CS553: Cloud Computing

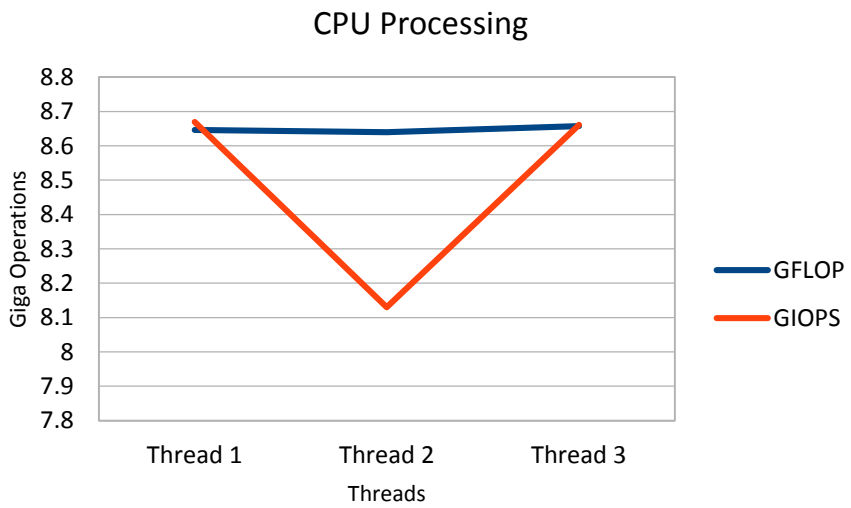
Evaluation Document

CPU Benchmarking:

- System Configuration
  - Amazon EC2 – t2-micro
  - Operating System: Linux Ubuntu 14*
  - RAM: 1 GB
  - 1 Core

Number Of Threads	GFLOPS	GIOPS
1	8.646125	8.669155
2	8.639707	8.130392
3	8.657069	8.660769

- Above Table shows CPU speed in terms of GIGA FLOPS (Giga Floating Point Operation per second) and GIGA IOPS (Giga Integer Operations per second respectively)
- We have taken 3 different threads (1,2, and 4).
- We have received almost same readings for GFLOPS and GIOPS irrespective of number of threads.
- But we have observed as we keep increasing number of threads to higher number performance start decreasing as threads share CPU and they need to switch among themselves more frequently than threads count with optimal number.

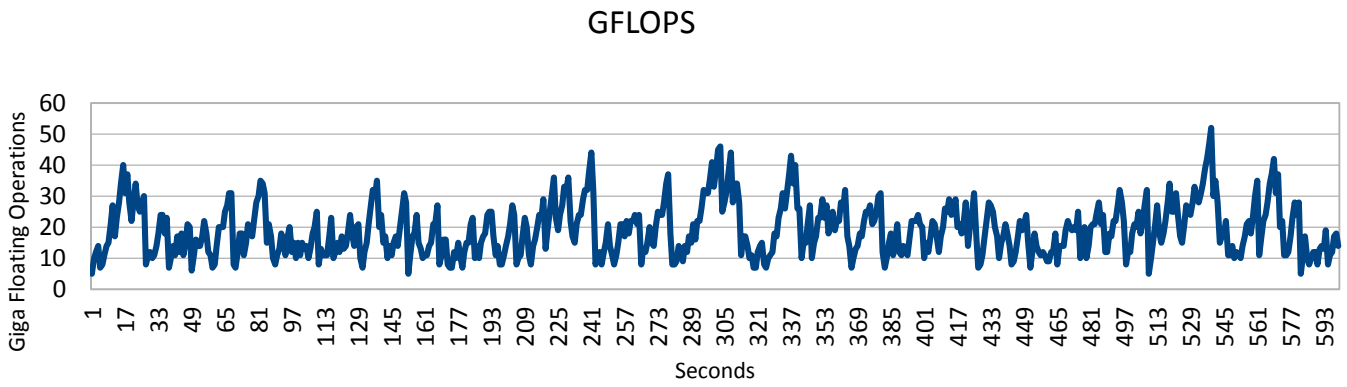


- **CPU Theoretical Performance –**

  = CPU Frequency * Number of Cores * Threads per core * IPC
  = 2.69 * 1 * 1 * 16
  = 43.04

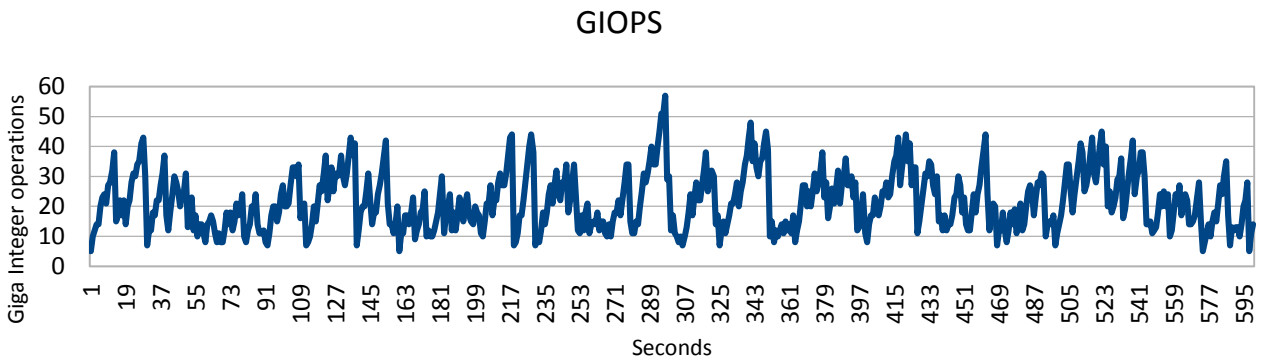
  - We have calculated CPU GFLOPS 8.6 and Theoretical Value for CPU is 43.04, so we have achieved 19.98% of Theoretical performance.

- GFLOPS for 10 min (sample plotted for every second)



  - We keep calculating GFLOPS over the period of 10 min (600 seconds) and keep the track of GFLOPS per second, Performance is fluctuating and giving the values between 10 to 30 GFLOPS on an average

- GILOPS for 10 min (sample plotted for every second)



- We keep calculating GIOPS over the period of 10 min (600 seconds) and keep the track of GIOPS per second, Performance is fluctuating and giving the values between 10 to 35 GFLOPS on an average

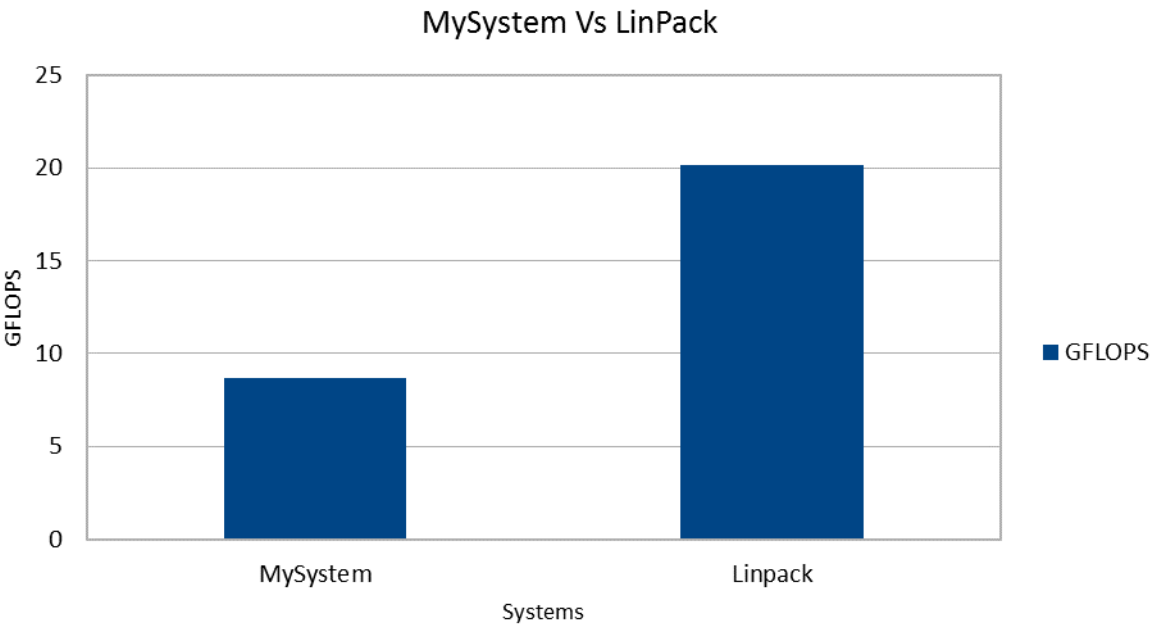
Conclusion-

- If we keep the execution running for 600 seconds and more, we found slight increase in the average number of GFLOPS and GIOPS.

- **Extra Credit:**
  - Here we have compared our results with Linpack benchmark. Below is screen shot for Linpack systems outputs

```
the correct number of CPUs/threads, problem input files, etc..
Fri Feb 12 00:53:14 UTC 2016
Intel(R) Optimized LINPACK Benchmark data

Current date/time: Fri Feb 12 00:53:14 2016

CPU frequency:    2.835 GHz
Number of CPUs: 1
Number of cores: 1
Number of threads: 1

Parameters are set to:

Number of tests: 15
Number of equations to solve (problem size) : 1000  2000  5000  10000 15000 1800
0 20000 22000 25000 26000 27000 30000 35000 40000 45000
Leading dimension of array                  : 1000  2000  5008  10000 15000 1800
8 20016 22008 25000 26000 27000 30000 35000 40000 45000
Number of trials to run                     : 4     2     2     2     2     2
  2     2     2     2     1     1     1     1     1
Data alignment value (in Kbytes)            : 4     4     4     4     4     4
  4     4     4     4     4     1     1     1     1

Maximum memory requested that can be used=800204096, at the size=10000

================== Timing linear equation system solver ==================

Size   LDA    Align. Time(s)    GFlops   Residual     Residual(norm) Check
1000   1000   4      0.039      17.1631  9.900691e-13 3.376390e-02   pass
1000   1000   4      0.037      18.0275  9.900691e-13 3.376390e-02   pass
1000   1000   4      0.037      18.1542  9.900691e-13 3.376390e-02   pass
1000   1000   4      0.037      17.9963  9.900691e-13 3.376390e-02   pass
2000   2000   4      0.276      19.3540  4.053480e-12 3.526031e-02   pass
2000   2000   4      0.275      19.3988  4.053480e-12 3.526031e-02   pass
5000   5008   4      4.009      20.7975  2.336047e-11 3.257429e-02   pass
5000   5008   4      4.004      20.8267  2.336047e-11 3.257429e-02   pass
10000  10000  4      31.436     21.2132  1.124127e-10 3.963786e-02   pass
10000  10000  4      30.908     21.5757  1.124127e-10 3.963786e-02   pass

Performance Summary (GFlops)

Size   LDA    Align.  Average  Maximal
1000   1000   4       17.8353  18.1542
2000   2000   4       19.3764  19.3988
5000   5008   4       20.8121  20.8267
10000  10000  4       21.3944  21.5757

Residual checks PASSED

End of tests

Done: Fri Feb 12 00:54:39 UTC 2016
ubuntu@ip-172-31-56-58:/usr/share/linpack$
```

- Analysis-
  - Below is graphical representation of results of my system with Linpack
  - My system has average GLOPS value to 8.6, Whereas Linpack has 20.5
  - So here we have achieved almost 42% of ideal value compared with Linpack.

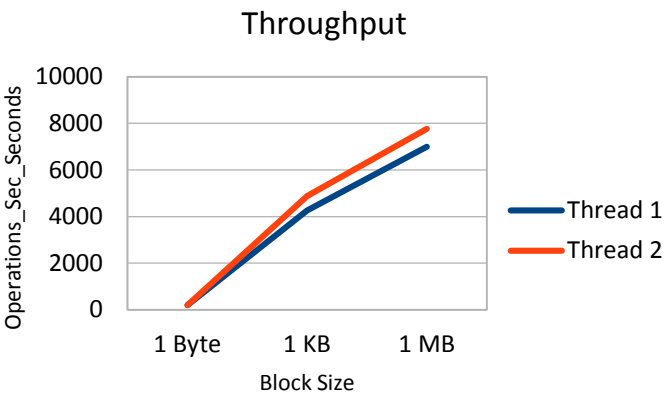


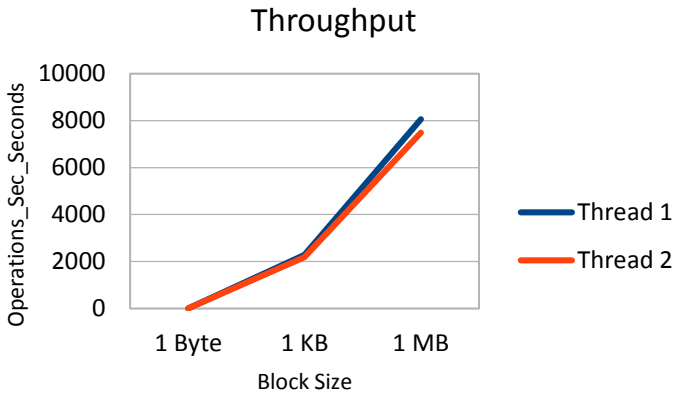
Memory Benchmarking:

- Write + Read Operations (Throughput)

	Sequential	
Block Size	Thread 1	Thread 2
1 Byte	201.207243	204.081633
1 KB	4266.66667	4876.19048
1 MB	6990.50667	7767.22963

	Random	
Block Size	Thread 1	Thread 2
1 Byte	8.719916	8.245723
1 KB	2275.55556	2178.7234
1 MB	8065.96923	7489.82857



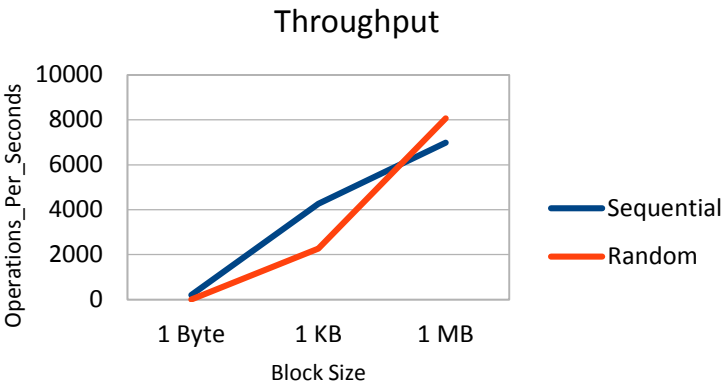


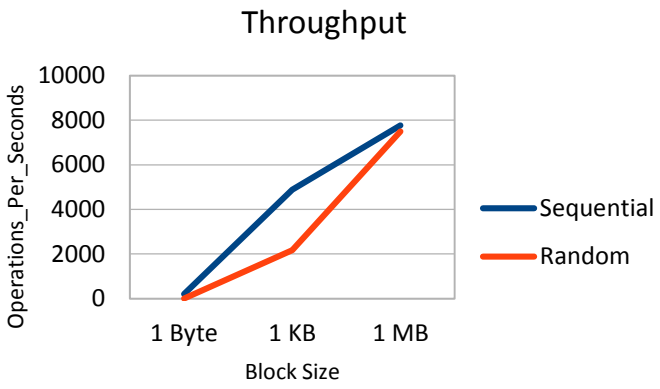
- Analysis-
  - From above tables and graph we can conclude that throughput for Read+Write Operations goes on increasing with increase in Size of block we are reading and writing by 1 Thread or 2 Threads
  - This is because as we increase block size, CPU write or Read more data at a time cause increase in throughput. It is not dependent on number of thread performing operation.

- Write + Read Operations (Throughput)

	Thread 1	
Block Size	Sequential	Random
1 Byte	201.207243	8.719916
1 KB	4266.66667	2275.55556
1 MB	6990.50667	8065.96923

	Thread 2	
Block Size	Sequential	Random
1 Byte	204.081633	8.245723
1 KB	4876.19048	2178.7234
1 MB	7767.22963	7489.82857



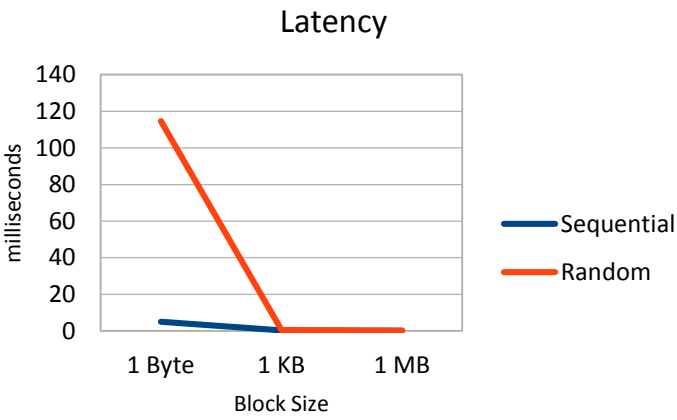


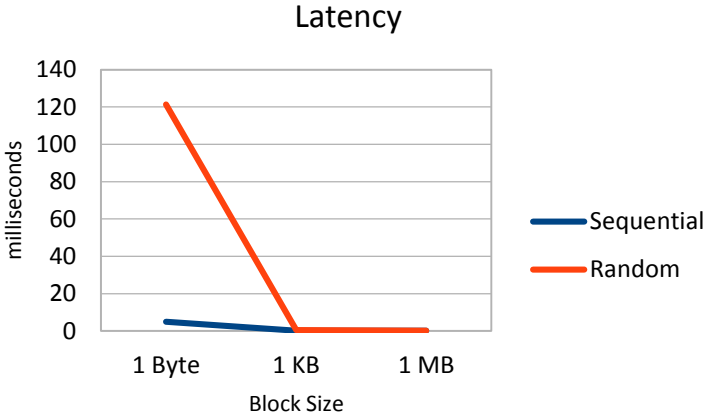
- Analysis –
  - From above tables and graph we can conclude that throughput for Read+Write Operations goes on increasing with increase in Size of block we are reading and writing it Sequentially OR Randomly.
  - This is because as we increase block size, CPU write or Read more data at a time cause increase in throughput. It is not dependent on how we are performing operations like Sequential or Random, eben there is slightly difference in their operations per seconds.

- Write + Read Operations (Latency)

	Thread 1	
Block Size	Sequential	Random
1 Byte	4.97	114.68
1 KB	0.234375	0.439453
1 MB	0.143051	0.123978

	Thread 2	
Block Size	Sequential	Random
1 Byte	4.9	121.275
1 KB	0.205078	0.458984
1 MB	0.128746	0.133514





- Analysis –
  - Tables and graphs above showing Latency for Sequential and Random operations for 1 Thread and 2 Threads.
  - Here we can conclude that Latency is high when we are reading or Writing Randomly with block Size of 1 Byte, in both the cases i.e. 1 thread or 2 thread.

- **Memory Theoretical Performance –**

  = Clock Frequency * Number of data transfer per clock * Memory bus interface width * number of interfaces
  = 1600MHz * 2 * 64 bits * 2
  = 409600 Byte
  = 51200 MB
  = 51200 MHz
  = 51.2 GHz

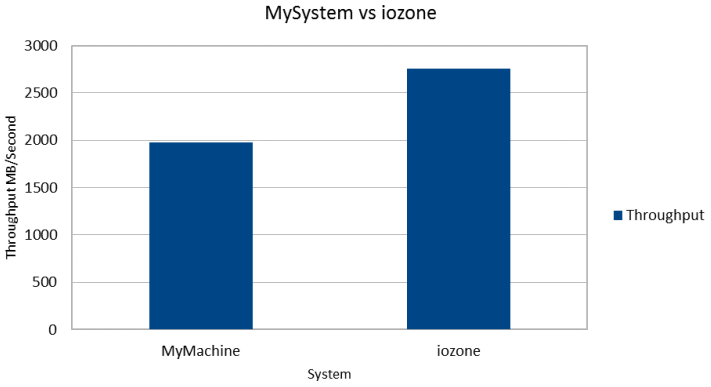
- **<u>Extra Credit</u>**
  - Here we have compared our results with iozone benchmark. Below is screen shot for iozone systems outputs.

```
Run began: Fri Feb 12 01:00:04 2016

Auto Mode
Command line used: ./iozone -a
Output is in Kbytes/sec
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 Kbytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.
                                                     random  random    bkwd   record   stride
      KB  reclen   write rewrite    read    reread    read   write    read  rewrite     read   fwrite frewrite   fread  freread
      64       4  855419 2006158 10821524 10402178 7100397 3203069 4988978  4018152  7100397  3203069  3363612 5860307 10821524
      64       8 1421755 2203800  7100397 1597288512902017 2662899 4274062  4274062  5389653  2561267  2662899 4897948 12310336
      64      16 1049372 2298136  3203069 1597288512310336 2662899 4274062  3363612  7940539  3057153  3165299 5389653  7940539
      64      32 1330167 2467108  9318832 1597288510821524 2801873 4564786  3057153  5735102  3203069  3588436 4564786 12310336
      64      64 1452528 2358717 15972885 1597288510402178 2662899 4274062  2772930  4018152  2689580  4274062 7940539 12902017
     128       4 1172704 2409592 11720614 11720614 8548124 3657016 7082197  4717434  8414153  3657016  3657016 8036304  9795896
     128       8 1598754 3657016 11720614 1420079411720614 3982553 7917784  5122535  8548124  3657016  4012317 9129573 12542043
     128      16 1727352 4407601 14200794 1420079412542043 4135958 8414153  4596273  8036304  3867787  4557257 9795896 10779307
     128      32 1827299 4557257 12842051  578489111470204 4717434 8036304  5122535 10779307  4267461  4557257 9129573 12842051
     128      64 1852520 4407601 14200794 1588107812842051 4717434 9129573  4717434  7582312  3218540  6727225 9129573 12842051
     128     128 1663139 4596273  5784891 1588107812842051 4934216 8548124  4759253  5325799  3468030  4557257 8548124 12542043
     256       4 1347557 3705040  8534922 10651598 8534922 3159869 5810112  4652146  5455847  3605511  3511189 8208677  9434868
     256       8 1407621 4264168 12228612 1345445010758322 3935921 9518507  4197489  9868433  3421677  1938842 3654598 13454450
     256      16 1675612 2413957  7314033 1345445012228612 3087188 6107548  5569035 11569783  2582316  2812272 4070199 12228612
     256      32 1354356 2812272  5455847 1281227712228612 2975955 5569035  5455847  9518507  3159869  3087188 6398720 12812277
     256      64 1588832 2509882  7314033 1516462412661199 3236056 5687020  4929815  8815202  2842047  3009318 3823789 13454450
     256     128 1610276 3009318  8271916 1416439512228612 3052087 4477549  3009318  4281170  2754556  3756894 4840911 12812277
     256     256 1673001 2911402 10758322  7314033 7571923 3326279 4929815  2486631  3705040  2665657  2000242 2984226 12812277
     512       4 1160923 2073249  5886650  8844453 8528336 2572435 5177083  4030519  6571132  2032051  2275345 4742616 10235583
     512       8 1418599 2089386  6571132 1424006910485468 2600470 5951911  5551840  9859630  3029721  2651850 3963567 12499490
     512      16 1514652 2483197  6821616 1378308811374040 3372274 5227492  6228097  9304292  2350044  2317080 6821616 12797441
     512      32 1723770 2786024  7115451 1424006911877300 2681653 6246213  6319739 10044089  2906696  2216629 4784885 13438092
     512      64 1610042 2317080  6931711 1310994411877300 3304808 5384786  5278892 11374040  2826358  2995907 6472112 13872122
     512     128 1689859 2497638  7647578 1352271113109944 3580298 4871723  3970896  8992598  3220553  3659616 6319739 13872122
     512     256 1815584 2235086  7115451 12215097 9107004 3029721 5509113  3346002  4494470  2497638  4340054 5566231  9468384
     512     512 1595686 2497638 11620224 12499490 5214798 3008498 6086873  2926501  5127637  3393590  3068685 6319739  8528336
    1024       4 1182598 1858644  5720477 11645609 7420395 2403712 5303701  5564829  8457895  2760606  2197132 4720752  9485232
    1024       8 1607514 2716948  6692005 12639479 8457895 2598591 5917516  6863100 10329265  2625597  1954207 6059442 11018226
    1024      16 1478064 2290886  5821271 1152065811645609 2960401 6093831  6402699  9485232  2716948  2173780 5853003  7220790
    1024      32 1648862 2348509  6819511 1066262812037272 2701567 5821271  6787181  9677584  2918161  2659742 7008693 13863422
    1024      64 1651398 2960401  6744549 1423090212828238 3529706 6918376  5720477 13472053  3066069  3191372 6519323 14668318
    1024     128 1503415 2751762  7208671 1330511610878686 2966535 5885083  4762630 10354166  3425544  2497356 6830356 12071103
    1024     256 1625154 2579861  7869040 1298335311903823 2944166 6650556  5390231  8914314  3655895  3458646 6819511  8895850
    1024     512 1865101 2297012  9402175 12983353 8840915 3618930 5958564  4594502  8822754  3149251  4826859 6874084 11249091
    1024    1024 1723636 2491561 10796646 1314226512313351 3191372 6819511  3281592  6519323  3314514  2640123 6787181 12983353
    2048       4 1354493 2321725  5626082 11502234 9101380 2283460 5293261  5785224  8292998  2438378  1905146 5159722 10607402
    2048       8 1445434 2120026  6807652 10240672112022231 3102397 6202989  7393606 11365268  3169944  2572756 6696208 12882215
    2048      16 1695335 3002637  6917203 1330111431187663 3287628 6114678  6225466 11090619  3042062  2767549 6308503 13488807
```

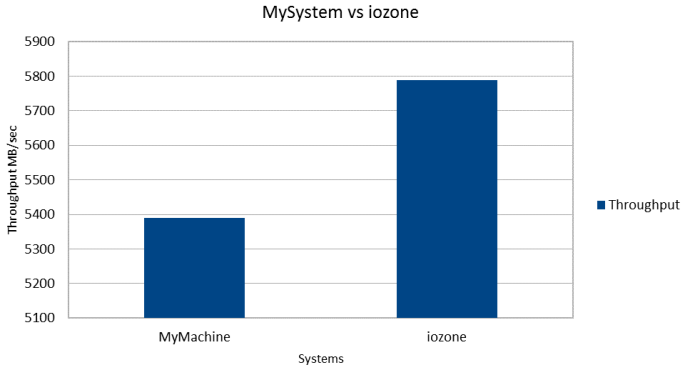
2048	4	1354493	2321725	5626082	11502234	9101380	2283460	5293261	5785224	8292998	2438378	1905146	5159722	10607402
2048	8	1445434	2120026	6807652	1024067211202231	3102397	6202989	7393606	11365268	3169944	2572756	6696208	12882215	
2048	16	1695335	3002627	6917293	1330111312187663	3287628	6114678	6225466	11900619	2942962	2767549	6298503	12488897	
2048	32	1563282	2476337	7311791	1413269812959958	2967361	5611381	7496851	12729493	2813783	2639154	6382745	13038646	
2048	64	1523901	2270783	6873016	1330111313639023	3538147	6207471	6945257	11564174	3180508	2959183	6401772	12635867	
2048	128	1713598	3089009	6608629	1374816911440955	2774700	5738844	6649556	12488897	3250308	2892423	6917293	13489098	
2048	256	1641240	3047366	5674397	1083485411502234	3025897	5832361	6041567	9944290	3038742	2457914	6690992	11835033	
2048	512	1806199	2933915	7698414	1218766311851361	3199462	6543188	4994712	9578367	3730203	3271351	6261772	11900619	
2048	1024	1731211	2709070	8749118	1303864611564174	2812862	6543188	4385291	8987113	3319389	5071381	6945257	12729493	
2048	2048	1958129	3180508	12882215	1313835911770166	3292669	6416117	3671217	6167360	3549844	3006831	6266340	11502234	
4096	4	1354045	1903416	6461498	10867793	8603675	4338563	5572803	5284803	8062616	3357537	2652931	6122971	9868918
4096	8	1882352	2992067	6758923	1288033811252164	5263753	6522830	6682678	10136778	4485833	2505520	6041154	10118866	
4096	16	1608179	2663212	7418527	1164101010985935	3470816	6178018	7277121	11447097	3571100	2740095	6861502	11311433	
4096	32	2131020	3116922	7625987	1296784013258059	3800178	6847827	7135072	11378861	4095510	2752828	6964419	10698597	
4096	64	1963457	3133979	7460407	1268068813518879	4231697	7209932	7935986	11135471	3750403	2910951	6839648	13176709	
4096	128	2058256	3199942	7656576	1256935713340420	5828019	7249484	7261741	11833449	3935097	2810931	7123239	12083134	
4096	256	1974968	2896717	7173805	1104242512765490	4008550	7123239	6916752	13207098	3501232	2790386	6804432	11348794	
4096	512	1955189	3286877	7249484	1150844211907264	3314142	6500617	5859825	10190895	2730514	2940849	6169144	11348794	
4096	1024	1923879	2878275	7892238	1252354412083134	3045641	6638778	5145513	10089154	3734100	2977031	5019235	11164417	
4096	2048	1703202	2483786	9163546	1347646012919082	3540193	4954103	4887857	7892238	3447830	5050220	6714018	12007129	
4096	4096	1718707	3009890	10215133	1211722313086376	3667146	5312585	3374022	5196878	3212508	3185110	6575257	10779149	
8192	4	1424958	2489149	6272536	8819674	8532752	2564005	4757117	5672914	8551866	2658022	2282461	5044886	9536939
8192	8	1485132	2226697	5285458	1155413011735662	3119697	5178710	7154097	11084481	3093575	2612151	7013901	8111736	
8192	16	1654311	2716226	7142200	1124040612409588	4213951	6854392	6351381	10981742	3676880	2649414	7453619	11084481	
8192	32	1863952	2722467	7218727	1145780712468128	4162896	7543623	8047143	12778810	3777121	2675407	7321790	10530733	
8192	64	1855497	2911150	7802283	1080560912642455	4471708	7346838	7500806	10115308	3503662	2756540	7307775	10291026	
8192	128	1781607	2718805	7155586	1110238911299550	3494753	5224380	8102172	11652086	3304518	2398115	7068733	10569606	
8192	256	1746382	3070628	6506533	1145780712642455	3491912	6425016	7167528	12302946	3102794	2318032	6786699	10393756	
8192	512	1792575	2792837	5919188	1090852611804201	3422010	6336155	6186701	11393220	3143385	2888633	6559949	10315743	
8192	1024	1612617	2544825	6670750	1056960610738070	3691893	7295362	6215800	10925870	3410461	2926772	6770651	9905351	
8192	2048	1730463	2788304	8242105	1162056012860118	3479888	6687628	5378957	10625170	3551098	3665505	7944805	10501766	
8192	4096	1908258	3280853	10422130	1294246813065504	5984134	7578564	6262248	9383277	4158362	4364357	7772279	10751510	
8192	8192	2006313	2957252	9002223	10315743	8577485	4625203	7224798	5264403	7367317	3978753	3364048	6224809	8725605
16384	4	1414860	2507478	6384903	7790612	7136122	2853347	5883111	5809993	8079126	2841079	2352294	6023899	6962596
16384	8	1531940	2646435	7192137	8881003	9204562	2811784	6537983	6551072	9372790	3016816	2547259	7147999	9153070
16384	16	1739477	2866320	7062054	921443610958635	3256722	6942900	7173368	9953854	2999042	2628116	7052632	8972610	
16384	32	1706518	2866320	7356911	984124111260308	3240597	7010899	7703281	12128809	3095359	2803068	7595993	10063176	
16384	64	1807988	3174578	7508843	1040289112449618	3802158	6993063	7775626	9991482	3196878	2869791	7379822	9740803	
16384	128	1790977	2988088	7403675	1020060211949538	3679205	5893706	7702417	11314072	3518156	2732514	7201936	9654587	
16384	256	1764761	2926249	6614125	8475718	9372790	4010753	6901759	6644181	10694297	3284743	2798274	5929815	8191806
16384	512	1746729	2673617	7245981	8687884	9438443	3019998	5917050	6939394	10618284	2889581	2679351	6716917	9834199
16384	1024	1833066	3119103	7278213	959259311643805	4487419	7656078	6806735	12109573	3385728	2912112	6787239	9097331	
16384	2048	1815631	2823801	7585093	889019410436067	3402660	7028827	5748753	11488082	3229024	3205079	7016626	7865515	
16384	4096	1763810	3041383	7950141	996684710577425	3946037	7353761	6097662	10182464	3715207	3425388	7349829	8991393	
16384	8192	1762091	2895303	7854726	8769931	8445510	3612858	6566722	5287006	8245869	4179557	4176255	6480642	9157949
16384	16384	1701701	2915325	6592551	6854943	7080244	3232822	4787877	3209419	5069376	3102906	2924630	4817414	6363028
32768	64	1786991	3148672	7100291	8173200	8999522	3244024	7459047	8180010	9029083	3491949	2992565	7546695	8064811
32768	128	1749036	3177058	7340332	8092829	8533503	3400960	7403597	8019635	8573962	3196938	3072167	7391652	8312602

- Analysis –
  - Below is graphical representation of results of my system with iozone
  - So here we have achieved almost below % of ideal value compared with iozone
  - Sequential Write Operation = 71.80%
  - Sequential Read Operation = 91.11%
  - Random Write Operation = 40.45 %
  - Random Read Operation = 118.36%

  - Sequential Operations

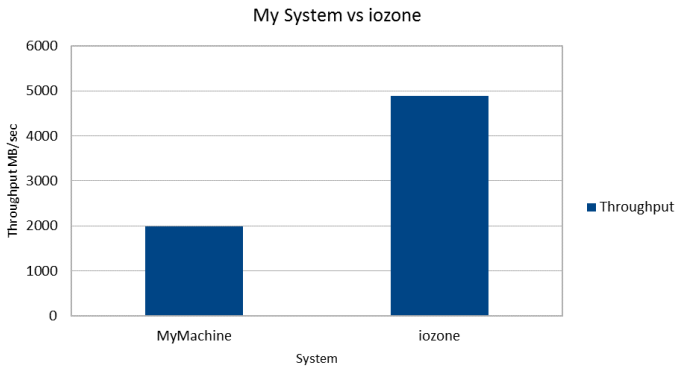


Sequential Write Operation

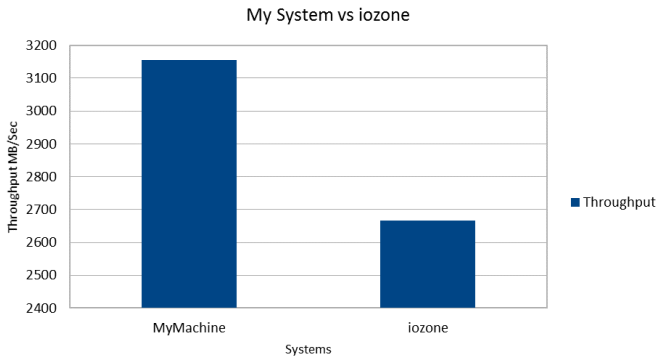


Sequential Read Operation

- Random Operations



Random Write Operation



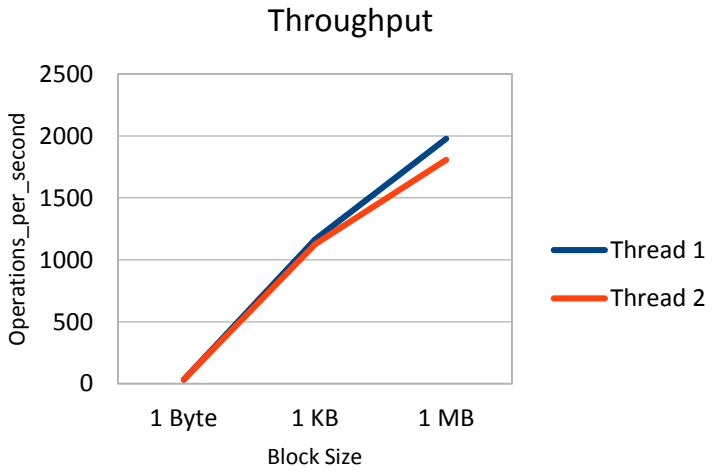
Random Read Operation

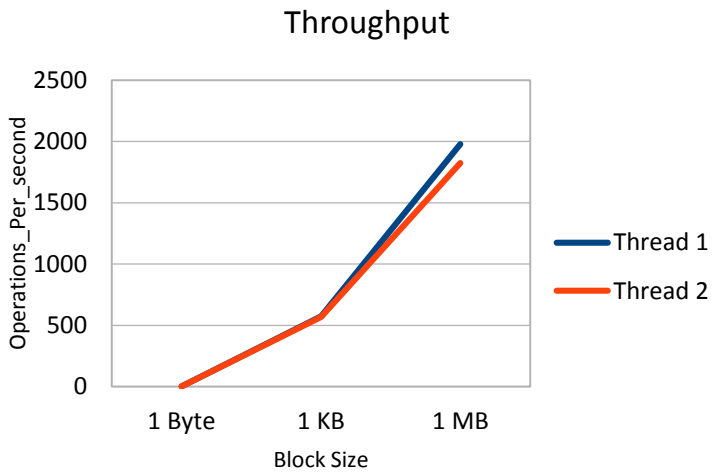
Disk Benchmarking:

- Write Operations (Throughput)

	Sequential	
Block Size	Thread 1	Thread 2
1 Byte	32.938076	32.615786
1 KB	1163.63636	1125.27473
1 MB	1978.44528	1807.88966

	Random	
Block Size	Thread 1	Thread 2
1 Byte	0.846762	0.845802
1 KB	572.067039	567.313019
1 MB	1978.44528	1823.61044



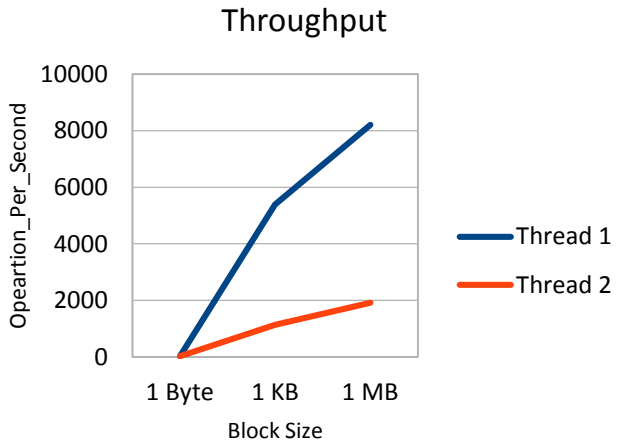


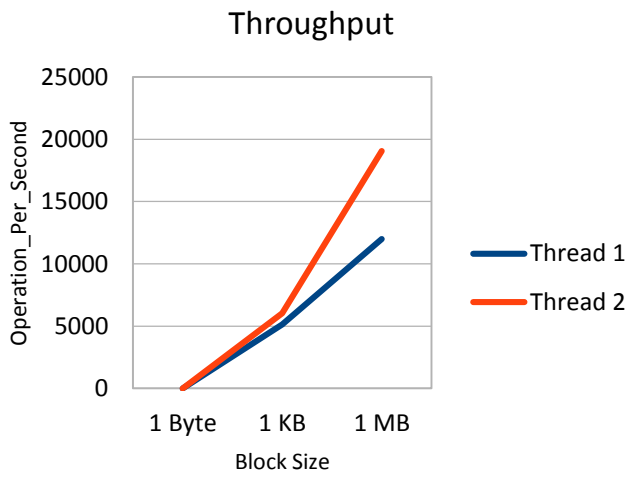
- Analysis –
  - Tables and graphs above are for Throughput of Write Operations for 1 thread and 2 threads.
  - From above graph we can conclude that throughput goes on increasing with increase in block size from 1 byte to 1 MB.
  - It increases in same fashion for both the operation i.e. with 1 thread and 2 thread.

- Read Operations (Throughput)

	Sequential	
Block Size	Thread 1	Thread 2
1 Byte	34.994756	33.090668
1 KB	5389.47368	1144.13408
1 MB	8206.542	1923.99266

	Random	
Block Size	Thread 1	Thread 2
1 Byte	6.4020487	6.4020487
1 KB	5120	6041.29794
1 MB	12000	19065.0182



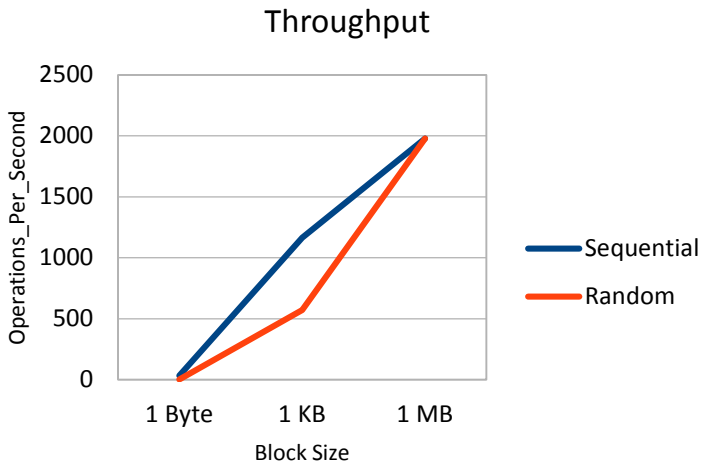


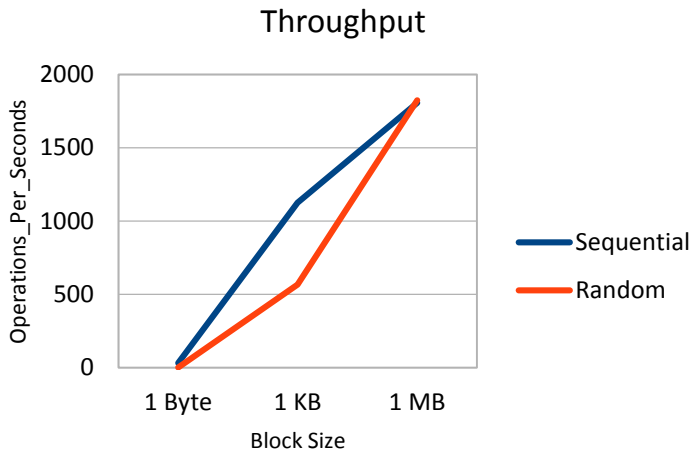
- Analysis –
  - Above figure represents the value of throughput for sequential and random read operation for 1 thread and 2 threads.
  - Here nature of graphs is different like in first graph the throughput for 1 thread is high than 2 threads, whereas in second graph throughput of 1 thread is lower than 2 threads.

- Write Operations (Throughput)

	Thread 1	
Block Size	Sequential	Random
1 Byte	32.938076	0.846762
1 KB	1163.63636	572.067039
1 MB	1978.44528	1978.44528

	Thread 2	
Block Size	Sequential	Random
1 Byte	32.615786	0.845802
1 KB	1125.27473	567.313019
1 MB	1807.88966	1823.61044



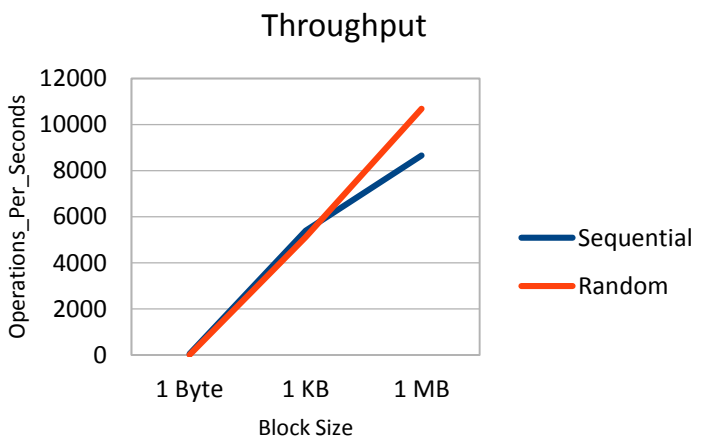


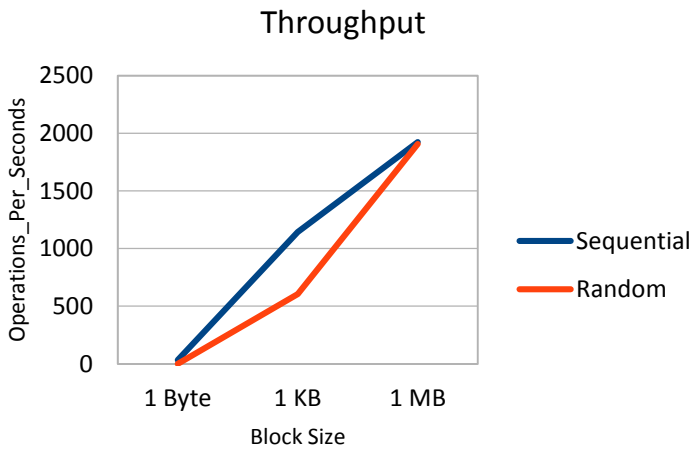
- Analysis-
  - Table and Graph above represent Throughput for Sequential and Random disk write operations
  - From above graph we can conclude that throughput keeps increasing with increase in block size, even it slightly varies for sequential and random operations. Its tread is same for sequential and Random.

- Read Operations (Throughput)

	Thread 1	
Block Size	Sequential	Random
1 Byte	44.99476	6.4020487
1 KB	5389.47368	5120.554
1 MB	8651.2545	10689.65

	Thread 2	
Block Size	Sequential	Random
1 Byte	33.090668	0.825631
1 KB	1144.13408	604.129794
1 MB	1923.99266	1906.50182



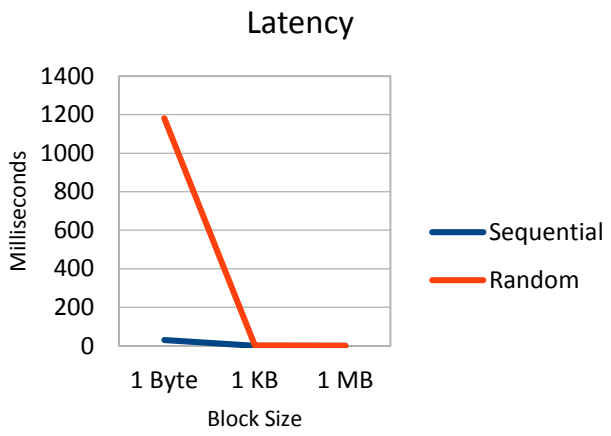


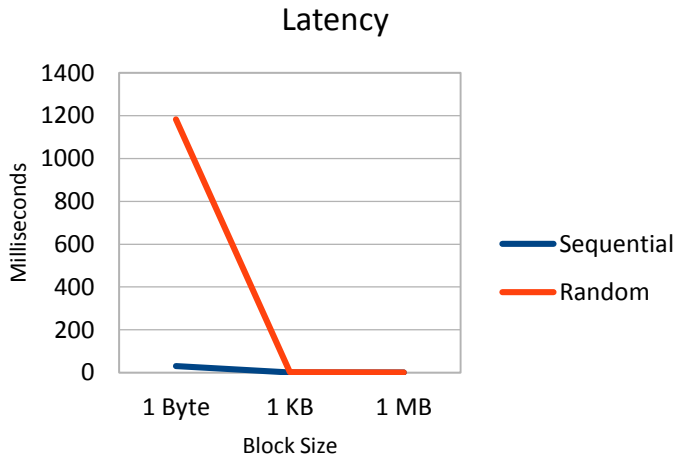
- Analysis-
  - Table and Graph above represent Throughput for Sequential and Random disk read operations
  - From above graph we can conclude that throughput keeps increasing with increase in block size, even it slightly varies for sequential and random operations. Its tread is same for sequential and Random.

- Write Operations (Latency)

	Thread 1	
	Sequential	Random
1 Byte	30.36	1180.97
1 KB	0.859375	1.748047
1 MB	0.505447	0.505447

	Thread 2	
	Sequential	Random
1 Byte	30.66	1182.31
1 KB	0.888672	1.762695
1 MB	0.553131	0.548363



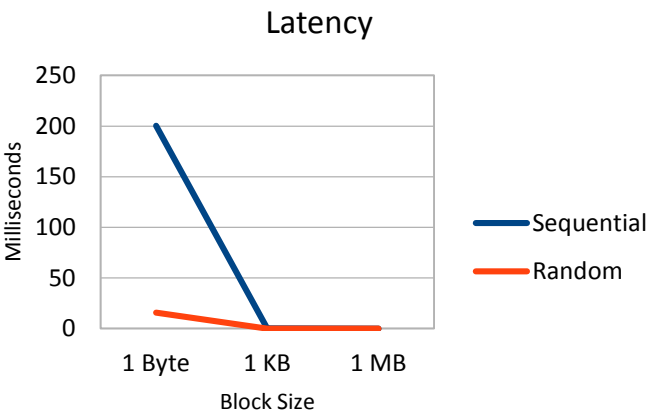


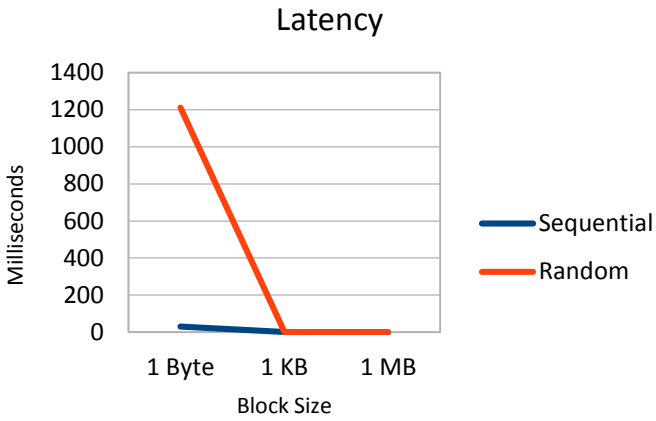
- Analysis-
  - Table and Graph above represent Latency for 1 Thread and 2 threads for disk write operations
  - From above graph we can conclude that Latency is high when we have block size of 1 Byte and it gets lower with increase in size of block. (In case of Random write operations)

- Read Operations (Latency)

	Thread 1	
	Sequential	Random
1 Byte	200.21	15.62
1 KB	0.185547	0.019531
1 MB	0.11053	0.019093

	Thread 2	
	Sequential	Random
1 Byte	30.22	1211.195
1 KB	0.874023	1.655273
1 MB	0.519753	0.524521

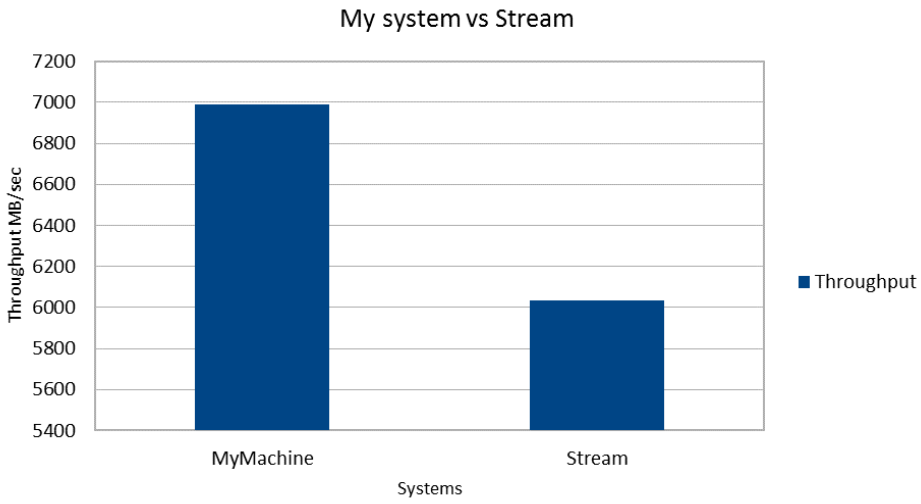




- Analysis-
  - Table and Graph above represent Latency for 1 Thread and 2 threads for disk read operations
  - From above graph we can conclude that Latency is high when we have block size of 1 Byte and it gets lower with increase in size of block.

- **Disk Theoretical Performance –**
  - As mentioned on Amazon Web Service Site Disk has performance of 160 mbps

- **<u>Extra Credit</u>**
  - Here we have compared our results with stream benchmark. Below is screen shot for stream systems outputs.
  - So here we have achieved almost below % of ideal value compared with stream
  - Read + Write Operation = 115%




```
ubuntu@ip-172-31-56-58: ~
ubuntu@ip-172-31-56-58:~$ ls
stream.c
ubuntu@ip-172-31-56-58:~$ sudo chmod 400 stream.c
ubuntu@ip-172-31-56-58:~$ gcc -o streamapp stream.c
ubuntu@ip-172-31-56-58:~$ ./streamapp
-------------------------------------------------------------
STREAM version $Revision: 5.10 $
-------------------------------------------------------------
This system uses 8 bytes per array element.
-------------------------------------------------------------
Array size = 10000000 (elements), Offset = 0 (elements)
Memory per array = 76.3 MiB (= 0.1 GiB).
Total memory required = 228.9 MiB (= 0.2 GiB).
Each kernel will be executed 10 times.
 The *best* time for each kernel (excluding the first iteration)
 will be used to compute the reported bandwidth.
-------------------------------------------------------------
Your clock granularity/precision appears to be 1 microseconds.
Each test below will take on the order of 26612 microseconds.
   (= 26612 clock ticks)
Increase the size of the arrays if this shows that
you are not getting at least 20 clock ticks per test.
-------------------------------------------------------------
WARNING -- The above is only a rough guideline.
For best results, please be sure you know the
precision of your system timer.
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:           6034.6     0.026914     0.026514     0.027427
Scale:          5955.9     0.027616     0.026864     0.028322
Add:            8750.5     0.028082     0.027427     0.028816
Triad:          8024.9     0.030805     0.029907     0.031178
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
-------------------------------------------------------------
ubuntu@ip-172-31-56-58:~$
```