Framework & Tracking @ LAL

Hadrien Grasland

LAL – Orsay

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 654168.

Framework Extensions

Context

- Task 3.4: Extend frameworks for **parallel event processing**
  - Initially aimed for common Gaudi/Marlin/Pandora solution
  - …but Gaudi went its own way before AIDA-2020 started
- LAL decided to refocus on **parallel condition handling**
  - No condition handling in Gaudi itself → effort duplication
  - Singleton storage must be re-thought for parallelism
  - An ATLAS-specific take on this exists, not generalizable
  - Proposal: Provide generic components at Gaudi level

Known use cases

- LHCb have a **30 MHz HLT** with **~unchanging** conditions
  - Condition readout should be fast
  - Per-event overhead should be minimal
- ATLAS have **heavy state**, changing **rarely** (<1/1000 evts)
  - Do not store one full detector state per event in flight
  - Bounded amount of states, sharing data with each other
- Everyone wants to remain compatible with **legacy code**
  - Support singleton storage as one possible implementation

What was proposed

- Bounded amount of detector states (“ConditionSlots”)
  - Can abstract all known condition storage schemes
  - Enables fast readout, event scheduling, usage tracking…
- Access metaphor akin to event data (“ConditionHandle”)
  - Tracks condition usage for the framework / detector desc.
  - Can serve as a cache to enable faster access
- Asynchronous storage allocation, raw I/O, derivation…
  - Allows buffering a few events instead of blocking right away
  - Good for processing shuffled IoV boundaries in reco.

What was done

- Enumerated known use cases for condition data
- Devised a generally applicable scheme
- Proved viability and performance via prototyping
- Investigated shortcomings of Gaudi infrastructure

- Implemented & submitted fixes for some, but…
  - Very hard to get feedback during development
  - Very hard to get review on the final work
  - One MR got merged under LHCb pressure… then reverted
  - Replacement has been awaiting review for months…

Meanwhile, on LHCb side...

- LHCb Run3 is getting close...
  - More pressing need for parallel condition handling
  - Difficulties of this project are obvious
- Solution: Take this work in experiment-specific direction
  - Re-use API design work from this project
  - Drop dependency on Gaudi infrastructure improvements
  - Implement everything as experiment-specific components
  - Support legacy LHCb conditions, DDCond being worked on

Where does that leave us?

- It is very hard to get any meaningful change into Gaudi
  - Wildly diverging visions across experiments
  - …so experiment-specific code almost always preferred
- No one will push for this work to be integrated now
  - No LHCb incentive anymore, ATLAS never really tried
  - Future prospects not meaningful on AIDA-2020 time scale
- The future of this project has never been less certain
  - Benefits keep shrinking, costs keep rising
  - Isn't there a better use for LAL's remaining time?

Advanced tracking tools

Context

- Task 3.6: Common developments on track finding & fitting
  - Initially envisioned in the context of aidaTT toolkit
  - …but since then, ACTS was announced and released
- ACTS is a deep rework/generalization of ATLAS Run2 code
  - Turns battle-tested code into an experiment-agnostic library
  - Enables new use cases like efficient multithreading
  - Used in FCC studies, TrackML, interest from LC community
- Consensus was that contributing to ACTS is best

Previously…

- Finished:
  - Implemented and integrated multi-threaded validation
  - Improved API const-correctness for easier multithreading
- Ongoing:
  - Optimizing key components (Runge-Kutta, Kalman Filter…)
  - Numerical analysis study of ACTS floating-point code
  - Analyzing Belle 2 perf. in view of piecewise integration

2018-2019 news

- Second attempt at Kalman Filter optimization via GSoC
  - Sadly, didn't get much further than previous one…
- Numerical study of ACTS via dynamic instrumentation
  - Double-precision part completed, presented at CHEP 2018
  - Single-precision study needs more input to continue
- Optimized interpolated magnetic field map
  - Being reviewed, will provide 10x faster random field lookups
- Spack packages for ACTS are now available
  - Installing dependencies & building has never been easier

Kalman Filter optimization context

- Kalman Filter is a significant share of tracking CPU
  - In extreme cases like LHCb, can be the main bottleneck
  - In ATLAS, CPU usage is ~comparable to Runge-Kutta
- Mostly linear algebra, for which ACTS is using Eigen
  - But Eigen is not very optimized for small matrix operations
  - Can be outperformed by large margin, ~2x speedup on 5x5…
  - …without a losing the high-level linear algebra interface
- Previous work at LAL did this using SIMD wrappers
  - An issue for ACTS acceptance: no desire to maintain a BLAS

GSoC 2018 attempt

- Tried to integrate LAL linear algebra work in xtensor library
  - More familiar than Eigen for physicists with numpy training
  - Larger developer base (Eigen is a ~1-person project)
  - Much cleaner codebase with a nice layered design
- Unfortunately, that attempt didn't pan out
  - Too ambitious for a 3-month student project?
  - Insufficient person-power to finish in-house
- Thus Eigen remains unchallenged in ACTS… for now.

Numerical analysis of ACTS

- Applied Verrou, a tool for floating-point instrumentation
  - Can randomize roundings, cast to f32… w/o recompilation
  - Very useful for numerical validation & single-precision ports
- Validation results presented at CHEP 2018
  - *Floating-point profiling of ACTS with Verrou* presentation[1]
  - Proceedings were just accepted, should be out soon
- Single precision study needs more input to move forward
  - Now we know how precise current ACTS code would be
  - Unclear what is good enough, what should change

[1] https://indico.cern.ch/event/587955/contributions/2938061/attachments/1683124/2705039/ACTS_Verrou.pdf

Interpolated magnetic field map

- ACTS has a very flexible field map / grid infrastructure
  - Supports fixed and variable binning
  - Supports fixed boundaries, infinite edges, periodic bounds
  - Supports arbitrary position and magnetic field transforms
- Many field queries during track propagation: must be fast
  - …but never underwent a proper performance review
  - So I optimized it and made it 10x faster at random queries
  - No loss of flexibility, minor API changes
  - Will provide microbenchmarks to spot future regressions

Spack packaging

- Installing HEP software can be a bit challenging
  - Especially when one is not using SL / CentOS
  - Especially when one wants unusual build options
- Spack is a source-based package manager
  - Main alternative to LCGCMake being investigated by HSF
- I contributed Spack recipes around ACTS & Gaudi
  - …including dependencies such as DD4hep, AIDA…
  - Now, building ACTS is as easy as a “spack install acts-core”
  - Will take care of installing dependencies automatically

Future Prospects

- Finish study of applicability of single-precision in ACTS
- Contribute to ACTS integration in ATLAS (and Belle 2?)
- Contribute microbenchmarks to perf-critical components
- Contribute more performance fixes
  - Hopefully resolve small matrix algebra integration puzzle

Questions? Comments?