Parallel Kalman Filter

25th May, Afternoon

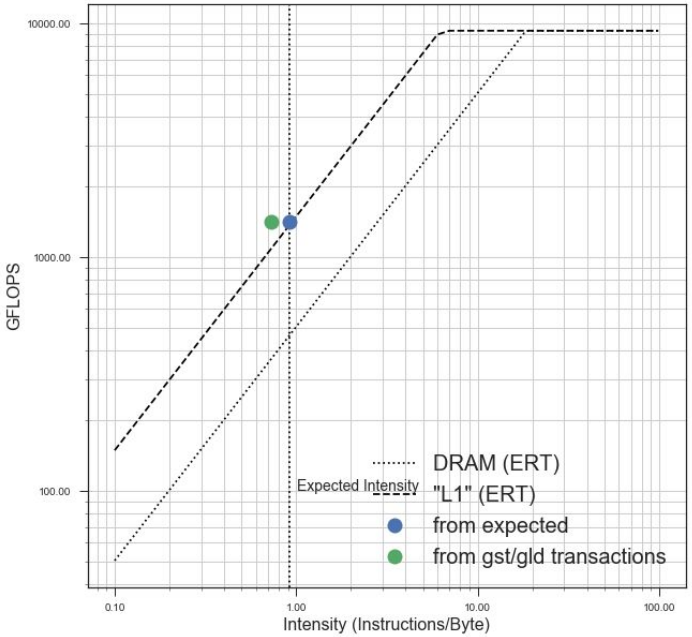
André, Dan, Matthieu, Slava

Objectives

- Understand performance bounds for KF-based track fitting and building and GPU
  - First, understand this bounds on a simpler, somewhat related problem, e.g. multiplying numerous 6x6 matrices
  - Explore how to get roofline performance from this simple problem
  - … and try to finally get a hold at what the profiler's numbers mean
- Relate the exploration of Matrix multiplication problem to track fitting
  - Estimates of tracking compute needs on GPUs
    - Both in term of the algorithm's intensity
    - And in term of input parameters based on existing x86 runs

6x6 Matrix Multiplication Proxy

- Obtained roofline-level performance
- Even a bit more with vector instructions
  - ERT seems to provide a conservative estimate of bandwidths (STREAM numbers are higher)
- Tried to max out floating point performance with a compute intensive example
  - Added a test with 100% FLOP efficiency. This test is highly artificial. It took an effort to tweak the code to make the compiler not ignore it.
- At least with CUDA 9.2, numbers from the profiler start to make sense
  - Gld_transactions, gst_transaction, flop_count_sp, ..



Progress

- Added a more realistic benchmark (simple Kalman track fit) to github repository
- From first principle operational intensity is ~2
  - If all data are read only once and kept in registers forever
- Profiler reports an operation intensity of 0.5
  - We will continue investigating to see if we can push this number closer to 2
  - Concomitantly, we also need to continue pushing performance vertically toward the roofline, even at a lower intensity

Observations from top-down approach

- Use a run of track building with CMS simulation PU70 on a KNL
- "Why KNL?"
  - This is pretty close in floating point power per unit compared to Pascal P100 GPU
    - 4.5 TFLOPS on KNL vs 9 TFLOPS on P100 GPU
- Implied FP32 operations derived from PU70 track building
  - one thread 1.3 implied GFLOPS. This corresponds to about 2 G FP32 operations per event.
  - full machine 90 implied GFLOPS
  - Compared to the declared specs FLOPS, this is about 2%
  - We actually see about 2% in our GPU version of the fitting code. Gives some hope for the analogy to work.
- Simple projections on how fast a GPU version can be is a factor of 2
  - Our KNL tests give 100 Hz processing power
  - Naive estimate for (yet to be completed) GPU version could be 200 Hz

Conclusions and plans