.23	635.5894077	212.3069439
WR	4	4KB	491.1379076	219.48	82.23	597.2733888	266.9098869
WR	8	4KB	487.0114058	284.48	82.23	592.2551451	345.9564636
WR	12	4KB	482.300535	541.2	82.23	586.5262495	658.1539584
WR	24	4KB	451.0176763	549.6	82.23	548.4831282	668.3692083
WR	48	4KB	614.2823094	448.8	82.23	747.0294411	545.7862094

Table 2b:

Workload	Concurrency	Record Size	MyDiskBench Measured Throughput (IOPS)	IOZone Measured Throughput (IOPS)	Theoretical Throughput (ops/sec)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
RR	1	4KB	226.4909218	164.1	82.23	275.4359	199.5622
RR	2	4KB	326.143819	221.8	82.23	396.6239	269.7312
RR	4	4KB	415.1761842	307.08	82.23	504.8962	373.4404
RR	8	4KB	466.6921792	338	82.23	567.5449	411.0422
RR	12	4KB	484.7497722	506.52	82.23	589.5048	615.9796
RR	24	4KB	4907.349246	441.84	82.23	5967.833	537.3221
RR	48	4KB	580.1810165	468.96	82.23	705.5588	570.3028