Performance Evaluation Benchmark

CS553 Fall 2014

1. Introduction

This report depicts the performance of various system components such as CPU, Memory, Disk, GPU and network. The experiments were performed three times and average and standard deviation were calculated. Each module describes the environment in which the experiment was performed, the theoretical values, formulas used and the respective graphs are attached in the datasheet to support the results.

2. CPU

This modules aims to measure the processor speed in terms of GFLOPs and GIOPs also allowing it to test the performance in different concurrency level (1, 2, 4 and 8 Threads).

2.1 System Environment

For CPU Benchmark we are performed our experiments on a VMware Player. Running environment is as follows:

OS: Ubuntu 14.04
RAM: 4 GB
HDD: 100 GB
Core: Intel Core i7-3630QM CPU @ 2.40 GHz x 4

2.2 Calculation

To calculate the speed in GFLOPs and GIOPs, the test performs floating point and integer operations **n** times. Here all operations are equally shared between multiple threads.

2.3 Theoretical value:

GFLOPs = GHz * IPC * No of Cores
=2.4 * 4 * 4 = 38.4

2.4 Experimental Calculation:

$$\text{GFLOPs} = \frac{(\#\text{Operations})}{(\text{totalTime} * 10^{\wedge 9})}$$

$$\text{GIOPs} = \frac{(\#\text{Operations})}{(\text{totalTime} * 10^{\wedge 9})}$$

**#Operations=No of Loop * no of Ops(Float/Int)

2.5 Results

With reference to the data sheet provided below the optimal number of threads to get the best performance is **8 Threads**. The average and standard deviation of GFLOPs and GIOPs are given in the datasheet and a comparative study is done with respect to Linpack Benchmark.

3. GPU

For GPU benchmark we have made use of CUDA C programming language to perform experiment on GPU processor and memory.
We have performed our experiments on a cluster "**jarvis**" with following environment parameters

3.1 System Environment

CUDA Capability <Major.Minor>: V2.0
15 Multiprocessors, 32 CUDA Cores/MP: 480 CUDA Cores
Total Global Memory: 1 GB
GPU Clock Rate: 1401 MHz (1.40 GHz)

3.2 Calculation

To calculate the speed in GFLOPs and GIOPs, the test performs floating point and integer operations **n** times.
To calculate the read and write memory bandwidth the "**cudamemcpy**" is used.

3.3 Theoretical value:

GFLOPs = GHz * IPC * No of Cores
= 1.40 * 4 * 480 = 2688

3.4 Experimental Calculation:

$$GFLOPs = \frac{(\#Operations)*(\#Multiprocessor)*(\#cores)*(\#Threads)}{Total\ Time * 10^{\wedge 9}}$$

$$GIOPs = \frac{(\#Operations)*(\#Multiprocessor)*(\#cores)*(\#Threads)}{Total\ Time * 10^{\wedge 9}}$$

$$Bandwidth = \frac{(\#Operations)}{Total\ time * 10^{\wedge 9}}$$

**#Operations=No of Loop * no of Ops (Float/Int/Memory Op)

3.5 Results

The GFLOPs and GIOPs were computed using the given formulas and the results are attached in the datasheet. The average and standard deviation of the results are also calculated.

4. Memory

The two parameters that are used to measure the performance of RAM are Throughput and Latency. A total of 12 experiments have been performed using various buffer sizes (1 byte, 1 Kb, 1 Mb), two operations (sequential read/write and random read/write) using 1 thread & 2 threads and the value of throughput and latency are determined by computing arithmetic mean.

4.1 System Environment

For Memory Benchmark, we have used a virtual machine with the configuration given below:

Operating System: Ubuntu 14.0.4
RAM: 4GB DDR3 SDRAM-PC12800
No of Cores Available: 4
HDD: 100 GB
Processor: Intel Core-i5 4210U 1600MHz

4.2 Theoretical Value

The theoretical peak was determined with reference to the Wikipedia page (DDR3_SDRAM). And the values are as follows:-
Throughput: 12800 Mb/s (Theoretical Peak)
Latency: 13.6 ns (Nanoseconds)

4.3 Experimental calculation

Throughput= (n/t_t)/ (1024*1024)) Mb/s

n = No of Operations
t_t = Difference Between end time and start time
Latency = The time taken to read and write one block of data (1b/1Kb/1Mb) - [t2-t1] ns
Where
t2 - t1 =Difference between end time and start time in nanoseconds

4.4 Results

With reference to the sheet below the best performance is given when we use **2 Threads.** The average and standard deviation of throughput and latency are given in the datasheet and a comparative study is done with respect to Stream Benchmark.

5. Disk

This module aims to calculate the read and write speed of Disk using various patterns such as Sequential read, Sequential write, Random Read, Random Write. All these operations are performed in varying concurrency of 1, 2 and 4 threads and also with varying block size of 1b, 1kb and 1Mb.

5.1 System Environment

Disk Benchmark experiments were performed on Lenovo G410. Running environment is as follows:

OS: Ubuntu 14.04
RAM: DDR3 6GB 1600
HDD: 500GB 5200
Processor: Intel i5 4200m

5.2 Calculations

1. For calculating throughput for Sequential read and Random read operation we are creating a FileChannel which reads inputstream using a buffer in form of byte[].
2. Timestamps are recorded before starting the reading operation and right after completing the reading operation.
3. To calculate the throughput we are calculating total time (t) taken to read total data (d) with multiple threads.

It is calculated as:

d= (totalReadOps * buffersize);

Throughput = (d*1000)/ (1024*1024*t) (MB/sec)

t = total time taken to read d bytes of data

6. Network

There are two different approaches through which a client can establish connection with the server they are

- TCP/IP
- UDP

TCP protocol uses stream and UDP uses datagrams for communication.

6.1 System Environment

Network Benchmark was performed on Lenovo G410. Running environment is as follows:

OS: Ubuntu 14.04
RAM: DDR3 6GB 1600
Processor: Intel i5 4200m

6.2 Calculations

Here we have to calculate Network bandwidth with TCP and UDP protocols.

For this experiment we send data to server and calculate the time to get the data back as response from server.

$$\text{Bandwidth (Mbits/sec)} = \frac{\text{Total Data sent (bytes)} * 8}{\text{Total Time Taken}}$$

Latency: We can get the network latency value by using following command in terminal.

$Ping 127.0.0.1
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=**0.048 ms**
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=**0.021 ms**

In our Experiment we calculate network latency by calculating the time taken to send 1 packet of data across nodes.

Average Latency Using TCP: 61.5 us
Average Latency Using TCP: 72.5 us

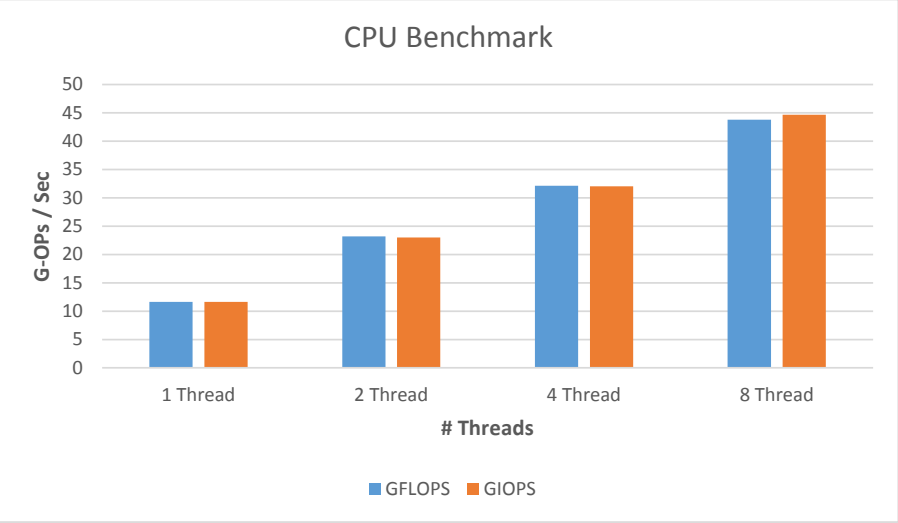
6.3 Results

The bandwidth and latency of the network were determined and the values are depicted on the datasheet. A comparative study was done using Iperf benchmark and the results that were determined after the experiment.

CPU Benchmark

GFLOPS & GIOPS TEST CASES								
	1 Thread		2 Thread		4 Thread		8 Thread	
	GFLOPS	GIOPS	GFLOPS	GIOPS	GFLOPS	GIOPS	GFLOPS	GIOPS
Test 1	11.644563	11.620039	23.193471	23.008026	32.124227	32.020999	43.797731	44.65497
Test 2	15.647424	15.14795	18.475003	18.633746	32.32649	31.693055	49.093017	44.085713
Test 3	15.577012	14.770259	31.240744	31.007882	31.566926	33.014216	46.637018	44.500381
Avg	**14.28966633**	**13.84608267**	**24.30307267**	**24.21655133**	**32.005881**	**32.24275667**	**46.50925533**	**44.413688**
Std Dev	**2.290997205**	**1.937037797**	**6.454800362**	**6.274966976**	**0.393368402**	**0.687930915**	**2.649953949**	**0.294363959**

Avg. CPU Benchmark w.r.t Threads				
	1 Thread	2 Thread	4 Thread	8 Thread
GFLOPS	11.644563	23.193471	32.124227	43.797731
GIOPS	11.620039	23.008026	32.020999	44.65497



Comparision(4 threads)	GFlops
Linpack Avg	39.4876
Benchmark	32.005881

linpack Result(v. 11.2.0.003)

Number of equations to solve (problem size): 10000
Leading dimension of array: 10000
Number of trials to run: 4
Data alignment value (in Kbytes): 4
Current date/time: Wed Sep 24 02:36:48 2014
CPU frequency: 2.389 GHz
Number of CPUs: 4
Number of cores: 4
Number of threads: 4
Parameters are set to:
Number of tests: 1
Number of equations to solve (problem size) : 10000
Leading dimension of array : 10000
Number of trials to run : 4
Data alignment value (in Kbytes) : 4
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
10000	10000	4	15.196	43.8847	9.603002e-11	3.386116e-02	pass
10000	10000	4	15.819	42.1559	9.603002e-11	3.386116e-02	pass
10000	10000	4	16.019	41.6309	9.603002e-11	3.386116e-02	pass
10000	10000	4	22.024	30.2789	9.603002e-11	3.386116e-02	pass

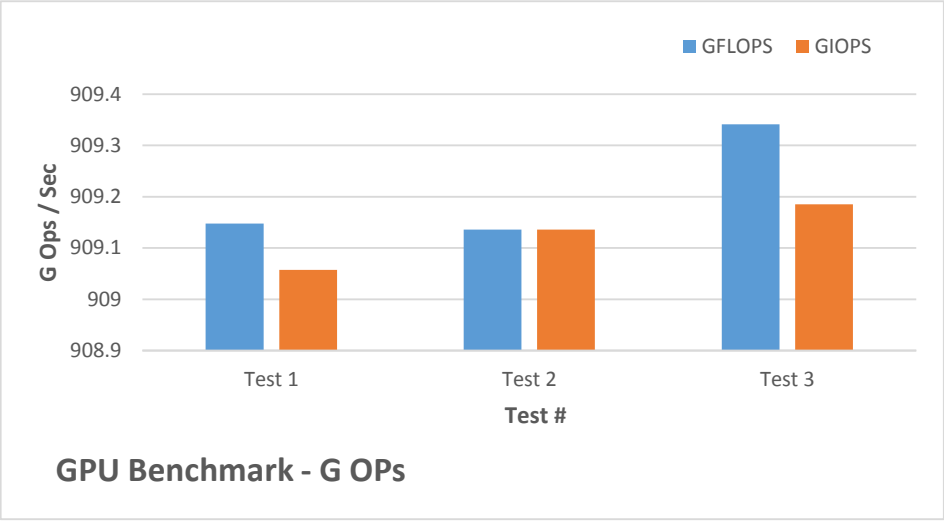
Performance Summary (GFlops)

Size	LDA	Align.	Average	Maximal
10000	**10000**	**4**	**39.4876**	**43.8847**

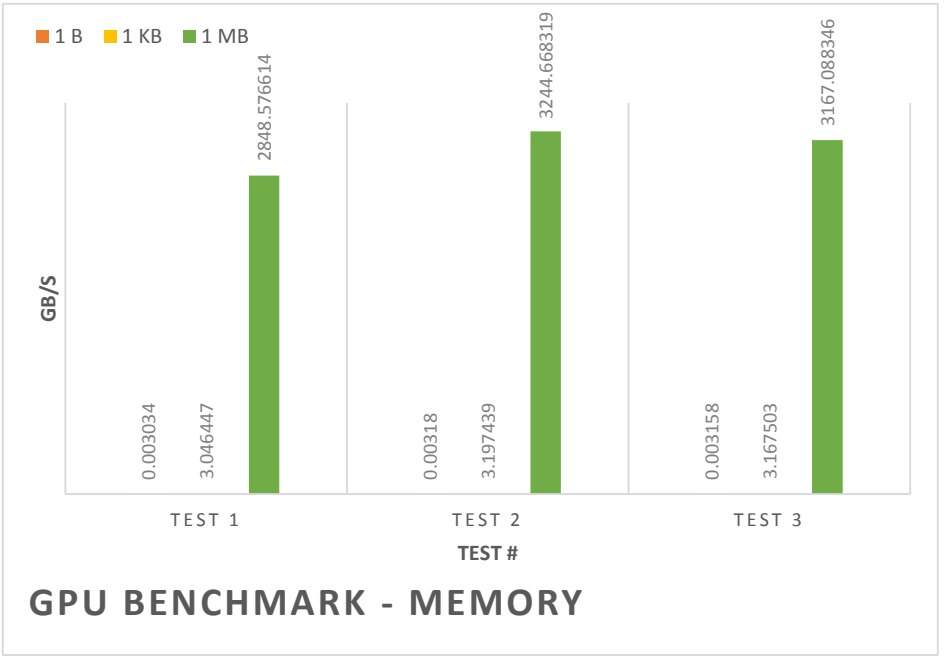
Residual checks PASSED

GPU Benchmark

GFLOPS & GIOPS Test Cases		
	GFLOPS	**GIOPS**
Test 1	909.14781	909.057099
Test 2	909.135815	909.135801
Test 3	909.341192	909.185434
Avg	**909.2082723**	**909.1261113**
Std Dev	**0.115267942**	**0.064713872**



Memory Performance			
	1 B	**1 KB**	**1 MB**
Test 1	0.003034	3.046447	2848.576614
Test 2	0.00318	3.197439	3244.668319
Test 3	0.003158	3.167503	3167.088346
Avg	**0.003124**	**3.137129667**	**3086.77776**
Std Dev	**7.8715E-05**	**0.079947173**	**209.903555**



MEMORY BENCHMARK

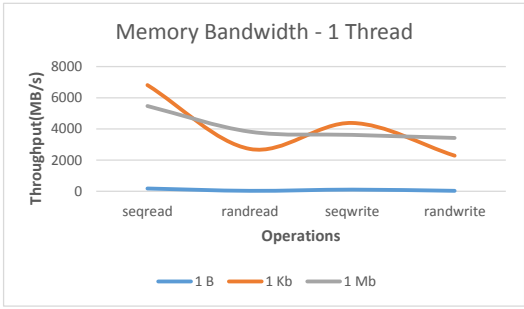
	1 B ~ Test Cases			
	Seq Read	Seq Write	Rand Read	Rand Write
Test 1	166.555	165.705	35.355	28.577
Test 2	210.137	87.868	30.822	25.068
Test 3	177.597	77.445	25.068	23.413
Avg	**184.763**	**110.3393333**	**30.415**	**25.686**
Std Dev	**22.65747885**	**48.23046253**	**5.155562918**	**2.636885853**

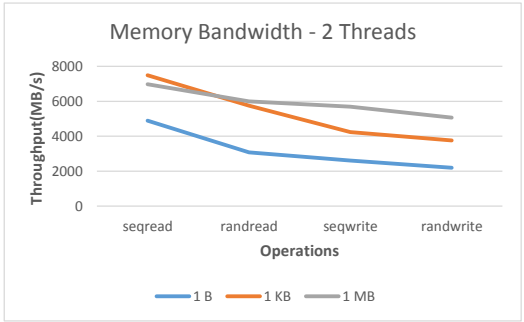
	1 KB ~ Test Cases			
	Seq Read	Seq Write	Rand Read	Rand Write
Test 1	6686.902	2602.571	2688.311	1988.198
Test 2	4212.84	3178.466	2585.565	1915.032
Test 3	5655.749	4595.993	2538.003	1986.98
Avg	**5518.497**	**3459.01**	**2603.959667**	**1963.403333**
Std Dev	**1242.728562**	**1025.895472**	**76.82380189**	**41.89523001**

	1 MB ~ Test Cases			
	Seq Read	Seq Write	Rand Read	Rand Write
Test 1	4187.314	1960.125	1978.442	1841.187
Test 2	4359.665	1945.262	1957.221	1741.252
Test 3	3650.445	2120.687	1981.113	1901.074
Avg	**4065.808**	**2008.691333**	**1972.258667**	**1827.837667**
Std Dev	**369.8932726**	**97.27537821**	**13.09129957**	**80.74293434**

1 Thread			
Mode	1 B	1 Kb	1 Mb
seqread	177.851	6813.568	5470.72
randread	30.23	2716.935	3823.39
seqwrite	105.703	4383.608	3619.039
randwrite	29.363	2284.563	3421.96

2 Threads			
	1 B	1 KB	1 MB
seqread	4891.17	7491.038	6976.568
randread	3073.024	5743.351	5993.933
seqwrite	2603.205	4230.956	5695.231
randwrite	2194.948	3758.589	5064.33





Comparision	Throughput ~ MB/s
Stream	**5471.2**
Benchmark	**5470.72**

Stream Result

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 62233 microseconds.
(= 62233 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:	5471.2	0.035321	0.029244	0.045979
Scale:	5274.4	0.034456	0.030335	0.042022
Add:	7411.2	0.035179	0.032383	0.039936
Triad:	7021.0	0.038899	0.034183	0.063289

Solution Validates: avg error less than 1.000000e-13 on all three arrays

DISK Benchmark

	1 B ~ Test Cases			
	Seq Read	Seq Write	Rand Read	Rand Write
Test 1	1.9234	0.6136	0.9986	0.2495
Test 2	0.2234	0.4549	0.7475	0.2434
Test 3	0.6234	0.9864	0.5495	0.3485
Avg	**0.9234**	**0.684966667**	**0.7652**	**0.280466667**
Std Dev	**0.888819442**	**0.272842378**	**0.225072588**	**0.058997486**

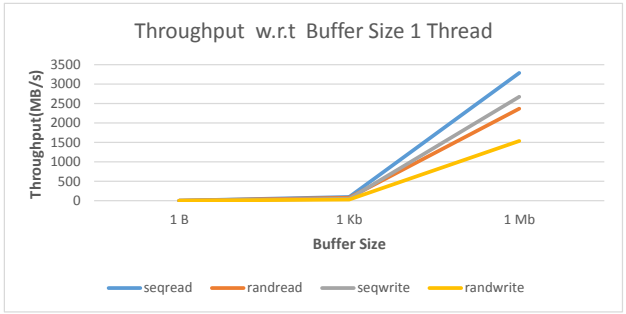
	1 KB ~ Test Cases			
	Seq Read	Seq Write	Rand Read	Rand Write
Test 1	95.34567	65.3456	75.3433	23.56789
Test 2	120.3454	49.3456	86.3456	34.09876
Test 3	75.34567	30.3456	60.3455	24.34567
Avg	**97.01224667**	**48.3456**	**74.01146667**	**27.33744**
Std Dev	**22.54610905**	**17.52141547**	**13.05111624**	**5.868374693**

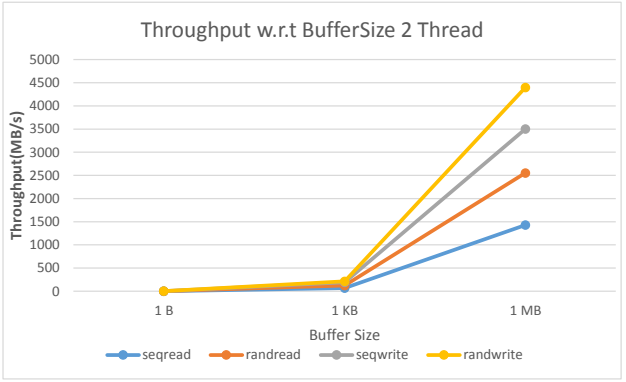
	1 MB ~ Test Cases			
	Seq Read	Seq Write	Rand Read	Rand Write
Test 1	3465.264	2987.264	2196.475	1475.187
Test 2	3200.346	2357.576	2240.867	1649.252
Test 3	3197.465	2678.576	2667.576	1485.074
Avg	**3287.691667**	**2674.472**	**2368.306**	**1536.504333**
Std Dev	**153.7888982**	**314.8640603**	**260.1241282**	**97.76740483**

Comparision	Throughput ~ MB/s
IOZONE	**3344**
Benchmark	**3287.691667**

1 Thread			
Mode	1 B	1 Kb	1 Mb
seqread	0.9234	97.0122	3287.6991
randread	0.7652	74.0114	2368.306
seqwrite	0.6849	48.3456	2674.472
randwrite	0.2804	27.33744	1536.504

2 Threads			
	1 B	1 KB	1 MB
seqread	0.7254	67.1254	1428.125
randread	0.2541	54.985	1125.415
seqwrite	0.6251	71.124	948.254
randwrite	0.1254	21.457	895.548





Network Benchmark

TCP Loopback Bandwidth(1 Thread)			
	1 B	1 KB	64kb
Test 1	1.58	356.24	8842.45
Test 2	0.57	394.87	7995.24
Test 3	1.98	346.46	8135.48
Avg	1.376667	365.8567	8324.39
Std Dev	0.726659	25.5977	454.0996

TCP Loopback Bandwidth(2 Thread)			
	1 B	1 KB	64kb
Test 1	1.67	158.58	7320.68
Test 2	0.95	135.49	7788.47
Test 3	0.97	180.79	7062.26
Avg	1.196667	158.2867	7390.47
Std Dev	0.410041	22.65142	368.1008

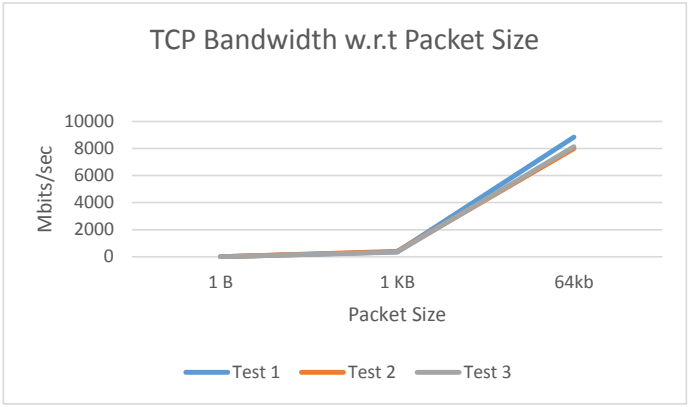
UDP Loopback Bandwidth(1 Thread)			
	1 B	1 KB	64 kB
Test 1	0.46	14.879	494.48
Test 2	0.47	28.25	528.46
Test 3	0.19	12.98	395.25
Avg	0.373333	18.703	472.73
Std Dev	0.15885	8.322287	69.21722

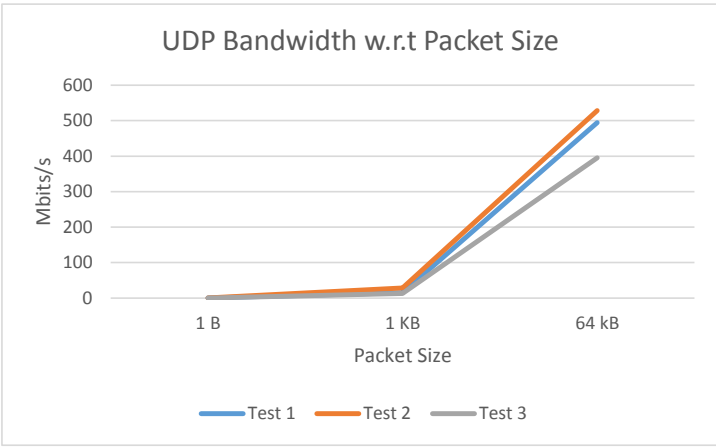
UDP Loopback Bandwidth(2 Thread)			
	1 B	1 KB	64KB
Test 1	1	7.56	156.46
Test 2	1	6.45	178.34
Test 3	1	3.47	97.34
Avg	1	5.826667	144.0467
Std Dev	0	2.115049	41.90248

Test	Latency(TCP)ms
1	76
2	44
3	28
4	39
5	55
6	65
7	47
Avg	61.5

Test	Latency(UDP)ms
1	76
2	59
3	57
4	63
5	42
6	39
7	69
Avg	72.5

Iperf Results (Mbits/sec)			
	1kb	64kb	1mb
Bandwidth	1518	12240	39485





Credits

1. Arihant Raj Nagarajan(**A20334121**)

    - CPU Benchmark
    - GPU Benchmark

2. Rahul Krishnamurthy(**A20330185**)

    - Memory Benchmark

3. Shashank Sharma(**A20330372**)

    - Disk Benchmark
    - Network Benchmark

References:

1.**http://docs.oracle.com**
2.**https://iperf.fr/**
3.**http://www.cs.virginia.edu/stream/**
4.**https://software.intel.com/en-us/articles/intel-math-kernel-library-linpack-download**
5.**http://www.iozone.org/**
6.**http://docs.nvidia.com/cuda/index.html#axzz3EHDMjsHc**