CS 553 - Cloud Computing, Spring 2018

PA 1 - Performance Report

Benchmarking

Saumil Pareshbhai Ajmera | A20397303 | sajmera4@hawk.iit.edu

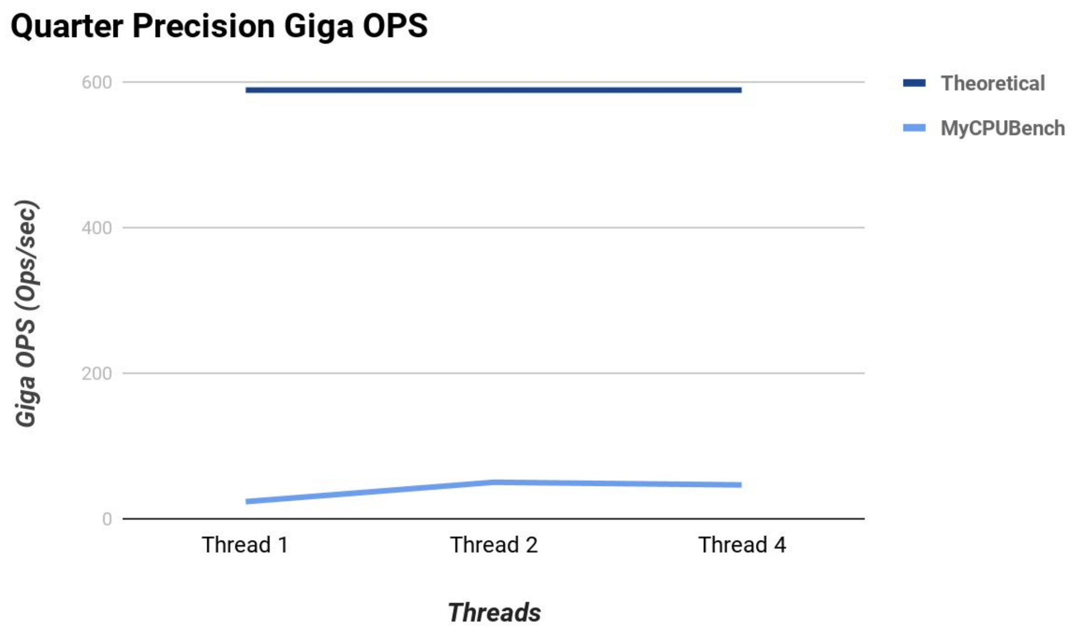
1. Processor Benchmarking

This benchmark measures the GigaOPS of a processor of chameleon compute node for combinations of data types and concurrency for 1 trillion arithmetic operations done in C language.

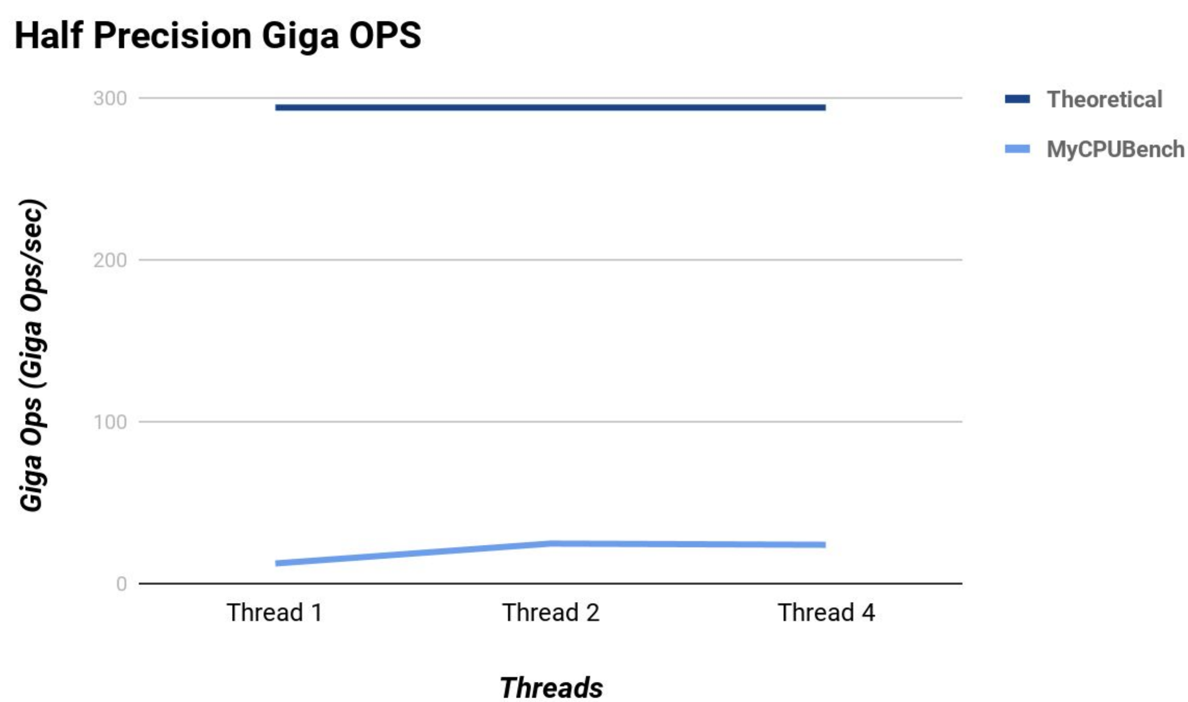
Type - Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz , Intel Haswell family

Description - I have used FMA vector instructions for double and single precision data types and AVX2 for half and quarter data types which executes respective operations per cycle which helps in faster computation and reduction in loop overhead.

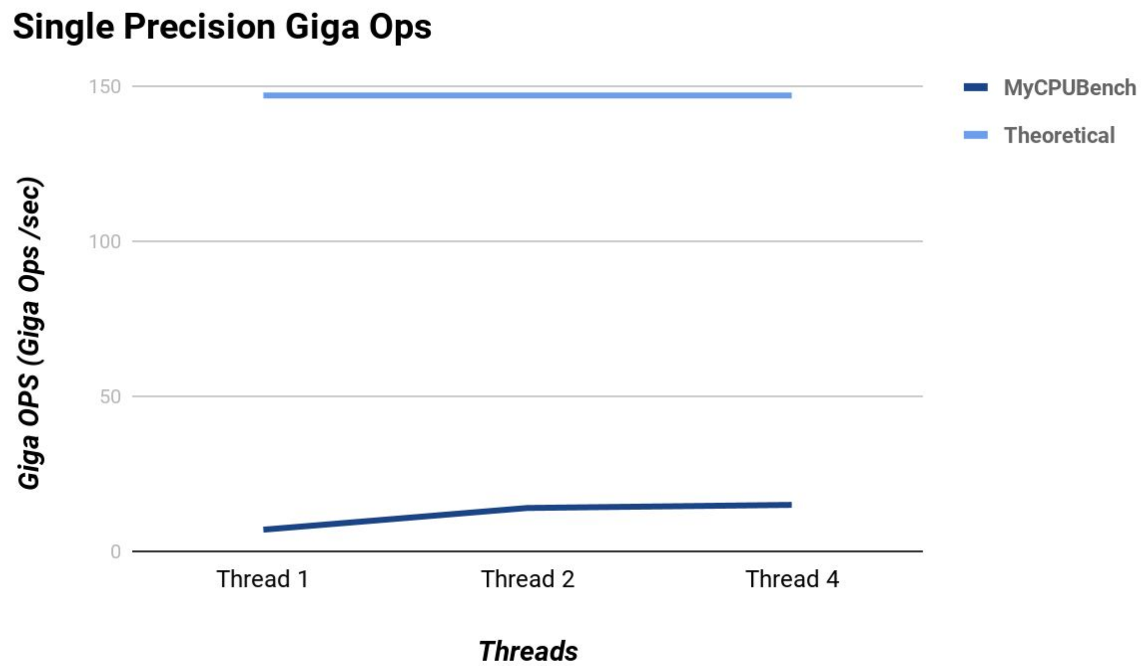
Work load	Concur rency	MyCPUBench (Giga OPS)	HPL Measured	Theore tical (Giga OPS)	MyCPUBench Efficiency (%)	HPL Efficiency (%)
QP	1	23.56	N/A	588.8	4.001	N/A
QP	2	50.18	N/A	588.8	8.52	N/A
QP	4	46.56	N/A	588.8	7.90	N/A
HP	1	12.44	N/A	294.4	4.22	N/A
HP	2	24.72	N/A	294.4	8.4	N/A
HP	4	23.96	N/A	294.4	8.1	N/A
SP	1	7.6	N/A	147.2	5.1	N/A
SP	2	14.77	N/A	147.2	10.0	N/A
SP	4	15.44	N/A	147.2	10.4	N/A
DP	1	3.72	36.43	73.6	5	49.4
DP	2	7.7	62.75	73.6	10.5	85.2
DP	4	7.3	71.10	73.6	9.9	98.9



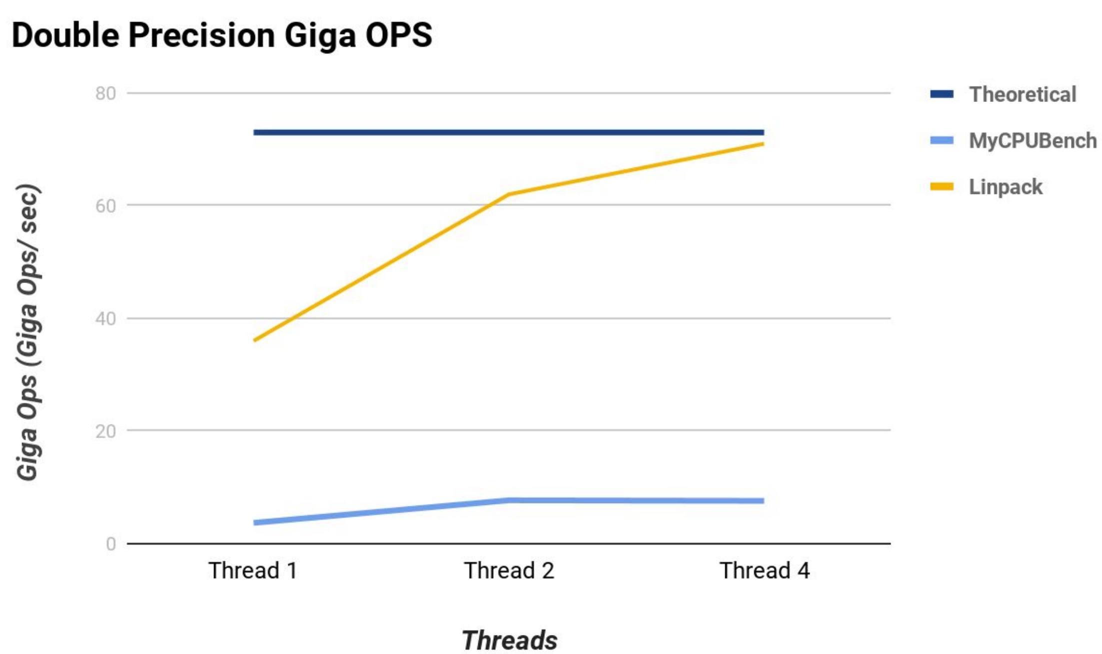
Above graph plots Giga Ops of quarter precision vs Threads for Theoretical and MyCPUBench



Above graph plots Giga Ops of half precision vs Threads for Theoretical and MyCPUBench



Above graph plots Giga Ops of single precision vs Threads for Theoretical and MyCPUBench



Above graph plots Giga Ops of double precision vs Threads for Theoretical an MyCPUBench

Observation - A the number of thread increases the CPU performance increases for both type of operations (double and integers). Here we can also show that the if number of thread is higher than the number of thread available in the system then the performance will remain the same or else it will decrease slightly. This kind of decrease is due to thread swapping in the CPU.

a) **Theoretical Values**

Giga Ops = CPU Frequency * Sockets * Cores per Socket * Instructions per cycle
For DP = 2.3 GHz*2*1* 16 = 73.6 Giga Ops
For SP = 2.3 GHz*2*1* 32 = 147.2 Giga Ops
For HP = 2.3 GHz*2*1* 64 = 234.4 Giga Ops
For QP = 2.3 GHz*2*1* 128 = 588.8 Giga Ops

b) **LinPack Performance**

Problem Size = 20,000 (¾ of memory)
Array Size = 20,000 30,000 50,000
Number of Trials = 3
Average Value for 1 Thread = 36.4 Giga Ops
Average Value for 2 Thread = 62.75 Giga Ops
Average Value for 4 Thread = 71.1 Giga Ops

c) **Conclusion**: From the above graphs, we can see that the CPU performance will drop at certain time intervals or after certain iterations. This might be due to the switching that happens between the threads and this drop is for very short interval (fraction of seconds).

2. Memory Benchmarking

This benchmark measures the throughput in (GB/sec) and latency in (us) for random and sequential read+write operations on the memory for multiple block size and different concurrency done in C language. It is done on Chameleon cluster.

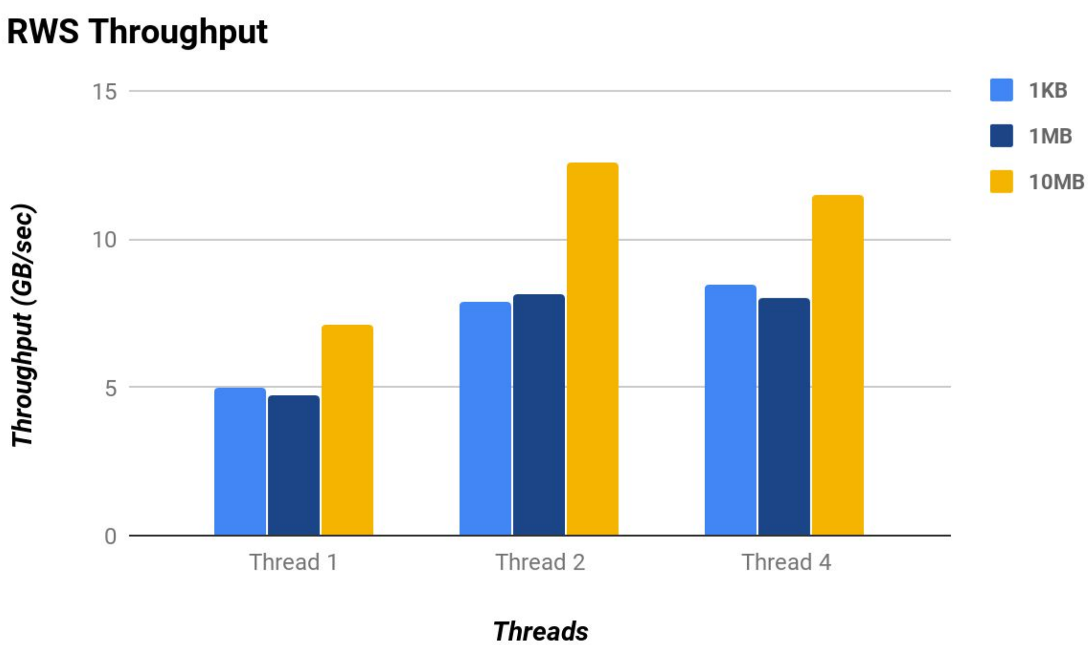
Type - DDR4 DIMM Synchronous 2133 MHz (CAS CL15)

Description - Entire 1GB of memory is divided into defined blocks and operation of sequential and random read+write are performed over 100 times for accessing over 100 GB of data for calculating throughput.

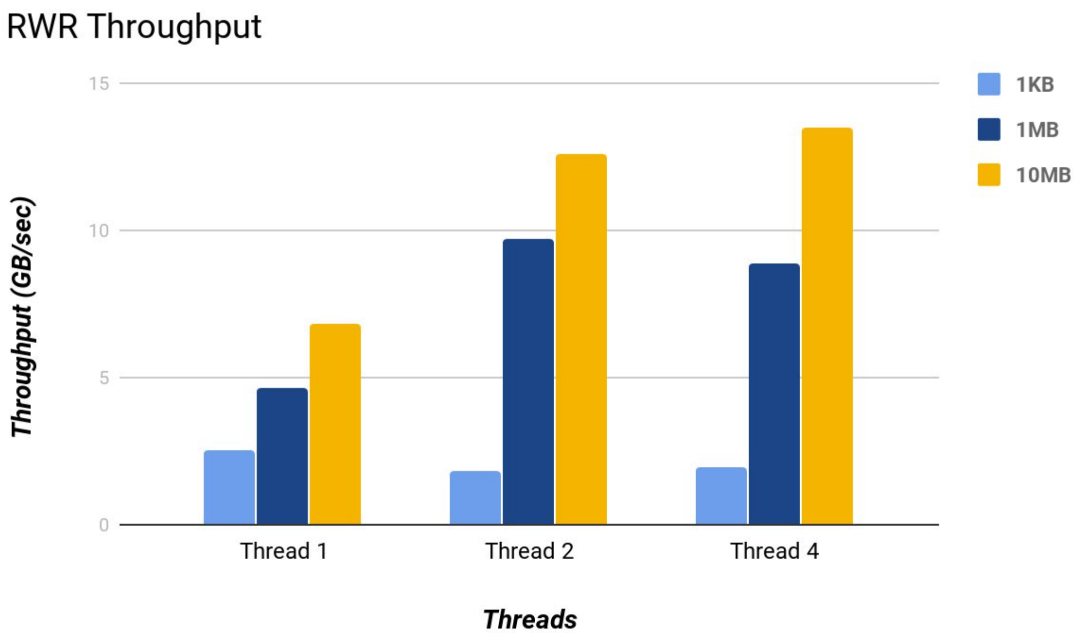
For latency calculation 100 million operations are performed on 1B of data in a sequential and random read+write pattern.

Workload	Concurrency	BlockSize	MyRAM Bench Throughput (MB/sec)	Theoretical Value (MB/sec)	Pmbw Throughput (MB/sec)	MyRAM Bench Efficiency(%)	Pmbw Efficiency(%)
RWS	1	1KB	4.97	68.25	15.5	7.2	32.1
RWS	1	1MB	4.7	68.25	19.3	6.9	24.3
RWS	1	10MB	7.1	68.25	17.5	10.4	40.5
RWS	2	1KB	7.9	68.25	23.4	11.5	33.76
RWS	2	1MB	8.13	68.25	19.5	11.9	41.69
RWS	2	10MB	12.6	68.25	38	18.4	33.5
RWS	4	1KB	8.43	68.25	27	12.35	31.5
RWS	4	1MB	7.99	68.25	40.1	11.7	19.9
RWS	4	10MB	11.46	68.25	35	16.7	32.7
RWR	1	1KB	2.5	68.25	4.9	3.66	51.02
RWR	1	1MB	4.63	68.25	0.48	6.77	964.5
RWR	1	10MB	6.8	68.25	0.44	9.9	1545.67
RWR	2	1KB	1.83	68.25	9.7	2.68	18.8

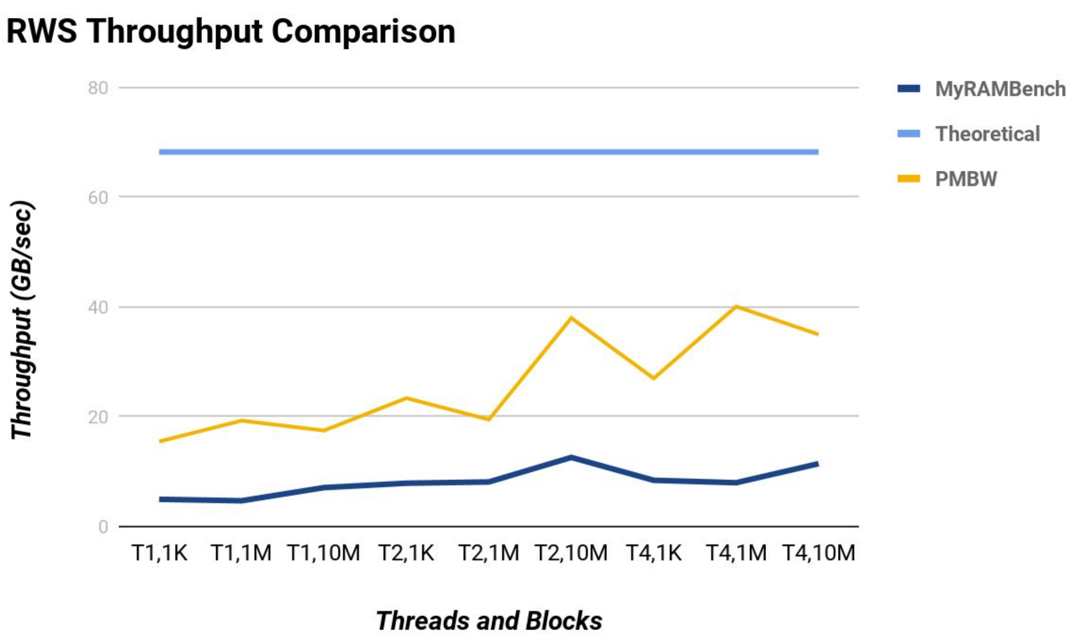
RWR	2	1MB	9.71	68.25	1.19	14.21	815.5
RWR	2	10MB	12.6	68.25	0.85	18.4	1482.1
RWR	4	1KB	1.93	68.25	21.4	2.82	9
RWR	4	1MB	8.9	68.25	2.77	13.04	321.5
RWR	4	10MB	13.5	68.25	0.86	19.78	1588.2



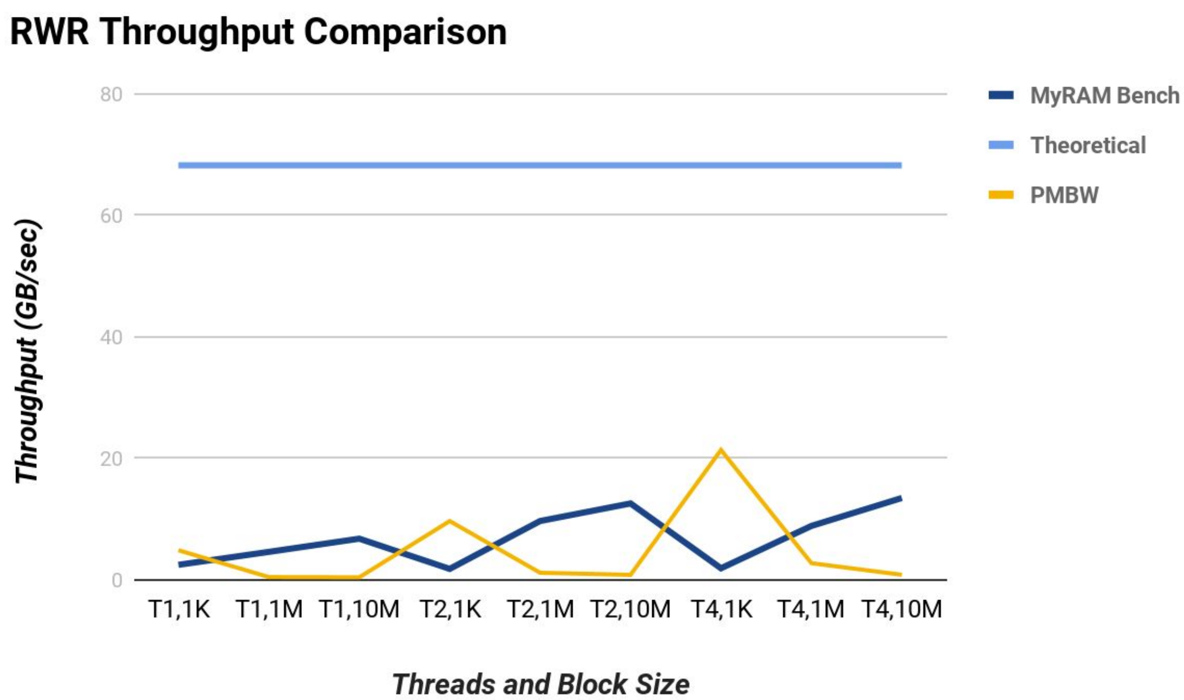
Above graph plots Throughput of RWS operation vs Threads for MyRAMBench for different block size.



Above graph plots Throughput of RWR operation vs Threads for MyRAMBench for different block size.

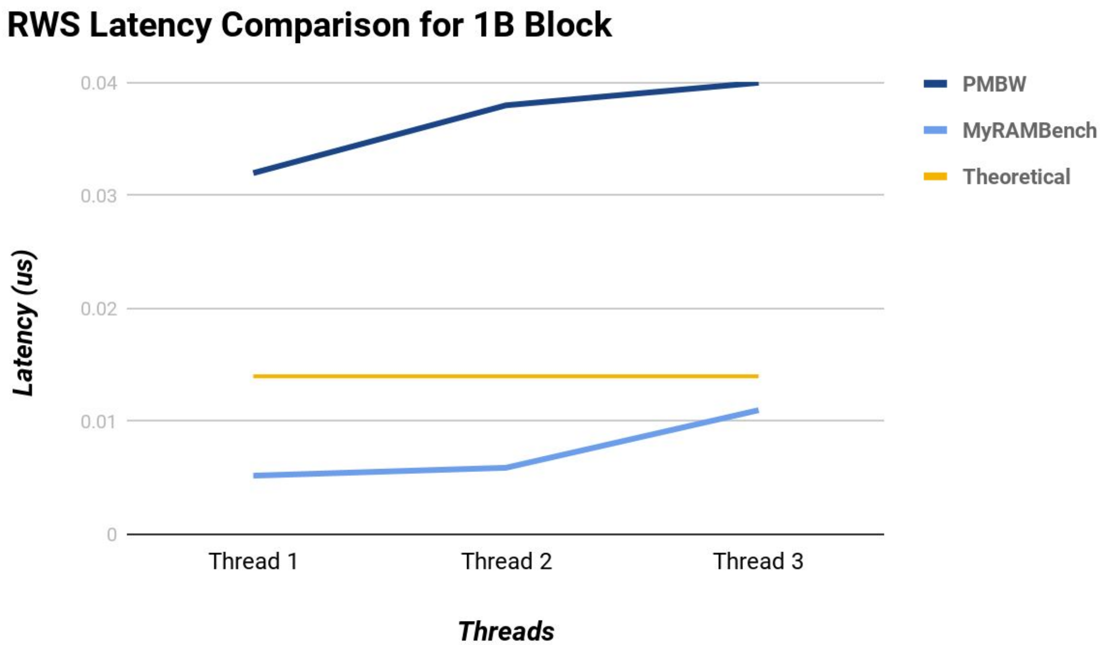


Above graph plots comparison of Throughput of MyRAMBench,Theoretical and PMBW values of RWS operation vs Threads and different Block sizes.

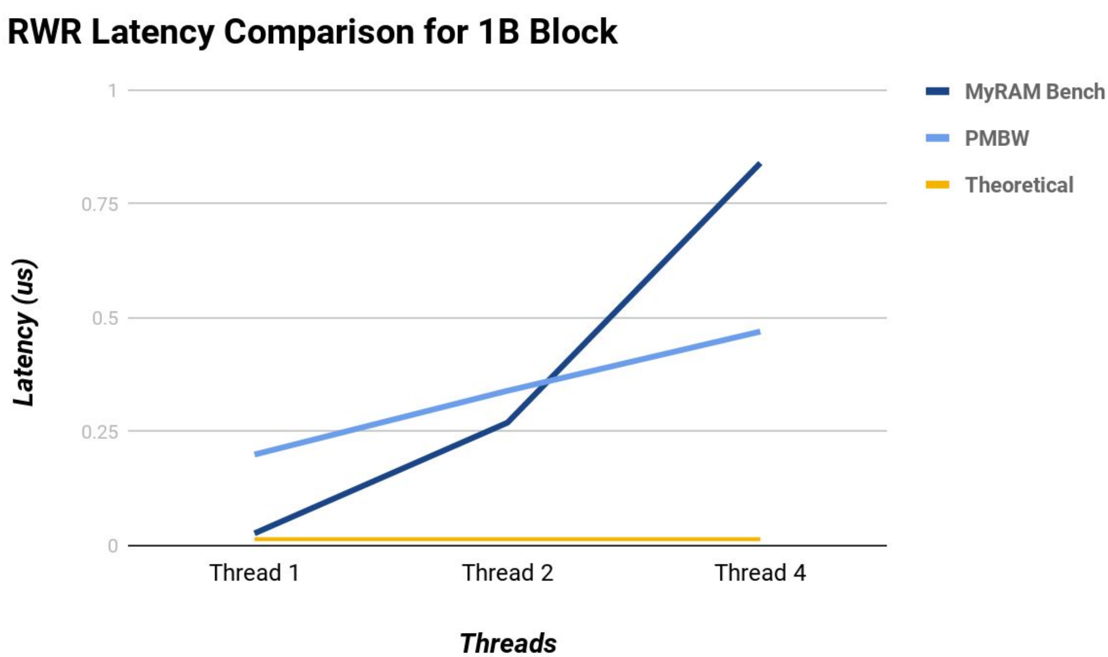


Above graph plots comparison of Throughput of MyRAMBench,Theoretical and PMBW values of RWR operation vs Threads and different Block sizes.

Workload	Concurrency	Block	MyRAM Latency (us)	Pmbw Latency (us)	Theoretical Latency (us)	MyRAM Efficiency (%)	Pmbw Efficiency(%)
RWS	1	1B	0.0052	0.032	0.014	62.8	-128.3
RWS	2	1B	0.0059	0.038	0.014	57.8	-171.42
RWS	4	1B	0.011	0.040	0.014	21.45	-185.45
RWR	1	1B	0.027	0.2	0.014	-92.1	-1328.57
RWR	2	1B	0.27	0.34	0.014	-1828.45	-2328.67
RWR	4	1B	0.84	0.47	0.014	-5900	-3257.14



Above graph plots comparison of latency of MyRAMBench,Theoretical and PMBW values of RWS operation vs Threads for 1B block size



Above graph plots comparison of latency of MyRAMBench,Theoretical and PMBW values of RWR operation vs Threads for 1B block size

Observation - From the above graph of throughput we can see that its value increases as threads are increased and so as block size. However lower latency is expected but we get higher throughput at the cost of latency so latency increases which is shown in latency graph.

a) **Theoretical Values**

Base DRAM clock frequency * Number of data transfers per clock * Memory bus (interface) width
* Number of interfaces
= 2133 MHz* 2* 64 * 2
= 546048 Mbits/second
= 68256 Mbyte/second
= 68.256 GB/second
While Theoretical latency is 14 ns given in memory specification
Efficiency % Throughput = (Actual /Theoretical) * 100
Efficiency % Latency = ((Theoretical - Actual) / Theoretical)*100

b) **Pmbw Performance**

It is already installed so I just ran binary by calling two methods with following parameters.

$pmbw -p 1 -P 1 -s 1024 -S 1024 -M 1G-f ScanRead64PtrSimpleLoop for RWS
$pmbw -p 1 -P 1 -s 1024 -S 1024 -M 1G -f PermRead64SimpleLoop for RWR

Here p and P are min and max threads, s and S are min and max block sizes which keeps varying and output is obtained in terms of bandwidth and time for throughput and latency calculation.

c) **Conclusion**: Here cache will play a role in random read+write operation as it may happen that we can hit caches before reaching original memory location which increases throughput and decreases latency

3. Disk Benchmarking

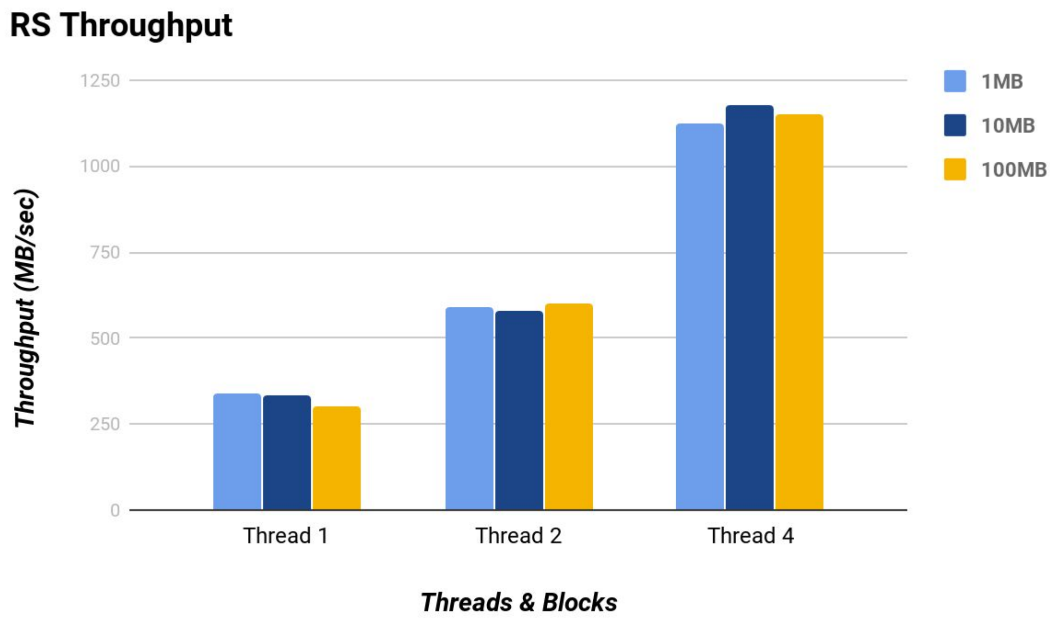
This benchmark measures the throughput in (MB/sec), IO operations in (IOPS) and latency in (ms) for combination of read, write along with random and sequential patterns on the memory for multiple block size and different concurrency done in C language. It is done on Prometheus cluster.

Type - Micron 5100 PRO 2.5"480GB,SATA,6Gb/s,3D NAND,7mm,1.5DWPD 2

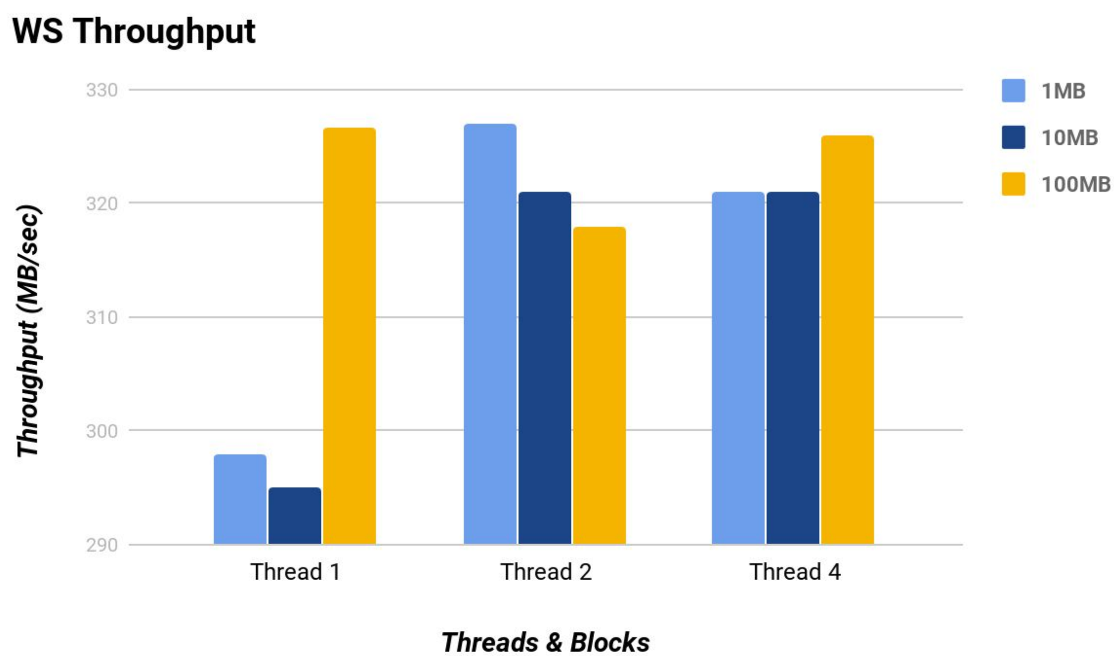
Description - Entire 10GB of file is divided into defined blocks for operation of sequential and random reading and writing to calculate throughput.
For latency and IOPS calculation 1 million operations are performed on 1KB of data in a random reading data from file and random writing data on file with multiple threads.

Workload	**Concurrency**	**Block**	**MyDiskBench Throughput (MB/sec)**	**IOZone Measured (MB/sec)**	**Theoretical Throughput (MB/sec)**	**My Disk Efficiency(%)**	**IOZone Efficiency (%)**
RS	1	1MB	337.78	351	540	62.1	65.2
RS	1	10MB	334.12	326	540	61.1	60.2
RS	1	100MB	303.17	325	540	56.49	60.3
RS	2	1MB	592.05	688	1080	54.57	63.5
RS	2	10MB	583.44	281	1080	55.5	26.7
RS	2	100MB	603.67	700	1080	54.6	64.1
RS	4	1MB	1127.38	1269	2160	52.7	58.1
RS	4	10MB	1187.89	1305	2160	54.4	60.2
RS	4	100MB	1153.12	1295	2160	55.8	59.1
WS	1	1MB	298.2	326	410	72.7	79.2
WS	1	10MB	295.61	298	410	72.1	72.1
WS	1	100MB	326.67	330	410	79.1	80.2
WS	2	1MB	327.8	370	820	39.8	45.4
WS	2	10MB	318.62	458	820	38.8	54.2

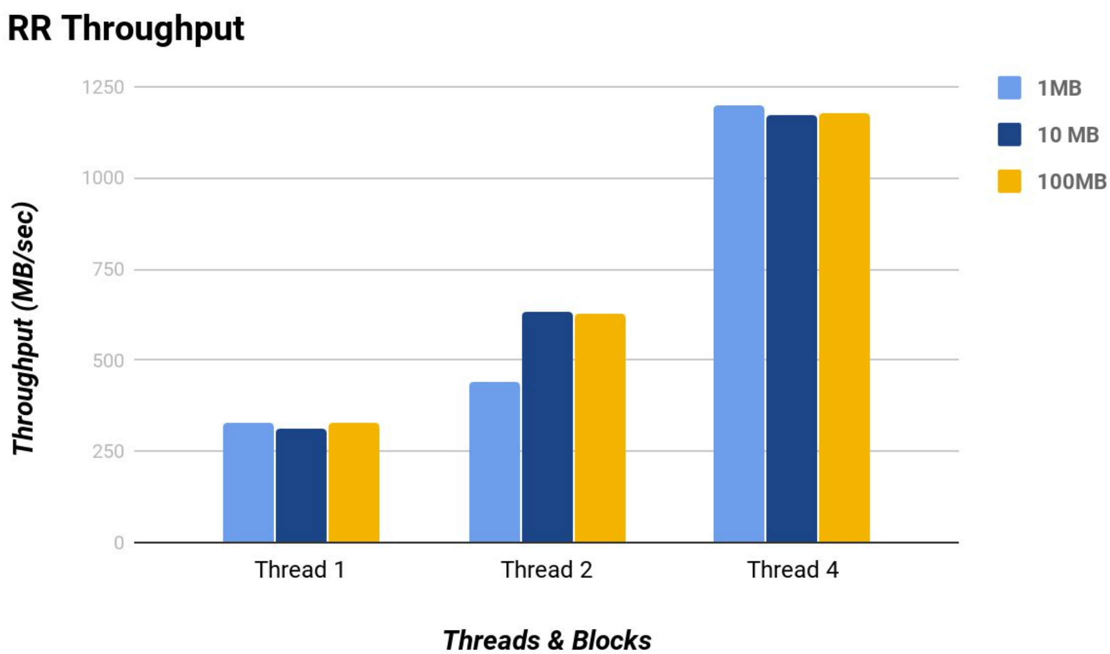
WS	2	100MB	321.7	530	820	39.6	64.4
WS	4	1MB	321.74	348	1640	19.5	21.2
WS	4	10MB	321.94	360	1640	19.2	21.2
WS	4	100MB	326.9	380	1640	19.1	23.4
RR	1	1MB	330.8	188	372	88.5	50.4
RR	1	10MB	310.2	315	372	83.2	84.5
RR	1	100MB	330.6	144	372	88.3	38.6
RR	2	1MB	439.7	451	744	59.2	60.3
RR	2	10MB	630.6	700	744	84.4	94.2
RR	2	100MB	626.6	690	744	84.3	92.3
RR	4	1MB	1197.2	693	1488	80.4	46.2
RR	4	10MB	1172.3	1190	1488	79.7	80.1
RR	4	100MB	1177.85	1350	1488	78.4	90.2
WR	1	1MB	333.1	298	172	194.4	173.4
WR	1	10MB	311.5	340	172	181	192.5
WR	1	100MB	308.2	330	172	179.1	191.6
WR	2	1MB	315.4	512	344	91.1	148.2
WR	2	10MB	303.5	500	344	88.2	145.3
WR	2	100MB	334.6	415	344	97.2	120.4
WR	4	1MB	333.6	550	688	48.4	80.1
WR	4	10MB	326.8	400	688	47.3	58.1
WR	4	100MB	308.5	410	688	45.07	59.55



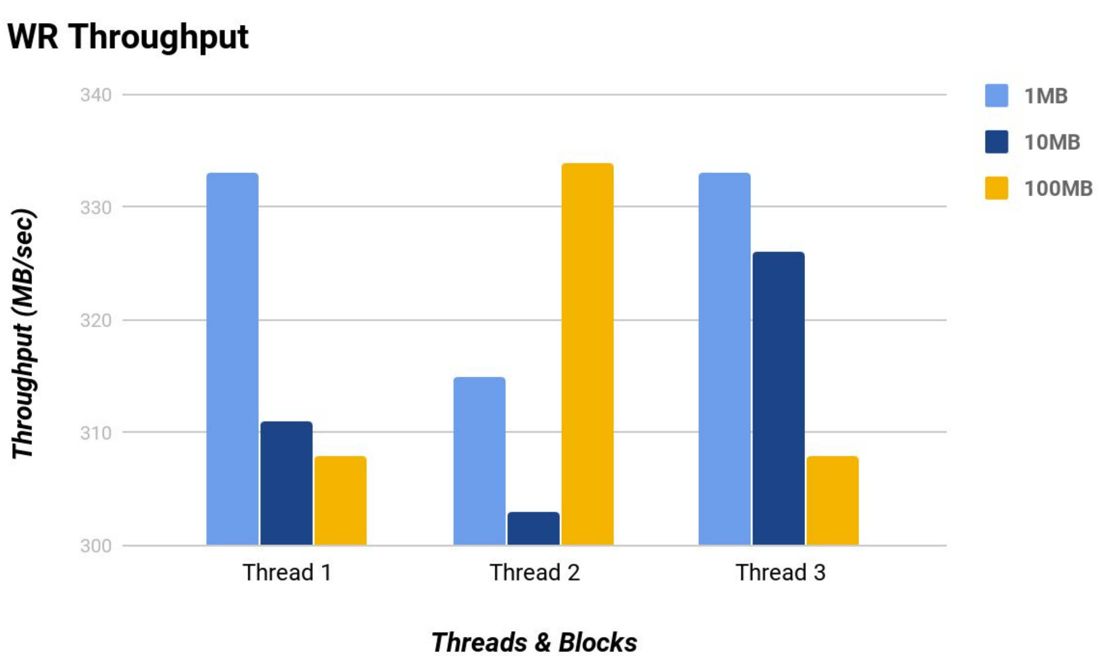
Above graph plots Throughput values of MyDiskBench of RS operation vs Threads for varying block sizes.



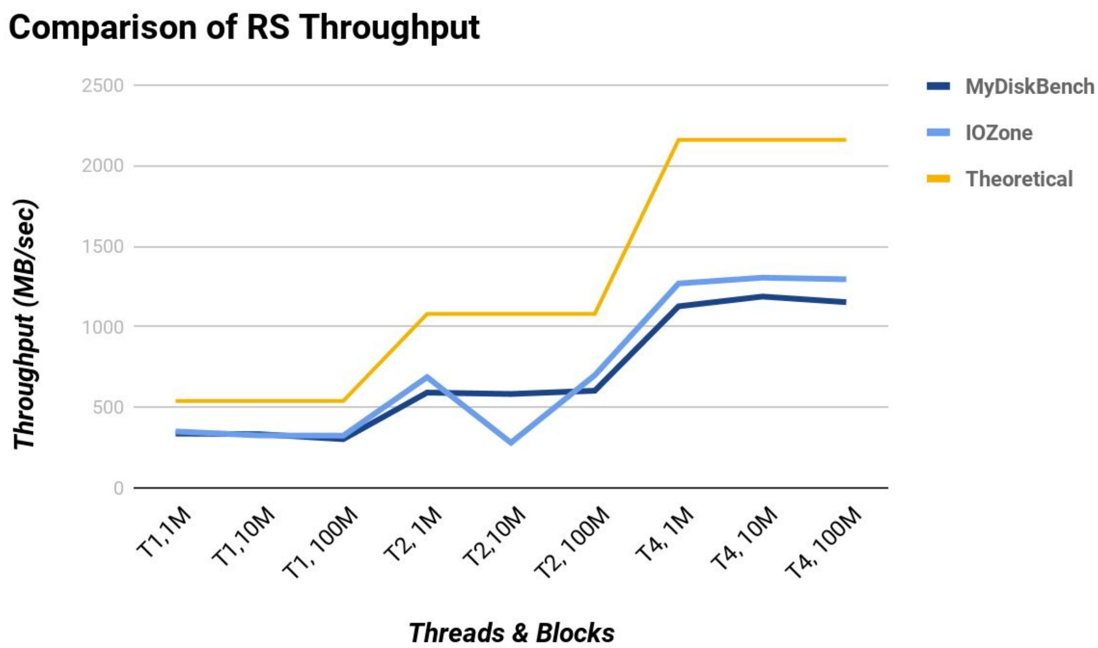
Above graph plots Throughput values of MyDiskBench of WS operation vs Threads for varying block sizes.



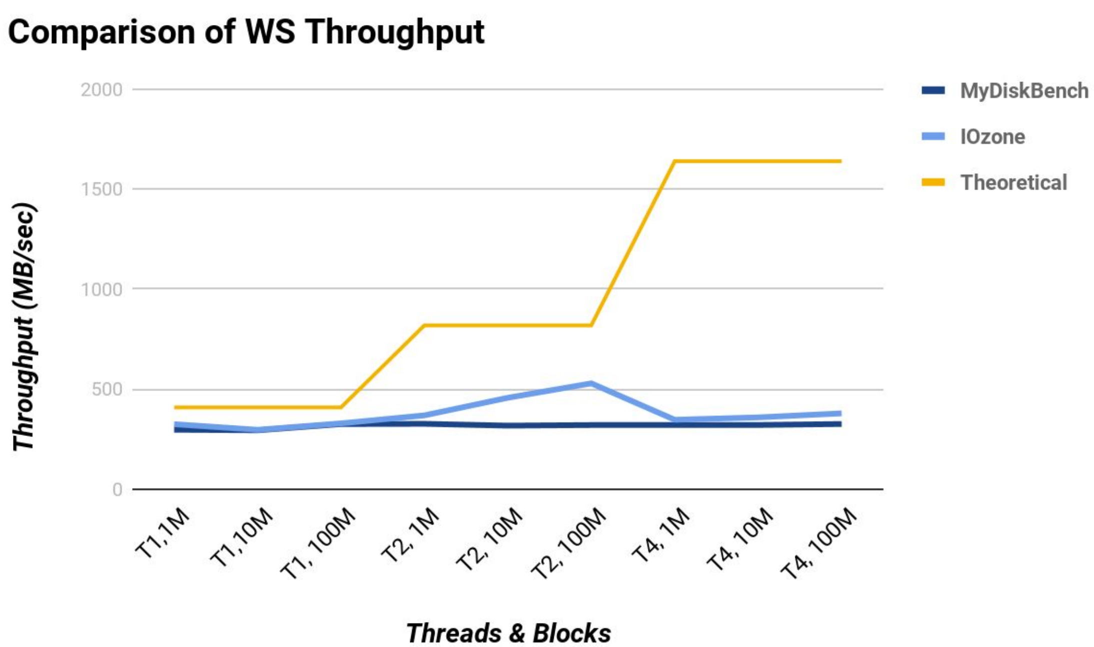
Above graph plots Throughput values of MyDiskBench of RR operation vs Threads for varying block sizes.



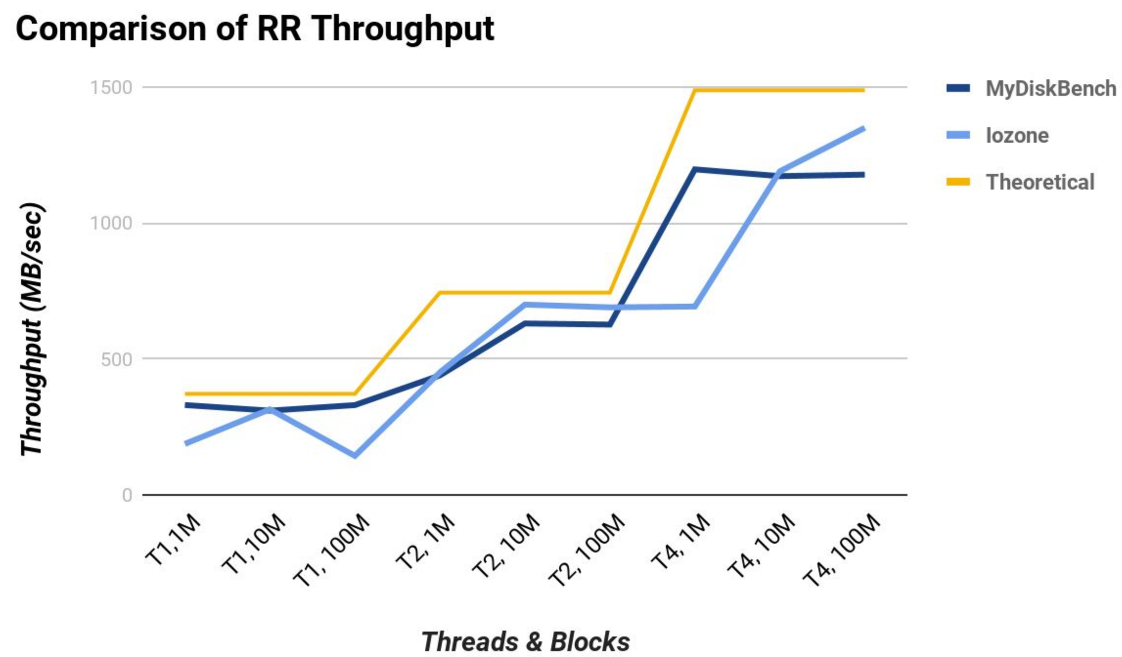
Above graph plots Throughput values of MyDiskBench of WR operation vs Threads for varying block sizes.



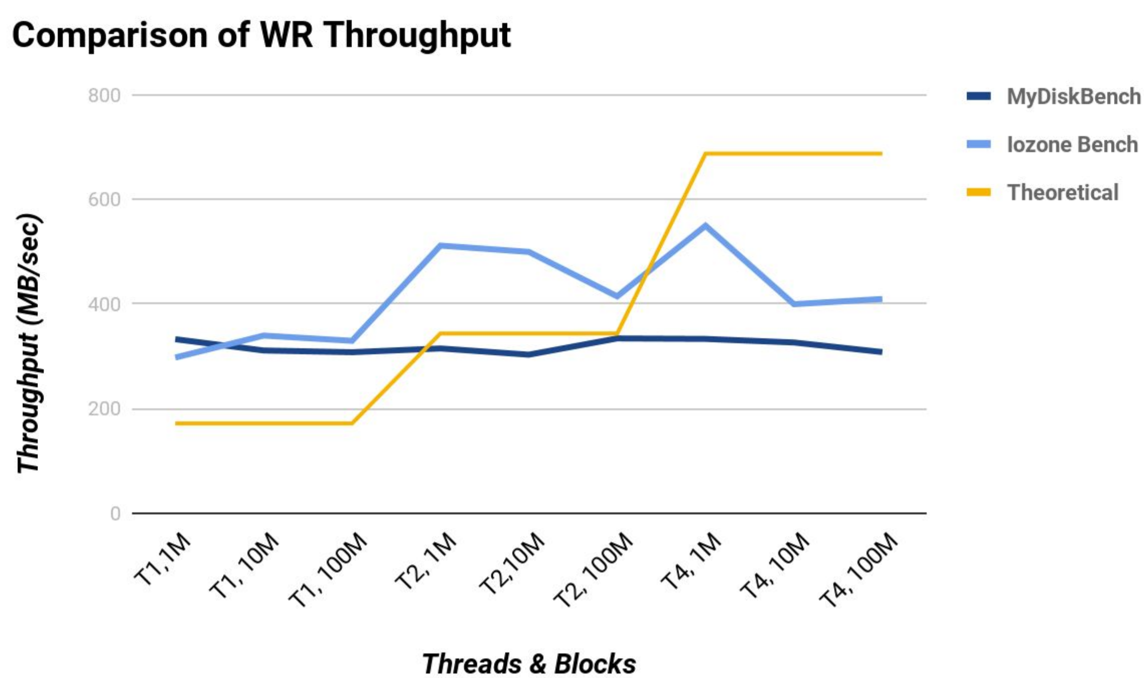
Above graph plots comparison of Throughput values of MyDiskBench, Iozone and Theoretical of RS operation vs Threads for varying block sizes.



Above graph plots comparison of Throughput values of MyDiskBench, Iozone and Theoretical of WS operation vs Threads for varying block sizes.

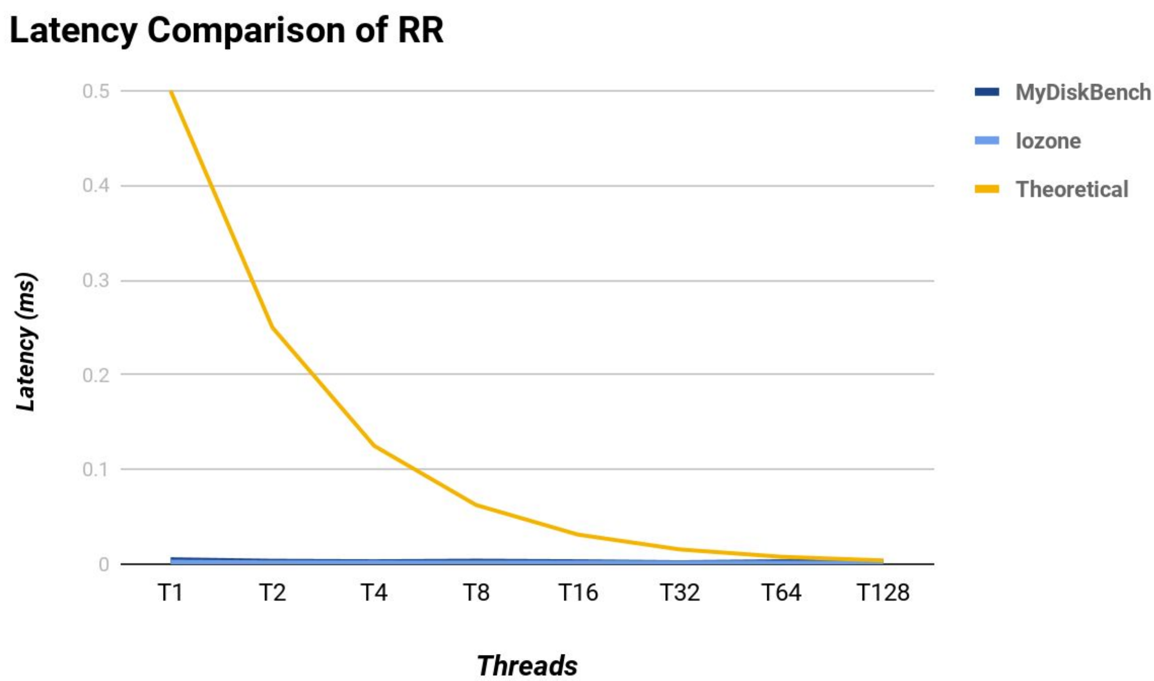


Above graph plots comparison of Throughput values of MyDiskBench, Iozone and Theoretical of RR operation vs Threads for varying block sizes.

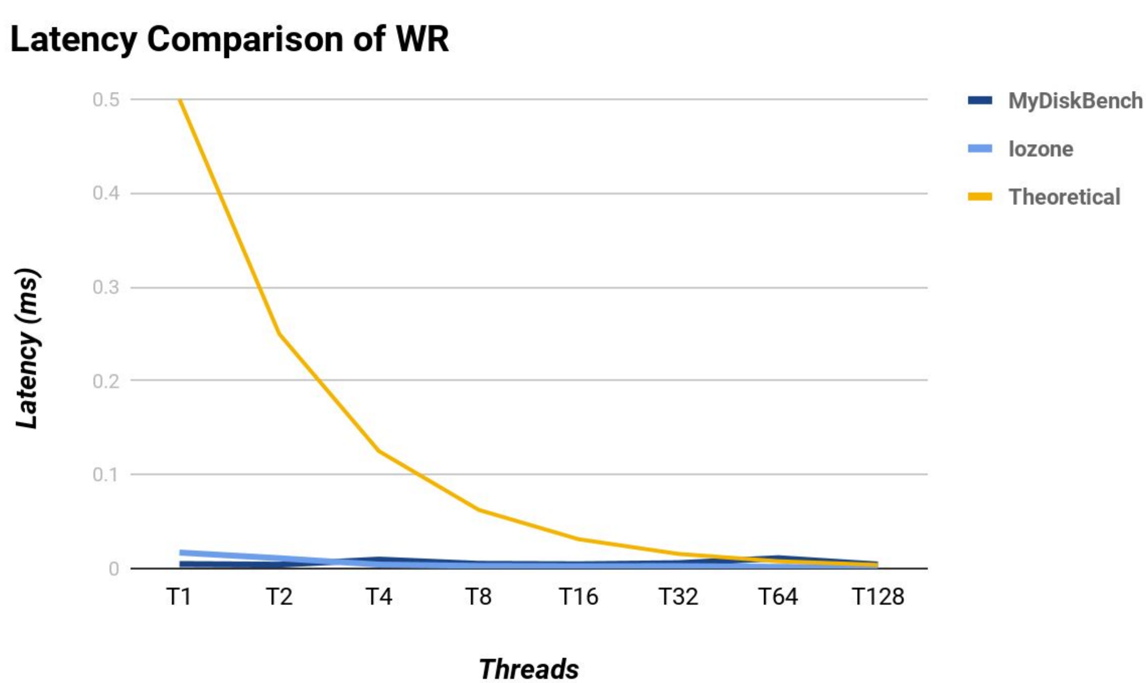


Above graph plots comparison of Throughput values of MyDiskBench, Iozone and Theoretical of WR operation vs Threads for varying block sizes.

Workload	Concurrency	Block Size	MyDiskBench Latency (ms)	IOZone Measured Latency (ms)	Theoretical Latency (ms)	MyDiskBench Efficiency(%)	IOZone Efficiency(%)
RR	1	1KB	0.0041	0.0016	0.5	99.2	99.4
RR	2	1KB	0.0025	0.00085	0.25	99	98.1
RR	4	1KB	0.0020	0.00086	0.125	98.4	99.1
RR	8	1KB	0.0026	0.00070	0.0625	93.8	98.8
RR	16	1KB	0.0020	0.00077	0.03125	93.1	97.1
RR	32	1KB	0.0010	0.00062	0.0156	74.2	96.6
RR	64	1KB	0.0020	0.00055	0.0078	99.6	92.4
RR	128	1KB	0.0022	0.00053	0.0039	43.1	86.1
WR	1	1KB	0.0048	0.017	0.5	99.7	96.5
WR	2	1KB	0.0041	0.011	0.25	98.6	95.3
WR	4	1KB	0.0095	0.0042	0.125	92.5	96.7
WR	8	1KB	0.0050	0.0028	0.0625	92.8	95.4
WR	16	1KB	0.0045	0.0025	0.03125	85.7	92.5
WR	32	1KB	0.0057	0.0026	0.0156	63.8	83.1
WR	64	1KB	0.011	0.0018	0.0078	-41.1	76.8
WR	128	1KB	0.0043	0.0015	0.0039	-10.9	61.9

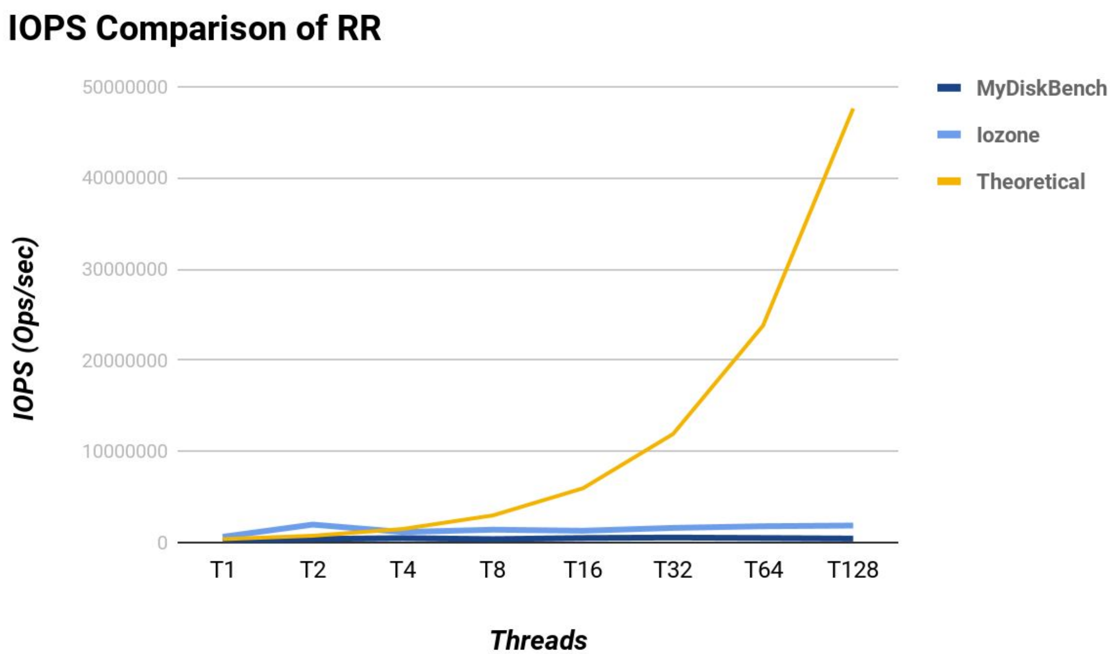


Above graph plots comparison of Latency values of MyDiskBench, Iozone and Theoretical of RR operation vs Threads for 1KB of block size.

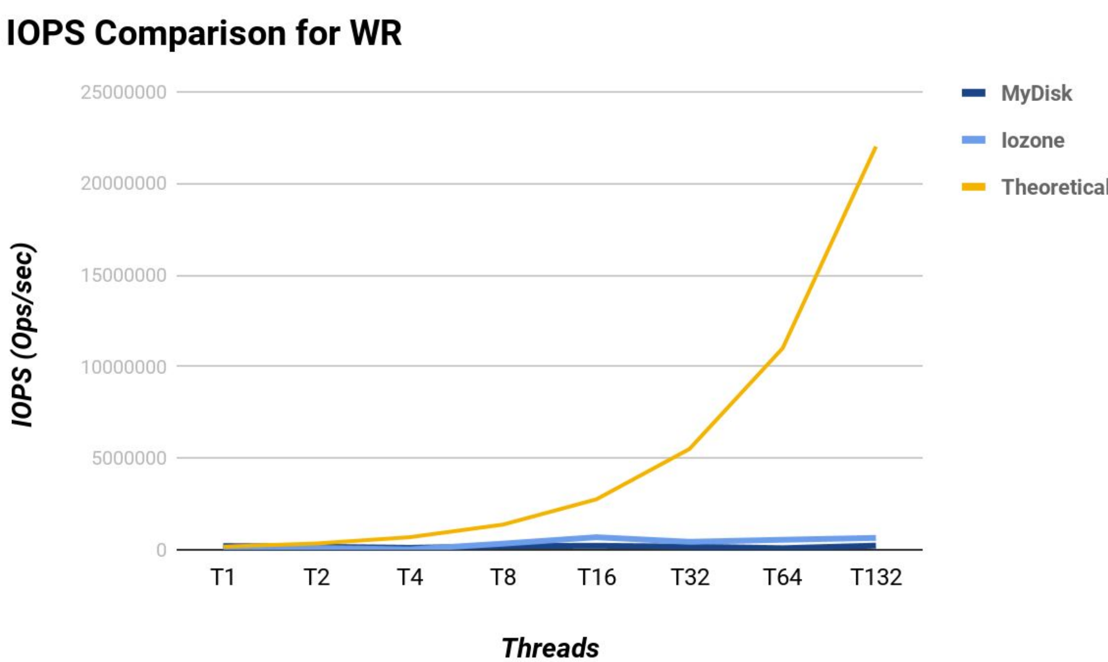


Above graph plots comparison of Latency values of MyDiskBench, Iozone and Theoretical of WR operation vs Threads for 1KB of block size.

Workload	Concurrency	Block Size	MyDiskBench IOPS	IOZone Measured IOPS	Theoretical IOPS	MyDiskBench Efficiency (%)	IOZone Efficiency (%)
RR	1	1KB	205488	625043	372000	55.5	168.4
RR	2	1KB	386301	1970547	744000	51.8	264.4
RR	4	1KB	485575	1151542	1488000	32.07	77.4
RR	8	1KB	373984	1408562	2976000	12.6	47.2
RR	16	1KB	495421	1292413	5952000	8.4	21.1
RR	32	1KB	547805	1605658	11904000	4.6	13.2
RR	64	1KB	495122	1789567	23808000	2.4	7.1
RR	128	1KB	435767	1865874	47616000	0.42	3.6
WR	1	1KB	205488	58724	172000	119.56	34.72
WR	2	1KB	169092	98161	344000	49.2	28.4
WR	4	1KB	105251	20665	688000	15.4	3.45
WR	8	1KB	198475	334674	1376000	14.3	24.23
WR	16	1KB	220450	685678	2752000	8.3	24.45
WR	32	1KB	172667	435676	5504000	3.1	7.8
WR	64	1KB	86614	546781	11008000	0.2	4.2
WR	128	1KB	228052	649899	22016000	1.1	2.9



Above graph plots comparison of IOPS values of MyDiskBench, Iozone and Theoretical of RR operation vs Threads for 1KB of block size.



Above graph plots comparison of IOPS values of MyDiskBench, Iozone and Theoretical of WR operation vs Threads for 1KB of block size.

a) Theoretical Values

As per specification given on data sheet of disk
Throughput RR - 372MB/sec WR-172 MB/sec RS-540 MB/sec WS- 410MB/sec
Throughput = IOPS * Block Size
Latency = Time of Execution / Total IOPS
Latency = 0.5 msec
Efficiency % Throughput = (Actual /Theoretical) * 100
Efficiency % Latency = ((Theoretical - Actual) / Theoretical)*100

b) Iozone Performance

It is already installed so I just ran binary by calling two methods with following parameters.

iozone -s 10G -r 1M -t 1 -i 0 -F //tmp/file.txt

Here, -s indicates file size, -r indicates block size, -t indicates throughput in parallel threads -i indicates operation type 0,1,2 for sequence read, sequence write and random read write, -F indicates file name to be generated.

c) Conclusion: Here disk cache will play a role in random read operation as it may happen that we can hit caches before reaching original memory location which increases throughput and decreases latency. We can also try to drop caches by generating system calls under privileged mode. We can also see that latency and throughput are inversely proportional as visible from the above graphs

4. Network Benchmarking

This benchmark measures the throughput in (Mb/sec) and latency in (ms) for network data transmission of 1GB, using TCP and UDP protocols over multiple block size and different concurrency for 100 times done in C language.

Type - 10 GBps connectivity with each node in a chameleon network.

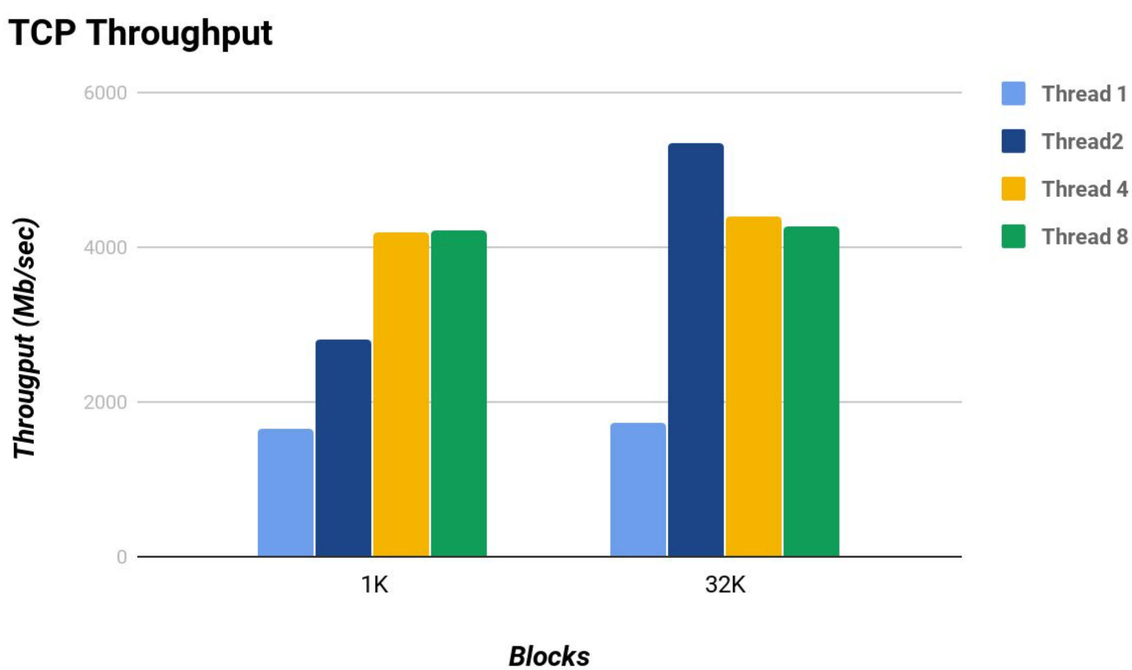
Description - Entire 10GB of data is divided into defined blocks for transfering over TCP and UDP protocols over a defined block size with multithreading.

Client will generate number of sockets as number of threads for both protocols and server will accept same number of sockets and data transfer process will begin. For throughput no acknowledgement is sent from server to client.

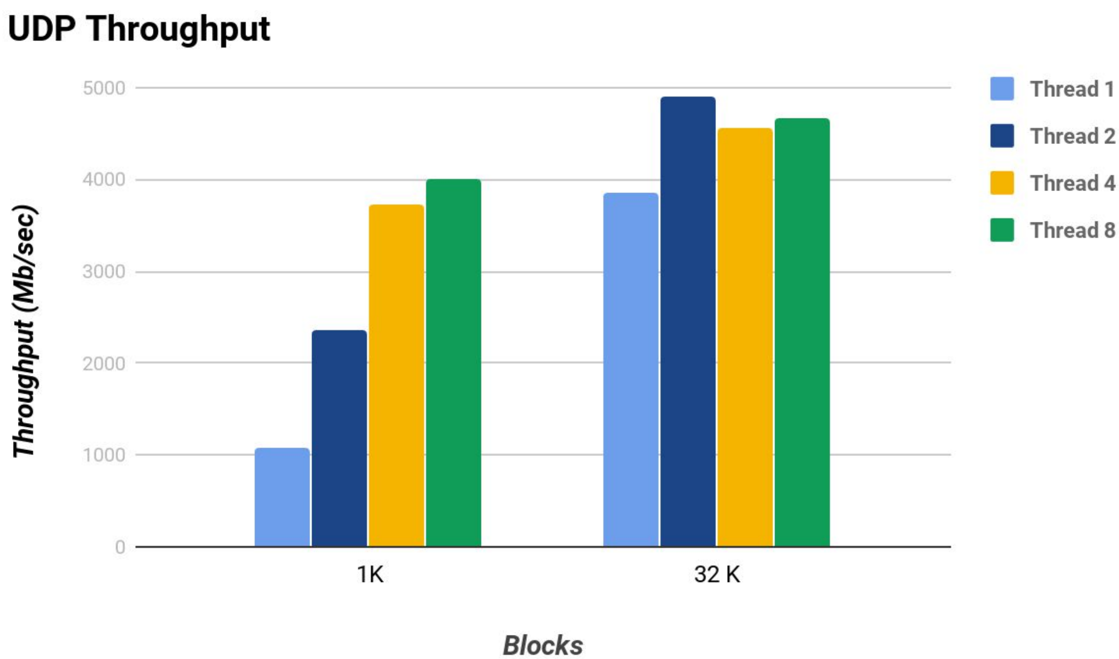
For latency calculation, 1 million transfer operations (ping pong) are performed on 1B of data send from client to server and server also responds back with same data. So it is round trip time.

Protocol	Concurrency	Block Size	MyNetBench Throughput (Mb/sec)	Iperf Throughput (Mb/sec)	Theoretical Throughput (Mb/sec)	MyNETBench Efficiency(%)	Iperf Efficiency(%)
TCP	1	1KB	1642.38	2007.32	10000	16.42	20.07
TCP	1	32KB	1727.21	2280.6	10000	17.27	22.80
TCP	2	1KB	2812.05	3500.6	10000	28.12	35
TCP	2	32KB	5321.67	8627.4	10000	53.21	86.27
TCP	4	1KB	4193.56	4507.5	10000	41.93	45.07
TCP	4	32KB	4401.41	4782.6	10000	44.1	47.82
TCP	8	1KB	4208.52	4400.6	10000	42.08	44
TCP	8	32KB	4277.08	5078.4	10000	42.77	50.78
UDP	1	1KB	1071.98	1200.8	10000	10.71	12
UDP	1	32KB	3845.76	3468.6	10000	38.41	34.68

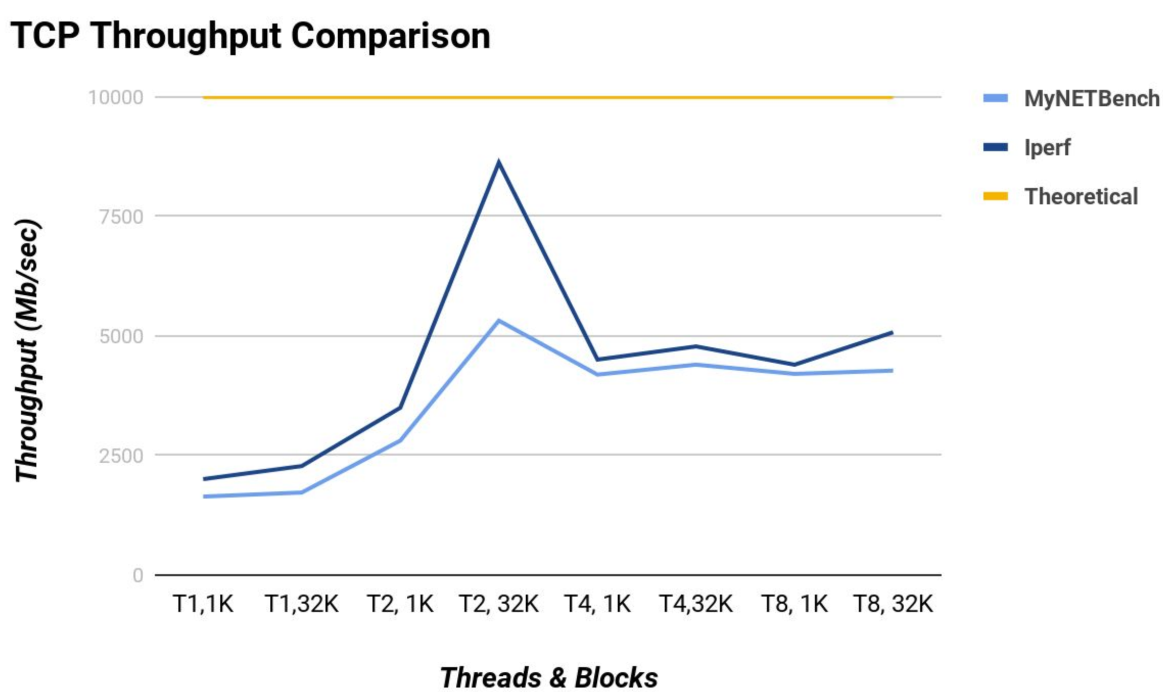
UDP	2	1KB	2352.4	1430.5	10000	23.52	14.30
UDP	2	32KB	4912.78	5200.6	10000	49.12	52
UDP	4	1KB	3735.5	5300.6	10000	37.35	53
UDP	4	32KB	4564.5	4800.6	10000	45.64	48
UDP	8	1KB	3995.6	5608.6	10000	39.95	56.08
UDP	8	32KB	4657.45	5001.4	10000	46.57	50.01



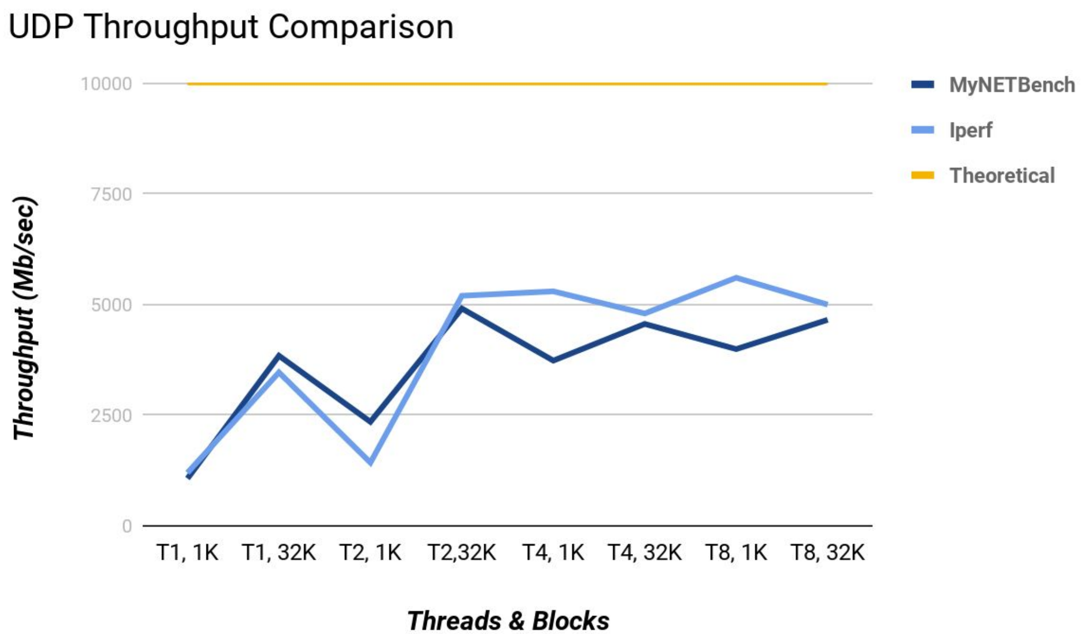
Above graph plots Throughput values of MyNETBench-TCP, Iperf and Theoretical of TCP operation vs Threads for varying block sizes.



Above graph plots Throughput values of MyNETBench-UDP, Iperf and Theoretical of UDP operation vs Threads for varying block sizes

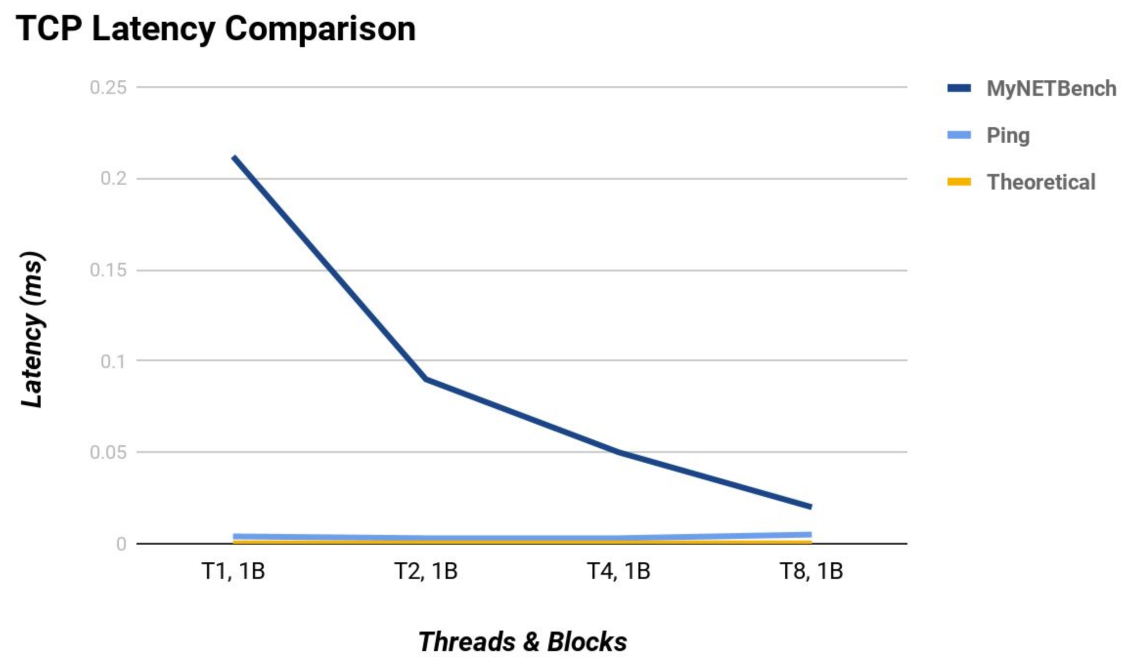


Above graph plots comparison of Throughput values of MyNETBench-TCP, Iperf and Theoretical of TCP operation vs Threads for varying block sizes

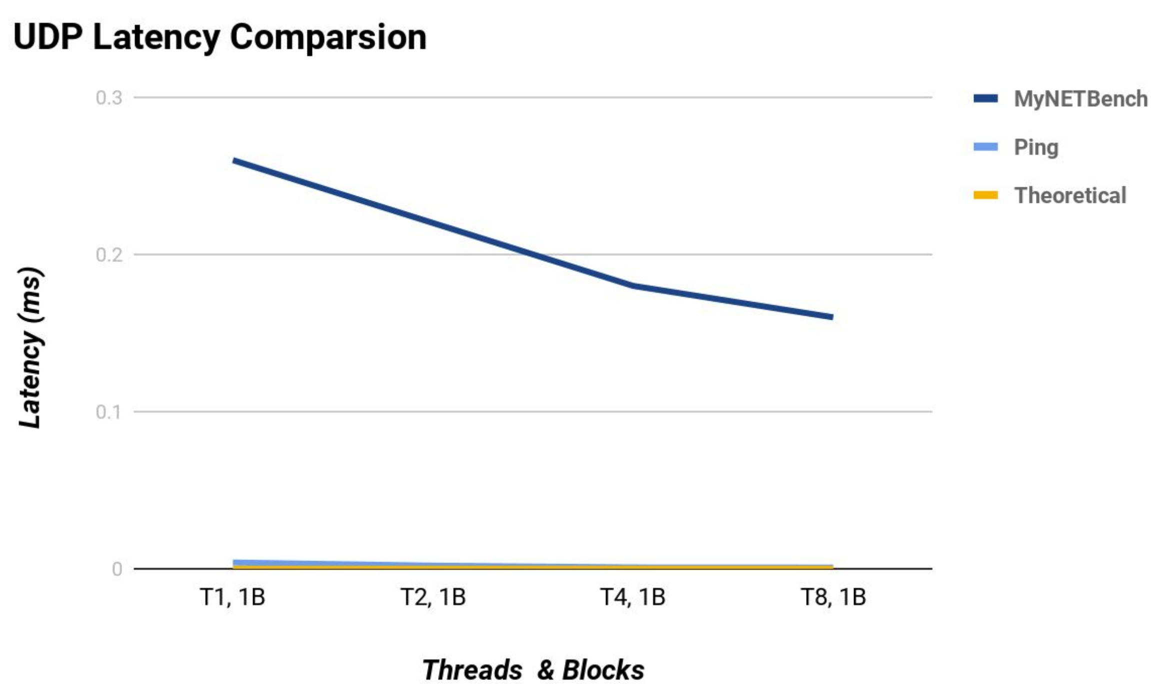


Above graph plots Throughput values of MyNETBench-UDP, Iperf and Theoretical of UDP operation vs Threads for varying block sizes

Protocol	Concurrency	Block Size	MyNetBench Latency (msec)	ping Latency (sec)	Theoretical Latency (msec)	MyNETBench Efficiency (%)	ping Efficiency(%)
TCP	1	1B	0.212	0.004	0.0007	-30185.2	-471.42
TCP	2	1B	0.09	0.003	0.0007	-12757.14	-328.57
TCP	4	1B	0.05	0.003	0.0007	-7042.85	-328.57
TCP	8	1B	0.02	0.005	0.0007	-2757.14	-614.2
UDP	1	1B	0.06	0.004	0.0007	-37042.4	-471.16
UDP	2	1B	0.02	0.002	0.0007	-31328.57	-185.34
UDP	4	1B	0.008	0.0009	0.0007	-25614.45	-28.34
UDP	8	1B	0.016	0.0008	0.0007	-22757.1	-14.28



Above graph plots Latency values of MyNETBench-TCP, Ping and Theoretical of TCP operation vs Threads for 1B block transferred 1 million operations.



Above graph plots Latency values of MyNETBench-UDP, Ping and Theoretical of UDP operation vs Threads for 1B block transfered 1 million operations.

Observation - From the above graph of throughput we can see that its value increases as threads are increased and so as block size. However latency for UDP is lower than TCP as TCP is connection oriented protocol and UDP is connectionless protocol so data loss is higher in UDP and it does not acknowledges server upon receiving data unlike TCP.

a) **Theoretical Values**

Theoretical Throughput value as defined in the Chameleon website as 10000Mb/s and Latency is 0.0007 msec

b) **IPerf Performance**

It is already installed so I just ran binary by calling two methods with following parameters.
iperf3 -s
iperf3 -c -u -l 32K -P 4
Here, -s and -c indicates server and client, -u for UDP and default is TCP, -P indicates number of parallel processes, -l indicates block size.
Output file is generated with bandwidth values after successful connection and transmission of data

c) **Conclusion**: Here TCP and UDP are transport layer protocol which handles data from network layer to transport layer at receiver end and from transport layer to network layer at the sender for reliable communication. High throughput and lower latency can be achieved in UDP than TCP at the cost of small data loss. So depending upon the requirement, any protocol can be implemented.