GPU Performance Analysis

Waleed Naveed
Yuvanesh Rajamani
Nico Maldonado

CS 3339
Spring 2022

Table of Contents

Abstract

GPUs which stand for Graphic Processing Unit, allows computers and many other technologies to render their graphics. With exponential growth of cores and threads within the GPU processors in this era. It is important to analyze the performance of these GPUs to ensure the upmost efficient performance is being achieved. All the optimizations that can be implemented are software based, as that it is easy to implement and transparent. We will run three benchmarks - CUDA, Matrix Benchmark, and a Unified Memory Test.

Introduction

There are limitations on the performance of a traditional CPU which is caused because of pipeline, scaling, and power constraints since they are single-core processors. This issue was solved using a multi-core architect, which places multiple cores on to a single chip. In addition to this GPUs are introduced as a method of splitting the workload that a CPU must deal with, since it also initially would render graphics. The introduction of GPU overall increased the performance of machines as the workload is being divided into factions. It also must be understood that GPUs are able to withhold higher memory bandwidth, floating-point throughput, and has more threads, allowing for more efficient pipelines and overall better performance. We today utilize GPU for playing video games, watching TV and even modern cellphone has a GPU which allow rendering of graphics on a surreal level. Our goal is to find which GPU performs better based on different qualifications and in comparison, with a machine that has a built-on GPU so it can be understood which is better for overall performance purposes. We compare a NVIDIA GeForce RTX 2070 Super and GeForce GTX 1060 3gb, along with the internal GPU, an Intel ® HD Graphics– Comet Lake-U GT2. We will run three benchmarks - CUDA (We use ZLUDA for Intel), Matrix Benchmark, and a Unified Memory Test in junction with these GPUs.

Hardware Specifications

These are the specifications for the GPUs on which we will be running our performance tests on:

Name/Brand:

Manufacturer: NVIDIA

Model: GeForce RTX 2070 Super

Architecture: Turing

Base Clock: 1605 MHz

Memory Specs:

Memory size: 8 GB	Memory type: GDDR6
Memory clock: 1750 MHz	Memory clock (effective): 14000 MHz
Memory interface width :256-bit	Memory bandwidth: 448 GB/s
L2 cache: 4 MB	

Cores:

CUDA: 7.5	CUDA cores: 2560
RT cores: 40	Tensor cores: 320
ROPs: 64	Texture units: 160

Name/Brand:

Manufacturer: NVIDIA

Model: GeForce GTX 1060 3gb

Architecture: Pascal

Base Clock: 1506 MHz

Memory Specs:

Memory size: 3 GB	Memory type: GDDR5
Memory clock: 2002 MHz	Memory clock (effective): 8008 MHz
Memory interface width: 192-bit	Memory bandwidth: 192.19 GB/s
L2 cache: 1.5 MB	

Cores:

CUDA: 6.1	CUDA cores: 1152
RT cores: NaN	Tensor cores: NaN
ROPs: 48	Texture units: 72

Name/Brand:

Manufacturer: NVIDIA

Model: Titan X

Architecture: Pascal

Base Clock: 1000 MHz

Memory Specs:

Memory size: 12 GB	Memory type: GDDR5
Memory clock: 1417 MHz	Memory clock (effective): 7 gbps
Memory interface width: 384-bit	Memory bandwidth: 336.6 GB/s
L2 cache: 3 MB	

Cores:

CUDA: 6.1	CUDA cores: 3584
RT cores: NaN	Tensor cores: NaN
ROPs: 96	Texture units: 224

Name/Brand:

Manufacturer: Intel

Model: Intel UHD Graphics (Comet Lake-U GT2) [V0] [ASUS]

Architecture: Generation 9.5 Architecture; x86

Base Clock: 300 MHz

Memory Specs:

Memory size: 1 GB

Memory type: LPDDR3/DDR4

Memory clock: 1330 MHz

Memory clock (effective): 665 MHz

Memory interface width: 64-bit

Memory bandwidth: 48.06 GB/s

L2 cache: 4 MB

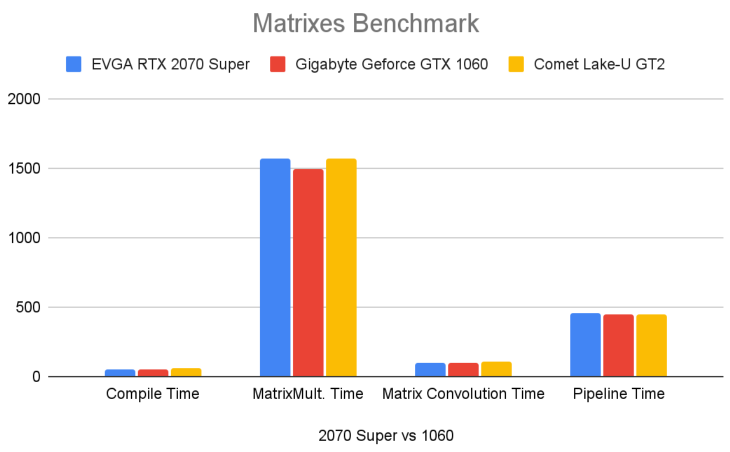
Cores: 4 / 8

Benchmark Results

The first benchmark that is being presented is a Matrixes Benchmark called "Benchmark". It will do three different types of benchmarks inside of this one program run. The first is called Matrix Multiplication, it takes two randomly generated matrices and multiplies them together (user sets the size of matrix). The next benchmark taken is the Matrix Convolution bench, this benchmark convolves a 3x3 kernel with a randomized matrix. Lastly, we have the pipelining benchmark which does the same as the matrix multiplication except is uses pipelining in the solving process.

We ran these tests three times each on the 2070 Super and the 1060 to get the most accurate data we could on each GPU. Here are our results (lower is better):

Matrices Benchmark				
2070 Super vs 1060 vs Lake-U	Compile Time	MatrixMult. Time	Matrix Convolution Time	Pipeline Time
EVGA RTX 2070 Super	46.8	1567.38	101.6	456.11
Gigabyte GeForce GTX 1060	52.29	1496.13	99.16	444.03
Comet Lake-U GT2	59.86	1567.475	110.65	450.56

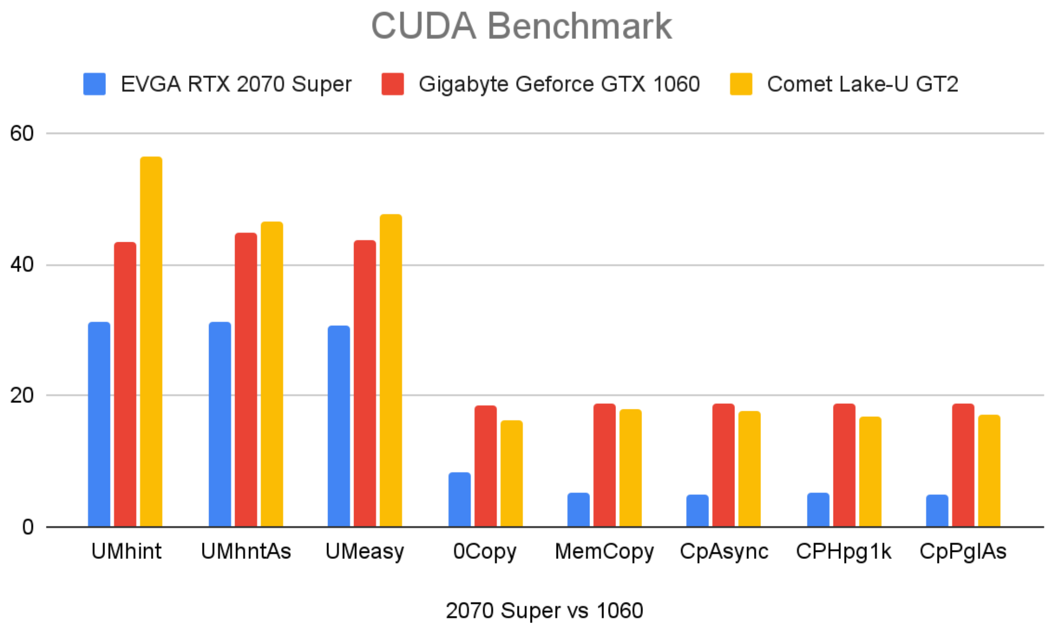


After looking at the data from all six tests, we found the results to be a little surprising with the 2070 Super only beating the 1060 out on the compile time for the matrices. The 1060 outperformed the 2070 in every other test done on the benchmark, this could be since the 1060 has a higher memory clock than the 2070 as seen in the specifications above. When reviewing the Comet you can see that it is significantly lacking in almost all of the tests ran, this is likely due to the fact that this GPU is threaded within its CPU, while parallel pipelining is still a resourceful technology , it is still lacking within the CPU and GPU — combined pipelining section.

The second benchmark done on the 1060 and 2070 is called "CUDA Benchmark", this is straight from NVIDIA, so it will only work with NVIDIA cards so, the other hardware specified above will have to use another benchmark like it. Now the CUDA Bench will do the same thing as the last benchmark (multiply a matrix), but this time using Unified Memory. We will see if this causes a bigger difference in speed between the two GPUs now. We will be running ZLUDA on the Intel GPU as CUDA is only for NVIDIA.

Here is the data found after, again three runs on each of the two GPU's (lower is better):

Cuda Benchmark	UMhint	UMhntAs	UMeasy	0Copy	MemCopy	CpAsync	CPHpg1k	CpPglAs
EVGA RTX 2070 Super	31.251	31.13	30.53	8.288	5.124	5.016	5.085	4.905
Gigabyte GeForce GTX 1060	43.265	44.761	43.681	18.437	18.931	18.89	18.894	18.735
Comet Lake	56.32	46.562	47.65	16.35	17.856	17.56	16.95	17.16



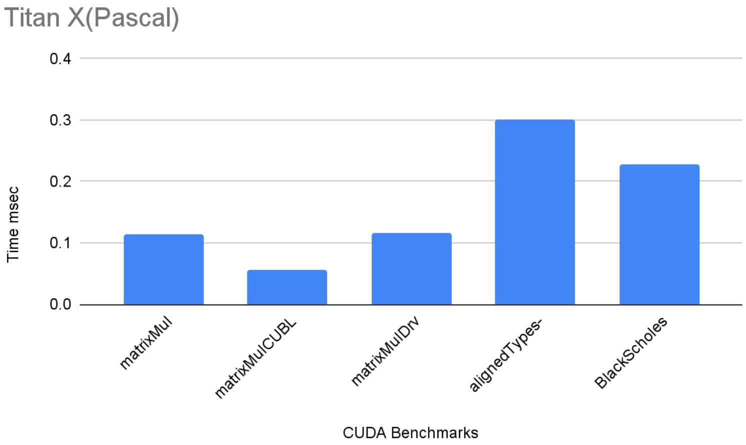
This time we can see that, as expected, the 2070 Super greatly outperforms the 1060 on every test in the benchmark. We can determine that this is caused by the 2070 Super having about 1,400 more Cuda cores than the 1060. The 2070 also has Tensor cores in which the 1060 has zero of. Tensor Cores enable mixed-precision computing, dynamically adapting calculations to accelerate throughput while preserving accuracy. The Comet once again lags in architecture to be able to keep up with the higher end - multicore processors.

ADA Texas State Server Benchmarks

The next system that we will perform our benchmarks on are the Texas State University Ada Server that has a Titan X(Pascal). For this server we will be using CUDA 10.0 samples to evaluate the performance. Since this is a university owned server, we couldn’t run the same benchmarks that we did for the other systems. Only CUDA 10.0 would run on this server due the fact that the GPU in the server has outdated drivers and that we are not authorized to make any changes to the server. The only test that is consistent across all system is the Unified Memory

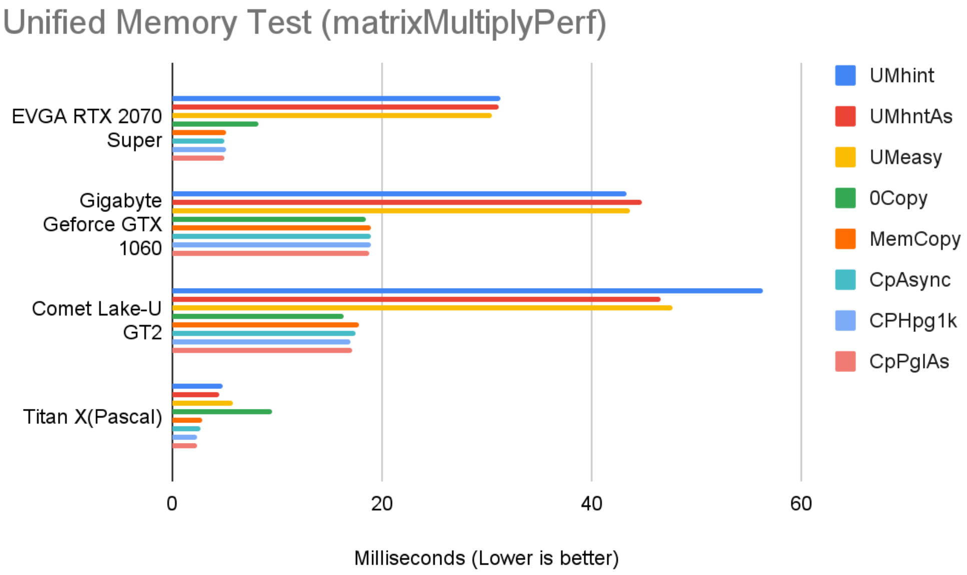
Test that uses the MatrixMultiplyPerf sample that we cover later. The samples that we did run on the server are matrixMulL, matrixMulCUBLAS, matrixMulDrv, alignedTypes-RGBA32_2, and BlackScholes. We ran each benchmark 3 times and then got the average. Below we have the execution times for each benchmark.

Titan X(Pascal)	matrixMulL	matrixMulCUBLAS	matrixMulDrv	alignedTypes-RGBA32_2	BlackScholes
Time msec	0.115	0.056666667	0.11666667	0.3007396667	0.228158333



The final benchmark being used is the Unified Memory Test, this is important as applications that iterate the same data over and over, need to be able to keep track and order to these values. Unified memory allows us to use zero-copy, so the GPU is allowed to access any page of the entire system and use that data for high bandwidth access. Essentially, it takes the Data from the CPU to th GPU and allows for that memory to be accessed all at once.

2070 x 1080 x Lake-U x Titan X	UMhint	UMhntAs	UMeasy	0Copy	MemCopy	CpAsync	CPHpg1k	CpPglAs
EVGA RTX 2070 Super	31.251	31.13	30.53	8.288	5.124	5.016	5.085	4.905
Gigabyte GeForce GTX 1060	43.265	44.761	43.681	18.437	18.931	18.89	18.894	18.735
Comet Lake-U GT2	56.32	46.562	47.65	16.35	17.856	17.56	16.95	17.16
Titan X(Pascal)	4.825	4.439	5.7344	9.6067	2.8513	2.7427	2.3867	2.4243



We can see in the data that the Titan X performs better than the rest of the GPUs, likely due to the number of CUDA cores on it, and its high amount of memory on the card. Although the Titan X came out almost 7 years ago, we can see the specs still make it a contender in the market for more computational applications.

Conclusion

This work allowed to us understand how different aspects of GPUs allow them to complete tasks quicker than others. We used a Matrix Benchmark, CUDA and ZLUDA Benchmarks, along with an extension of CUDA which would be Unified Memory Benchmark. We used these on an NVIDIA RTX 2070, GTX 1060, Titan X and Intel Comet Lake-U. We didn't necessarily apply any performance tuning to these GPUs, so we understand them from a stack value point of view. Out of all the GPUs the Titan X performed better than the rest, with the Comet Lake coming last. The Titan X has the most amount of memory almost a 50% increase among the rest at 12 GB. It also has the highest core count out the rest of the GPUs.While, the Intel Comet Lake performs the slowest as it has the least number of cores and memory out of the rest of the GPUs the only throttle that the Titan X could've faced would've been its outdated architecture. This was proven incorrect because the Titan X was the most efficient at all the benchmarks, likely due to its memory size and number of cores, which are at an almost 50% increase compared to the other GPUs. Ideally, increasing memory and cores allow the GPUs to perform tasks at the same time with efficient parallelism. We plan to extend our testing beyond the CUDA and Matrix Benchmarks to other benchmarks, which will likely utilize different parts of the GPU compared to these benchmarks.

Works Cited

Info on Unified Memory:
https://developer.nvidia.com/blog/unified-memory-cuda-beginners/#:~:text=is%20Unified%20Memory%3F-,Unified%20Memory%20is%20a%20single%20memory%20address%20space%20accessible%20from,on%20either%20CPUs%20or%20GPUs.

Info on Tensor Cores:
https://www.nvidia.com/en-us/data-center/tensor-cores/

ZLUDA Benchmark :
https://github.com/vosen/ZLUDA

CUDA Benchmark:
https://github.com/NVIDIA/cuda-samples

“Benchmark” :
https://github.com/gpujs/benchmark

Comparison of Parallel Performance :
https://github.com/GMAP/NPB-CPP

2070 Super Specs. :
https://www.gpuzoo.com/GPU-NVIDIA/GeForce_RTX_2070_Super.html

1060 Specs. :
https://www.gpuzoo.com/GPU-NVIDIA/GeForce_GTX_1060_3GB.html