Madgraph on GPUs

W. Hopkins

Introduction: future computing resources

- Future computing resources will include High Performance Computers (HPCs).
- Future HPCs will include "compute accelerators" (GPU-like processors).
- Many HPCs foreseen to require/incentivize use of GPUs.
- HPCs =/= HTCs: may not be able to handle high input/output throughput.
- Generators may be well-suited for HPCs: compute intensive with little input.

Introduction: Madgraph GPU

- (NVIDIA) GPU version of Madgraph developed, last updated in 2013.
  - Code developed by group at KEK. Arxiv: arXiv: 1010.2107
  - Used CUDA 4 (NVIDIA)
- 3 parts of MC generation separately CUDAfied:
  - Helicity amplitude (matrix element) evaluation (HELAS).
  - Phase-space Integration: (g)VEGAS/(g)BASES.
  - Event generation is missing: (g)SPRING
- No single Madgraph implementation that has all parts working (as far as we know).
- Each part is modularized: can implement any single one in Madgraph.

Summer 2019 Madgraph GPU revival

- Summer student at ANL (K. Fielman) worked on reviving the 2013 version to work with modern CUDA (CUDA >6).
  - No part (HELAS,VEGAS,BASES/SPRING) worked out of the box.
- Several functions had to be changed (new interface in modern CUDA).
- Succesfully revived gVEGAS (GPU version of the VEGAS integration algorithm).
  - gVEGAS worked on test function in stand-alone mode (not integrated int Madgraph).
- More info in HSF Generator WG meeting talk
- Modernized code lives here: https://xgitlab.cels.anl.gov/whopkins/MadgraphGPU

Next steps for gVEGAS

- Integrate useful function: actual matrix element.
- Make Madgraph work with stand-alone gVEGAS (you could say "integrate gVEGAS" but that gets confusing).
- Study gVEGAS enabled Madgraph within the ATLAS software environment.

Main question

How much effort and what kind of work is needed to have a useful GPU-enabled algorithm work within ATLAS?

Not aiming at building production version! CUDA is too narrow for that and requires parallel code base!

ATLAS qualification for GPU-enabled Madgraph

- Documenting the GPU implementations of VEGAS (gVEGAS)
- Profile Madgraph to understand how much time is spent on integration.
- Investigate using gVEGAS in a Madgraph version used in ATLAS.
- Integrate a gVEGAS enabled Madgraph into the ATLAS workflow and document the process of integrating GPU-enabled code into ATLAS.

QT aim: study and document effort needed to have GPU-enabled algorithm in ATLAS.

Person power accounting

- Current effort:
  - 50% of ATLAS postdoc working on qualification task.
  - WH: supervision of postdoc.
- Olivier Mattelaer is interested in learning CUDA and how to offload Madgraph parts to a GPU.
  - Olivier is one of the main developers of Madgraph.