Analysis

- **Introduction**:
  - This document shows the result of all the benchmarks that are executed for CPU, Memory, Disk and Network. Graphical Visualization has been shown to understand the performance clearly and more accurately.
  - Experiments for CPU, Memory and Network are performed on Hyperion.
  - Experiments for Disk are performed on Prometheus.
  - For calculating the time, the most feasible function that I could find after careful analysis was gettimeofday() which gives accuracy upto microseconds.

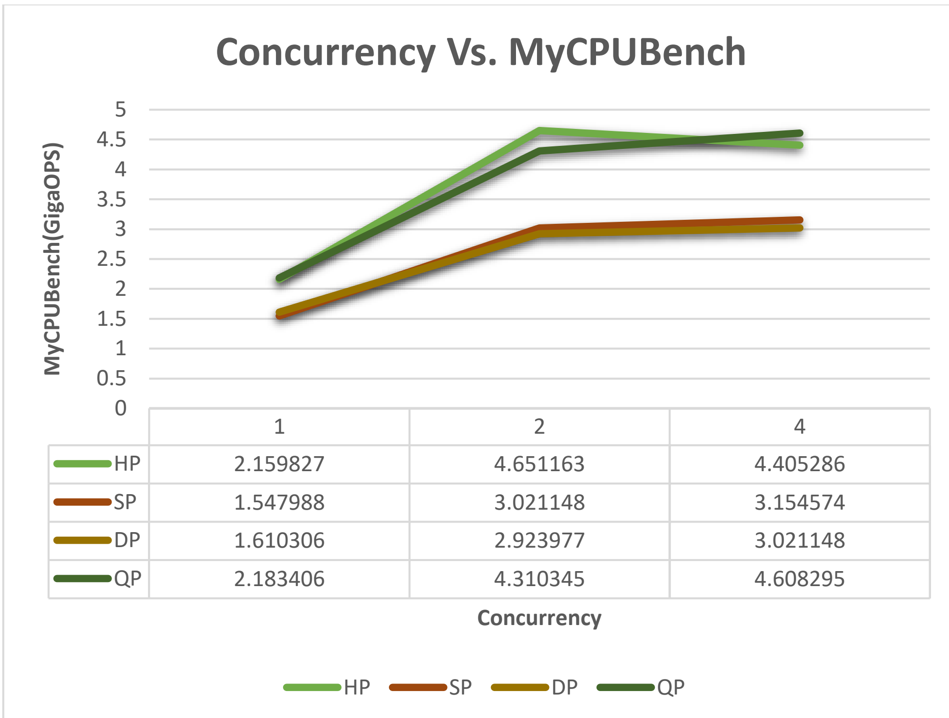
1. CPU Benchmark:

- Performance Parameters that were considered are:
  - Number of operations performed: **1 trillion.**
  - Type of operations:
    - Quarter Precision (char)
    - Half Precision (short)
    - Single Precision (int)
    - Double Precision (double)
  - Concurrency: 1,2 and 4.
  - Flops is calculated by: 2*1*2.3* operationType
    - OperationType: DP = 16
    - SP = 32
    - HP = 64
    - QP = 128
- The following table shows the comparison of Throughput/efficiency when different parameters are passed as input.

Processor Performance

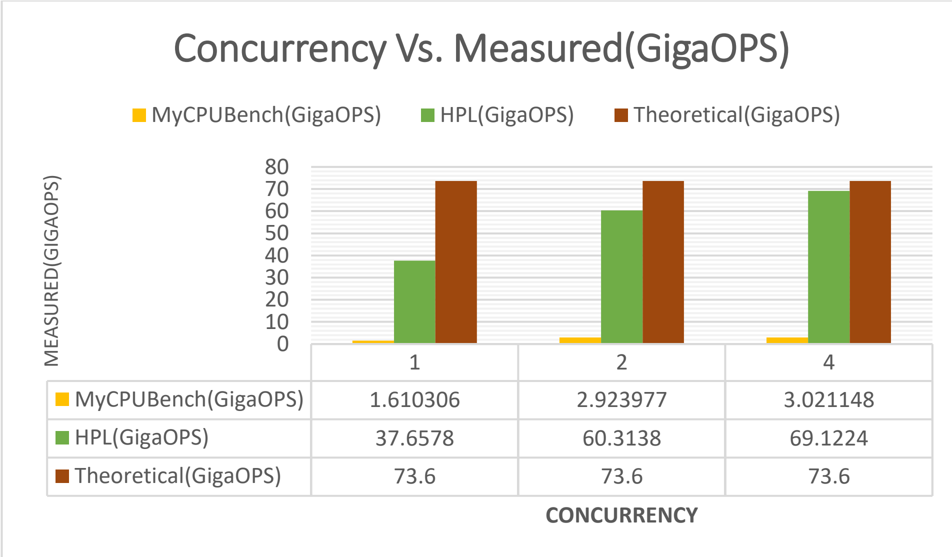
Workload	**Concurrency**	**MyCPUBench Measured Ops/Sec (GigaOPS)**	**HPL Measurerd Ops/Sec (GigaOPS)**	**Theoritical Ops/Sec (GigaOPS)**	**MyCPU Bench Efficiency(%)**	**HPL Efficiency(%)**
QP	1	2.183406	N/A	588.8	0.37082303	N/A
QP	2	4.310345	N/A	588.8	0.732055876	N/A
QP	4	4.608295	N/A	588.8	0.782658798	N/A
HP	1	2.159827	N/A	294.4	0.733636889	N/A
HP	2	4.651163	N/A	294.4	1.579878736	N/A
HP	4	4.405286	N/A	294.4	1.496360734	N/A
SP	1	1.547988	N/A	147.2	1.051622283	N/A
SP	2	3.021148	N/A	147.2	2.052410326	N/A

SP	4	3.154574	N/A	147.2	2.143052989	N/A
DP	1	1.610306	37.6578	73.6	2.187915761	51.16548913
DP	2	2.923977	60.3138	73.6	3.972794837	81.94809783
DP	4	3.021148	69.1224	73.6	4.104820652	93.91630435



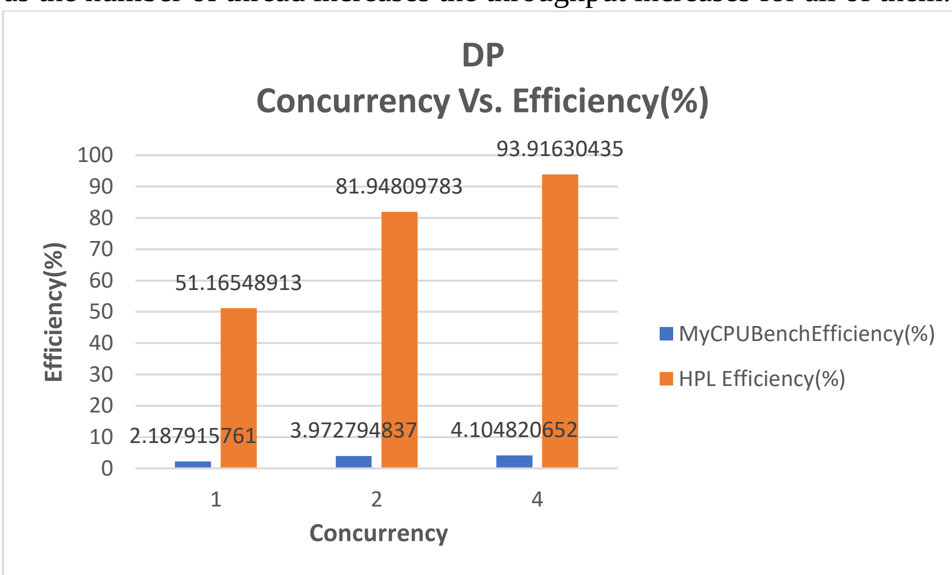
QP vs HP vs SP vs DP for GigaOps

- The following graph shows the comparison of throughput between QP, HP, SP and DP by varying concurrency.
- As we can clearly see that QP has the most throughput means more number of operations are performed in given time as compared to all other precision’s.
- Also, as the thread count increases, the throughput also increases but after certain thread count the scaling is flat where no further efficiency can be achieved.



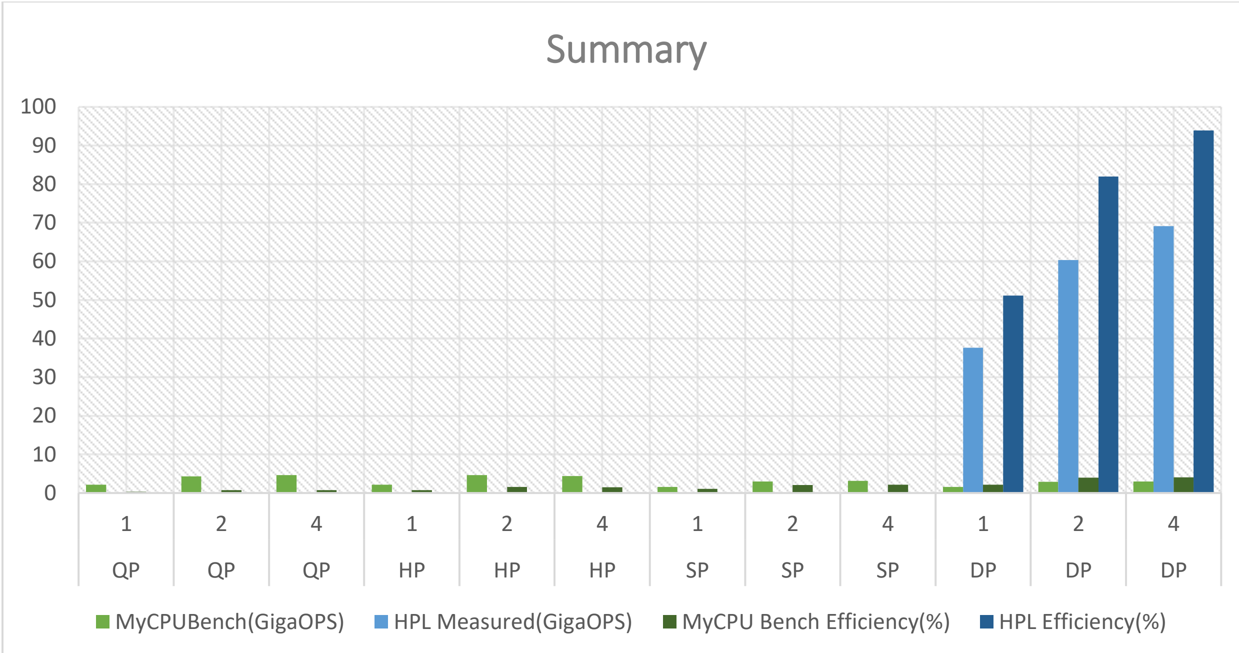
Throughput : MyCPUBenchmark vs HPL vs Theoretical

- The above graph shows comparison of MyCPUBenchmark vs HPL vs Theoretical values.
- It shows that the HPL values are pretty close to theoretical values as it uses AVX ofcourse which executes more instructions per cycle.
  - Also, as the number of thread increases the throughput increases for all of them.



Efficiency comparison for DP for MyCPUBench vs HPL

- From the graph, HPL benchmark has higher efficiency as compared to MyCPUBench.
- As the thread count increases, the number of operations executed per cycle also increases.
- Comparison is with respect to Theoretical values.



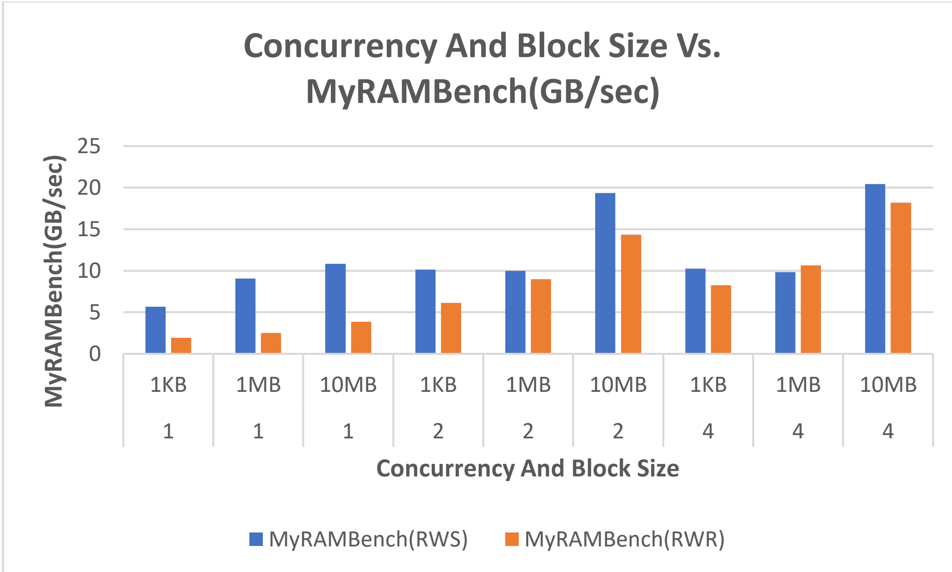
<u>Summarizing entire throughput scenario of all the Precisions and Benchmarks</u>

- Concluding the overall results for the CPU benchmark would be:
  - As the number of threads increase, the throughput increases, but after certain threshold value, the scaling is flat or decreases with increase in parallelism.
  - The value for HPL is quite close to Theoretical values of Benchmark.
  - I have tried to maximize cpu usage by giving large number of operations.
  - All the results are in GigaOps.
- Possible Improvements:
  - For Similar operations, CPU may cache the result and provide an output quickly as compared to different operations.
  - Use of AVX instructions to increase the number of instructions per cycle will help in increasing the values.

2. **<u>Memory</u>**:
   - Performance Parameters that were considered are:
     - Size of data used: **100GB**
     - Type of operations:
       1. RWS – Read Write Sequential.
       2. RWR – Read Write Random.
     - Blocksize: 1KB, 1MB, 10MB
     - Concurrency: 1,2 and 4.
     - For latency:
       1. Data: 1byte
       2. Operations: 100 million

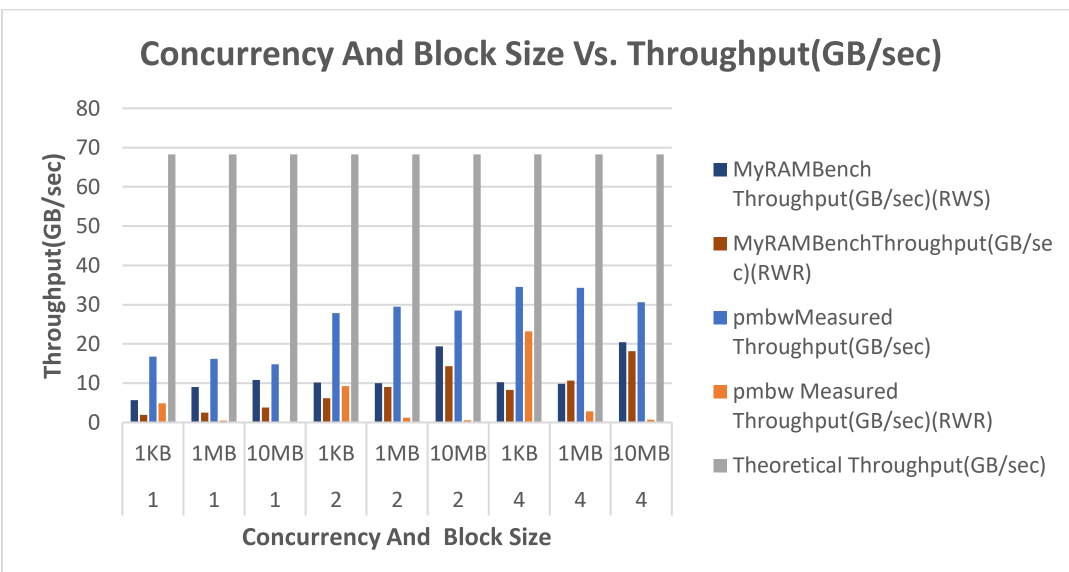
- The following table shows the comparison of Throughput/efficiency when different parameters are passed as input.

Memory Throughput							
Work-Load	**Con-currency**	**Block Size**	**MyRAMBench Measured Throughput (GB/sec)**	**pmbw Measured Throughput(GB/sec)**	**Theoretical Throughput (GB/sec)**	**MyRAMBench Efficiency(%)**	**pmbw Efficiency(%)**
RWS	1	1KB	5.671611	16.75573811	68.256	8.309322257	24.5483739
RWS	1	1MB	9.053908	16.16164552	68.256	13.26463315	23.6779851
RWS	1	10MB	10.849624	14.84214746	68.256	15.89548758	21.7448246
RWS	2	1KB	10.143139	27.83418639	68.256	14.86043571	40.7791057
RWS	2	1MB	9.990358	29.45914195	68.256	14.63660045	43.1597837
RWS	2	10MB	19.335467	28.46489784	68.256	28.32786422	41.7031438
RWS	4	1KB	10.272503	34.55994028	68.256	15.04996337	50.6328239
RWS	4	1MB	9.826079	34.27369587	68.256	14.39591977	50.213455
RWS	4	10MB	20.426348	30.64544943	68.256	29.92608415	44.8978103
RWR	1	1KB	1.920878	4.843576176	68.256	2.814225856	7.09619107
RWR	1	1MB	2.520235	0.452953134	68.256	3.692327414	0.66360926
RWR	1	10MB	3.845289	0.255613064	68.256	5.633627813	0.37449171
RWR	2	1KB	6.143139	9.262095201	68.256	9.000145042	13.5696425
RWR	2	1MB	8.99014	1.186063668	68.256	13.17120839	1.73766946
RWR	2	10MB	14.337416	0.584671468	68.256	21.00535631	0.85658619
RWR	4	1KB	8.270351	23.17289516	68.256	12.1166652	33.9499753
RWR	4	1MB	10.625441	2.868451133	68.256	15.56704319	4.20248935
RWR	4	10MB	18.189149	0.683363412	68.256	26.64842505	1.00117706



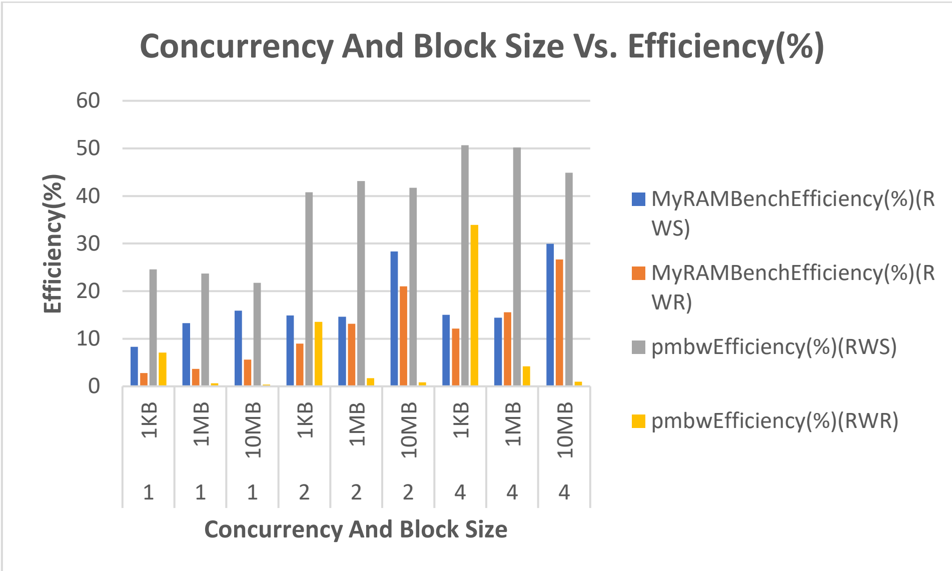
Throughput: RWR vs RWS

- The above graph shows comparison between RWR and RWS for various block sizes and threads for their throughputs.
- It clearly shows that **RWS** always has more throughput as compared to **RWR** throughput in any case provided.
- Also, as the block size increases, the throughput also increases.
- Considering the number of threads, the throughput increases as the number of threads increases for RWS and RWR also.



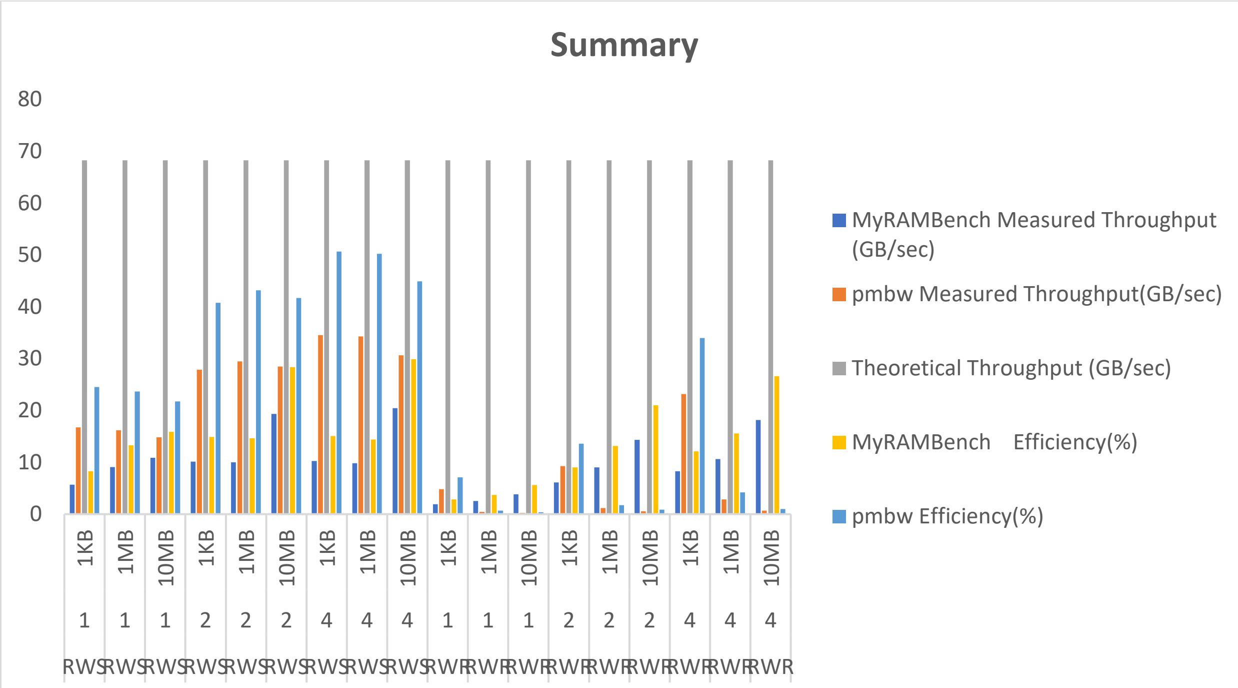
Throughput: MyRamBench vs Theoretical vs pmbw

- The above graph shows the comparison between MyRamBench vs pmbw vs Theoretical values for various threads and block sizes for RWS and RWR.
- We can easily make out that sequential execution has more read+write speed as compared to random one.



Efficiency: pmbw vs MyRamBench

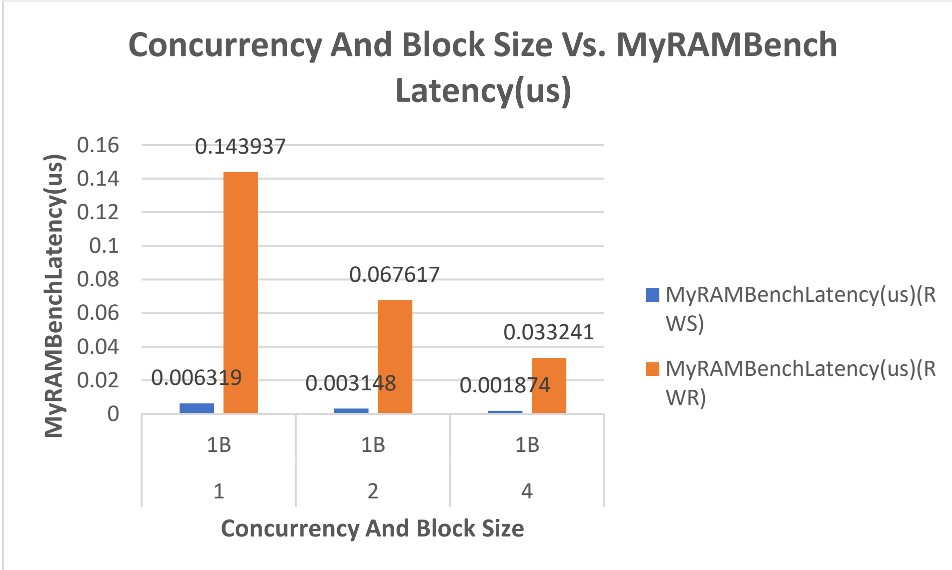
- The above graph shows comparison between pmbw RWS vs MyRamBench RWS and also pmbw RWR vs MyRamBench RWR and among RWR vs RWS for each of them.
- RWS is more efficient as compared to RWR as depicted above.
- The efficiency increases as block size increases.
- Also, as we increase thread, the efficiency increases for all of them.



<u>Summarizing the entire Memory operations.</u>

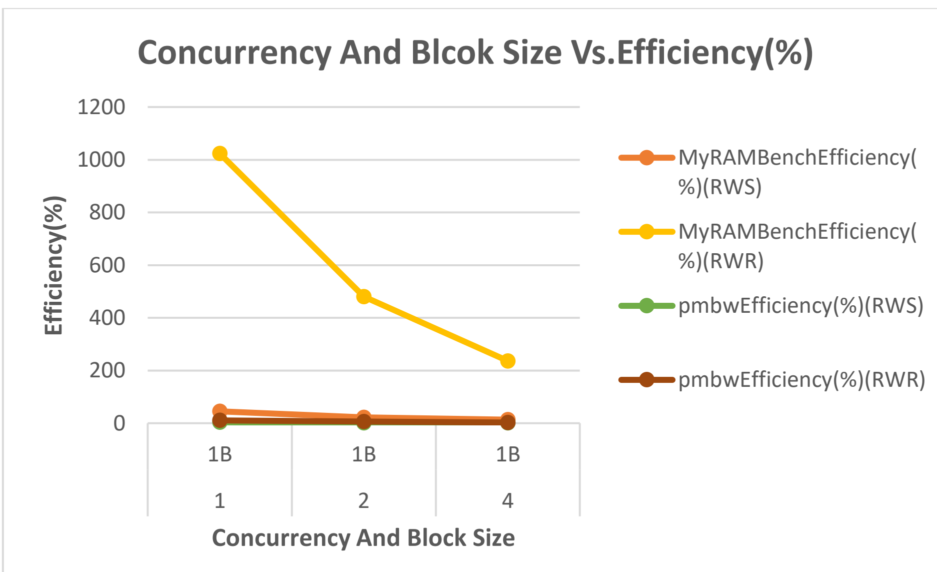
- Based on the observations above in tabular as well as visualization charts, we ca clearly state that RWS has higher throughput as compared to RWR.
- The throughput increases as we increase the block size from 1KB to 10MB.
- Also, if the concurrency is increased, the throughput increases to great extent.
- RWS has achieved higher efficiency, which is close to theoretical one.

Memory Latency							
Work-load	**Con-currency**	**Block Size**	**MyRAMBench Measured Latency(us)**	**pmbw Measured Latency (us)**	**Theoritical Latency (us)**	**MyRAMBench Efficiency (%)**	**pmbw Efficiency (%)**
RWS	1	1B	0.006319	0.000475662	0.01406	44.943101	3.3830867
RWS	2	1B	0.003148	0.000271911	0.01406	22.38975818	1.93393059
RWS	4	1B	0.001874	0.000210701	0.01406	13.32859175	1.49858542
RWR	1	1B	0.143937	0.00148148	0.01406	1023.733997	10.536843
RWR	2	1B	0.067617	0.000907748	0.01406	480.9174964	6.45624607
RWR	4	1B	0.033241	0.000353173	0.01406	236.4224751	2.51190038



Latency: RWS vs RWR

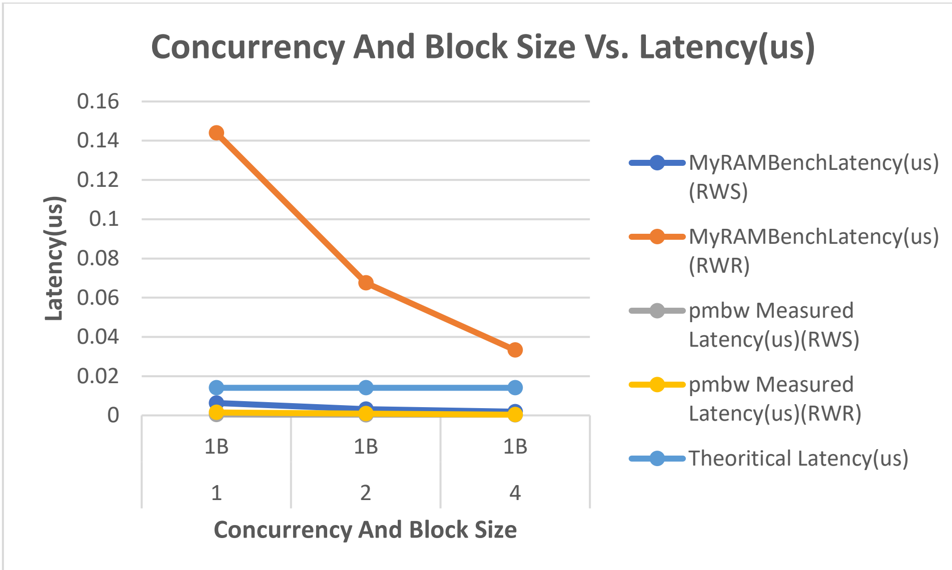
- The above graph depicts the latency comparison between RWS vs RWR for different threads.
- As we can clearly see that the latency is more for RWR as compared to RWS.
- The latency also decreases as we increase the concurrency.
-



Latency: MyRamBench vs pmbw

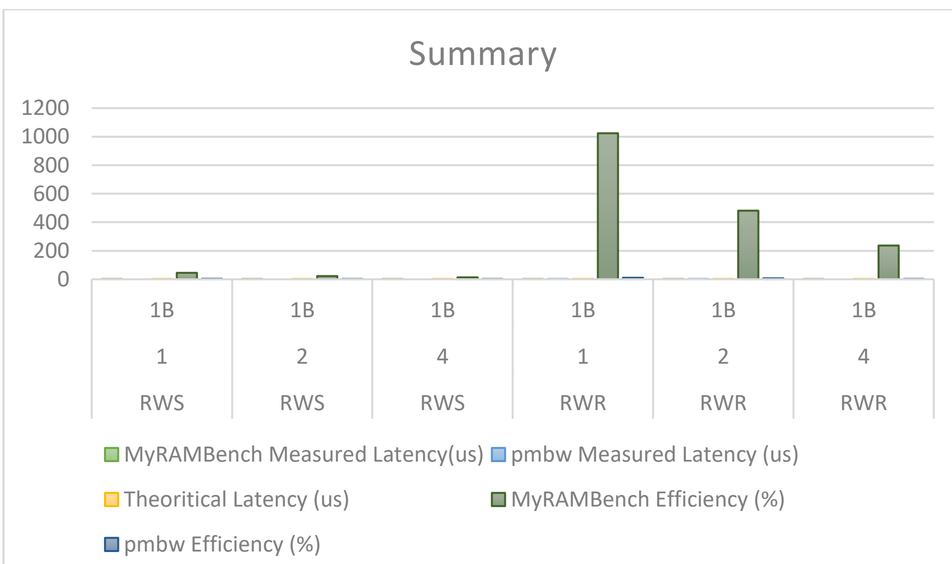
- The above graphs displays comparison between MyRamBench efficiency and pmbw efficiency.
- As we can see that RWR has more latency as compared to RWS for both pmbw as well as MyRamBenchEfficiency.

- MyRamBench has more latency as compared to pmbw for RWS as well as for RWR.



Latency: Theoretical vs pmbw vs MyRamBench

- Theoretical latency is constant for most of the cases.
- But in MyRamBench and pmbw, the latency decreases as we increase the number of threads.



Summary: Latency for RWS vs RWR

Conclusion:

- The latency is more for RWR as compared to RWS.
- Latency decreases as we increase the number of threads.

Possible Improvements:

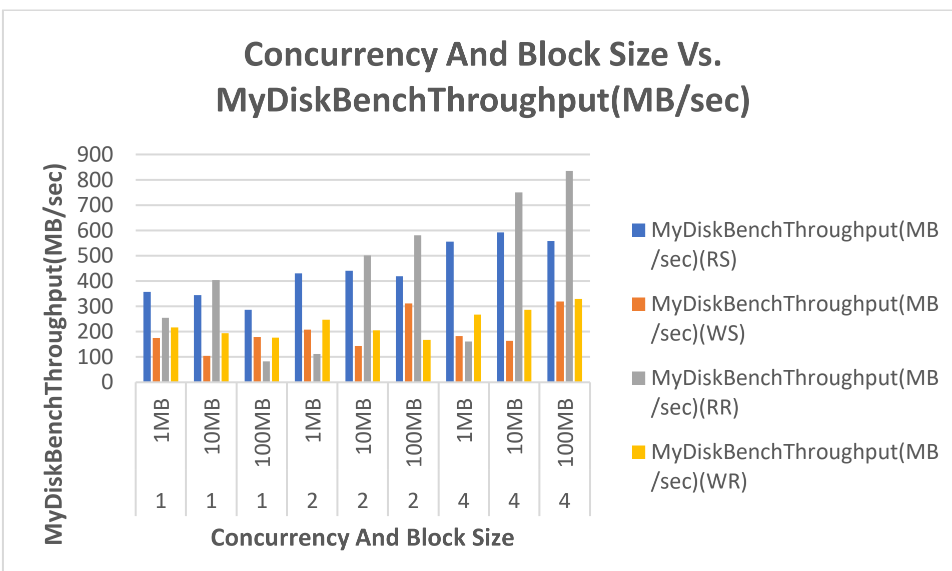
- Use of other functions instead of memcpy for finding the values and then each of those values can be compared.

3. **Disk**:
   - Performance Parameters that were considered are:
     - Size of data used: **10GB**
     - Type of operations:
       1. RS – Read Sequential.
       2. WS – Write Sequential.
       3. WR – Write Random.
       4. RR – Read Random.
     - Blocksize: 1MB, 10MB, 100MB
     - Concurrency: 1,2 and 4.
     - For latency:
       1. Data: 1KB
       2. Operations: 1 million
   - The following table shows the comparison of Throughput/efficiency when different parameters are passed as input.

Disk Throughput

Work-load	Con-currency	Block Size	MyDisk Bench Measured Throughput(MB/sec)	IOZoneMeasured Throughput (MB/sec)	Theoretical Throughput (MB/sec)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
RS	1	1MB	357.675026	313.18	540	66.23611593	57.996296
RS	1	10MB	344.8586	353.18	540	63.8627037	65.403704
RS	1	100MB	286.549619	303.18	540	53.06474426	56.144444
RS	2	1MB	430.297117	626.36	1080	39.84232565	57.996296
RS	2	10MB	440.020285	706.36	1080	40.74261898	65.403704
RS	2	100MB	419.57819	766.36	1080	38.84983241	70.959259
RS	4	1MB	555.173874	1252.72	2160	25.70249417	57.996296
RS	4	10MB	592.034449	1332.72	2160	27.40900227	61.7
RS	4	100MB	558.697172	1532.72	2160	25.86560981	70.959259
WS	1	1MB	174.804174	310.52	410	42.63516439	75.736585
WS	1	10MB	104.608384	325.52	410	25.51424	79.395122
WS	1	100MB	178.503212	301.52	410	43.53736878	73.541463
WS	2	1MB	207.73182	621.04	820	25.33314878	75.736585
WS	2	10MB	143.231427	701.04	820	17.4672472	85.492683
WS	2	100MB	311.662903	561.04	820	38.0076711	68.419512
WS	4	1MB	182.341382	1242.08	1640	11.11837695	75.736585

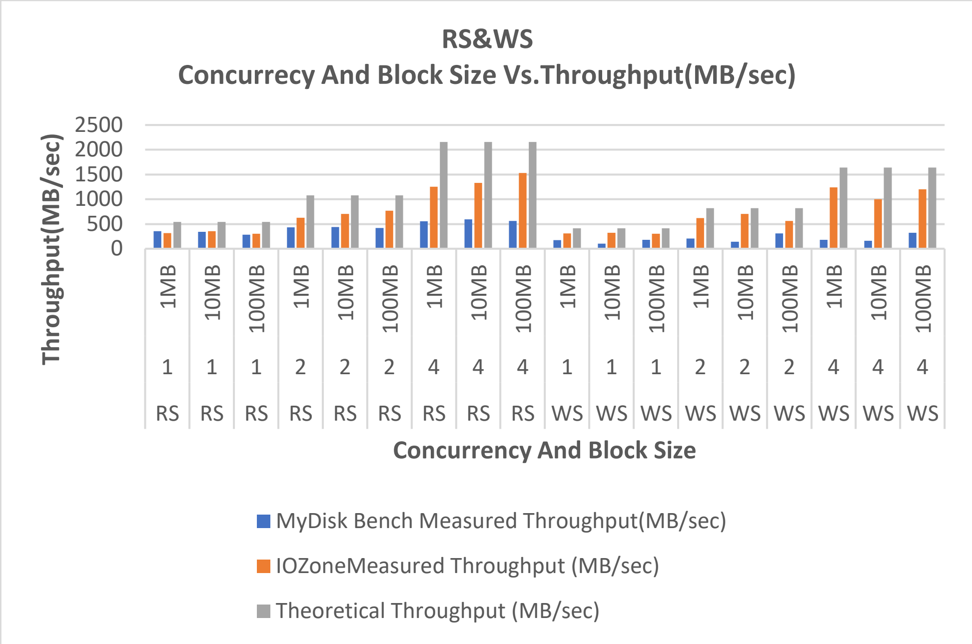
WS	4	10MB	163.669314	1002.08	1640	9.97983622	61.102439
WS	4	100MB	319.108477	1202.08	1640	19.45783396	73.297561
RR	1	1MB	255.134688	284.25	372	68.58459355	76.41129
RR	1	10MB	404.119856	219.25	372	108.6343699	58.938172
RR	1	100MB	83.021784	184.25	372	22.31768387	49.52957
RR	2	1MB	112.332482	598.5	744	15.09845188	80.443548
RR	2	10MB	501.069156	468.5	744	67.34800484	62.97043
RR	2	100MB	580.715709	588.5	744	78.05318669	79.099462
RR	4	1MB	160.807581	1137	1488	10.80696109	76.41129
RR	4	10MB	750.315658	1057	1488	50.42443938	71.034946
RR	4	100MB	835.204819	1161	1488	56.12935612	78.024194
WR	1	1MB	216.29527	185.63	172	125.753064	107.92442
WR	1	10MB	193.719832	195.63	172	112.6278093	113.73837
WR	1	100MB	176.218729	105.63	172	102.4527494	61.412791
WR	2	1MB	247.095913	371.26	344	71.83020727	107.92442
WR	2	10MB	205.80559	351.26	344	59.8272064	102.11047
WR	2	100MB	167.339458	251.26	344	48.64519128	73.040698
WR	4	1MB	267.018973	410.469	688	38.81089724	59.661192
WR	4	10MB	285.84357	562.52	688	41.54703052	81.761628
WR	4	100MB	329.19745	540.458	688	47.84846657	78.554942



Throughput: RS vs RR vs WR vs WS

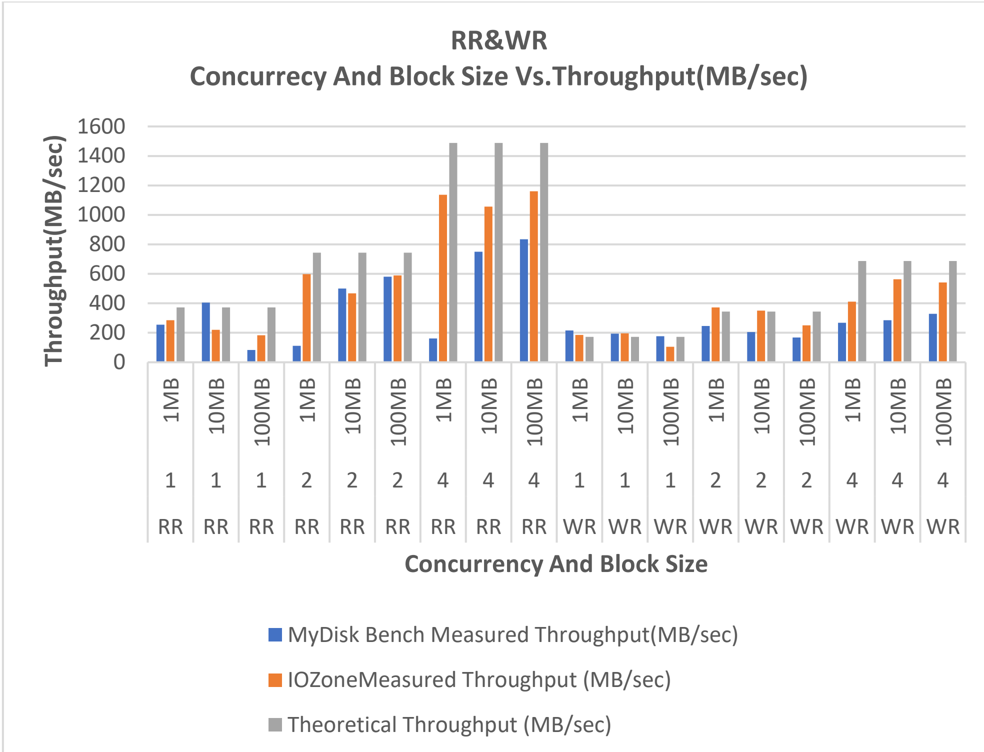
- It shows the comparison between values of MyDiskBench throughput for RS, RR, WS and WR.
- As we see, RS has the maximum throughput among all of them.

- As the thread increases the throughput also increases.
- As the block size increases the throughput also increases.



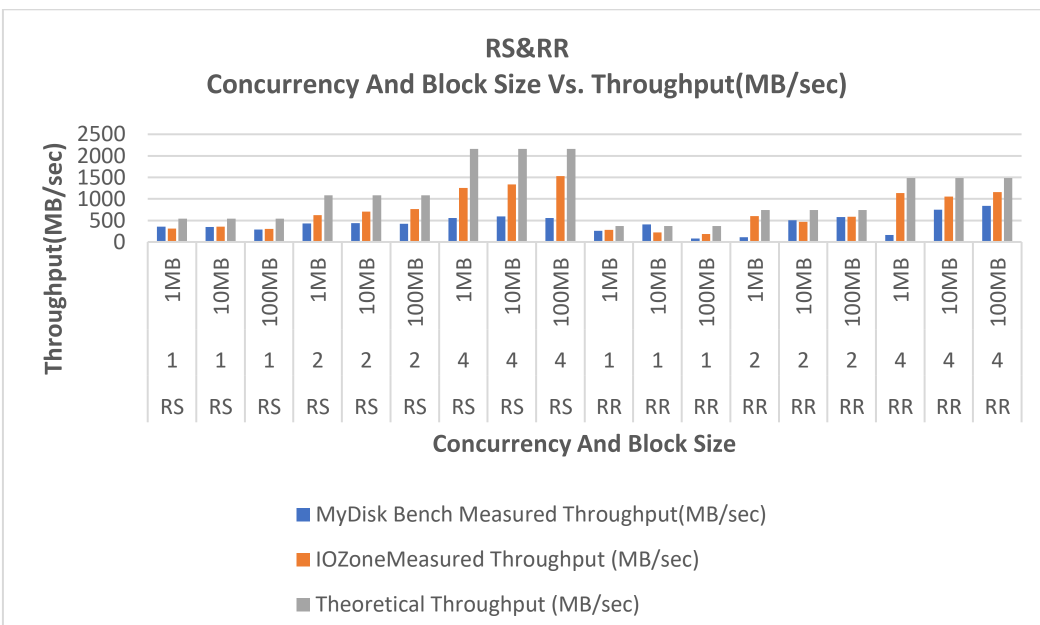
Throughput: RS vs WS

- It shows comparison between RS vs WS for MyDiskBench, IOZONE and Theoretical.
- We can clearly observe that RS performs well as compared to WS.
- As the block size increases, both of their throughput increases.
- As the thread count increases, throughput increases for all of them.



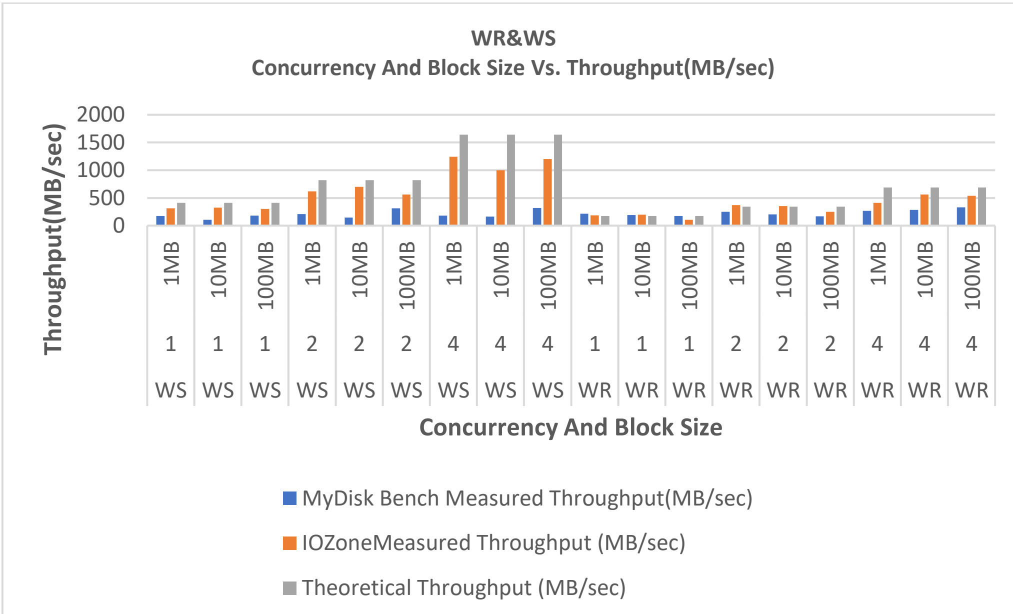
Throughput :RR vs WR

- It shows comparison between RR vs WR for MyDiskBench, IOZONE and Theoretical.
- We can clearly observe that RR performs well as compared to WR.
-



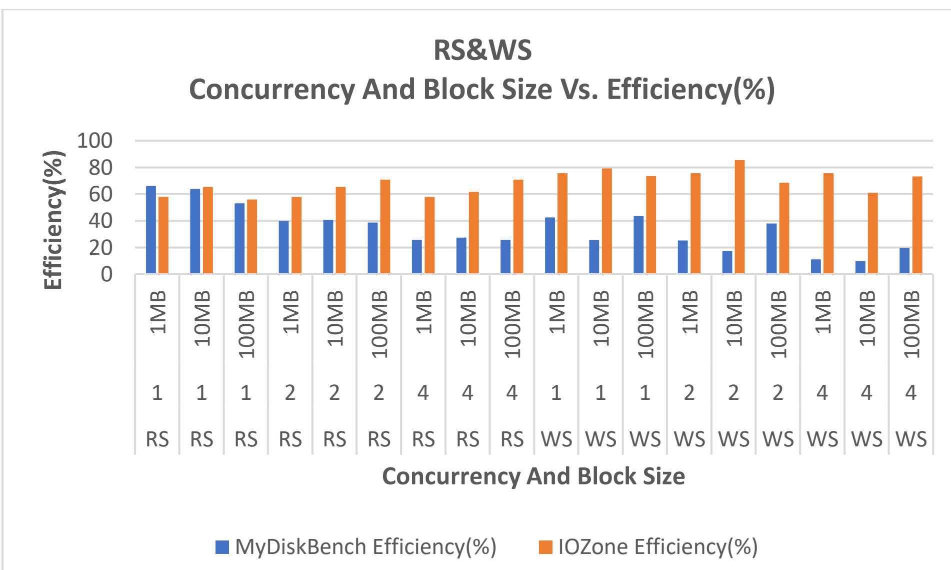
Throughput : RS vs RR

- As compared above, sequential read has more throughput as compared to random read operation.
- It is measured in terms of MyDiskBench, IOZONE and Theoretical.
-



Throughput : WR vs WS

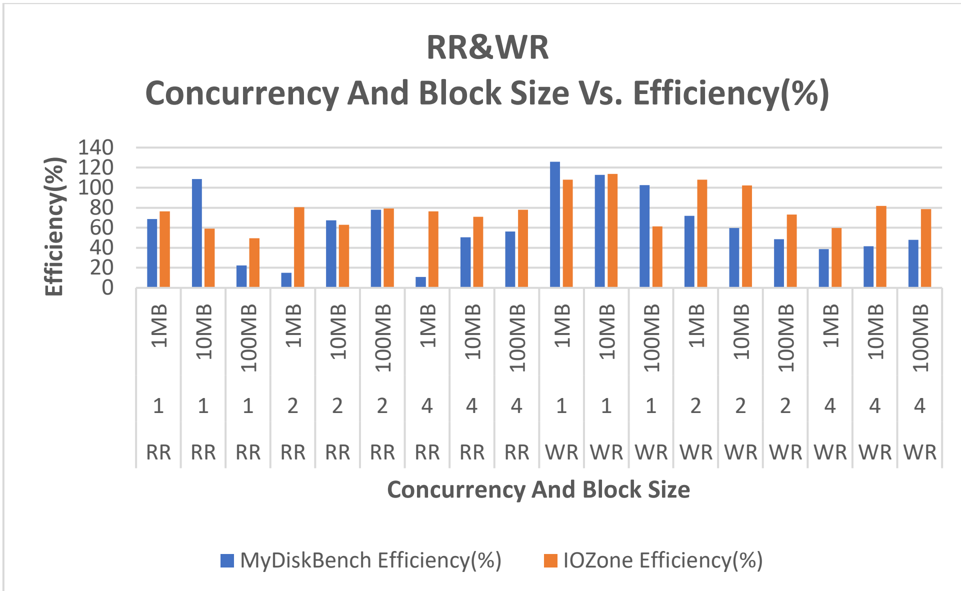
- This graph shows visualization difference between WR and WS for the values of MyDiskBench, IOZONE and Theoretical.
- Sequential write has more throughput as compared to random write.



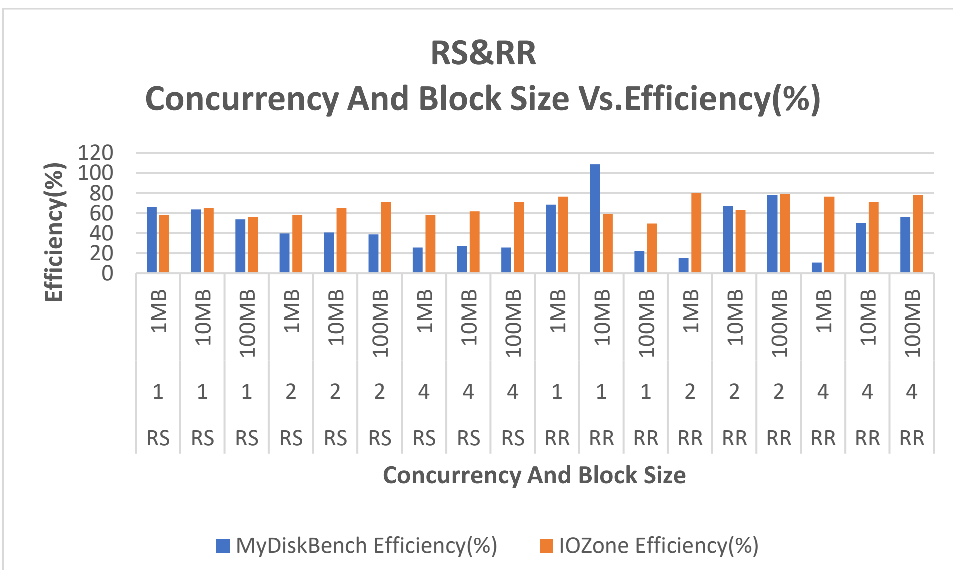
Efficiency: RSvsWS

- The graph shows efficiency between sequential read and sequential write.

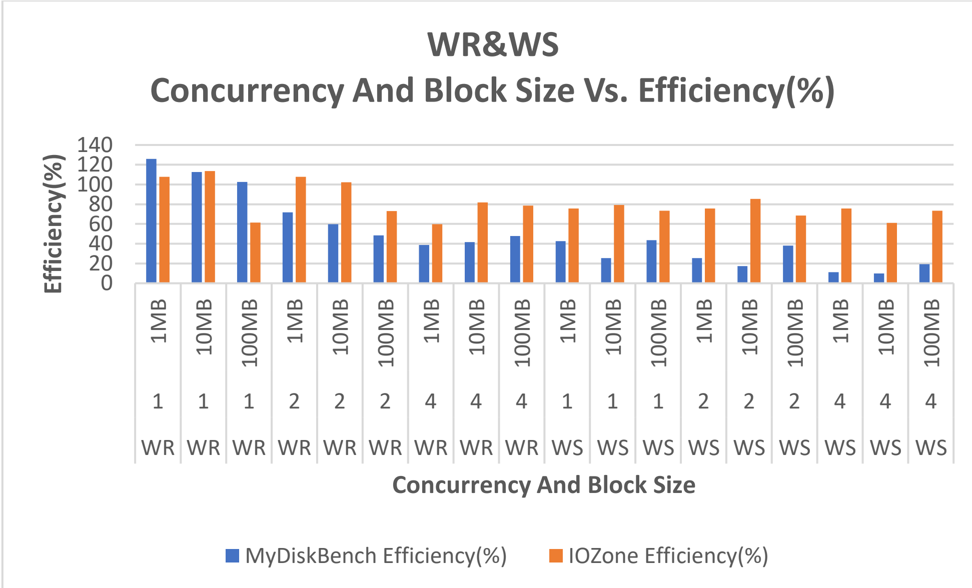
- The efficiency of sequential read is always greater as compared to sequential write.
- MyDiskBench efficiency is compared to IOZONE efficiency for sequential read and write.



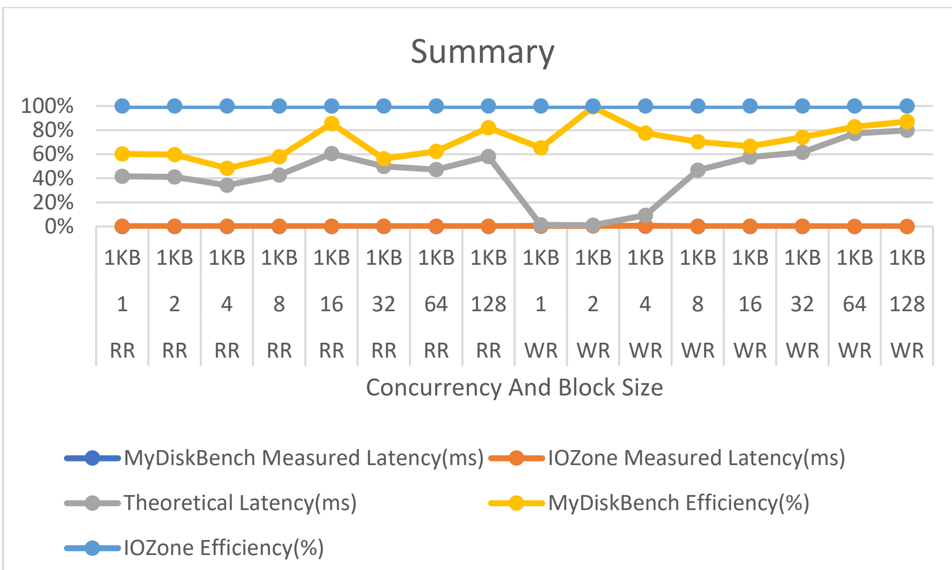
<u>Efficiency: RRvsWR</u>



<u>Efficiency: RSvsRR</u>



Efficiency: WRvsWS

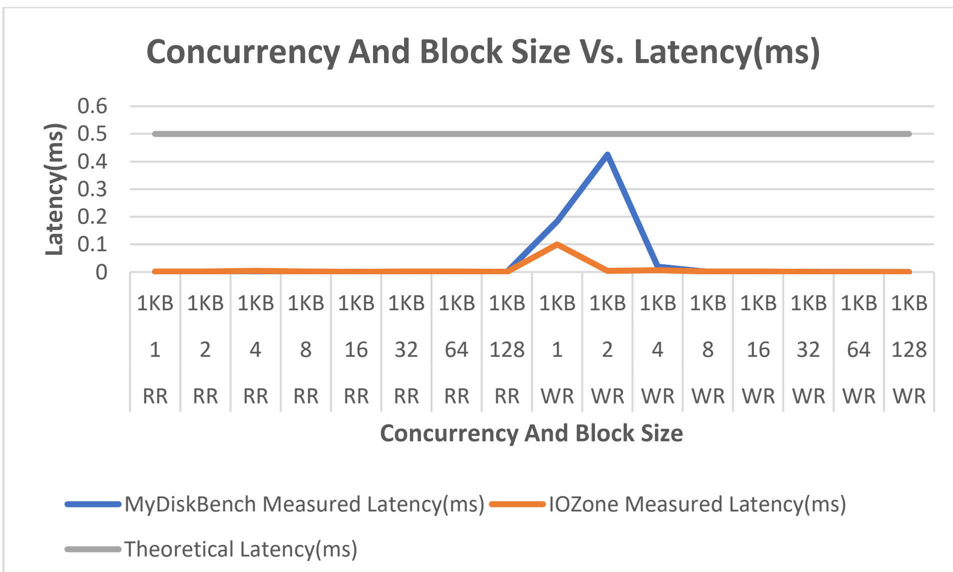


Summary

-Disk Latency:

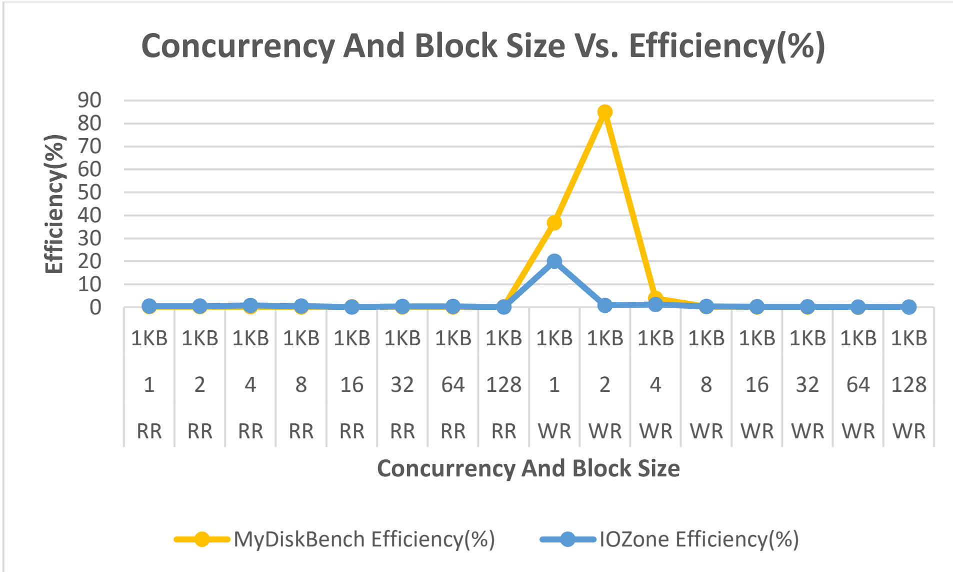
Disk Latency(Measured in ms)							
Work-load	Con-currency	Block Size	MyDiskBench Measured Latency(ms)	IOZone Measured Latency(ms)	Theoretical Latency(ms)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)

RR	1	1KB	0.00113	0.0024	0.5	0.226	0.48
RR	2	1KB	0.001127	0.00247	0.5	0.2254	0.494
RR	4	1KB	0.00104	0.0038	0.5	0.208	0.76
RR	8	1KB	0.000888	0.00249	0.5	0.1776	0.498
RR	16	1KB	0.001041	0.0006	0.5	0.2082	0.12
RR	32	1KB	0.000321	0.0022	0.5	0.0642	0.44
RR	64	1KB	0.000813	0.002	0.5	0.1626	0.4
RR	128	1KB	0.001031	0.00078	0.5	0.2062	0.156
WR	1	1KB	0.184161	0.1	0.5	36.8322	20
WR	2	1KB	0.425145	0.0038	0.5	85.029	0.76
WR	4	1KB	0.019427	0.0064	0.5	3.8854	1.28
WR	8	1KB	0.001269	0.0016	0.5	0.2538	0.32
WR	16	1KB	0.000402	0.001456	0.5	0.0804	0.2912
WR	32	1KB	0.000514	0.00106	0.5	0.1028	0.212
WR	64	1KB	0.000179	0.000556	0.5	0.0358	0.1112
WR	128	1KB	0.000225	0.000409	0.5	0.045	0.0818



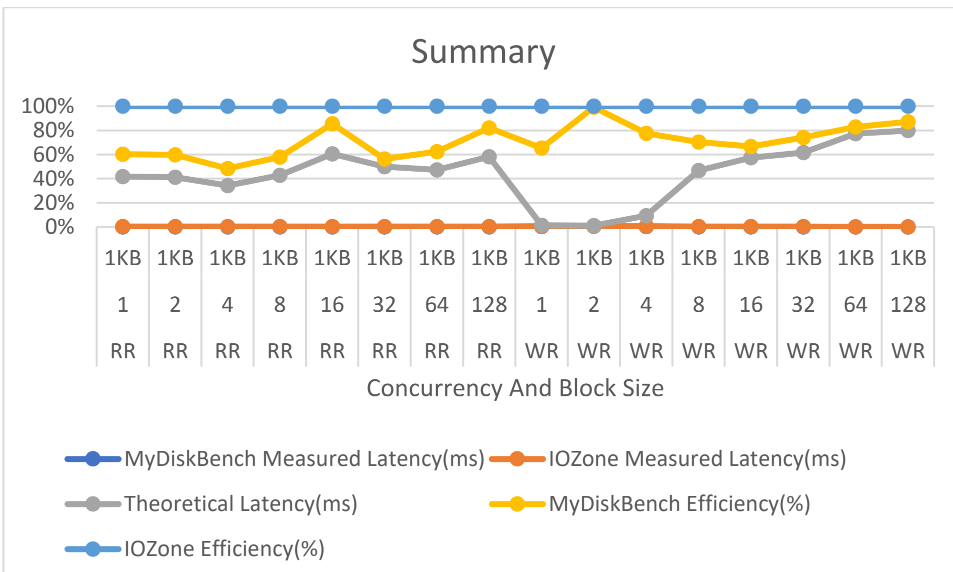
Disk Latency: RR vs WR

- It shows comparison between various threads for Random read and random write.
- As the thread count increases, the latency decreases.
- The Random Read has less latency as compared to random write.
- It shows comparison between MyDiskBench, iozone and Theoretical latencies.



Efficiency: RR vs WR – MyDiskBench vs IOZONE

- It shows efficiency change between RR and WR for MyDiskBench and IOZONE.

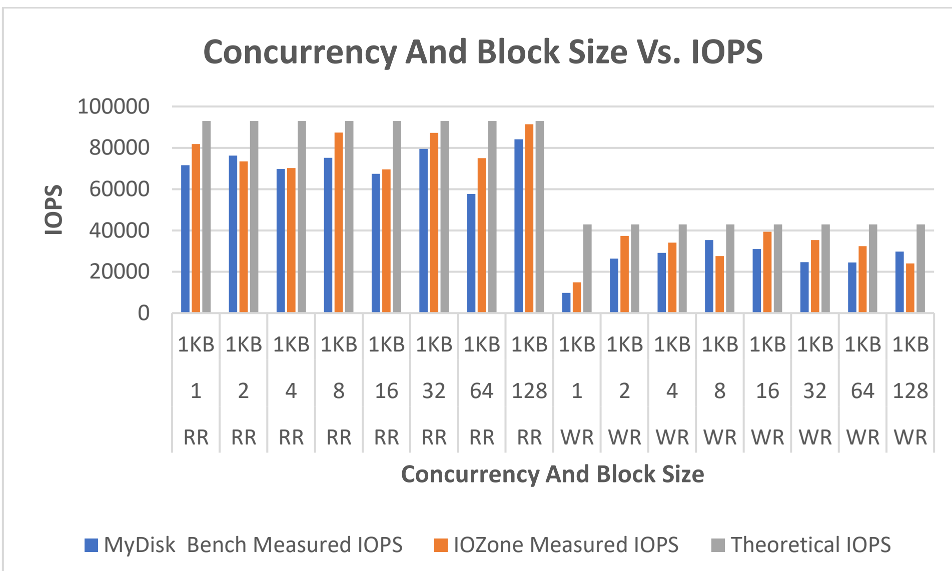


Summary

Disk Latency (Measured in IOPS)

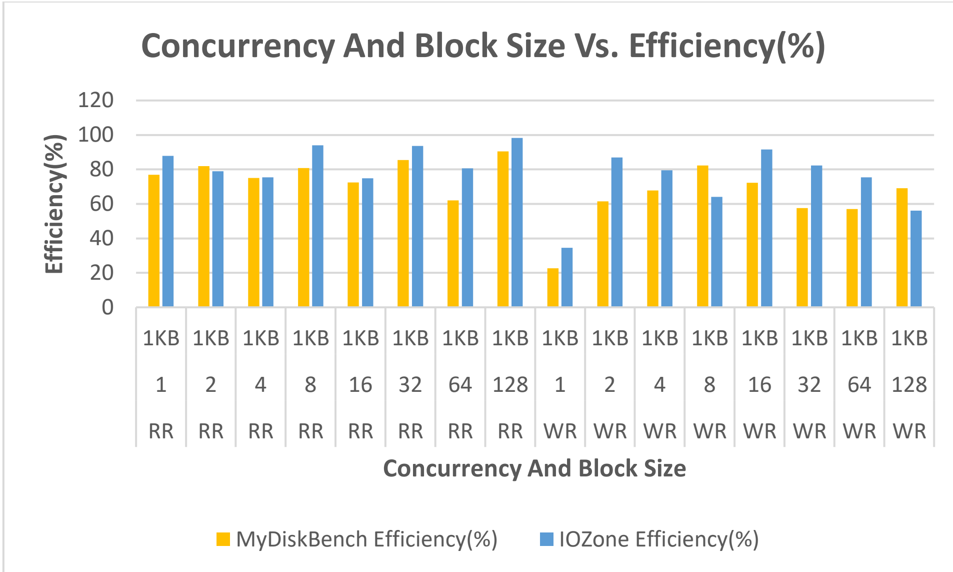
Work-load	Con-currency	Block Size	MyDisk Bench Measured IOPS	IOZone Measured IOPS	Theoretical IOPS	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
RR	1	1KB	71546	81790	93000	76.9311828	87.946237

RR	2	1KB	76182	73410	93000	81.91612903	78.935484
RR	4	1KB	69710	70182	93000	74.95698925	75.464516
RR	8	1KB	75149	87410	93000	80.80537634	93.989247
RR	16	1KB	67410	69621	93000	72.48387097	74.86129
RR	32	1KB	79410	87149	93000	85.38709677	93.708602
RR	64	1KB	57610	74954	93000	61.94623656	80.595699
RR	128	1KB	84105	91405	93000	90.43548387	98.284946
WR	1	1KB	9715	14840	43000	22.59302326	34.511628
WR	2	1KB	26419	37410	43000	61.43953488	87
WR	4	1KB	29183	34150	43000	67.86744186	79.418605
WR	8	1KB	35410	27541	43000	82.34883721	64.048837
WR	16	1KB	31050	39410	43000	72.20930233	91.651163
WR	32	1KB	24720	35410	43000	57.48837209	82.348837
WR	64	1KB	24510	32450	43000	57	75.465116
WR	128	1KB	29710	24105	43000	69.09302326	56.05814



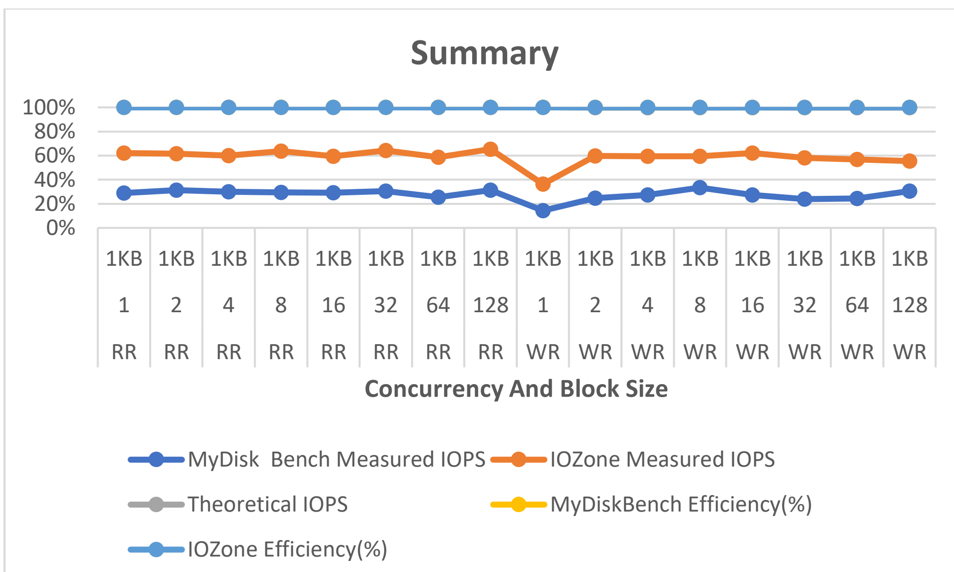
IOPS: MyDiskBench vs IOZONE vs Theoretical

- The graph shows comparison between MyDiskBench, iozone and Theoretical values.
- RR has more IOPS as compared to WR.
- As the thread increases, the iops also increases.



Efficiency: MyDiskBench vs Iozone

- Random read has more efficiency as compared to random write.
- As thread increases, the efficiency also increases.



Summary: IOPS

Conclusion:

- Throughput increases as the concurrency increases.
- Throughput increases as the block size increases.
- Latency Decreases as the concurrency increases.

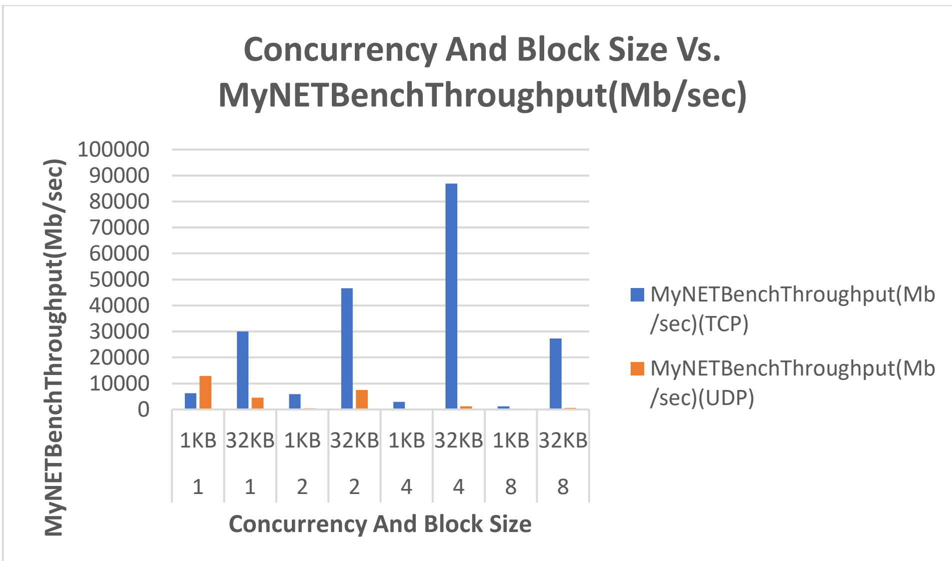
- Throughput is more for sequential operations as compared to random operations.
- Latency is less for sequential operations as compared to random operations.

4.**Network**:

- Performance Parameters that were considered are:
  - Size of data used: **1GB for 100x times.**
  - Type of Protocol:
    1. TCP
    2. UDP
  - Blocksize: 1B, 1KB, 32KB
  - Concurrency: 1,2,4 and 8.
  - For latency:
    1. Data: 1B
    2. Operations: **1 million**
- The following table shows the comparison of Throughput/efficiency when different parameters are passed as input.

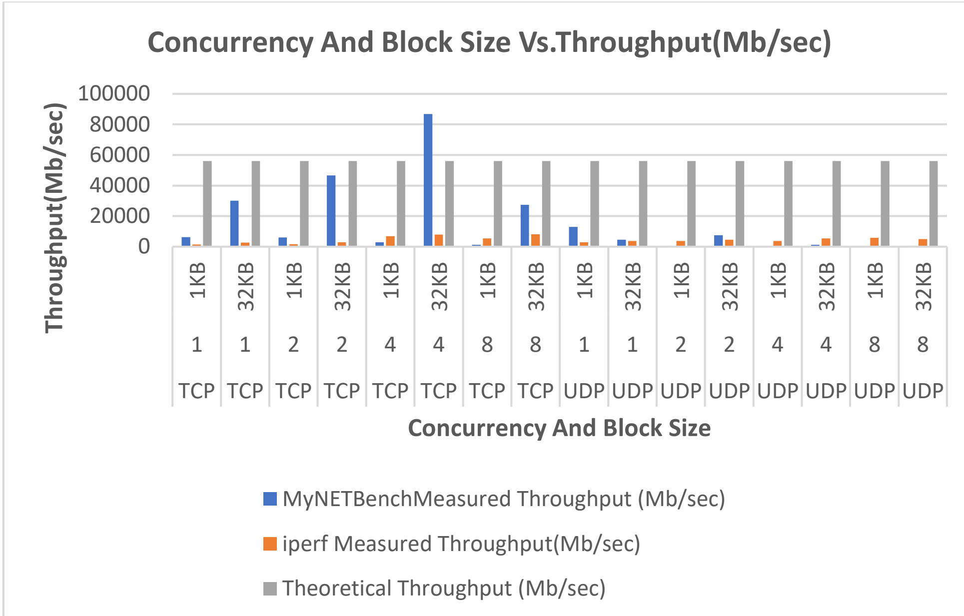
Network Throughput							
Protocol	**Con-currency**	**Block Size**	**MyNETBenchMeasured Throughput (Mb/sec)**	**iperf Measured Throughput(Mb/sec)**	**Theoretical Throughput (Mb/sec)**	**MyNETBench Efficiency (%)**	**iperf Efficiency (%)**
TCP	1	1KB	6199.673064	1485.25	56000	11.07084476	2.6522321
TCP	1	32KB	30029.32551	2756.25	56000	53.62379556	4.921875
TCP	2	1KB	5930.773853	1706.13	56000	10.59066759	3.0466607
TCP	2	32KB	46593.10659	2910.12	56000	83.20197605	5.1966429
TCP	4	1KB	2878.830475	6941.24	56000	5.140768706	12.395071
TCP	4	32KB	86871.68611	7970.15	56000	155.1280109	14.232411
TCP	8	1KB	1249.532874	5418.12	56000	2.231308704	9.6752143
TCP	8	32KB	27338.56166	8214.19	56000	48.8188601	14.668196
UDP	1	1KB	12905.36644	2941.18	56000	23.04529721	5.2521071
UDP	1	32KB	4488.459996	3719.14	56000	8.015107135	6.6413214

UDP	2	1KB	511.3977369	3641.25	56000	0.913210244	6.5022321
UDP	2	32KB	7551.463429	4648.49	56000	13.48475612	8.300875
UDP	4	1KB	285.4062604	3751.04	56000	0.509654036	6.6982857
UDP	4	32KB	1177.613766	5419.27	56000	2.102881725	9.6772679
UDP	8	1KB	133.9018594	5717.18	56000	0.239110463	10.20925
UDP	8	32KB	534.7648972	4971.13	56000	0.954937317	8.8770179



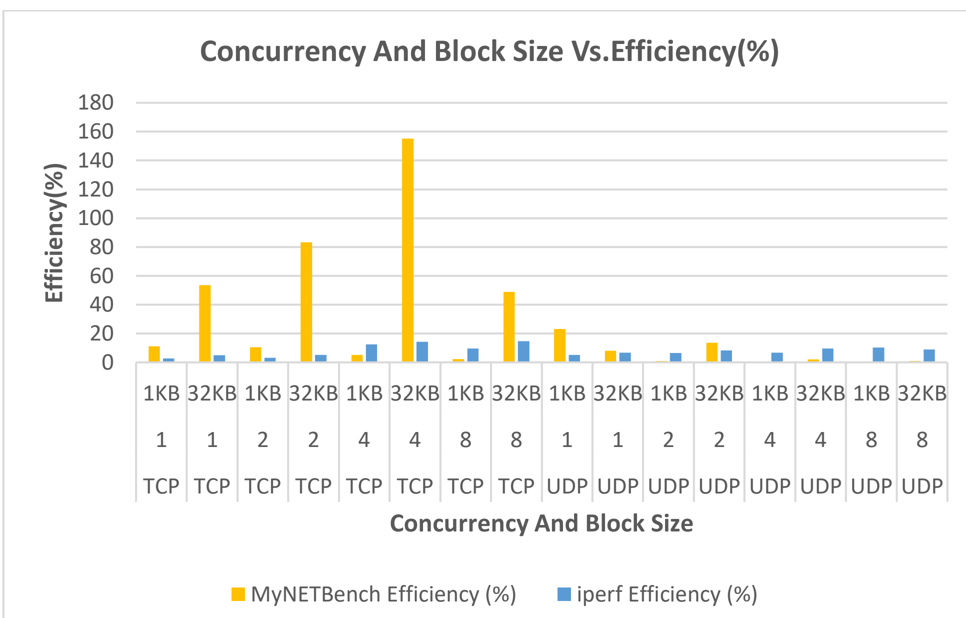
Throughput: TCP vs UDP

- The graph represents the comparison between TCP and UDP's throughput values for various block sizes and number of threads.
- As the threads increase, the throughput increases.
- Also, as the number of threads increases, the throughput increases.
- UDP has more throughput as compared to TCP in most of the scenarios.



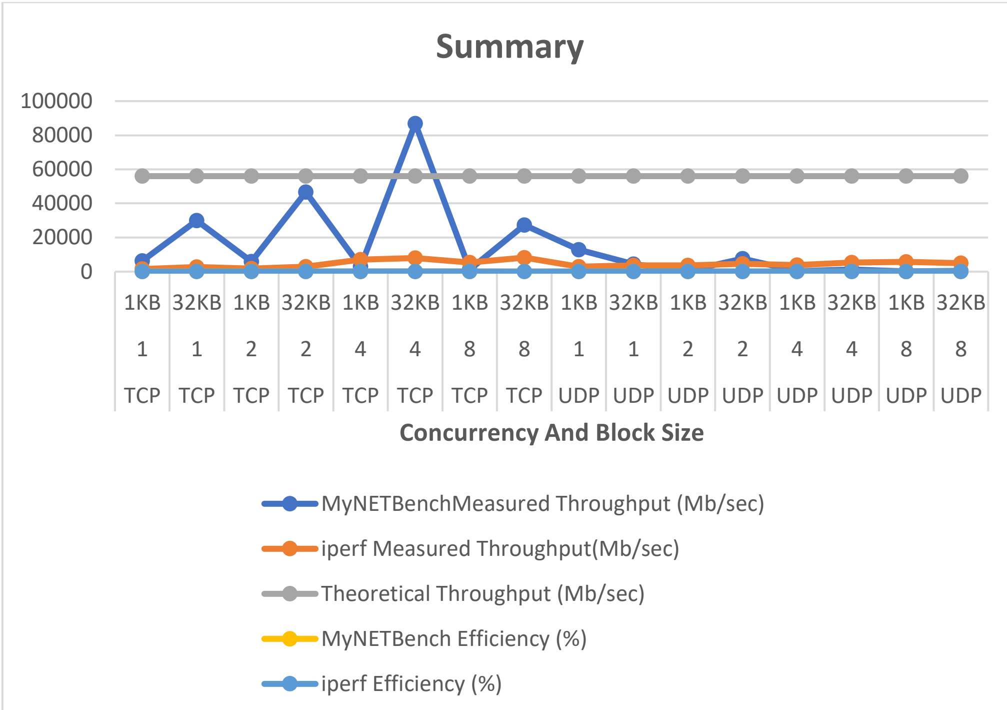
<u>MyNetBench vs Theoretical vs iperf : TCP vs UDP</u>

- This graph shows comparison between TCP and UDP.
- This graph also shows as thread increases the throughput increases.
- Also, as the packet size increases the throughput increases.
- TCP has lower throughput as compared to UDP because of the latency in establishing connection and sending an acknowledgement.



<u>Efficiency: MyNetBench vs iperf : TCP vs UDP</u>

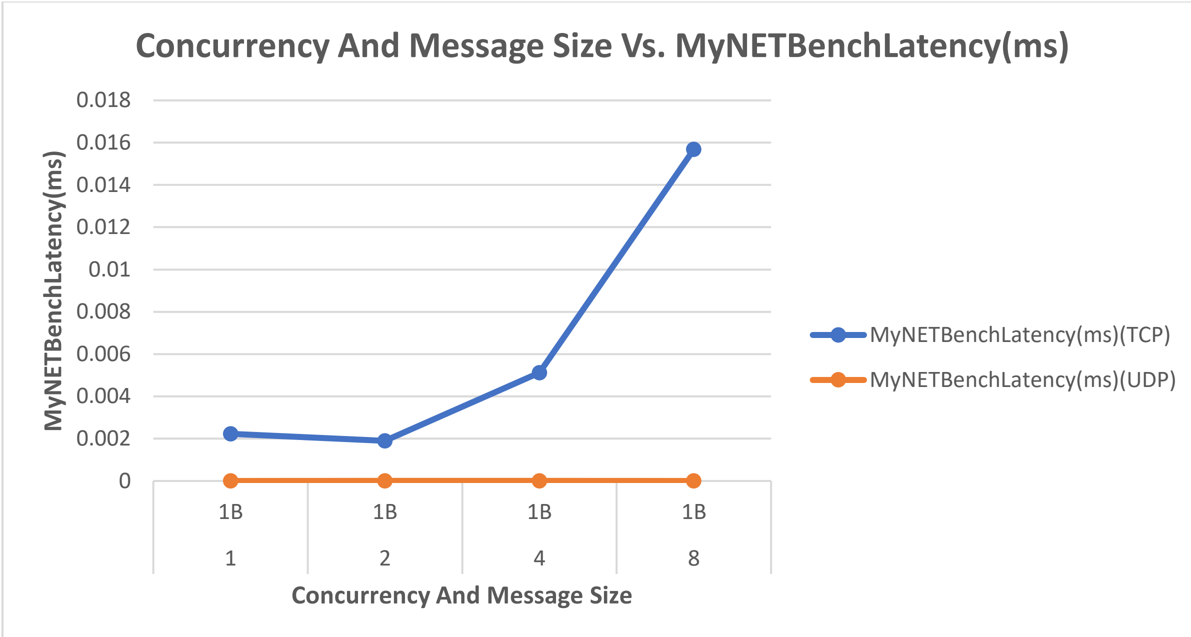
- The graph shows comparison between TCP and UDP for different block sizes and different threads.



Summary

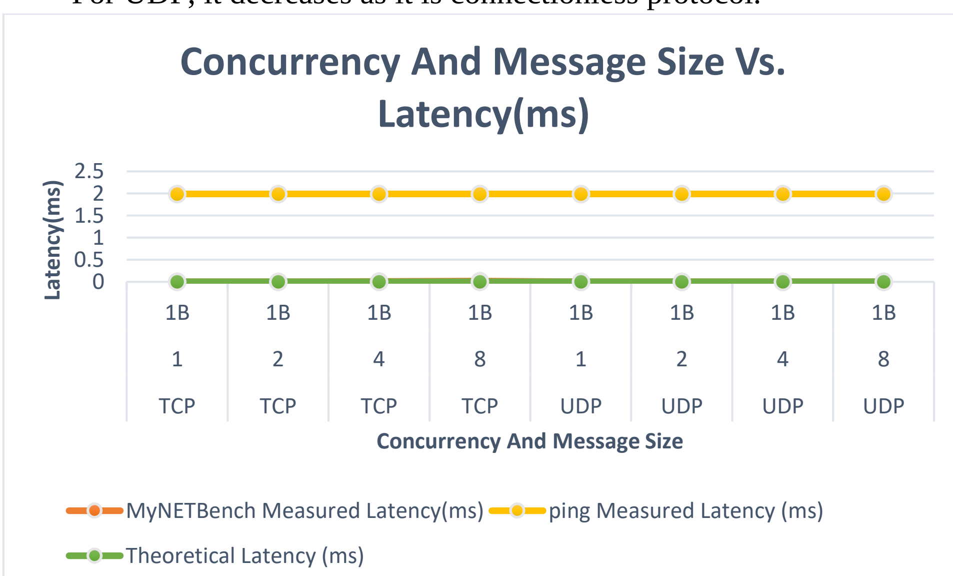
Network Latency:

Network Latency							
Protocol	**Con-currency**	**Message Size**	**MyNETBench Measured Latency(ms)**	**ping Measured Latency (ms)**	**Theoretical Latency (ms)**	**MyNETBench Efficiency(%)**	**iperf Efficicency(%)**
TCP	1	1B	0.002236106	1.99	0.0007	319.4436431	284285.7143
TCP	2	1B	0.001898035	1.99	0.0007	271.1479153	284285.7143
TCP	4	1B	0.005125068	1.99	0.0007	732.1525897	284285.7143
TCP	8	1B	0.01568161	1.99	0.0007	2240.22993	284285.7143
UDP	1	1B	2.08E-07	1.99	0.0007	0.02969	284285.7143
UDP	2	1B	1.69E-07	1.99	0.0007	0.0242	284285.7143
UDP	4	1B	1.30E-07	1.99	0.0007	0.018601429	284285.7143
UDP	8	1B	8.95E-08	1.99	0.0007	0.012784286	284285.7143



Latency: TCP vs UDP

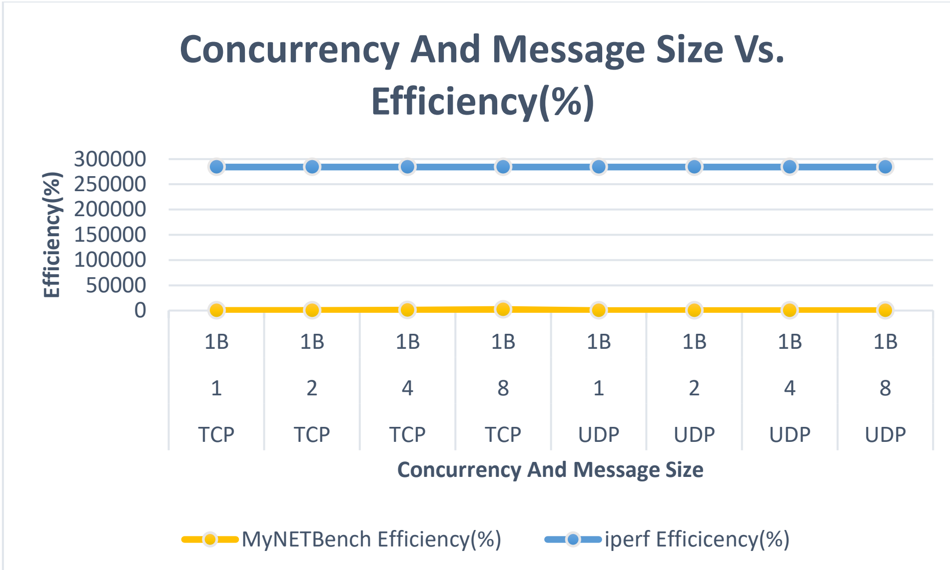
- This graph shows relation between TCP and UDP for various concurrency levels.
- TCP has always more latency as compared to UDP.
- For TCP, latency increases with number of threads.
- For UDP, it decreases as it is connectionless protocol.



Latency: MyNetBench vs ping vs Theoretical

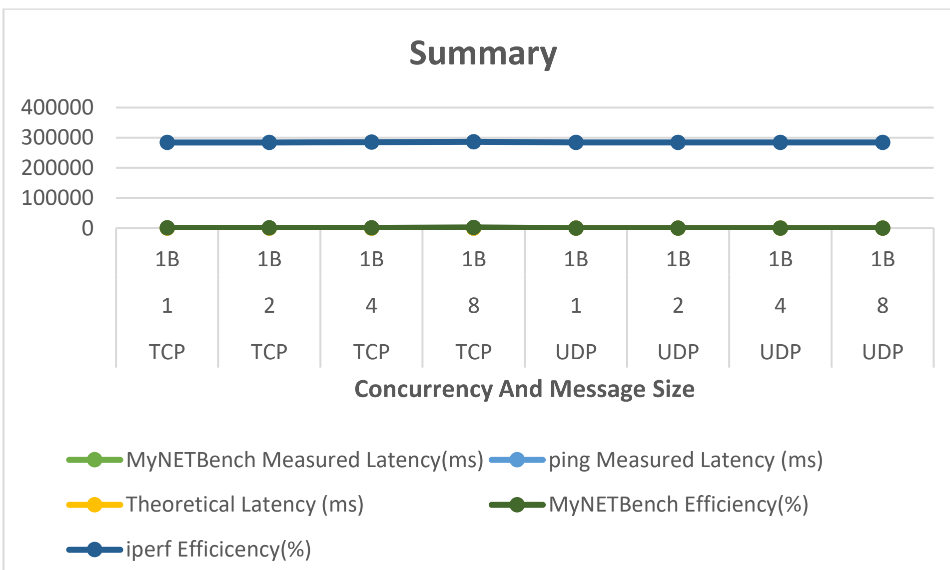
- This graph represents the difference in latency values of ping, theoretical and MyNetBench.
- MyNetBench has almost similar values as of Theoretical.

- Ping has bit more latency comparatively due to acknowledgement sent.



Efficiency: iperf vs MyNetBench

- This shows the efficiency difference between ping and MyNetBench with respect to theoretical values.



Summary

Conclusion:

- TCP is slower as compared to UDP.
- Throughput for TCP is more than UDP.

- As thread increases the throughput also increases.
- As packet size increases, network throughput also increases.
- Latency decreases as the concurrency increases.
- For more packet size, latency is smaller comparatively.