EvtGen status report

Michal Kreps, John Back, Tom Latham,
Fernando Abudinen, (Alex Ward)

WARWICK
THE UNIVERSITY OF WARWICK

EvtGen status

- Generator package specialised for heavy-flavour hadron decays
- Used as well inside simulation of b jets
- Contains about 130 decay models implementing specific dynamics of various decays
- Maintains detailed decay table with large number of explicit decays
- Known decay branching fractions do not add up to 100%, remainder is filled up by generating quark configurations and passing those to Pythia8 for fragmentation
- Fraction of decays passed to Pythia8 depends on particle (b-baryons rely more on Pythia8 than others)
- τ decays simulated using TAUOLA
- PHOTOS used for simulation of final-state radiation (FSR)
- Source code stable over past 10 years (most changes due to addition of new models)
- Recently went through some modernisation and clean-up

Code issues

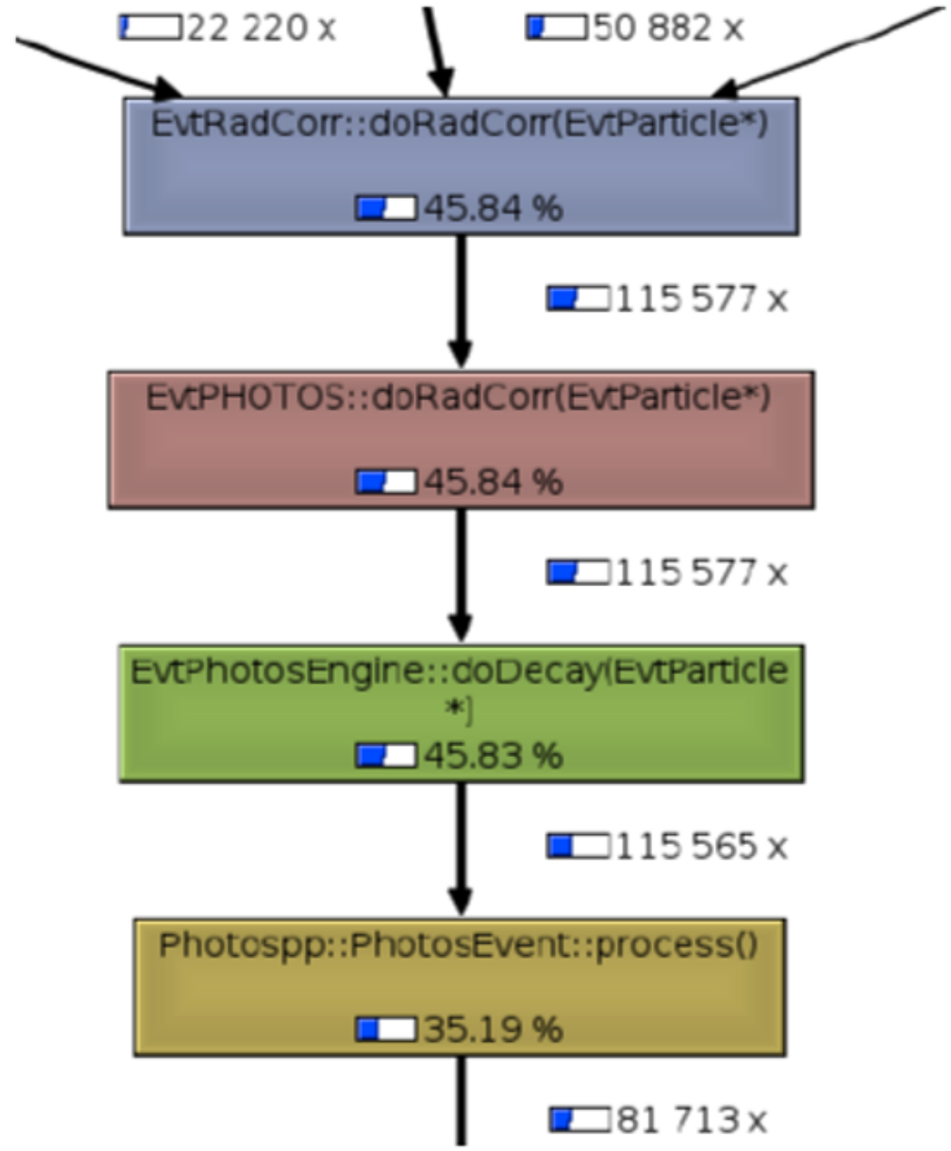
- Reasonably good design
  - based on interaction with SW engineer looking through code to make it thread safe
- But it has lot of small things, which can be improved with modern C++ features
- Code uses lot of different styles as contributions came from different people
- There is quite some duplication in code for some models
  - Often same physics model just different form-factors
- Documentation definitely requires improvements
  - Lot of decay models added after initial phase are missing from any documentation

Code issues

- Modern CPUs do not necessarily increase speed of single core, but rather pack more cores to CPU
  - Particle physics simulation can be easily parallelised by doing different events on different cores
  - But with more core, memory per core decreases and often one needs multithreading to benefit
- Experiments (main users) are moving their simulation frameworks towards multithreading and so they need all components to be thread safe
  - Does not help much EvtGen itself, but aims at decreasing memory footprint
- Some challenges from external dependencies like PHOTOS and TAUOLA
  - With understanding that Pythia8 is thread safe

PHOTOS in EvtGen

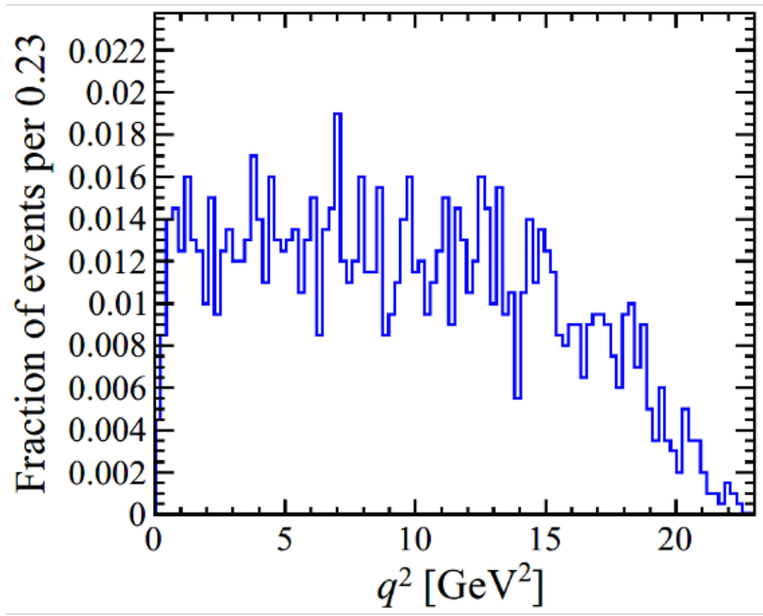
- PHOTOS is commonly used in almost every decay
- Profiling shows a significant amount of CPU time consumption in PHOTOS itself
- Conversion EvtGen « HepMC also significant
- Similar conversion happens inside PHOTOS
- Probably half of CPU time effectively spent on conversion
- Need to try bypassing HepMC to estimate possible gain
  - Usually ~1/3 of EvtGen CPU time spent on FSR simulation



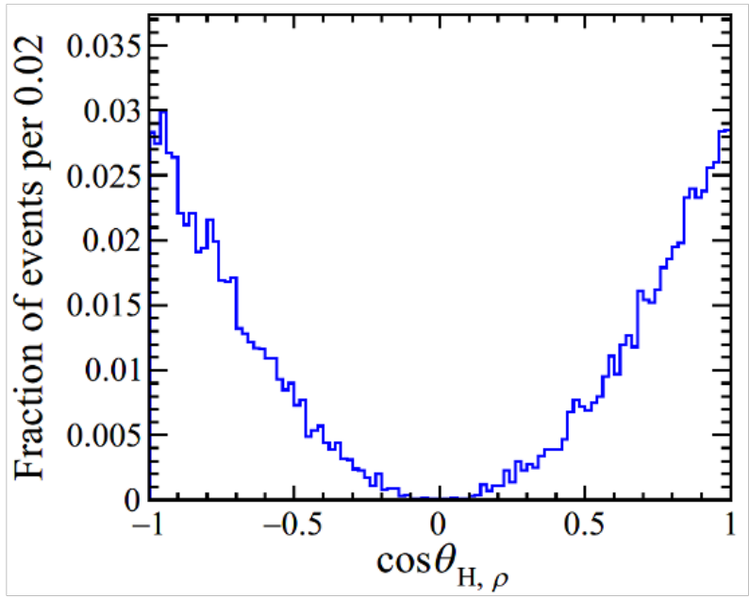
Testing framework

- Simulation needs testing and validation after structural changes due to code consolidation and implementation of thread-safety
- Tests (in different formats) existed only for about 40% of the 130 decay models
- Migrated all tests and added new ones to a common testing framework
- Implemented automatic recognition of tests to be run depending on changes (still to be refined)
- Finalised first working version with tests for all models
- Will require to add new tests for each new model
- Was holding some other changes
- While developing this, we discovered couple of issues with some decay models

$B^+ \to K^+\mu^+\mu^-$



$B^+ \to \overline{D}^0\rho^+(\to \pi^+\pi^0)$

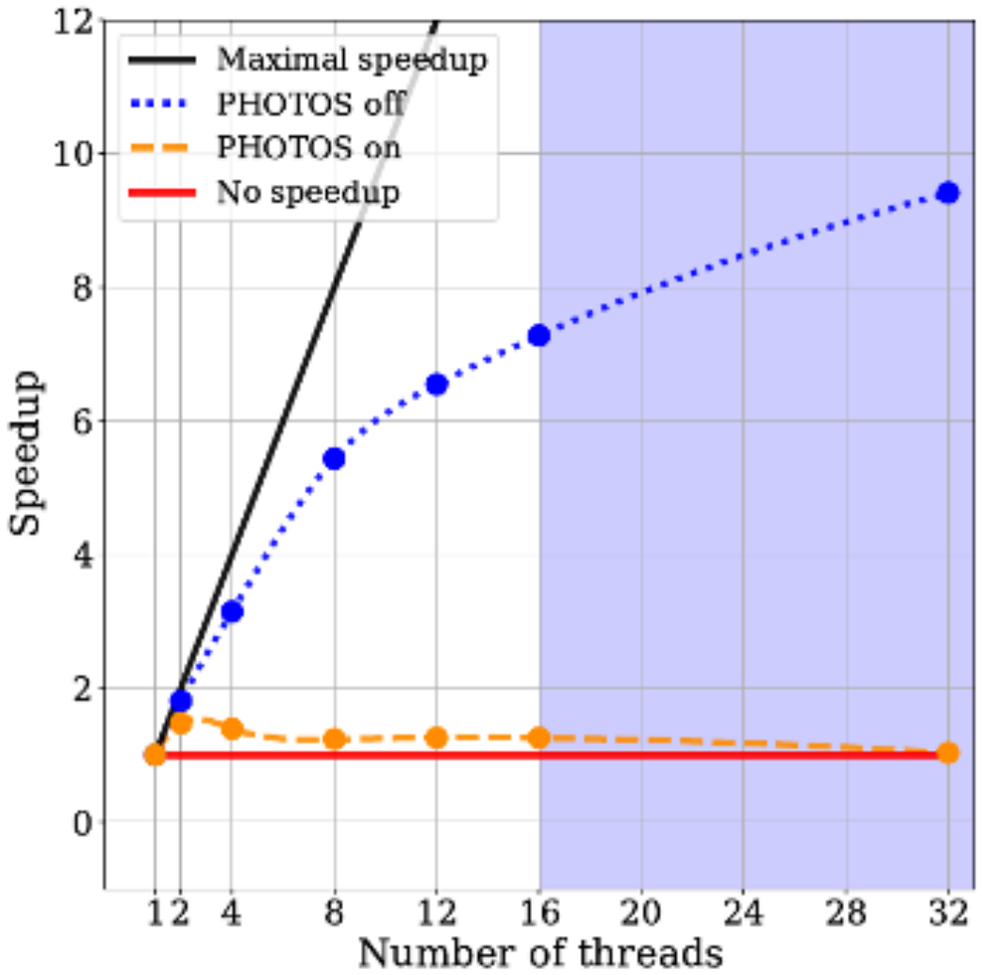


Thread safety

- Help from Heather Ratcliffe and Chris Brady (SW engineers)
- Code executed in parallel cannot modify its state
- Internal limitations:
  - Global random number generation
  - Global instance of particle properties and decay table
  - Decay models modify their state
- External limitations
  - PHOTOS and TAUOLA are not thread safe (authors are exploring options, but without clear timeline)
  - TAUOLA could be potentially replaced with Pythia8
  - Need to think about alternatives for FSR

Thread safety

- First version in hands
  - Simplest possible modifications
  - Change static objects to static const where possible
  - Global singletons made thread-local
  - Serialised (muttered) calls to PHOTOS and TAUOLA
- All tests we have has passed
- Performance gain very limited by external dependencies
- With per event/particle seeding of random number generator fully reproducible results independent of number of threads
- Further work needed and identified



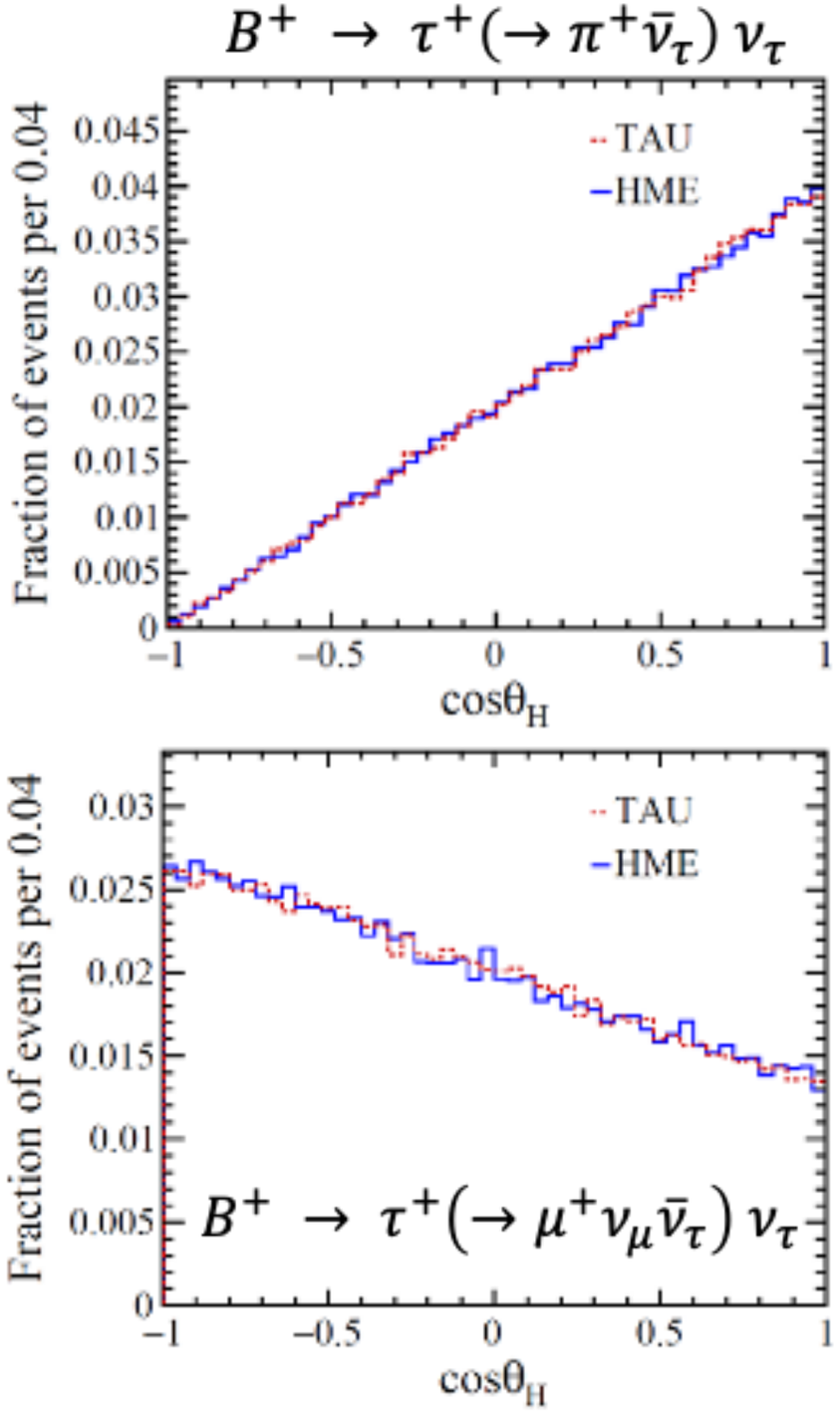
τ decays with Pythia8

- In addition to multithreading limitations, spin-state information of τ not propagated between EvtGen and TAUOLA:
  - TAUOLA expects τ from W, Z, γ or H, not from B
  - needed for analyses sensitive to τ polarization

- Simulation of τ decays with spin-state propagation possible with PYTHIA8 using HME (helicity-matrix element) amplitude model.

- Main EvtGen « Pythia interface ready

- Need to iron out conversion of helicity/spin basis (and initialization)



Further improvements

- Decay table would be better global rather than thread local but it contains decay models which modify their state
- All currently have to provide two functions:
  virtual void decay(EvtParticle* p) = 0;
  virtual void makeDecay(EvtParticle* p, bool recursive = true) = 0;
- These need to be made const
- The first one is called only from the second one
- makeDecay function is non-const only because of decay function
- Have proof-of-principle that we can actually achieve what we need, but will have to touch every decay model, requires lot of work

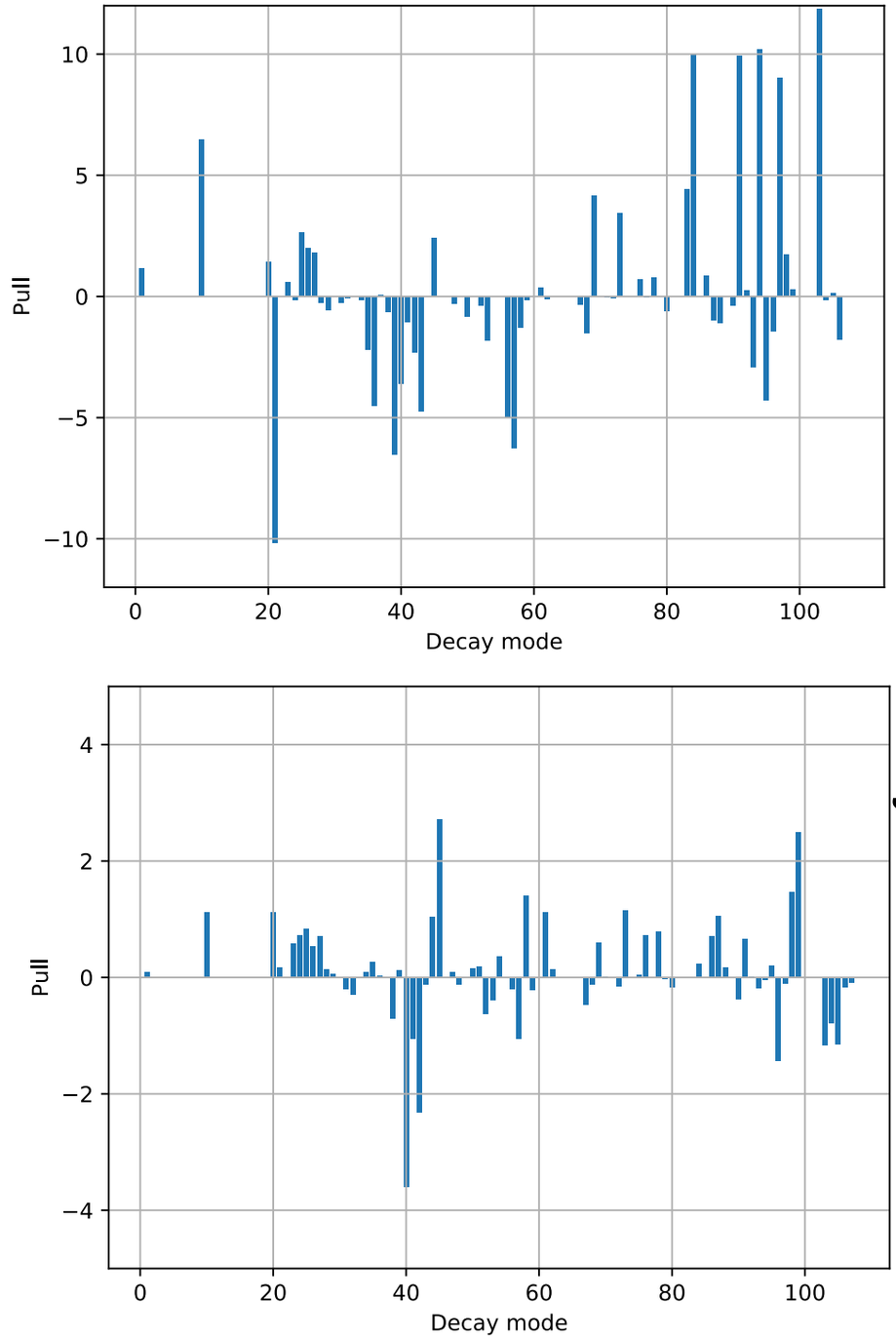
- Finish τ decays with Pythia8
- Investigate alternatives to PHOTOS (Sherpa/Herwig++ have implementation of FSR)

Decay table update

- Update of decay table difficult due to unavailability of good input data
  - PDG does good job of collecting all measurements
  - Does not have good metadata and often one has to go back to papers to understand exact meaning of the measurement
- Basically impossible to make automatic update
- Some time ago I made test of possible way out by tuning D_s decay table
- Rather then update all branching fractions, generate D_s decays and compare individually measured decays to WA (PDG)
- Tune worst offenders until χ^2 becomes reasonable or does not improve because of conflicting information
- Ignore inclusive branching fractions in this, but check them at the end of process

Decay table update

- Before tuning χ2/ndf=963.1/72
- After tuning total χ2/ndf=57.8/72
- Significant improvement
- Decays with pulls above 2σ are:
- $D_s \rightarrow f_0(980)\pi^+$ with $f_0(980) \rightarrow K^+K^-$
- $D_s \rightarrow 2K_s\pi$ due to tension with $D_s \rightarrow K^*(892)^+K^0$
- $D_s \rightarrow K_S 2\pi^+\pi^-$
- $D_s \rightarrow f_0(1710)\pi^+$ with $f_0(1710) \rightarrow K^+K^-$
- Idea works reasonably on charm
- Would be good to repeat exercise for all bottom and charm
  - Have to see how it will scale to larger number of decays



Summary

- Mainly consolidation and bug fixes in past few years
- Almost finalised testing framework which allows to cross-check that changes do not change things (unless expected)
  - Just need bit of work on setup to run tests automatically
- Have now first thread-safe version with clear path in next steps to improve
- Have an idea how to move forward with decay table update, but requires lot of work to perform full update