Development of `traccc` as-a-Service (aaS)

Miles Cochran-Branson[1], Yuan-Tang Chou[1], Xiangyang Ju[2], Haoran Zhao[1]

[1]University of Washington Department of Physics, [2]Lawrence Berkeley National Lab

Objective

Implement a custom backend to run the rule-based tracking accelerator `traccc` as-a-service. The main goals of this approach are to:

- Increase throughput
- Optimize coprocessor resource utilization
- Decrease per-event cost

Upcoming HL-LHC Challenges

Increased collisions at the interaction point in the HL-LHC will lead to massive challenges in resolving dense detector environments such as below:

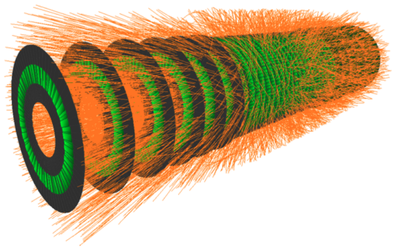


Figure: $\langle\mu\rangle = 200$ pileup event in Track-ML detector

Even if we do not reach $\langle\mu\rangle = 200$, computation times scales superlinearly with pileup:

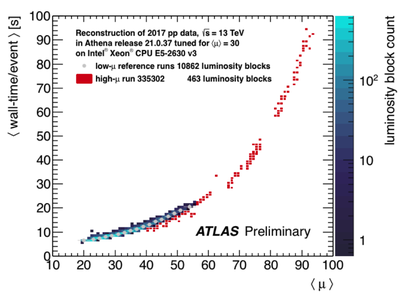


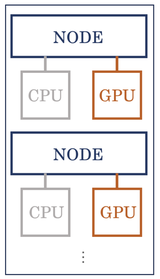
Figure: CPU compute time scales exponentially with $\langle\mu\rangle$.

This provides an excellent environment for the use of coprocessors to parallelize track reconstruction.

Heterogeneous Computing

The standard computing paradigm for coprocessors is *heterogeneous computing* where CPUs and co-processors such as GPUs are connected on the same node.

- Advantages of heterogeneous regime
  1. Many working examples
  2. Most coprocessor code written with this architecture in mind
- Disadvantages of heterogeneous regime
  1. Can be harder to implement new tools in existing frameworks
  2. Can be inefficient in use of resources



Traccc

- A possible solution to tracking problem using coprocessors within ACTS
- Set of standalone tools developed to demonstrate effective track reconstruction with accelerators such as GPUs
- Uses a combinatorial Kalman filter for track finding/fitting

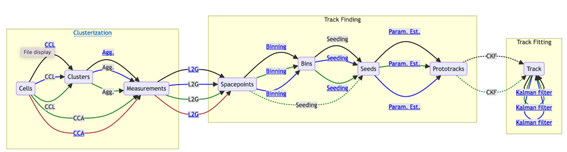


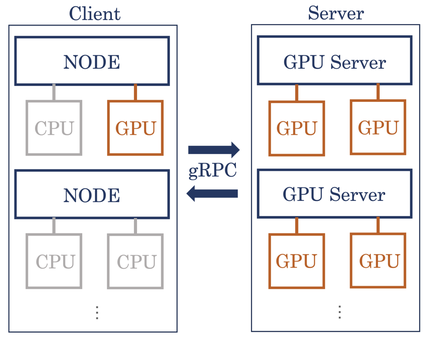
Figure: Traccc-architecture

Current architecture of `traccc` includes the full-chain track reconstruction chain implemented on GPUs

As-a-service Paradigm

The *as-a-service* (aaS) paradigm offloads expensive coprocessor operations to a dedicated GPU server

- Advantages of aaS regime
  1. Can be easier to integrate with production framework (e.g. Athena)
  2. Potentially improve scalability and resource utilization
- Disadvantages of aaS regime
  1. Often no working implementations
  2. Can introduce server latency



Traccc as-a-service

The main components of the as-a-service approach:

- A standalone version of the `traccc` tracking algorithm
- A custom backend using `NVIDIA` Triton inference server
- A client to send data and receive fitted tracks

Performance metrics and ways to increase throughput:

- Send multiple client requests to the server to reduce latency
- Increase the number of `Triton` model instances loaded onto the server to sustain multiple current requests

Testing was done on the perlmutter HPC NERSC at LBNL with data sent through localhost. Future iterations could include other HPC servers, cloud-based servers, or a dedicated GPU cluster at P1.

Results

Standalone tests of the server performance with data sent but without fitted tracks received are shown below.

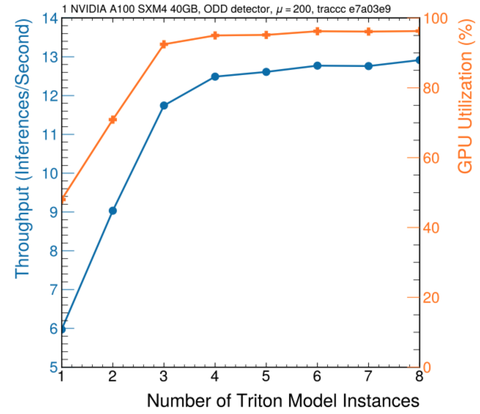


Figure: Throughput and GPU utilization improvement with an increase in Triton model instances per GPU

Conclusion

We observe an increase in GPU utilization from 45% to near 100% corresponding to an increase in throughput of ∼1.5 times.

Acknowledgements

This research was only possible with funding from the WATCHEP fellowship and computing resources provided by Lawrence Berkeley National Laboratory (LBNL). We also wish to thank the `traccc` team for their help and guidance.

References

Traccc ACTS HL-LHC Tracking

Email: miles.cb@cern.ch, **Contribution ID 60**