Parallel algorithms accelerated by GPUs for the processing of real time collision data of the ATLAS experiment at CERN

Diogo Weigel
Bruno Rodrigues

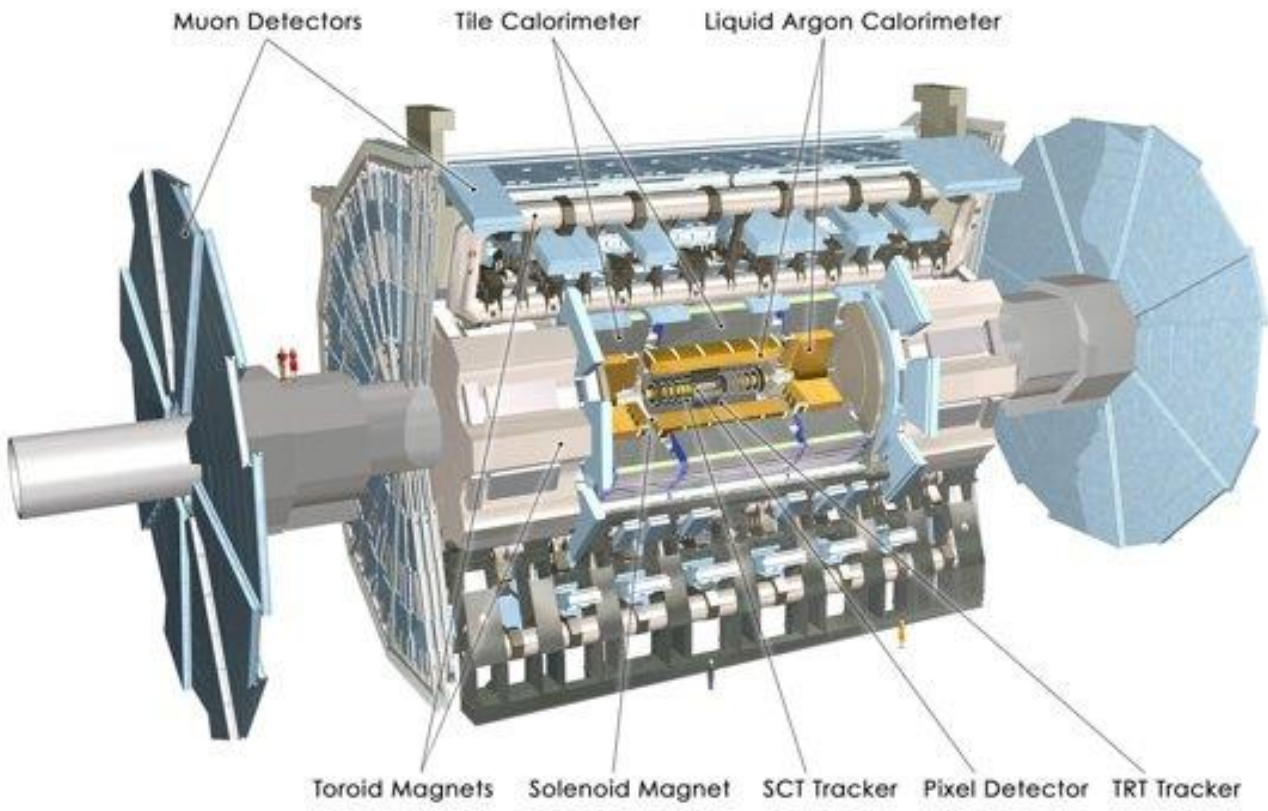
Objectives

- Study how the Trigger works and how it can be improved.
- Compare Trigger running times between different GPUs and different configurations on the same GPU.
- Understand the difference between using standard computing and parallel computing.
- Learn the basics of CUDA and its implementations.

Cuda

- Cuda is a parallel programming architecture which uses GPUs as the main source of computing power.
- It can be integrated in programming languages such as C, C++, Fortran etc.

ATLAS and its Triggers

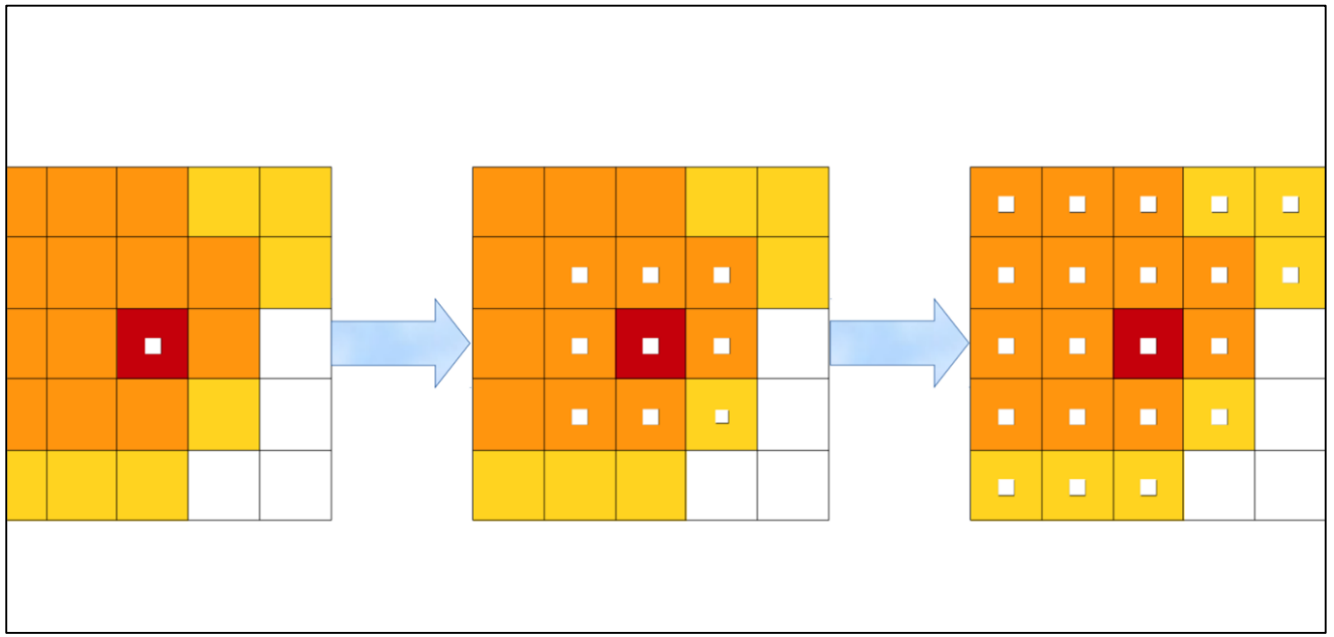


- ATLAS collects 64 TB/s of data by producing over 40 Million events per second in proton/proton collisions.
- TDAQ (Trigger and Data Acquisition) system is divided into three levels: first level Trigger, High Level Trigger (HLT) and Dataflow system.
- The first Level Trigger is divided into three parts: Trigger Calorimeter, Trigger Muon and the Central Trigger Processor.

Trigger Calorimeter

- Sends e-/γ, hadron/τ, candidate jets and overall energy information's to the CPU Trigger, which combines the input signals of up to 256 trigger types.
- Uses Topological Clustering and Cluster Splitting.
- Clusters are the reconstructed energy deposits of particles in the calorimeter.

Topological Clustering



- Reconstructs the energy deposited by electrons, hadrons and photons into clusters – proto-clusters.
- The characteristics of the proto-clusters are defined by three parameters: S (primary seed threshold), N (threshold for growth control) and P (main cell filter).

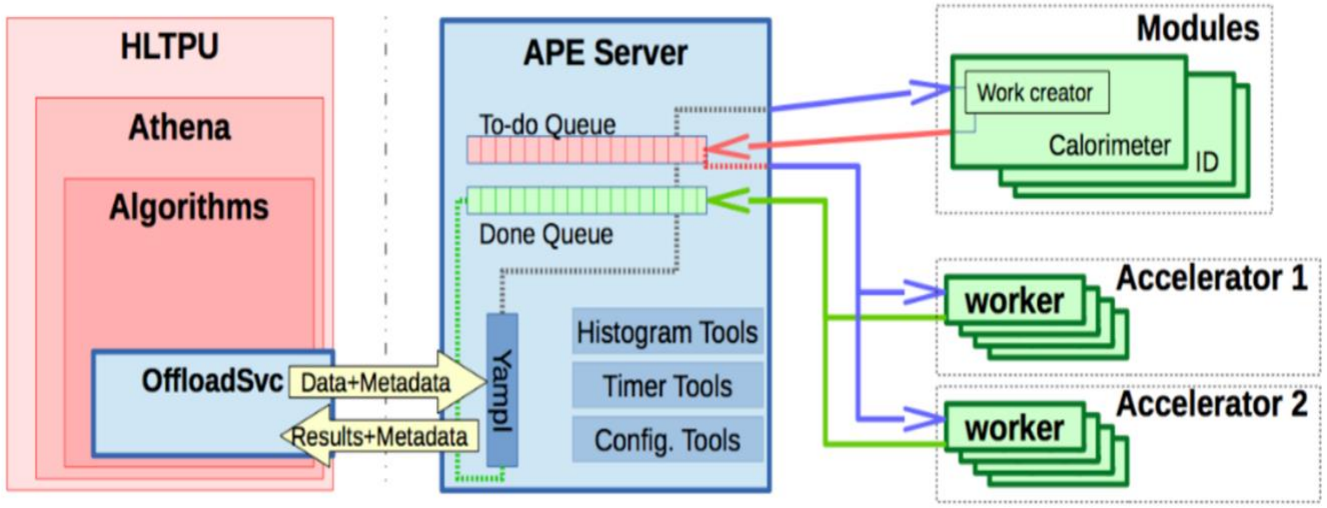
TAC (Topo-Automaton Clustering)

- Parallel version of the TopoCluster which uses a cellular automaton.
- Algorithm implemented to accelerate the cluster growing step in Trigger Calorimeter.

Cluster Splitter

- Proto-Clusters can be too large to give a meaningful measure of the energy flow from the particles.
- Clusters with two or more local maxima are divided between the signal peeks.

Server/Client Architecture



- The Accelerator Process Extension (APE) makes the communication of the Algorithm with the GPGPU.
- Independent of Athena.
- The offloading service was reading directly from the files, independently from Athena.
- Serves multiple clients.

Hardware

- AMD FX(tm)-8320 8-core processor
- 8 GB RAM
- NVIDIA GeForce GTX- 650
- 2 GB RAM

- Intel Xeon E3-12xx v2 24-core processor
- 20 GB DDR3 RAM
- NVIDIA Tesla K20
- 5 GB RAM

Events used

- 755 ttbar events
- 137 pileup
- 48000 loops ran each time

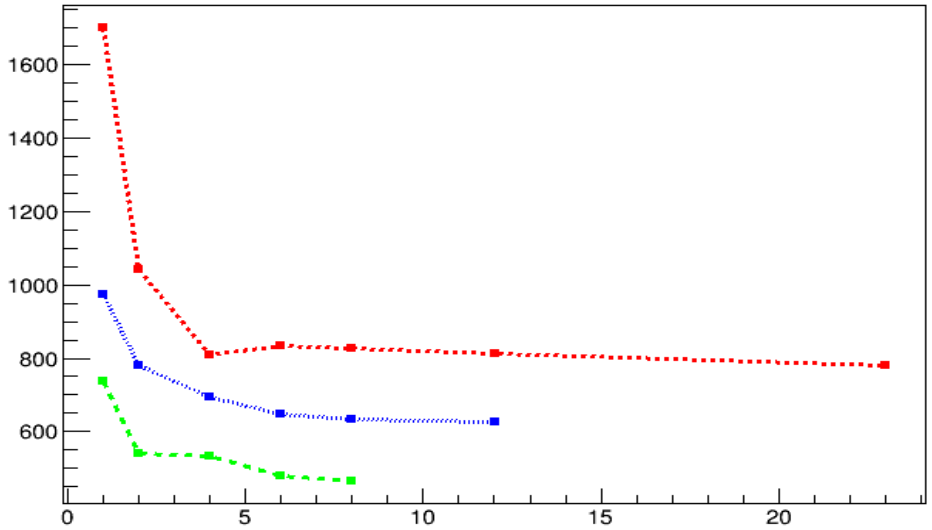
CPU running times for Tesla K20

Clients	Growing (s)	Combined (s)
23	--	782.037
12	628.034	811.325
8	634.760	828.523
6	648.123	833.037
4	693.240	809.647
2	779.917	1042.820
1	973.515	1700.830

CPU running times for GTX-650

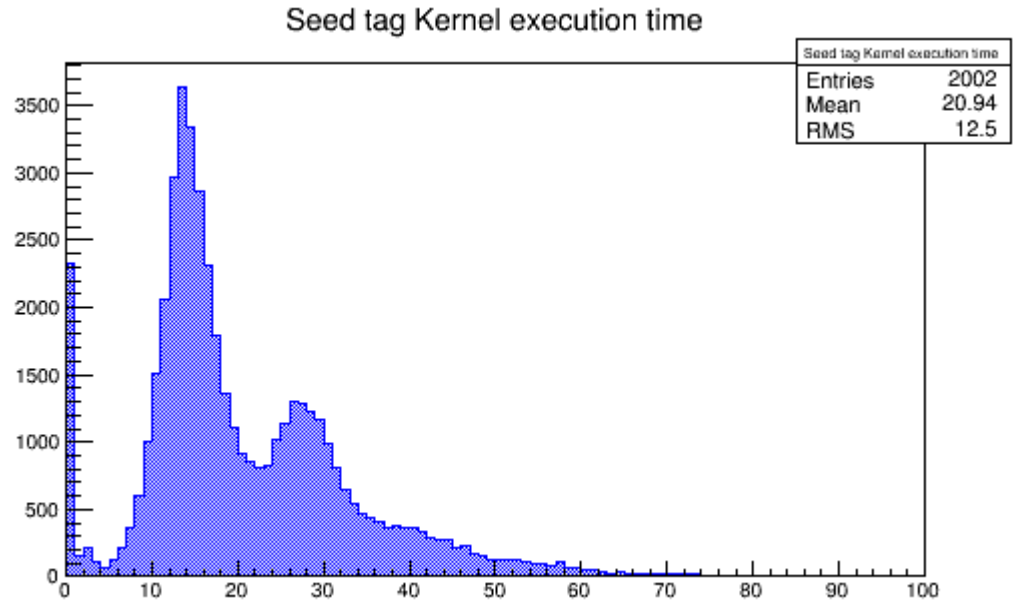
Clients	Growing(s)
12	--
8	465.473
6	478.000
4	532.152
2	541.508
1	739.113

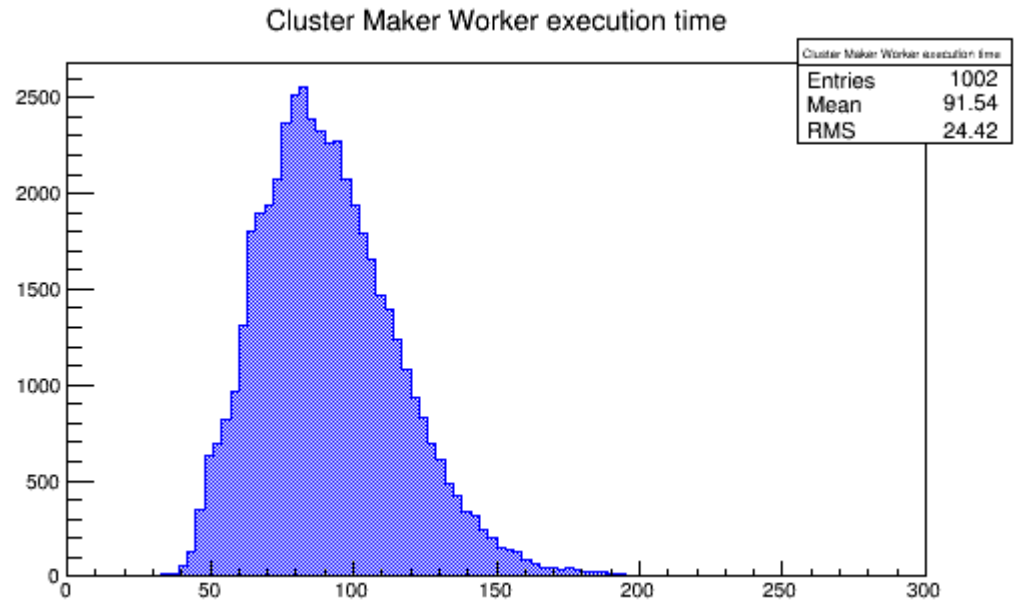
Running times / Number of clients

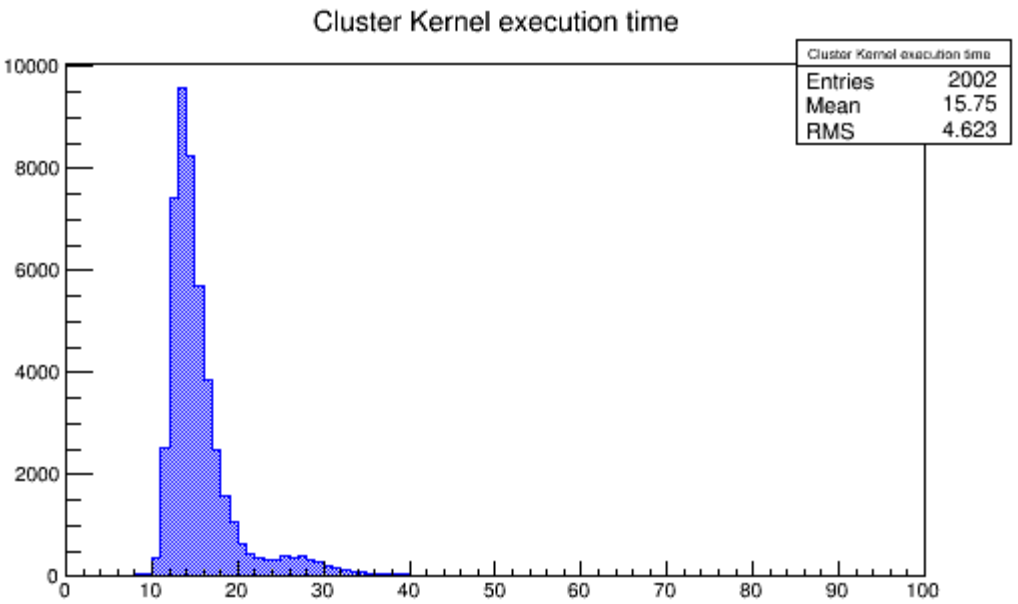


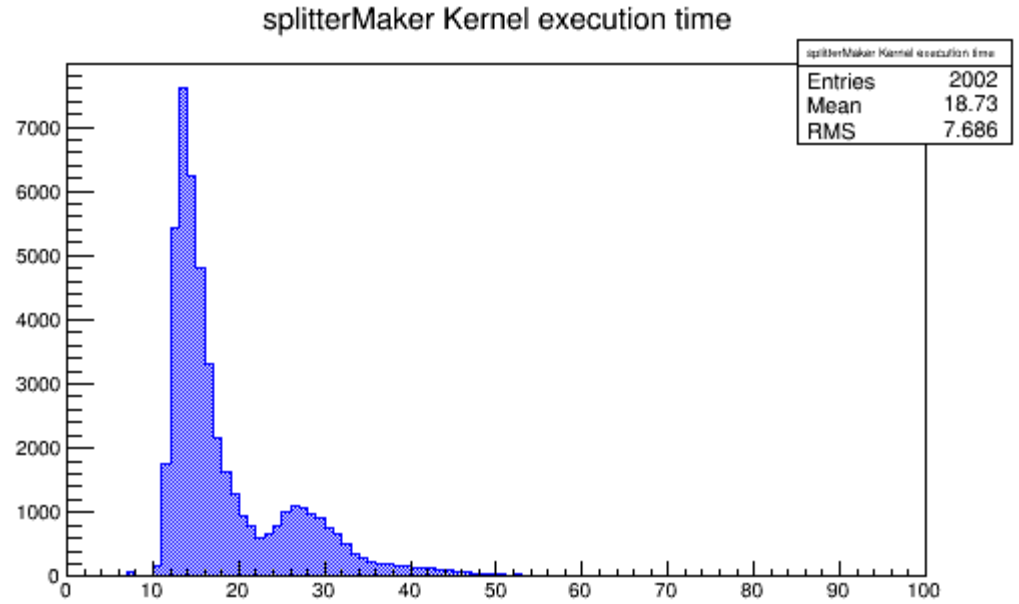
- Tesla K20 Combined
- Tesla K20 Growing
- GTX- 650 Growing

Tesla K20 | 6 runners and 6 clients | 8000 loops in 833.037 s| Combined



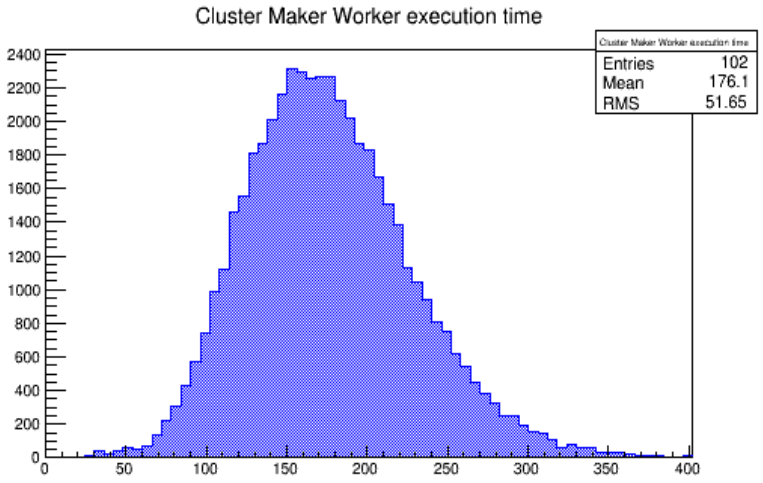




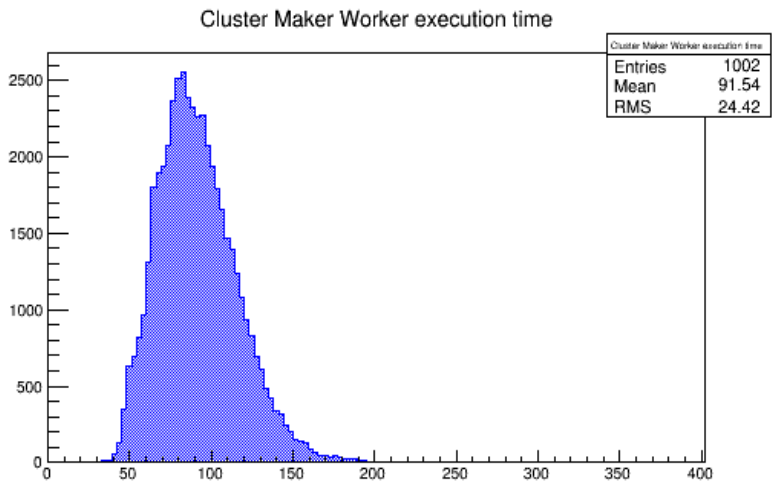


Tesla K20 | Combined |Cluster Maker Worker execution time

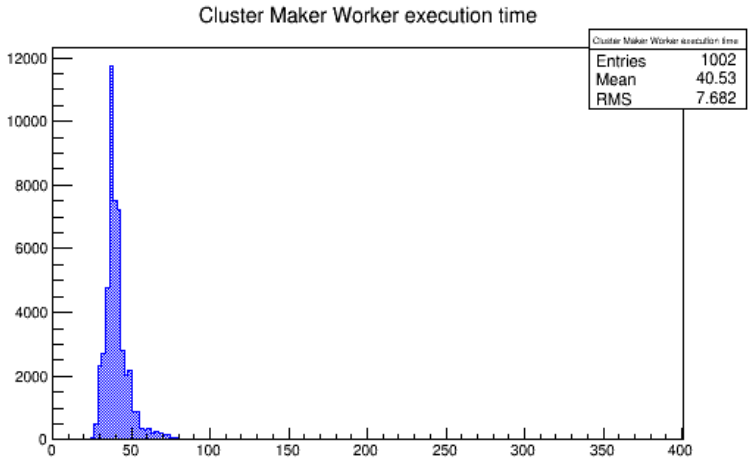
12 clients/runners |4k loops



6 clients/runners |8k loops



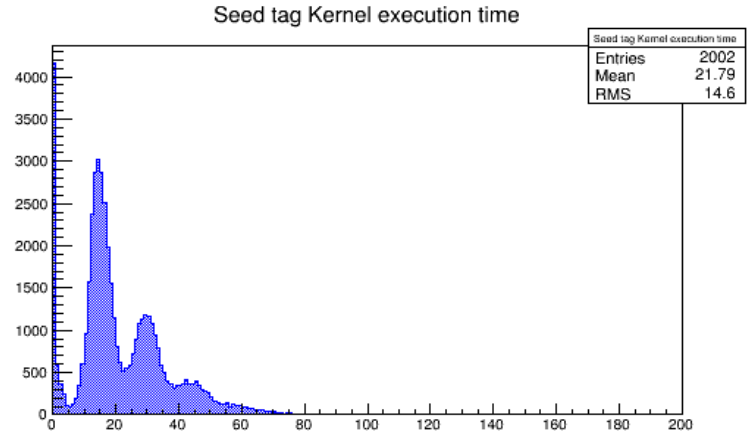
2 clients/runners |24k loops



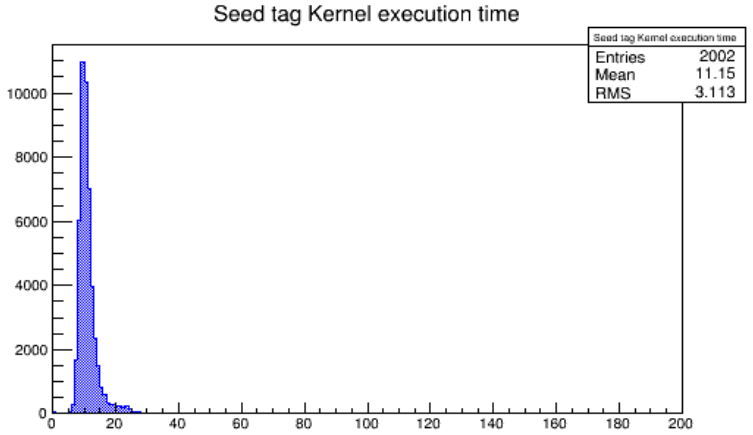
- The more clients are used, the longer each individual process takes to run.
- Since more processes run in parallel, even though each individually takes longer, overall it finishes faster.

Tesla K20 vs GTX 650
Growing

Tesla K20

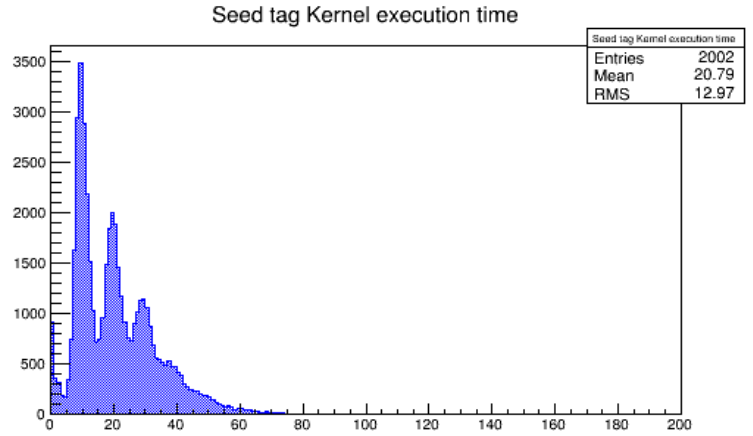


6 clients

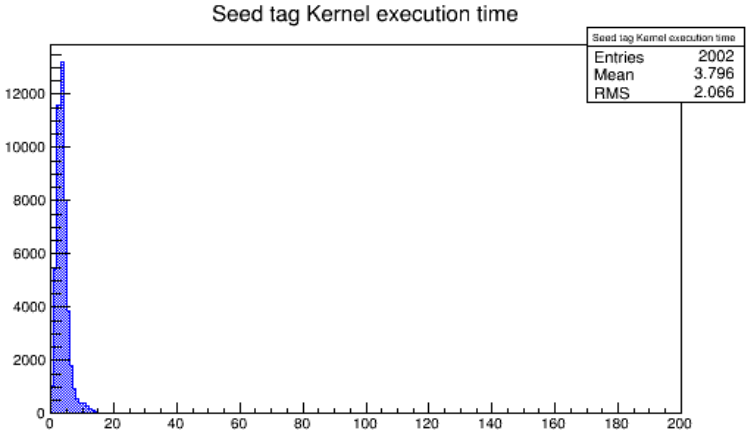


2 clients

GTX-650



6 clients



2 clients

Conclusions

- The decrease of time taken will stagnate with higher numbers of clients on the same GPU.
- The time it takes for each iteration to process rises with the number of clients but the overall time it takes decreases.