CS553 Cloud Computing
Programming Assignment 1

Evaluation

submitted by:

Chiranjeevi Ankamreddy
A20359837

The assignment carries out benchmark different parts of a computer system, from the CPU, memory, disk, and network .
I've evaluated the Benchmark different parts of a computer system, from the CPU, disk, and network on Amazon AWS cloud t2.micro instances.

CPU BENCHMARKING:

I've evaluated the processor speed in terms of floating point operations per second (Giga FLOPS, 109 FLOPS) and integer operations per second (Giga IOPS, 109 IOPS).I've evaluated the processor speed at varying levels of concurrency (1 thread, 2 threads, 4 threads).Results mentioned in the table below are averages of GFLOPS and GIOPS for their respective threads.
Configuration of t2.micro instance is as bellow:Operating System : Ubuntu .Ram:1,core:1

One Thread :
GFLOPS : 9.84
GIOPS : 8.91

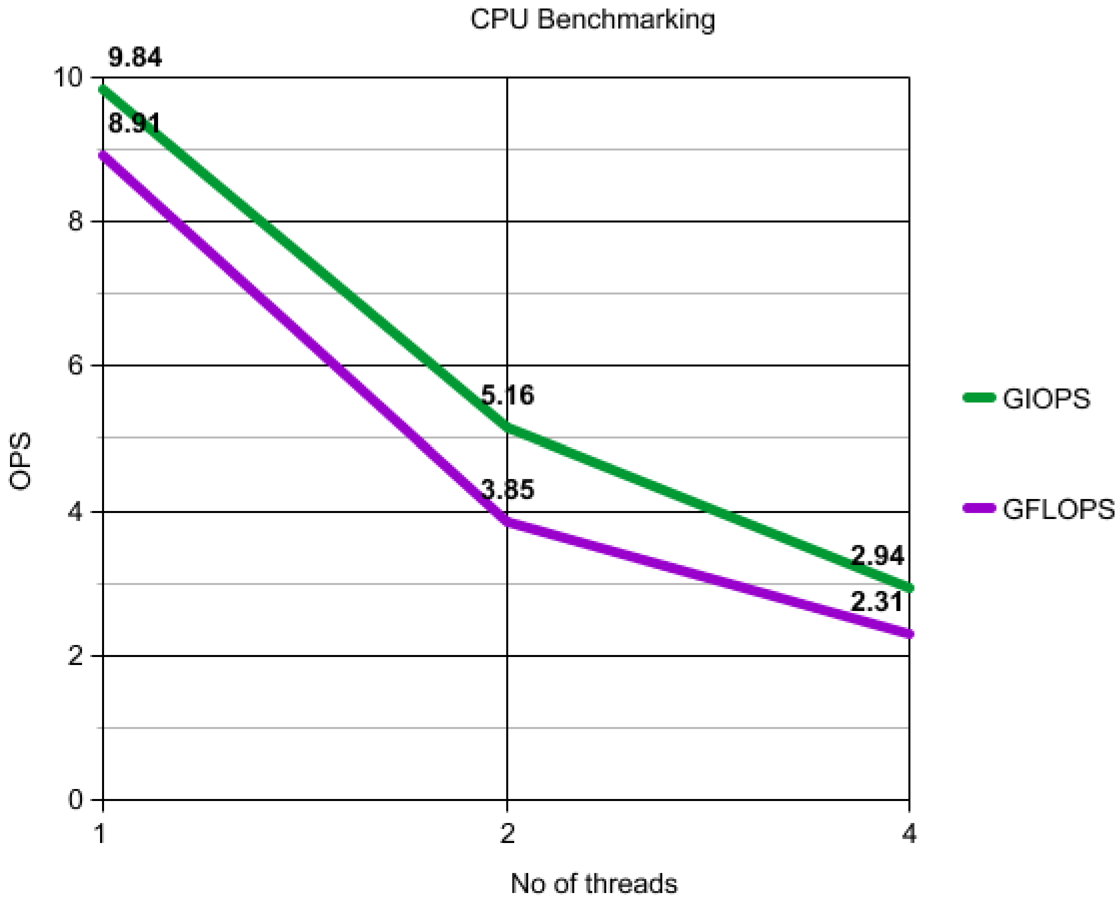
Two Threads :
GFLOPS : 5.16
GIOPS : 3.85

Four Threads:
GFLOPS : 2.94
GIOPS : 2.31

GRAPH :

This graph plots the GFLOPS and GIOPS versus the number of threads. I have evaluated 1,2,4 threads and it's been observed that the GFLOPS and GIOPS are highest when run on a single thread.

X -axis: Number of Threads
Y-axis: OPS



Theoretical Peak Performance = No.of.cores*cpu Frequency*threads*Ins/cycles

= 1 *2.46 * 1 * 16

= 41.06 GFLOPS

Conclusion :

Performance of Integral operation is higher than performance for floating operation.

The optimal number of threads to get the best performance is 1.

Theoretical peak performance of the processor is 41.6 GFLOPS

Efficiency obtained with respect to theoretical performance is 26.8%

Separate Experiment:

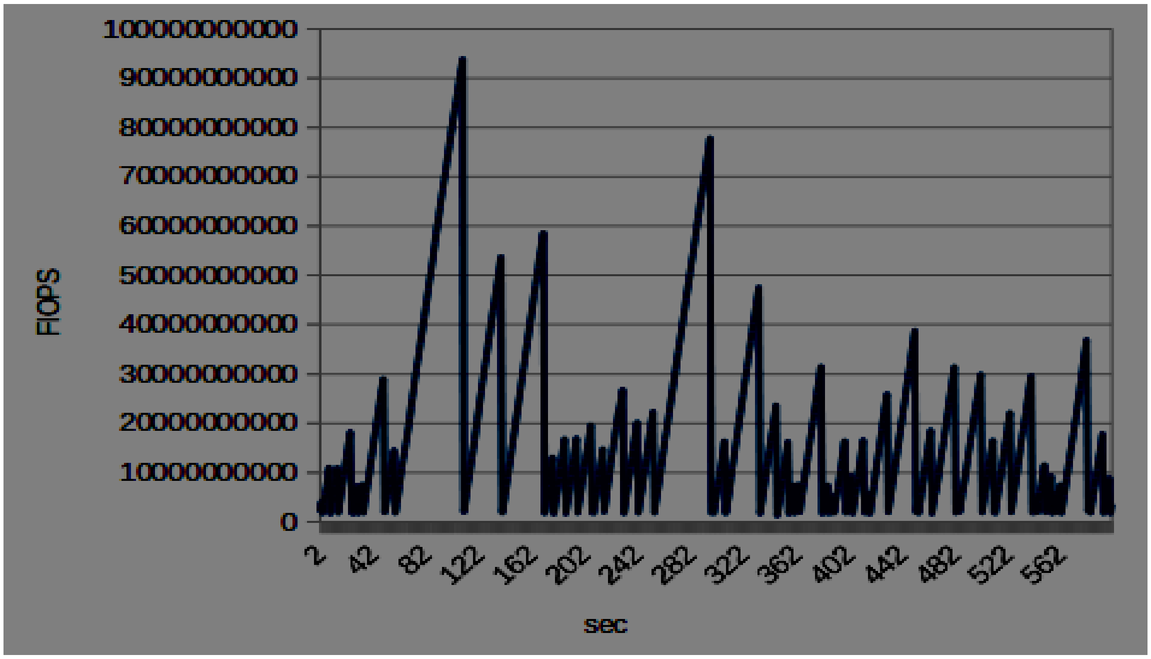
I have evaluated benchmark on floating point and integer instructions and 4 threads for a 10-minute period for each one and took samples every second on how many instructions per second were achieved during the experiment.

1.FLOPS

Number of threads:4

Graph: This graph plot the FLOPS versus time (0 to 10 min) . I have evaluated 4 threads for every Second, Observed that the FLOPs were changing when time increases.

X -axis: Time(seconds)
Y-axis: FLOPS



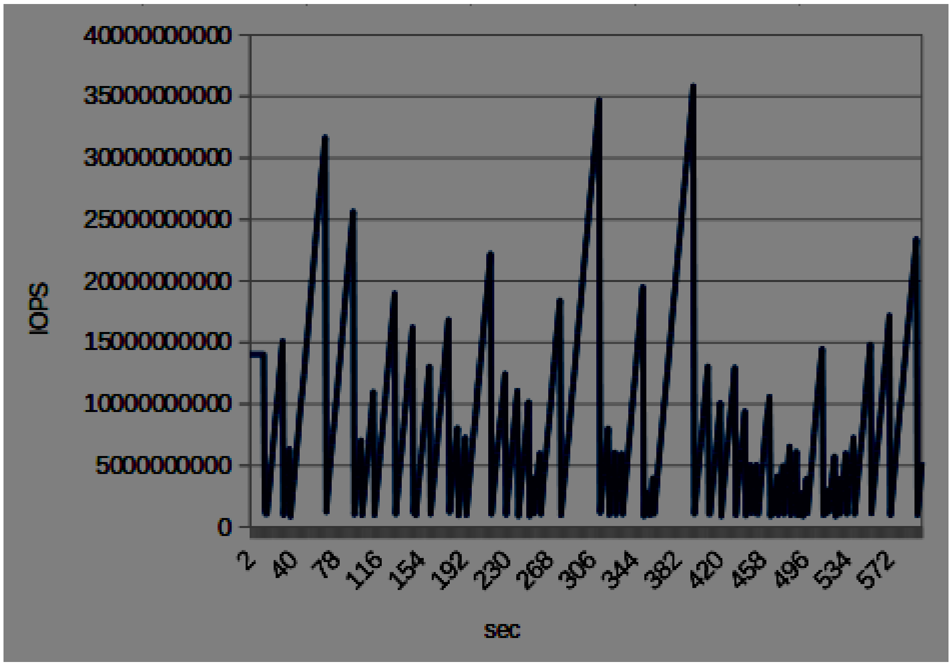
2.IOPS

Number of threads:4

Graph:

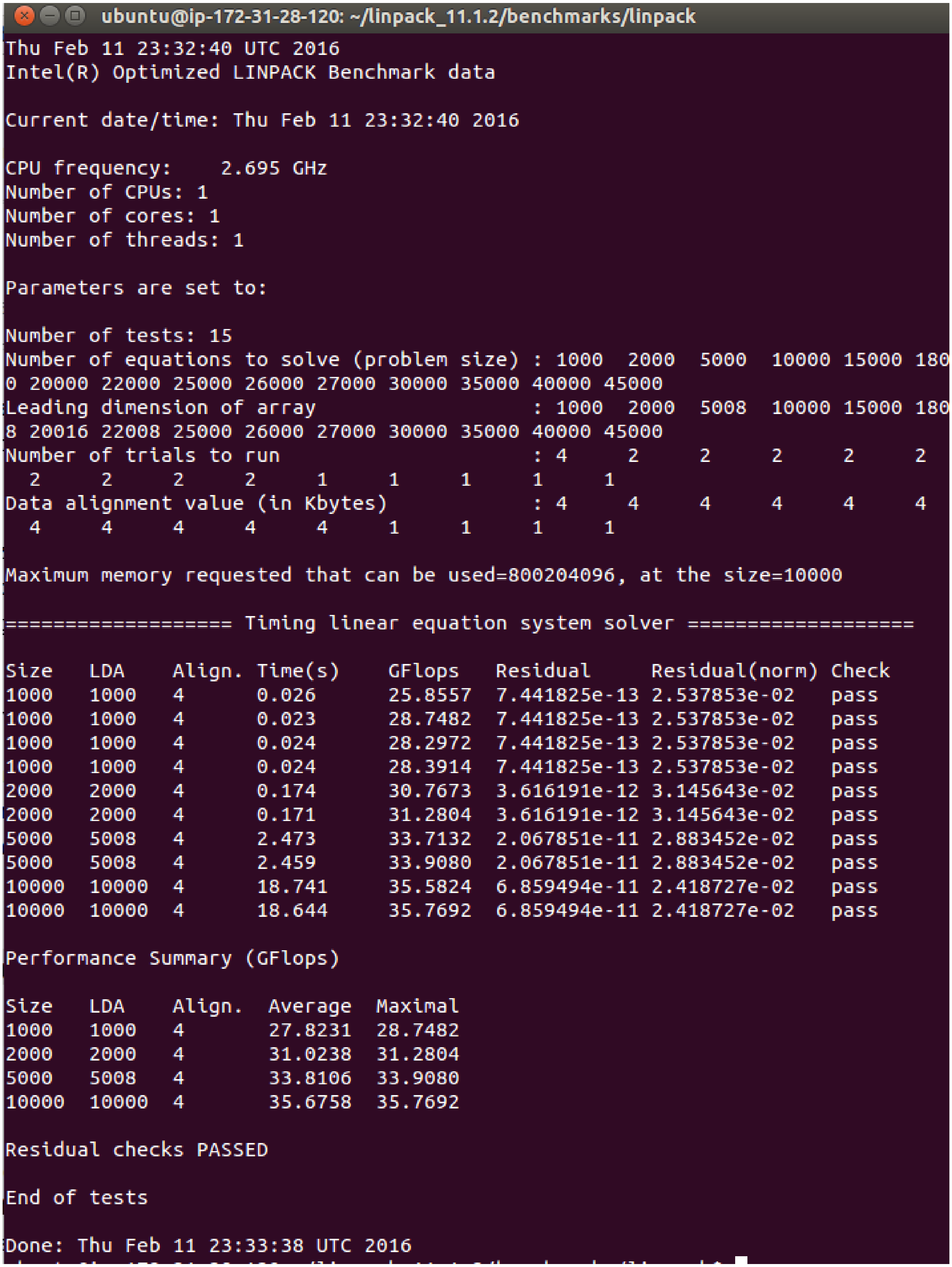
Graph: This graph plot the IOPS versus time (0 to 10 min) . I have evaluated 4 threads for every Second, Observed that the IOPS are increasing and decreasing as time increases.

X -axis: Time(seconds)
Y-axis: IOPS



EXTRA CREDIT :

LINPACk benchmark:



ubuntu@ip-172-31-28-120: ~/linpack_11.1.2/benchmarks/linpack

```
Thu Feb 11 23:32:40 UTC 2016
Intel(R) Optimized LINPACK Benchmark data

Current date/time: Thu Feb 11 23:32:40 2016

CPU frequency:    2.695 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 180
0 20000 22000 25000 26000 27000 30000 35000 40000 45000
Leading dimension of array                  : 1000  2000  5008  10000 15000 180
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      0.026      25.8557  7.441825e-13 2.537853e-02   pass
1000   1000   4      0.023      28.7482  7.441825e-13 2.537853e-02   pass
1000   1000   4      0.024      28.2972  7.441825e-13 2.537853e-02   pass
1000   1000   4      0.024      28.3914  7.441825e-13 2.537853e-02   pass
2000   2000   4      0.174      30.7673  3.616191e-12 3.145643e-02   pass
2000   2000   4      0.171      31.2804  3.616191e-12 3.145643e-02   pass
5000   5008   4      2.473      33.7132  2.067851e-11 2.883452e-02   pass
5000   5008   4      2.459      33.9080  2.067851e-11 2.883452e-02   pass
10000  10000  4      18.741     35.5824  6.859494e-11 2.418727e-02   pass
10000  10000  4      18.644     35.7692  6.859494e-11 2.418727e-02   pass

Performance Summary (GFlops)

Size   LDA    Align.  Average  Maximal
1000   1000   4       27.8231  28.7482
2000   2000   4       31.0238  31.2804
5000   5008   4       33.8106  33.9080
10000  10000  4       35.6758  35.7692

Residual checks PASSED

End of tests

Done: Thu Feb 11 23:33:38 UTC 2016
```

Best performance achieved on running linpack benchmark is 35.76 GFLOPS

The efficiency achieved by linpack when compared to theoretical performance is 34.4%

DISK BENCHMARKING

Sequential Access:
I have Evaluated the disk speed, it includes read operations, write operations, sequential access, varying block sizes (1B, 1KB, 1MB), and varying the concurrency (1 thread, 2 threads) ,And Measured throughput (MB/sec) and latency (ms).
Number of Threads: 1

Buffer Size : 1 Byte
Read Speed : 0.032353 Mb/Sec
Write Speed : 0.0497560 Mb/Sec
Latency : 0.017329 msec

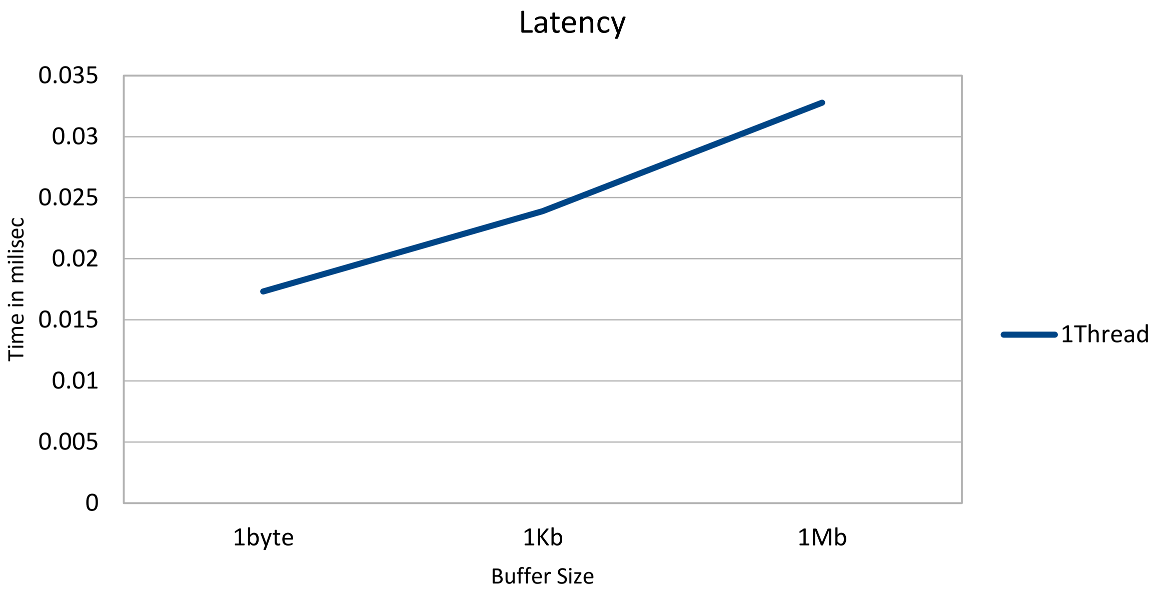
Buffer Size : 1 kiloByte
Read Speed : 31.76 Mb/Sec
Write Speed : 40.38 Mb/Sec
Latency : 0.02389 msec

Buffer Size : 1 Megabytes
Read Speed : 55.43 Mb/Sec
Write Speed : 53.13 Mb/Sec
Latency : 0.032769 msec

GRAPH

Latency:

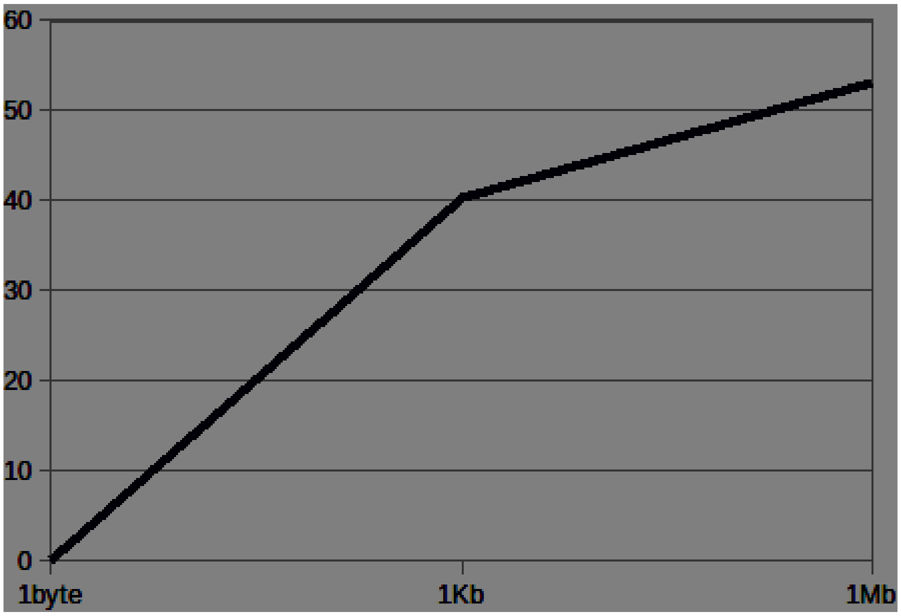
X-axis: Buffer Size
Y-axis: Time(ms)



Throughput:

X-axis: Buffer Size

Y-axis: Write Speed



Number of Threads: 2

Buffer Size : 1 Byte
Read Speed : 0.087625 Mb/Sec
Write Speed : 0.1323 Mb/Sec
Latency : 0.0072755 msec

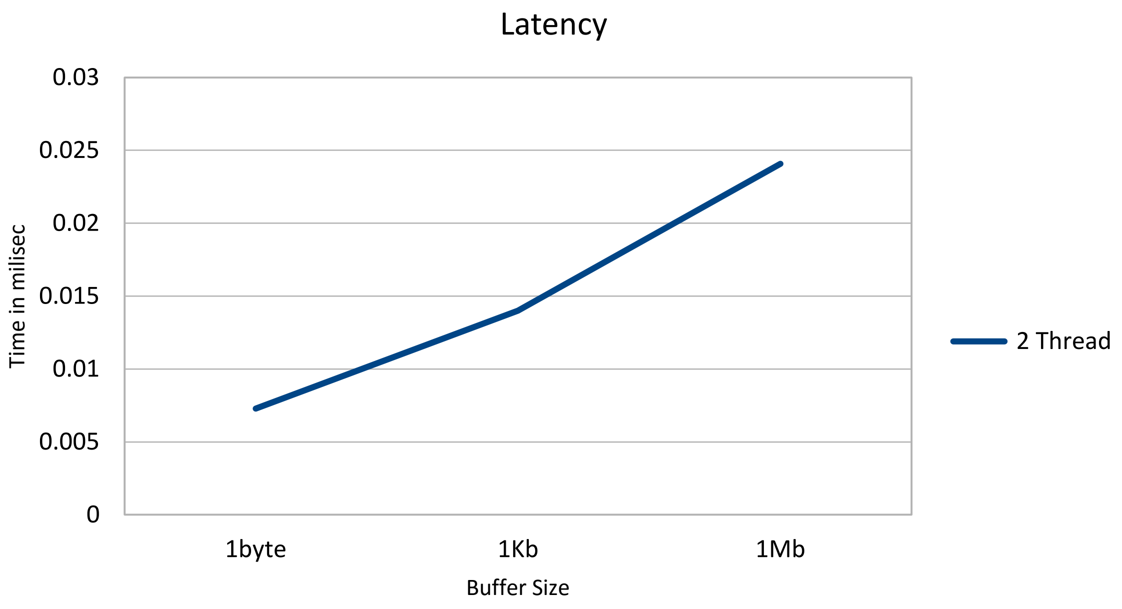
Buffer Size : 1 kiloByte
Read Speed : 53.93Mb/Sec
Write Speed : 69.29 Mb/Sec
Latency : 0.014 msec

Buffer Size : 1 MegaByte
Read Speed : 192.55 Mb/Sec
Write Speed : 206.45 Mb/Sec
Latency : 0.0240875 msec

GRAPH

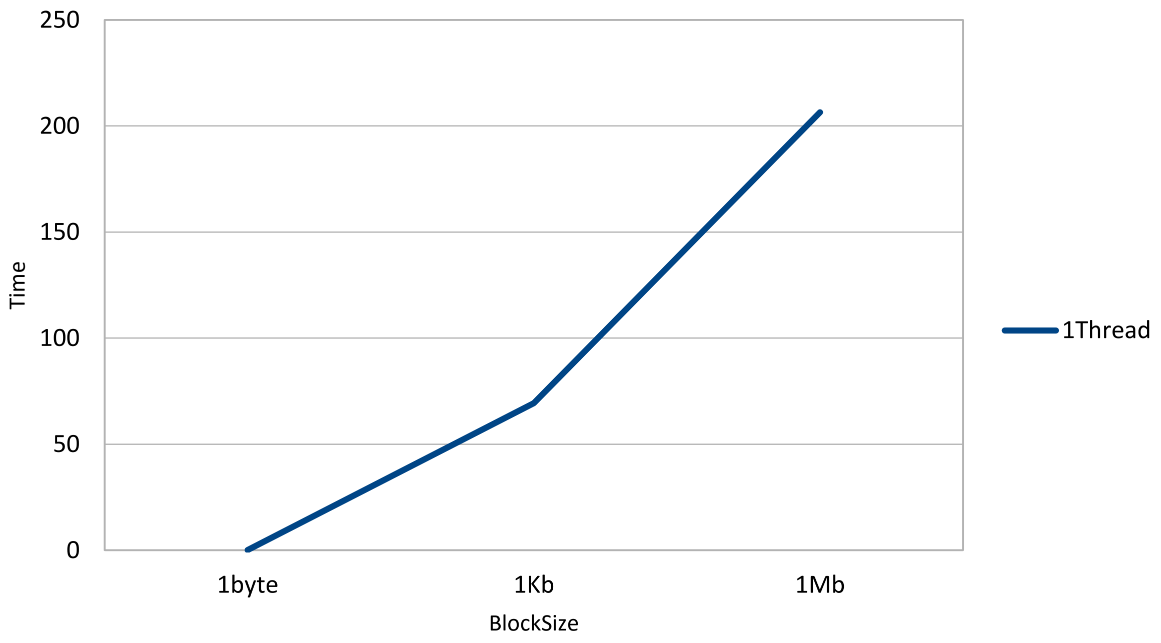
Latency:

X-axis: Buffer Size
Y-axis: LTime



Throuhput:

X-axis: Buffer Size
Y-axis: Write Speed



Random Access:

I have Evaluated the disk speed, it includes read operations, write operations, random access, varying block sizes (1B, 1KB, 1MB), and varying the concurrency (1 thread, 2 threads) ,And Measured throughput (MB/sec) and latency (ms).

Number of Threads: 1

Buffer Size : 1 Byte
Read Speed : 0.20564 Mb/Sec
Write Speed : 0.08153 Mb/Sec
Latency : 0.009937 msec

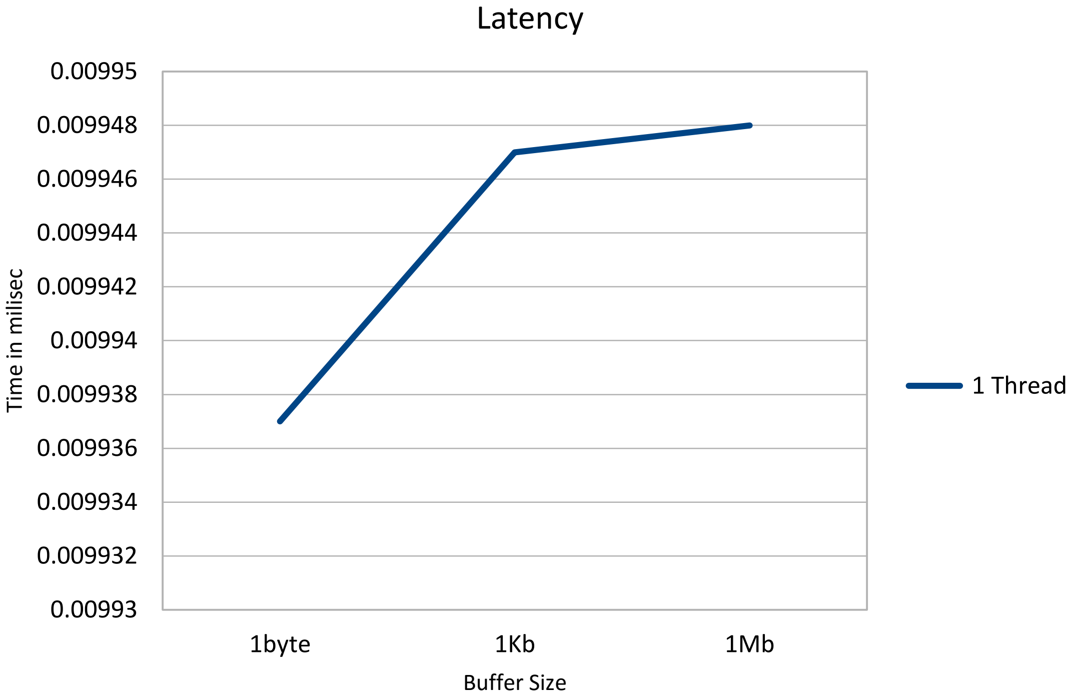
Buffer Size : 1 kiloByte
Read Speed : 4.0243 Mb/Sec
Write Speed : 14.549 Mb/Sec
Latency : 0.009937 msec

Buffer Size : 1 MegaByte
Read Speed : 83.49 Mb/Sec
Write Speed : 210.195 Mb/Sec
Latency : 0.009937 msec

GRAPH

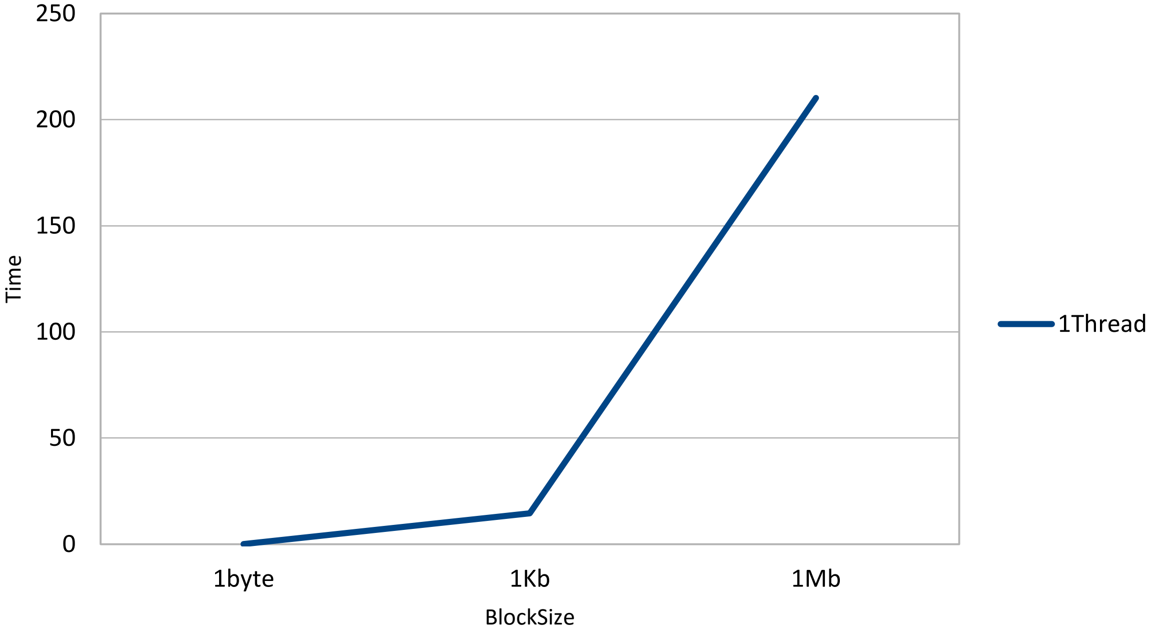
Latency:

X-axis: Buffer Size
Y-axis: Time



Throuhput:

X-axis: Buffer Size
Y-axis: Write Speed



Number of Threads: 2

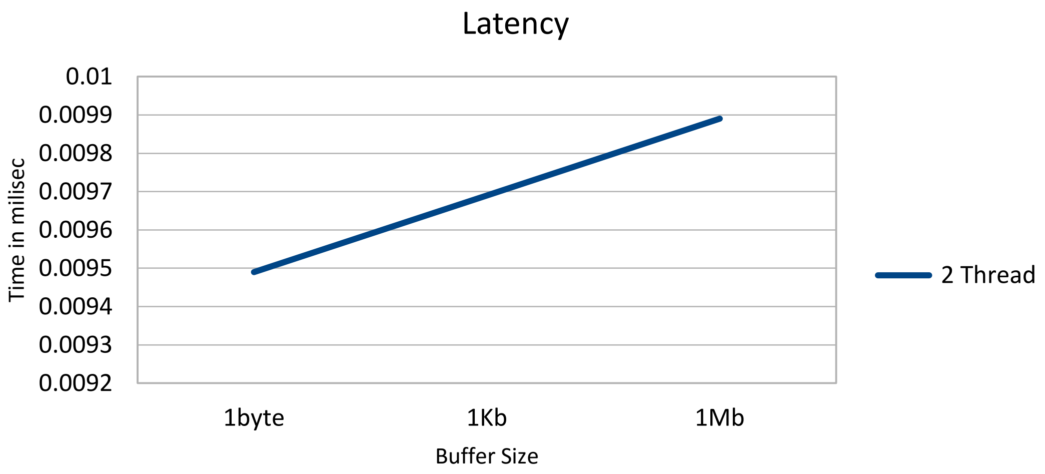
Buffer Size : 1 Byte
Read Speed : 0.144245 Mb/Sec
Write Speed : 0.17461 Mb/Sec
Latency : 0.00949 msec

Buffer Size : 1 kiloByte
Read Speed : 0.26457 Mb/Sec
Write Speed : 0.35461 Mb/Sec
Latency : 0.00949 msec

Buffer Size : 1 MegaByte
Read Speed : 147.92 Mb/Sec
Write Speed : 178.803 Mb/Sec
Latency : 0.00949 msec

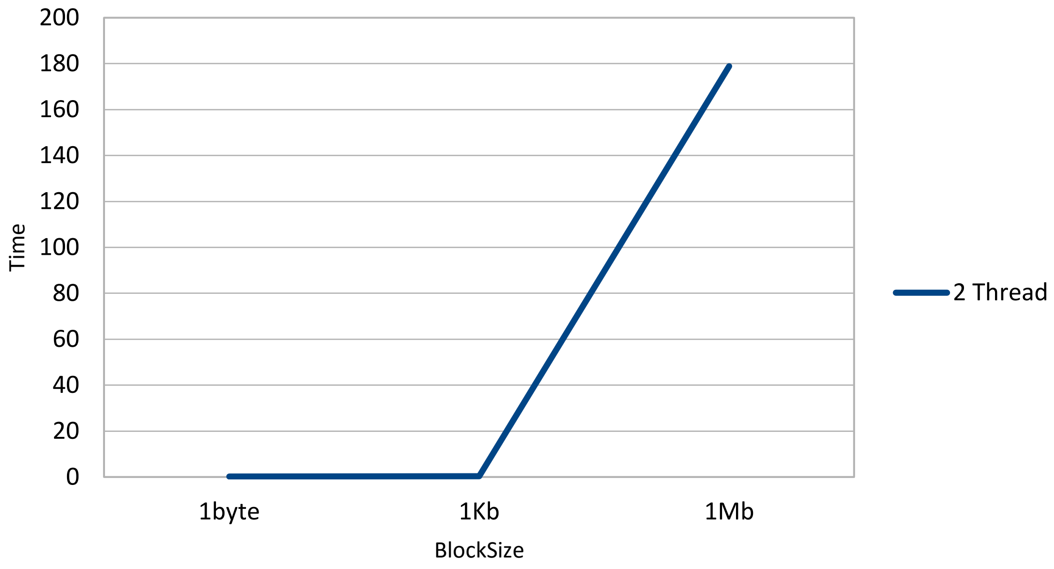
GRAPH

Latency:
X-axis: Buffer Size
Y-axis: Time



Throughput:

X-axis: Buffer Size
Y-axis: Write Speed



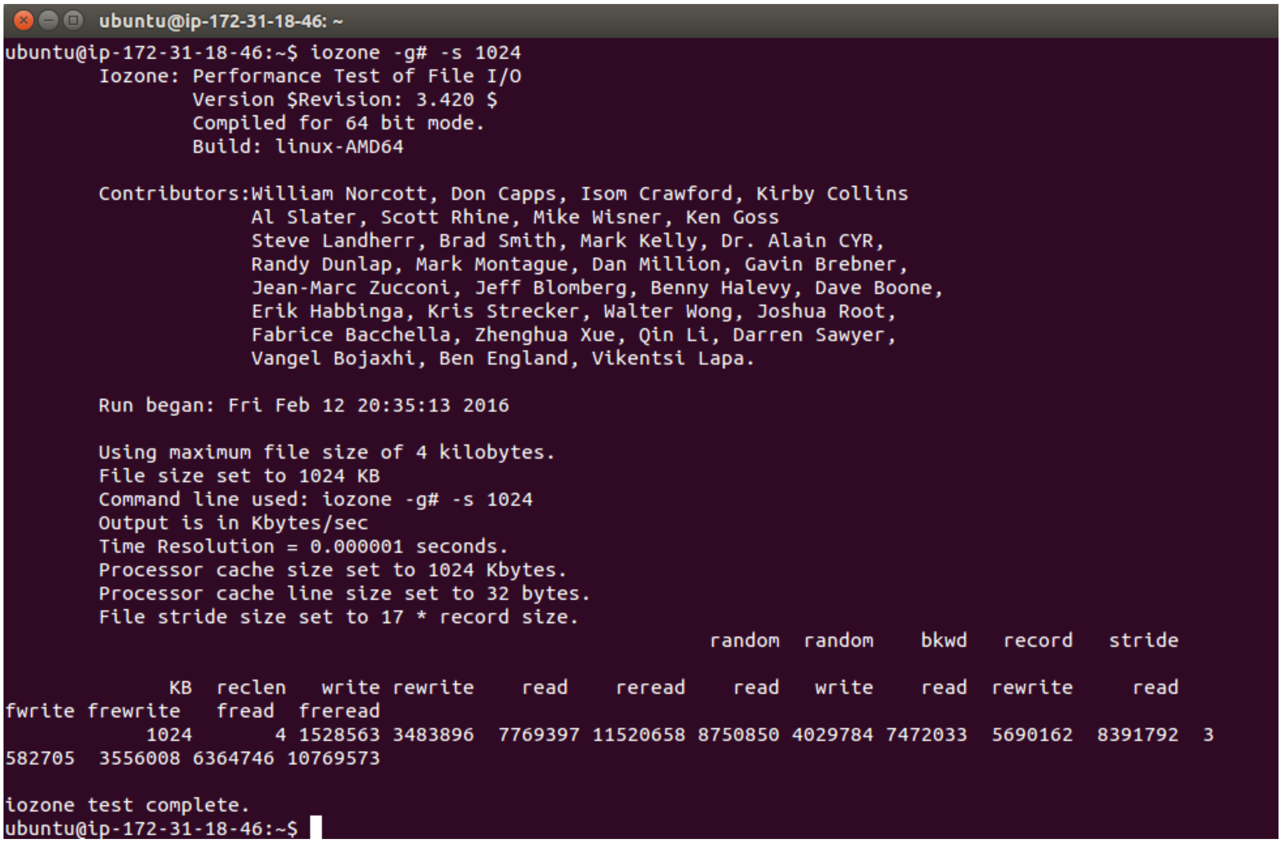
Conclusion:

1. Sequential Write operation give higher performance than Random Write.

2. Theoretical performance of disk as given on Amazon t2.instance: 158 Mbps.

EXTRA CREDIT:

IOZone benchmark:



```
ubuntu@ip-172-31-18-46: ~
ubuntu@ip-172-31-18-46:~$ iozone -g# -s 1024
        Iozone: Performance Test of File I/O
                Version $Revision: 3.420 $
                Compiled for 64 bit mode.
                Build: linux-AMD64

        Contributors:William Norcott, Don Capps, Isom Crawford, Kirby Collins
                     Al Slater, Scott Rhine, Mike Wisner, Ken Goss
                     Steve Landherr, Brad Smith, Mark Kelly, Dr. Alain CYR,
                     Randy Dunlap, Mark Montague, Dan Million, Gavin Brebner,
                     Jean-Marc Zucconi, Jeff Blomberg, Benny Halevy, Dave Boone,
                     Erik Habbinga, Kris Strecker, Walter Wong, Joshua Root,
                     Fabrice Bacchella, Zhenghua Xue, Qin Li, Darren Sawyer,
                     Vangel Bojaxhi, Ben England, Vikentsi Lapa.

        Run began: Fri Feb 12 20:35:13 2016

        Using maximum file size of 4 kilobytes.
        File size set to 1024 KB
        Command line used: iozone -g# -s 1024
        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
            1024       4 1528563 3483896  7769397 11520658 8750850 4029784 7472033  5690162  8391792  3
582705  3556008 6364746 10769573

iozone test complete.
ubuntu@ip-172-31-18-46:~$
```

Best performance achieved on running IOZone benchmark is 4029.784

Theoretical Performance is 568.

Throughput achieved by My system when compared to IOZone for Sequential Write and read is 32% ,40.6 ,And random is 18.6% ,22.9%.

The efficiency achieved by IOZone when compared to theoretical performance is 14.09%

NETWORK

I have Evaluated the network speed between 2 instances ,it includes the TCP protocol stack, UDP, varying packet/buffer size (1B, 1KB, 64KB), and varying the concurrency (1 thread & 2 threads),measured are throughput (Megabits per second, Mb/sec) and latency (ms).

TCP:

One Thread:

Buffer Size : 1 Byte
Speed : 47.6 Mb/Sec
Latency : 21 msec

Buffer Size : 1 kiloByte
Speed : 58.082 Mb/Sec
Latency : 17 msec

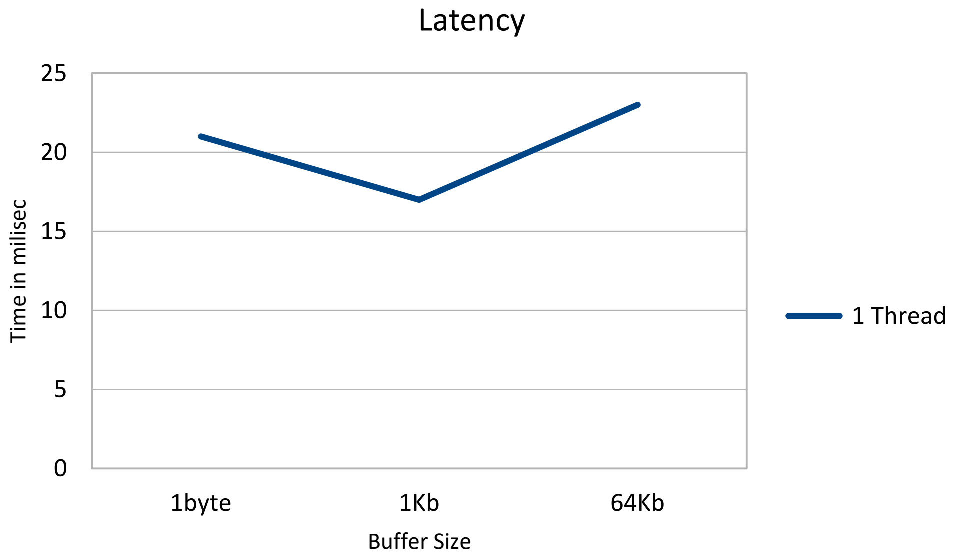
Buffer Size : 1 MegaByte
Speed : 43.47Mb/Sec
Latency : 23 msec

GRAPH

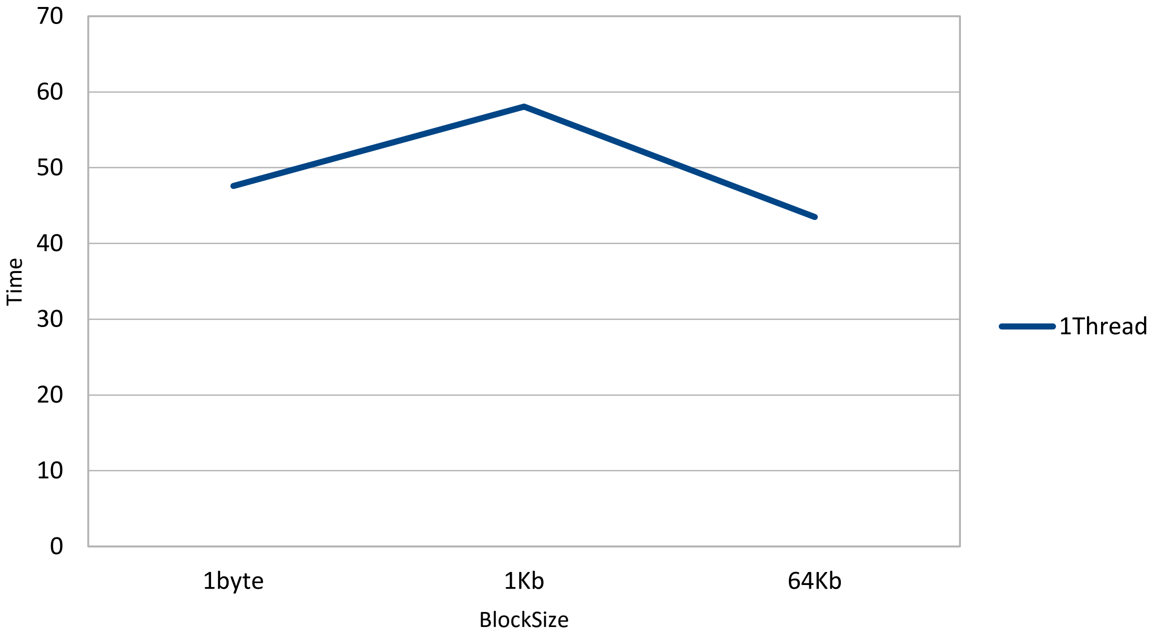
Latency:

1.Latency

X -axis: Buffer Size
Y-axis: Time(ms)



Throughput:
X -axis: Buffer Size
Y-axis: Speed



Two Threads:

Buffer Size : 1 Byte
Speed : 37 Mb/Sec
Latency : 30 msec

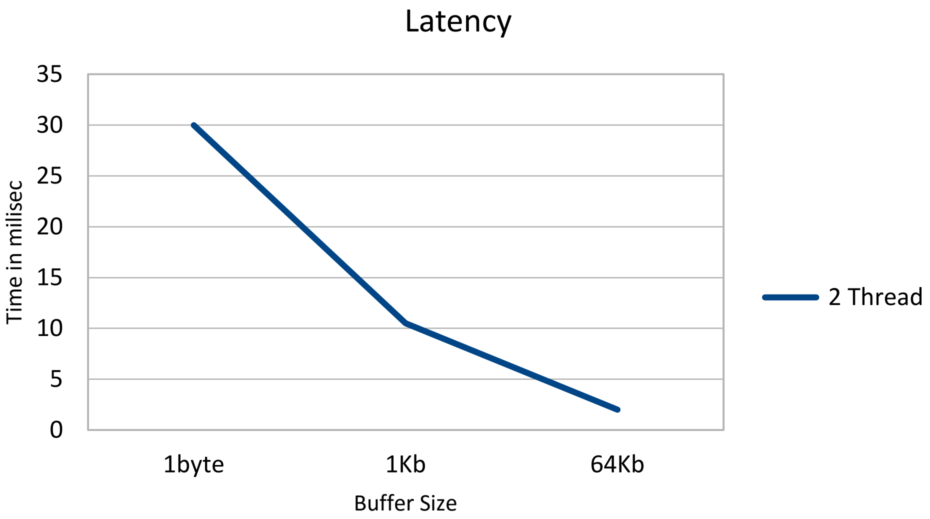
Buffer Size : 1 kiloByte
Speed : 91 Mb/Sec
Latency : 10.5 msec

Buffer Size : 1 MegaByte
Speed : 499.9 Mb/Sec
Latency : 2 msec

GRAPH

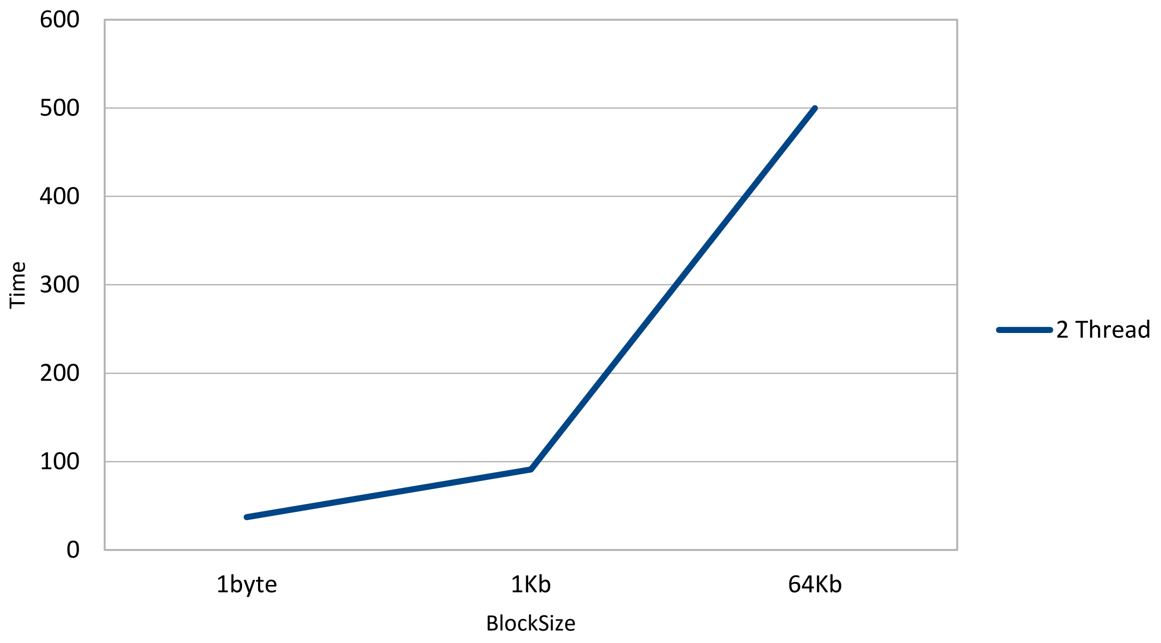
Latency:

X-axis: Buffer Size
Y-axis: Time



Throughput:

X-axis: Buffer Size
Y-axis: Speed



UDP:

I have Evaluated the network speed between 2 instances ,it includes the UDP Protocol, varying packet/buffer size (1B, 1KB, 64KB), and varying the concurrency (1 thread & 2 threads),measured are throughput (Megabits per second, Mb/sec) and latency (ms).

Buffer Size : 1 Byte
Speed : 71Mb/Sec
Latency : 0.124 msec

Buffer Size : 1 kiloByte
Speed : 114 Mb/Sec
Latency : 0.087 msec

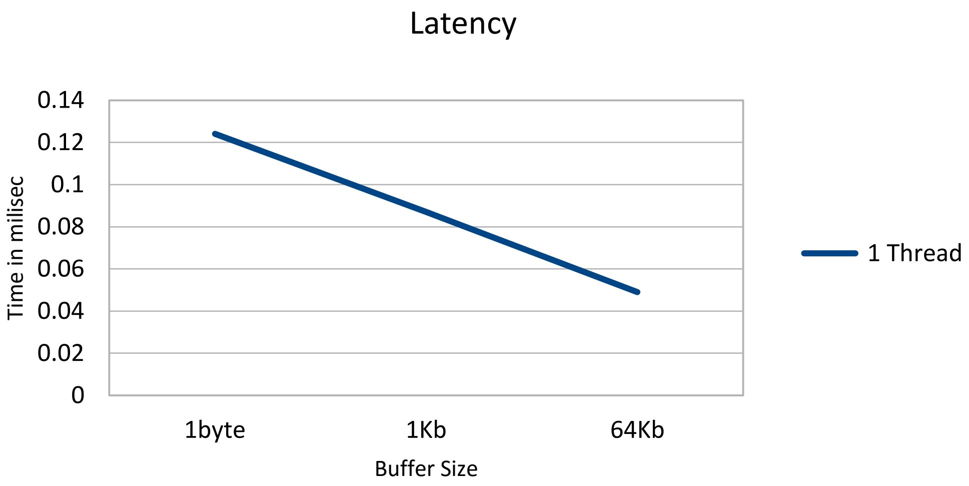
Buffer Size : 1 MegaByte
Speed : 168Mb/Sec
Latency : 0.049 msec

GRAPH

1.Latency

X -axis: Buffer Size

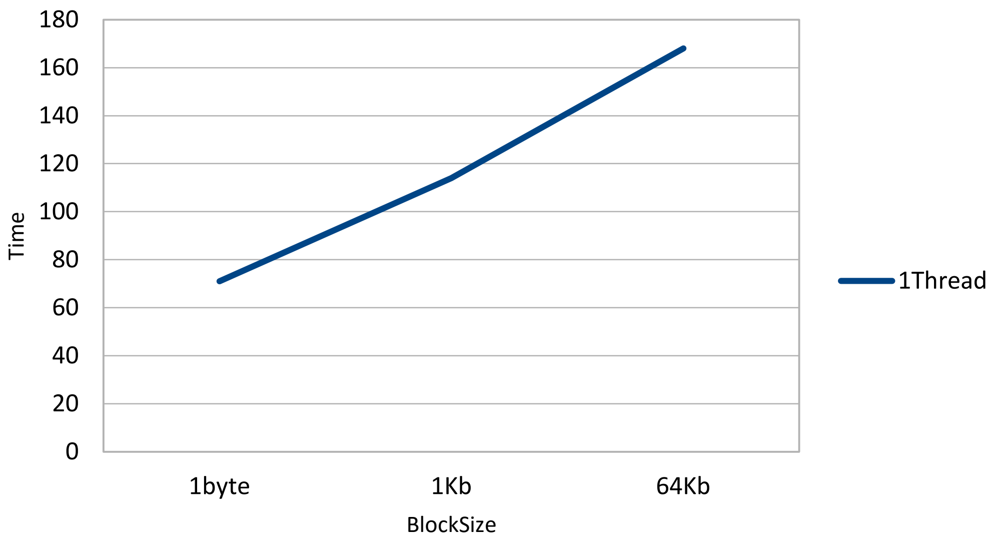
Y-axis: Latency and Speeds



2.Throughput

X-axis: Buffer Size

Y-axis: Speed



Two Threads:

Buffer Size : 1 Byte

Speed : 131Mb/Sec

Latency : 0.097 msec

Buffer Size : 1 kiloByte

Speed : 162 Mb/Sec

Latency : 0.0678 msec

Buffer Size : 1 MegaByte

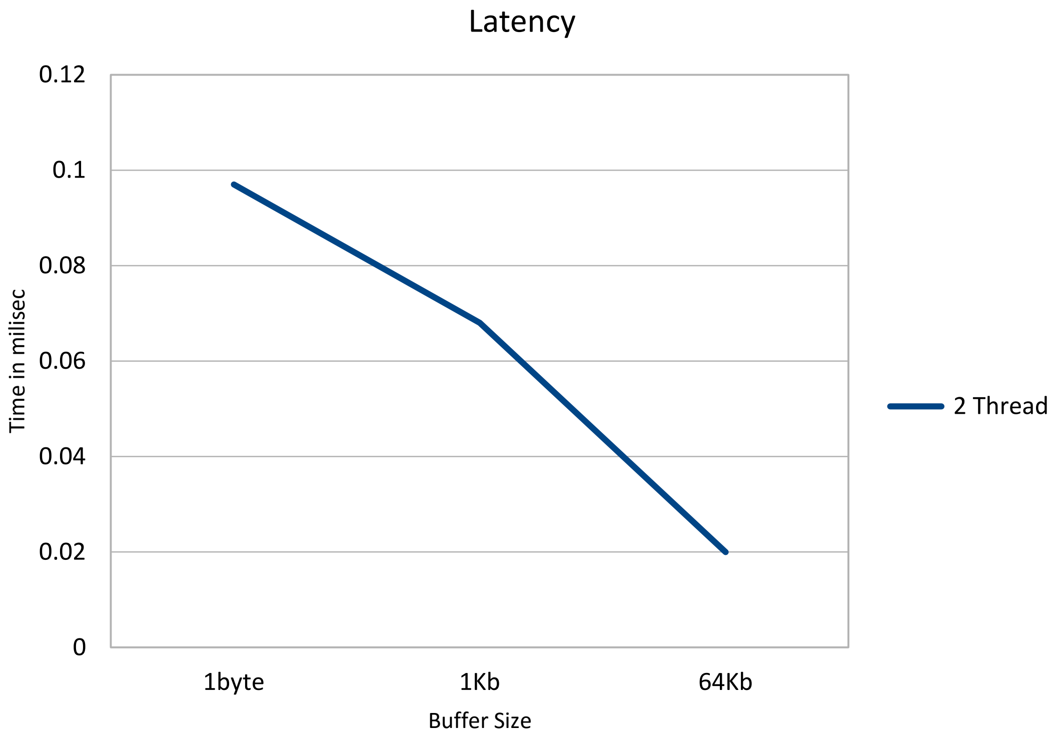
Speed : 248Mb/Sec

Latency : 0.02 msec

<u>GRAPH</u>

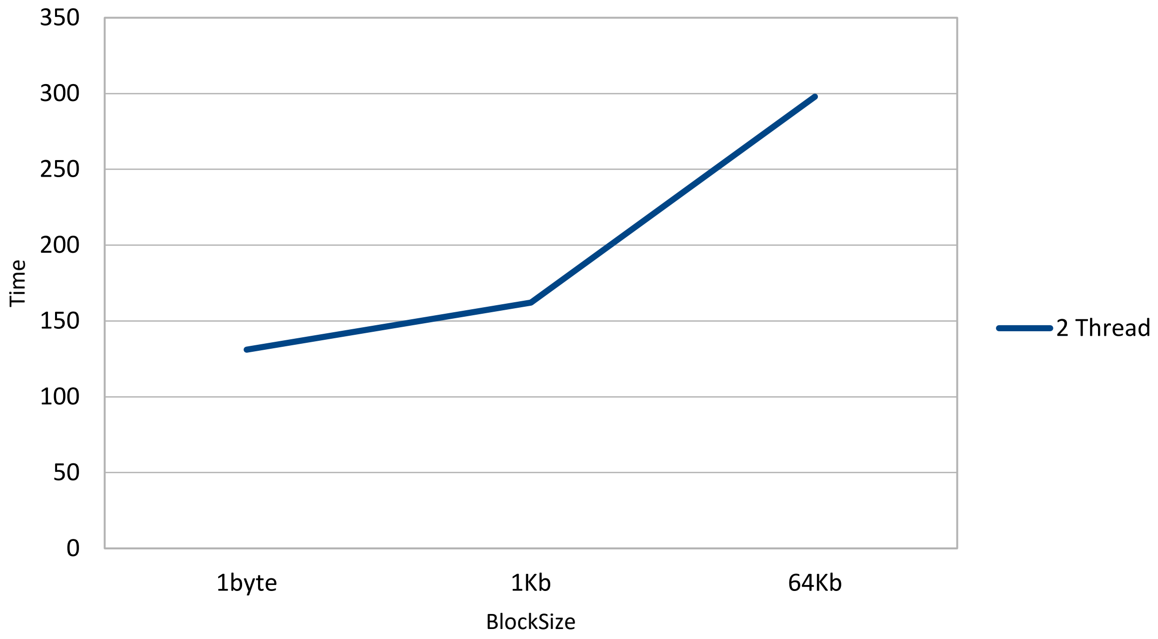
Latency:

X -axis: Buffer Size
Y-axis: Latency and Speeds



Throughput:

X -axis: Buffer Size
Y-axis: Speed



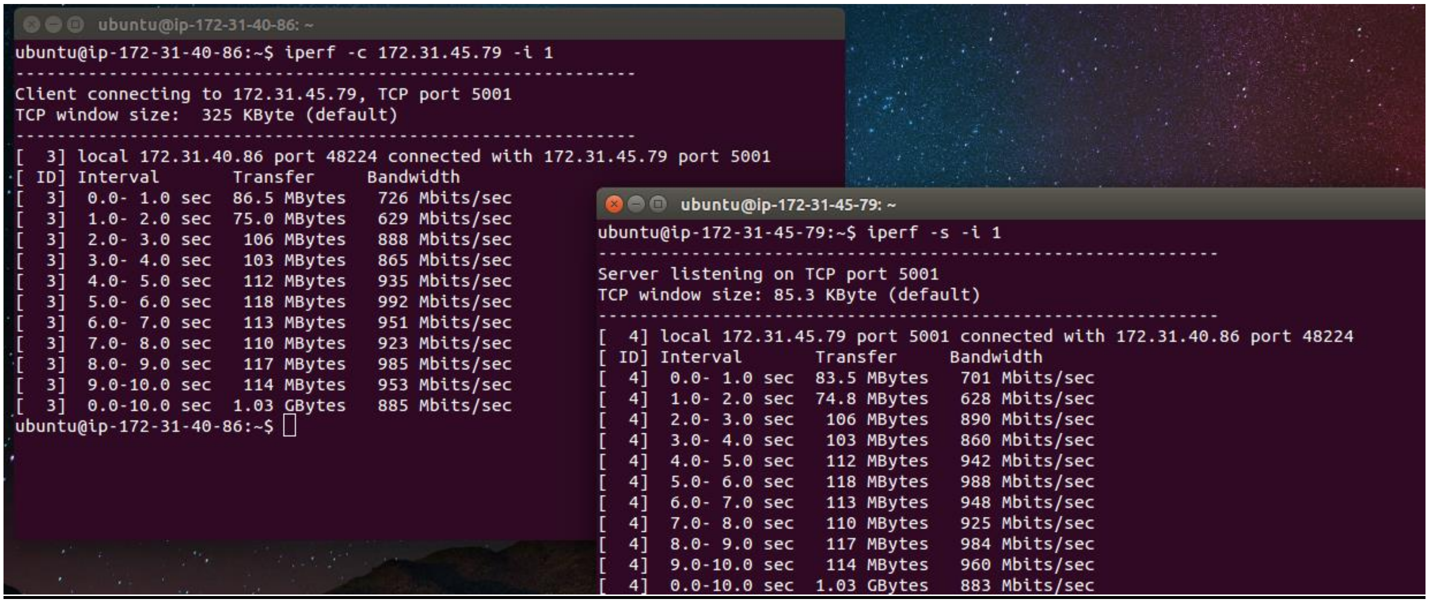
Conclusion:

1. UDP gives higher performance than Tcp.

2. Theoretical performance of network is as given on Amazon t2.instance: 164 Mbs/sec.

EXTRA CREDIT:

IPerf Benchmark:



```
ubuntu@ip-172-31-40-86:~$ iperf -c 172.31.45.79 -i 1
------------------------------------------------------------
Client connecting to 172.31.45.79, TCP port 5001
TCP window size:  325 KByte (default)
------------------------------------------------------------
[  3] local 172.31.40.86 port 48224 connected with 172.31.45.79 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  86.5 MBytes   726 Mbits/sec
[  3]  1.0- 2.0 sec  75.0 MBytes   629 Mbits/sec
[  3]  2.0- 3.0 sec   106 MBytes   888 Mbits/sec
[  3]  3.0- 4.0 sec   103 MBytes   865 Mbits/sec
[  3]  4.0- 5.0 sec   112 MBytes   935 Mbits/sec
[  3]  5.0- 6.0 sec   118 MBytes   992 Mbits/sec
[  3]  6.0- 7.0 sec   113 MBytes   951 Mbits/sec
[  3]  7.0- 8.0 sec   110 MBytes   923 Mbits/sec
[  3]  8.0- 9.0 sec   117 MBytes   985 Mbits/sec
[  3]  9.0-10.0 sec   114 MBytes   953 Mbits/sec
[  3]  0.0-10.0 sec  1.03 GBytes   885 Mbits/sec
ubuntu@ip-172-31-40-86:~$ 
```

```
ubuntu@ip-172-31-45-79:~$ iperf -s -i 1
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 172.31.45.79 port 5001 connected with 172.31.40.86 port 48224
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  83.5 MBytes   701 Mbits/sec
[  4]  1.0- 2.0 sec  74.8 MBytes   628 Mbits/sec
[  4]  2.0- 3.0 sec   106 MBytes   890 Mbits/sec
[  4]  3.0- 4.0 sec   103 MBytes   860 Mbits/sec
[  4]  4.0- 5.0 sec   112 MBytes   942 Mbits/sec
[  4]  5.0- 6.0 sec   118 MBytes   988 Mbits/sec
[  4]  6.0- 7.0 sec   113 MBytes   948 Mbits/sec
[  4]  7.0- 8.0 sec   110 MBytes   925 Mbits/sec
[  4]  8.0- 9.0 sec   117 MBytes   984 Mbits/sec
[  4]  9.0-10.0 sec   114 MBytes   960 Mbits/sec
[  4]  0.0-10.0 sec  1.03 GBytes   883 Mbits/sec
```

Best performance achieved on running IPerf benchmark is 984Mb/Sec

The efficiency achieved by IPerf when compared to theoretical performance is 51.48%