Scientific Software: Reconstruction

Giuseppe Cerati

Computational Science Working Group Pre-meeting 1

April 12, 2017

Introduction & Old/current model

Goal of this talk is to help the discussion, not meant to be a full review!

This is what could be today's typical model (of course there are exceptions!):

- Reconstruction code written by average physicist, integration in framework supervised by experts
- Key reconstruction components/algorithms optimized by computing experts
- Code developed and run on CPUs (single thread, no vectorization)
- Workflows run at grid centers, one job per core (sharing memory for cores on the same multicore processor)
  - CMS can parallelize at event and module level; main advantage: better use of memory
- Machine learning and multivariate tools used for tagging and selection of candidates/events
- Trigger: low level trigger on hardware/FPGAs, high level on CPU is a simplified version of offline reco

Major issues and challenges for the future

All are opportunities for better physics, we 'just' need to learn how to take advantage of them!

- Higher accelerator intensities: more data (both online and offline)
- New detectors: larger, more granular, new technologies
- End of frequency scaling with Moore's law, emerging of highly parallel architectures
  - parallelism both at SIMD/vectorization and thread level
- Heterogeneous systems
  - CPU/MIC/GPU/…, computing centers/supercomputers, …
- Big progress of deep learning techniques

Strategies we could pursue (I)

Many of the points below are already being studied, the question is how a specific solution can become standard practice?

- Explore new architectures and parallelization (SIMD and thread) at sub-module level (algorithms)
  - vectorization is more difficult and for some systems more important than thread-level parallelization
  - of course efforts for event level and module-level parallelization should continue
- On demand reco on accelerators/co-processor
  - definitely for trigger, and also offline?
- Identify systems for processing of standard and non-standard workflows
  - grid-like farms, supercomputers, commercial clouds, … ?
- Explore portable solutions and/or automatic code generation/optimization
  - same origin for different code on heterogeneous systems
    - identify data structures efficient for different architectures and models
    - what about auto tuning techniques? average physicist would write 'simple' code with algorithm flow, actual optimized code auto generated

Strategies we could pursue (II)

- Study deep learning techniques
  - can machine learning be used for full event reconstruction?
  - or should we envision a heterogeneous reconstruction in terms of algorithms, e.g.:
    - ML for early event tagging, traditional algorithms for different reconstruction paths based on tag
    - ML for pattern reco, traditional fit
    - ...
  - how can we effectively input the physics to machine learning? so that training does not waste cycles on things we already know how to do, nor picks up unphysical solutions
- Development model may need to be revisited
  - centralized work gives higher quality code and consistent data products
  - distributed work gives access to more resources, closer to analyses
  - need a proper balance between the two modes!

We are not the only ones asking ourselves these questions:

Community White Paper (CWP) Working Groups, Software Trigger and Event Reconstruction WG:

http://hepsoftwarefoundation.org/cwp/cwp-working-groups.html