ATLAS Simulation: an Update on Performance and Related Topics

J. Apostolakis, J. Chapman, A. Dotti, S. Farell, A. Haas, Z. Marshall
for the ATLAS Simulation team

Geant4 Collaboration Meeting - September 2017

Outline

- Current Production & its version of Geant4
- CPU Performance Improvement
  - Porting G4Solids improvements for production (based on Geant4 10.1)
  - Exploring use of Geant4 static builds with Athena
  - Additional topics under investigation
- Impact of new infrastructure (git) on testing & performance tuning
- Preparation for MC18 using Geant4 10.3
- Geant4MT in AthenaMT
- Non-reproducibility between Intel & AMD

Current Production

- Running 13 TeV MC production to compare with 2015/16/17 data (MC16)
  - Geant4 10.1 patch03, CLHEP 2.1, 64-bit, gcc 6.2, SLC6, C++14
  - Only minor changes on top of vanilla G4 10.1.p03:
  - Using our “ISF” infrastructure by default
  - Using the FTFP_BERT_ATL physics lists. (Thanks!)
  - Running production on more exotic machines: HPCs, Amazon cloud, BOINC
- Still running tails of (much) older productions:
  - Geant4 9.6+ patches for “MC15” production
  - Geant4 9.4+ patches for “MC12” production

Patches on top of Geant4 10.1.patch 03

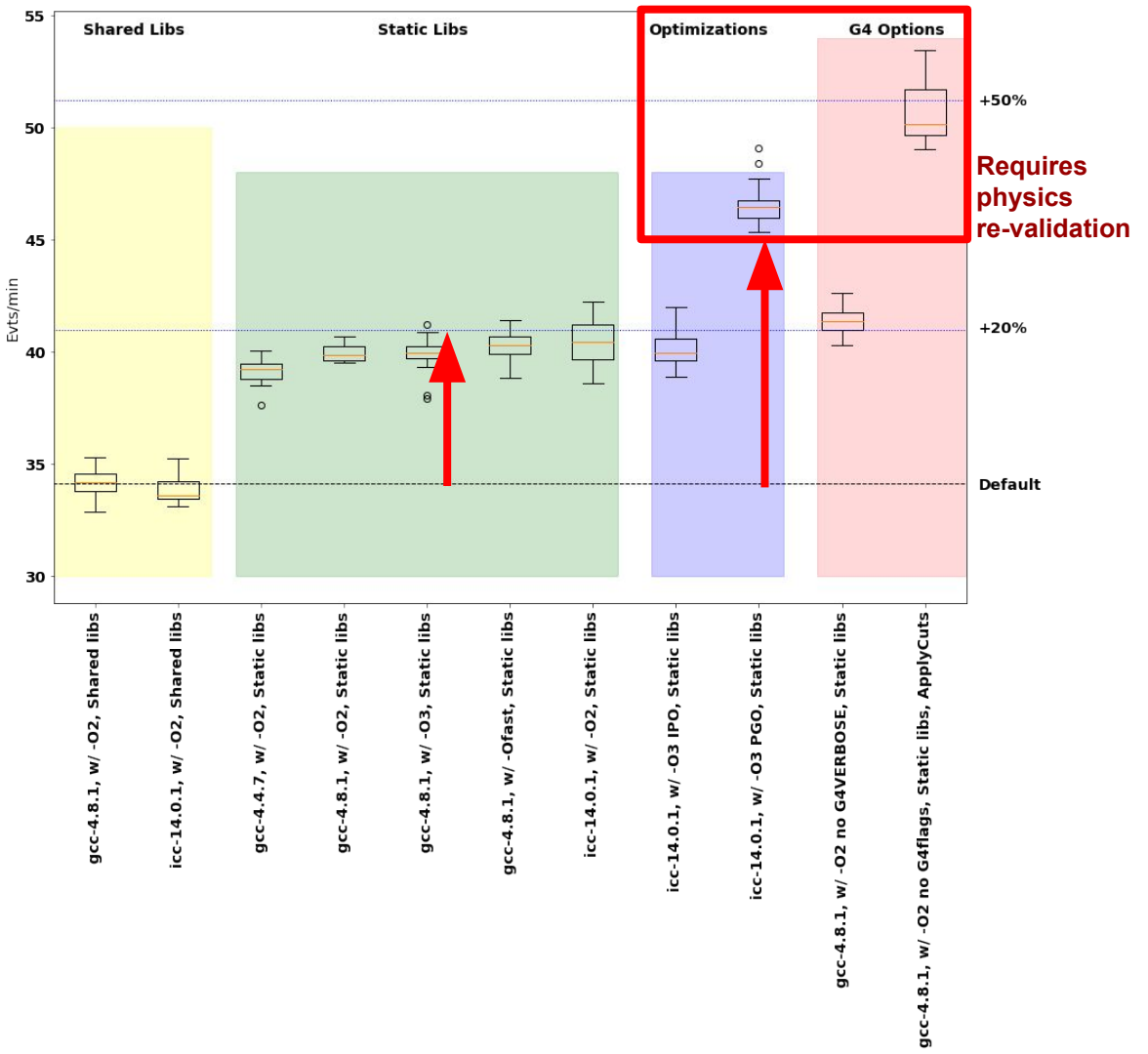
- Evgueni Tcherniaev suggested optimizations to some G4Solid methods, from his recent efforts (G4 development for 10.4)
- Recent tests using G4 10.1.patch03 + revised G4Solid implementations from G4 trunk yielded ~4% speed up in simulation time
  - This version has already passed our internal “physics validation” checks.
  - Doing more detailed checks of the G4 geometry before adopting this for production.
- Looking again at Runge Kutta steppers
  - Can recently patched G4NystromRK4 be used instead of ATLAS-specific G4AtlasRK4 stepper ? (suggestion of John Apostolakis.)

Statically-linked “Big Library”

- Aim to collect all the Geant4 dependencies in ATLAS code into a single shared object library, and statically linked it against Geant4.
  - This approach yielded ~10% speed improvements in initial tests with an old CMT-based release.
  - Andrea Dotti now working to reproduce it in our current CMake-based releases.
- This is also a starting point for further optimisation
  - profile guided optimisation (PGO) - as it requires static libraries.

Potential of static libraries

Static builds seen to provide a boost 10-20% in CPU performance in ‘simpler’ G4 benchmarks



Quick testing with Simulation-only Athena

- The switch to CMake has finally allowed simulation-only Athena releases
  - had been under development for some time.
- “AthSimulation” releases can be built quickly
  - 40 mins, including building Geant4, on a suitable machine.
  - can then be distributed on CVMFS and used on the grid.
- Enables us to easily test new Geant4 patches with Athena.
  - Used to test a few iterations of the G4Solid patches
  - Used to build the ATLAS software on top of Geant4 10.2 for recent comparisons with CMS
  - Used also for an initial test of Geant4 10.3.

Investigating performance on KNL

Initial tests on KNL (Cori @ NERSC) found cycles per instruction (CPI) rate of 3

- Investigation saw 60% frontend bound stalls; instruction cache thrashing
- Similar results in G4 standalone – we want to understand this!

Seen larger improvement from use of static libraries (than Xeon).

Geant4MT in AthenaMT

Substantial progress in converging on the prototype ‘Geant4MT’ simulation in AthenaMT with the ISF framework

- Some limited issues with ATLAS-side calorimeter code remain to be ironed out
- Could enter validation in the near future

Progress made on a random number streams.

Preparing the next Production MC18

- Getting ready for launching of MC18 (!)
  - Aim to have the release production-ready by mid 2018.
  - Geant4 10.3 patch 03, CLHEP 2.4, 64-bit
  - Expecting this to be the main production platform through LS2 (late 2021).
- Compilers & OS
  - Plan to use gcc 6.2 in C++14 ‘mode’, currently on SLC6
  - More serious tests of clang and AMD porting.
  - Still testing ICC, Mac OS X builds (but no production plans)

Intel vs AMD non-reproducibility

- Thanks to Alberto for his detailed studies into the output differences between the same simulation jobs run on Intel and AMD machines.
- The differences are related to whether the *sincos* function vs separate *sin* and *cos* functions are used on each CPU.
- Effects were found to be much stronger for hadronic physics than EM physics. This is now understood.