EvtGen - Recent developments and prospects

CHEP 2023

MONASH University

Fernando Abudinén, John Back,
Michal Kreps, Thomas Latham

- Introduction
- Testing framework
- Enabling multithreading
- Other improvements
- Outlook

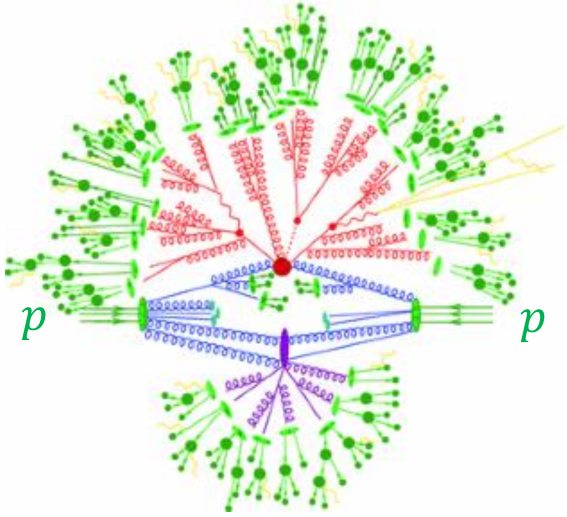
26th CHEP conference
Norfolk, Virginia, USA
May 9, 2023

WARWICK
THE UNIVERSITY OF WARWICK

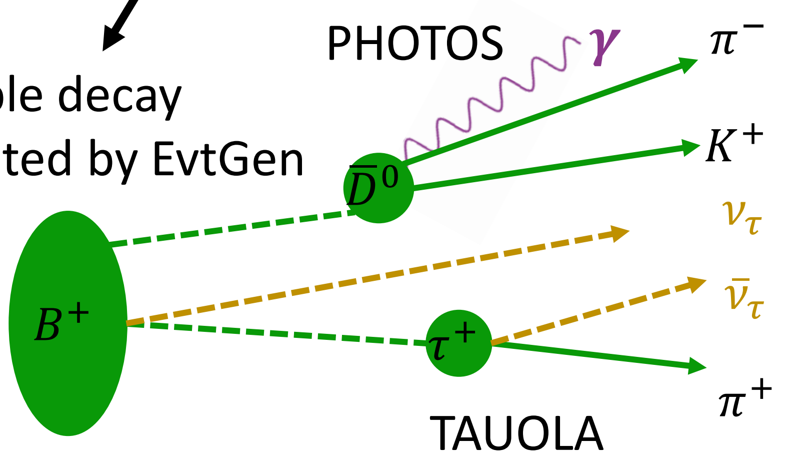
Introduction

- EvtGen: 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
- Final-state radiation (FSR) simulated using PHOTOS

Example collision simulated by Pythia8



Example decay simulated by EvtGen



Status

- Developed in the 90’s, stable over past 10 years
  (changes mostly additions of new models)

Challenges for updates

- Various code styles across models (due to contributions from various authors)
- Several code duplications across models (often same kinematics but different form factors)
- Experiments (main users) need generators to be thread-safe as they are moving their simulation frameworks towards multithreading to exploit modern CPUs

Recent developments

- Work on modernisation and clean-up
- First adaptation of core code towards thread safety (with help of software engineers)
- Implemented global testing framework for validation

Plans

- Physics wise no plan for changes in near future

- Currently working on code consolidation
  - Unify coding style, C++ modernisation
  - Plan to decrease code duplication within decay models
  - Improve/Update documentation (Doxygen and paper/guide)
  - Improve method to update decay table

- Continue work towards thread safety
  - Exploring alternatives for external dependencies that are not yet thread safe
  - Implementing full adaptation of internal code redesign

Testing framework

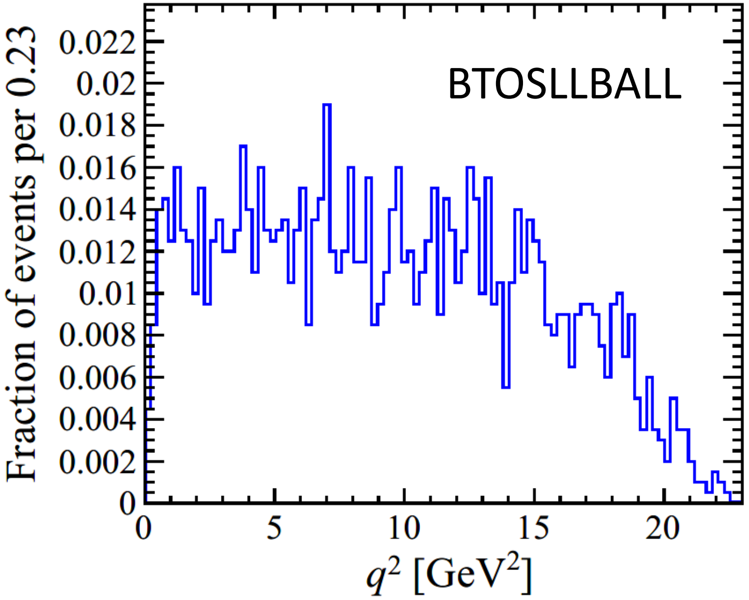
Testing framework

- 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
  - ⇒ With common testing module and configuration files
- Finalized first working version with tests for all models

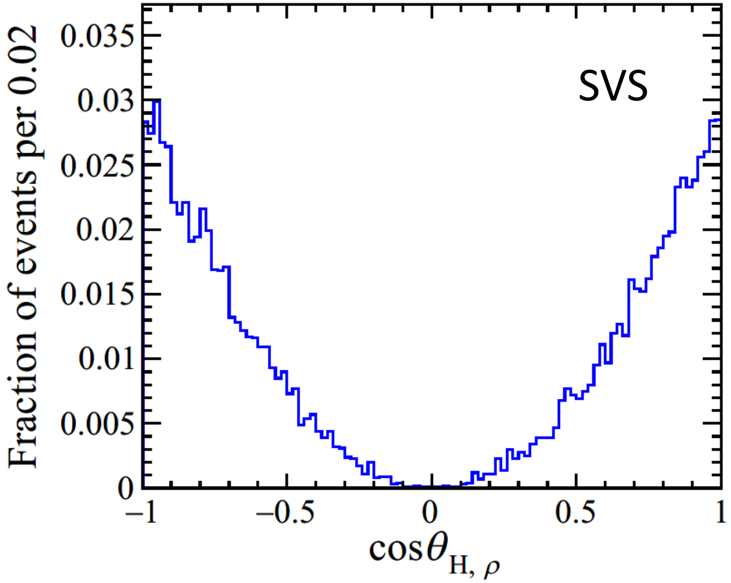
⇒ Served to discover and fix issues with existing models

⇒ Will require to add new tests for each new model

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



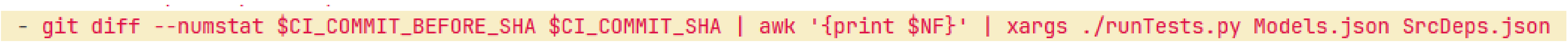
$B^+ \to \overline{D}^0\rho^+(\to \pi^+\pi^0)$



Testing framework

Implemented automatic recognition of tests to be run depending on changes

- Identify files modified in a commit
- If files associated with a model changed ⇒ run respective tests
- If framework files changed ⇒ run all tests
- Issue: Gitlab BEFORE_SHA variable not always set (for example when new branch created)



```
 - git diff --numstat $CI_COMMIT_BEFORE_SHA $CI_COMMIT_SHA | awk '{print $NF}' | xargs ./runTests.py Models.json SrcDeps.json
```

⇒ Need to decide what to compare to in such cases

⇒ Comparing with master branch could be a solution
(but is probably not what is needed in all cases)

Making EvtGen thread-safe

Challenges for multithreading in Evtgen

- Internal: structural limitations for multithreading inside EvtGen
  - Global instance of random number generator
  - Global instance of particle properties and decay table

⇒ Needed structural changes identified and first combination of solutions found

- External: limitations from dependences
  - TAUOLA
  - PHOTOS

⇒ Overcoming limitations from dependences are more challenging as they are external

  - TAUOLA and PHOTOS authors currently exploring ways to enable thread safety
  - Exploring use of Pythia8 as alternative to TAUOLA
  - Exploring use of Sherpa’s PHOTONS++ as alternative to PHOTOS

Progress on thread safety

With help from software engineers at Warwick University

Heather Ratcliffe
Chris Brady

Set of solutions to reach thread-safety (preliminary):

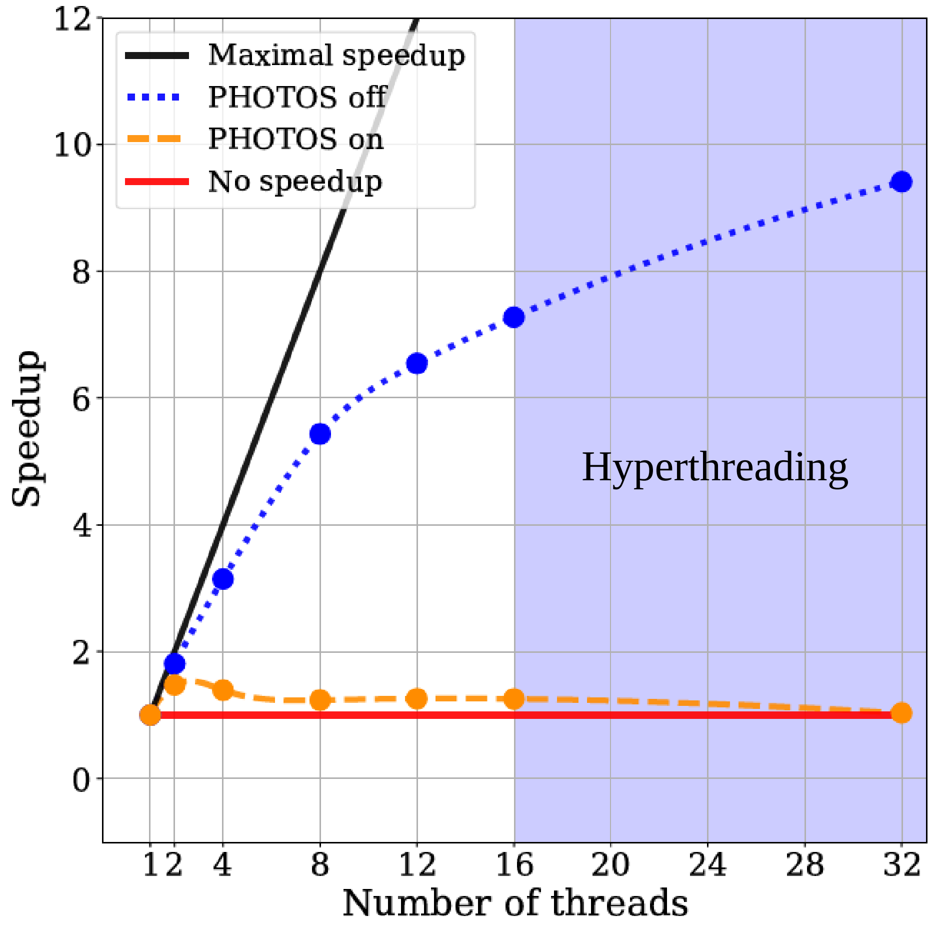
- Converted static objects to static const where possible
- Global singleton objects made thread-local
- Serialized calls (using mutex) to PHOTOS and TAUOLA

⇒ Deeper structural changes needed to fully exploit multi-threading (plan to continue working on it)

⇒ Reproducible results independent of number of threads

⇒ Current preliminary status reached thread-safety, passing tests for all decay models

⇒ But performance limited by external dependencies



Pythia 8 for τ decays

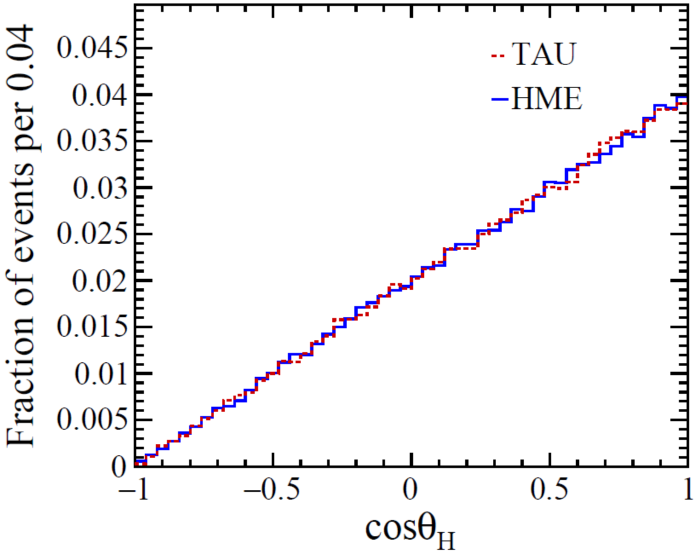
- In addition to multithreading limitations, spin-state information of τ not propagated between EvtGen and TAUOLA:
  - Needed for analyses sensitive to τ polarization

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

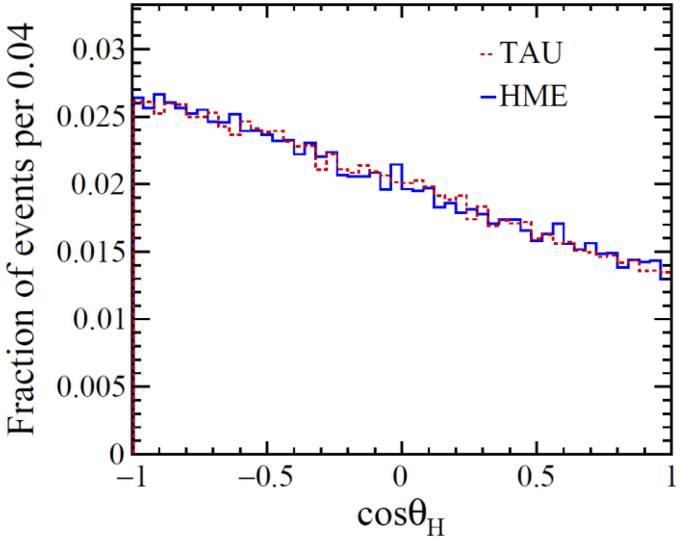
- Main EvtGen ↔ Pythia interface ready

- Need to iron out conversion of helicity/spin basis (interesting also for interface with TAUOLA)

$B^+ \to \tau^+(\to \pi^+\bar{\nu}_\tau)\,\nu_\tau$



$B^+ \to \tau^+\left(\to \mu^+\nu_\mu\bar{\nu}_\tau\right)\nu_\tau$



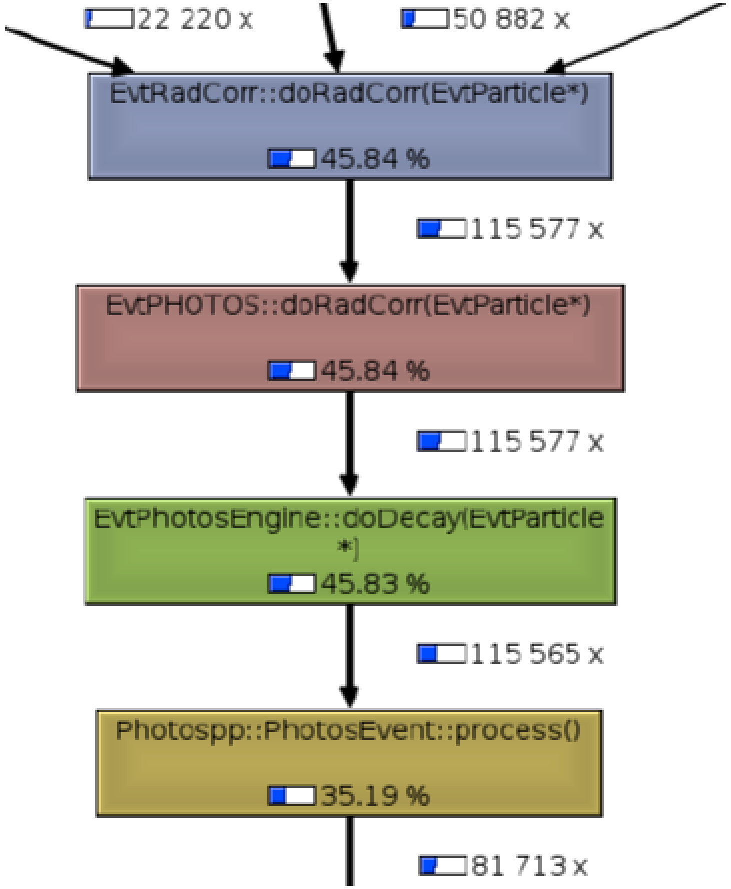
PHOTOS in EvtGen

- EvtGen does not consider soft photon emission from charged particles (final-state radiation)
- Full event is passed to PHOTOS and retrieved for FSR simulation

⇒ PHOTOS is commonly used in almost all decays

- 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



22 220 x

50 882 x

EvtRadCorr::doRadCorr(EvtParticle*)

45.84 %

115 577 x

EvtPHOTOS::doRadCorr(EvtParticle*)

45.84 %

115 577 x

EvtPhotosEngine::doDecay(EvtParticle *)

45.83 %

115 565 x

Photospp::PhotosEvent::process()

35.19 %

81 713 x

Sherpa’s PHOTONS++ for final-state radiation

- PHOTONS++ in Sherpa can simulate emission of soft photons (to higher orders of perturbation theory)
- If switched on, also hard photons (to first order)
- Algorithm implementation enables thread safety

$\Rightarrow$ Can be explored as alternative to PHOTOS

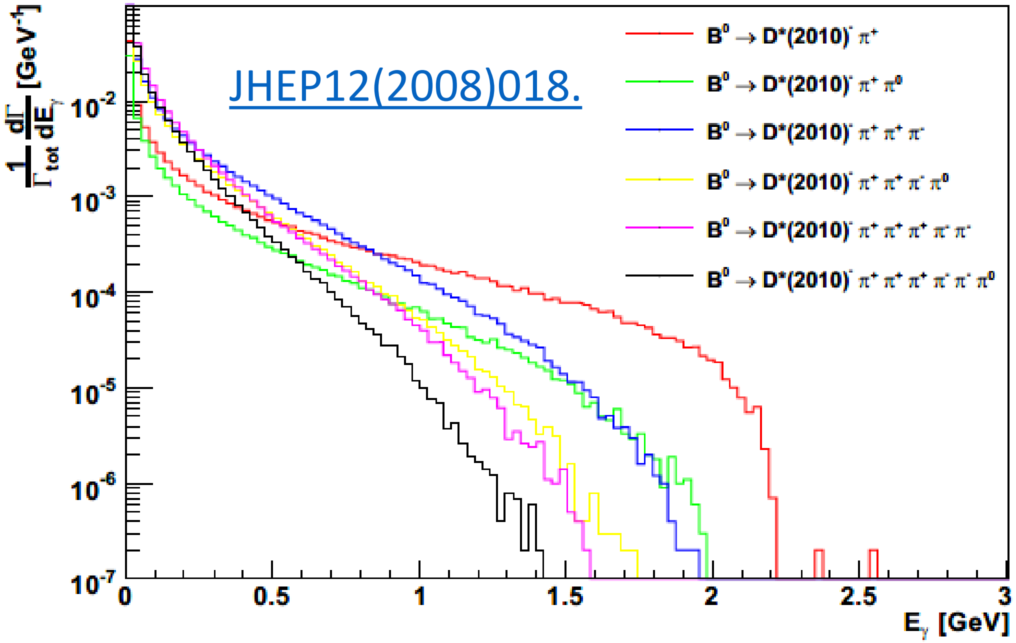
- Recently started work on EvtGen ↔ Sherpa interface

$\Rightarrow$ Implementation in progress

$\Rightarrow$ Requires tuning (for instance of cut-off energy)

$\Rightarrow$ And validation of physics output

Energy of radiated soft photon



JHEP12(2008)018.

Other improvements

Improving the decay table handling

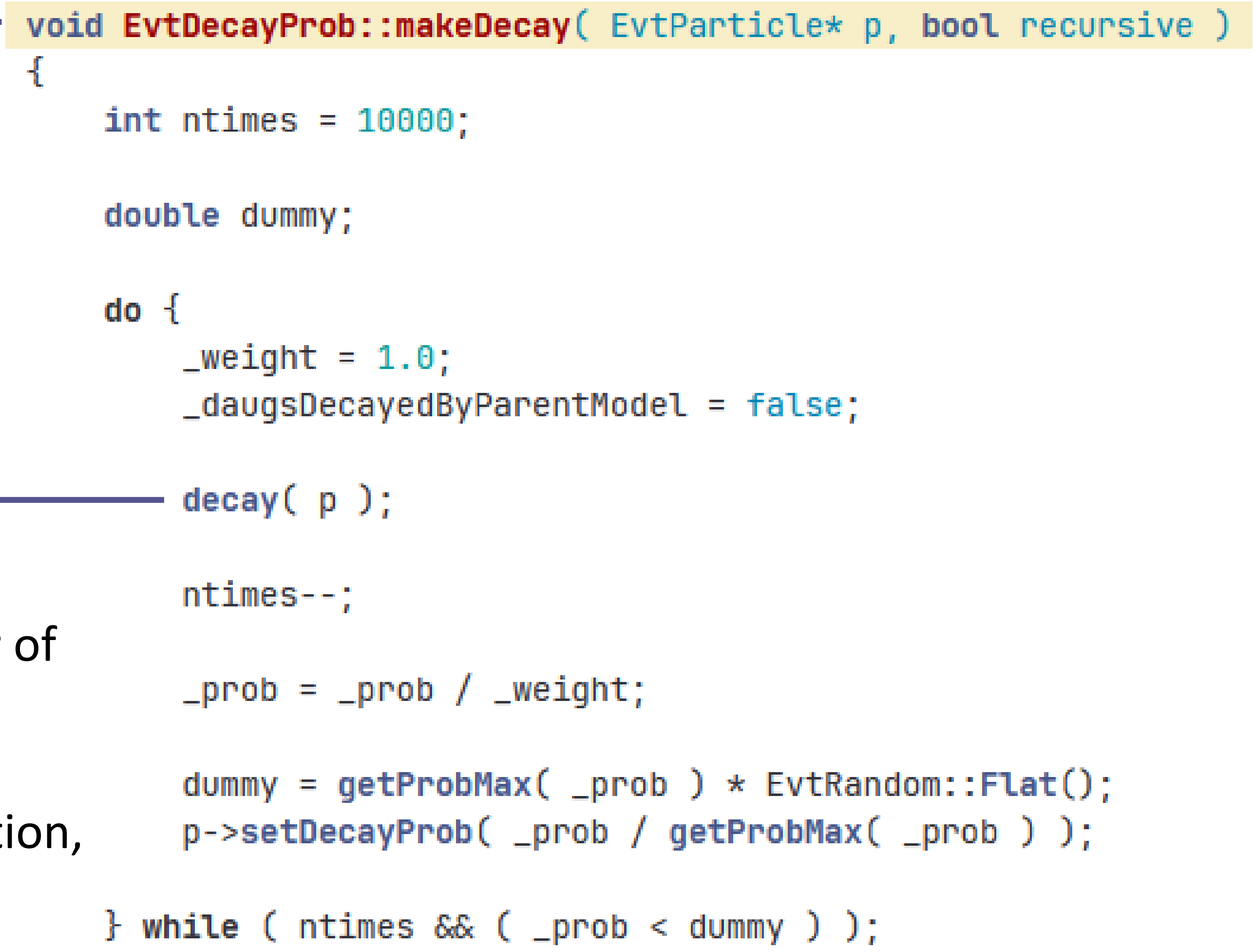
Decay table instance should be made const (rather than thread-local)

Class member function accepting/rejecting events and part of decay table instance (EvtDecayTable ↔ EvtModel ↔ DecayProb)

Function calling calculation of decay probability inside decay model and modifying **_prob**

⇒ Should return value instead of modifying member of decay table instance

⇒ Solution appears straightforward on initial inspection, but needs intervention in all decay models



```
void EvtDecayProb::makeDecay( EvtParticle* p, bool recursive )
{
    int ntimes = 10000;

    double dummy;

    do {
        _weight = 1.0;
        _daugsDecayedByParentModel = false;

        decay( p );

        ntimes--;

        _prob = _prob / _weight;

        dummy = getProbMax( _prob ) * EvtRandom::Flat();
        p->setDecayProb( _prob / getProbMax( _prob ) );

    } while ( ntimes && ( _prob < dummy ) );
```

Updating the decay table content

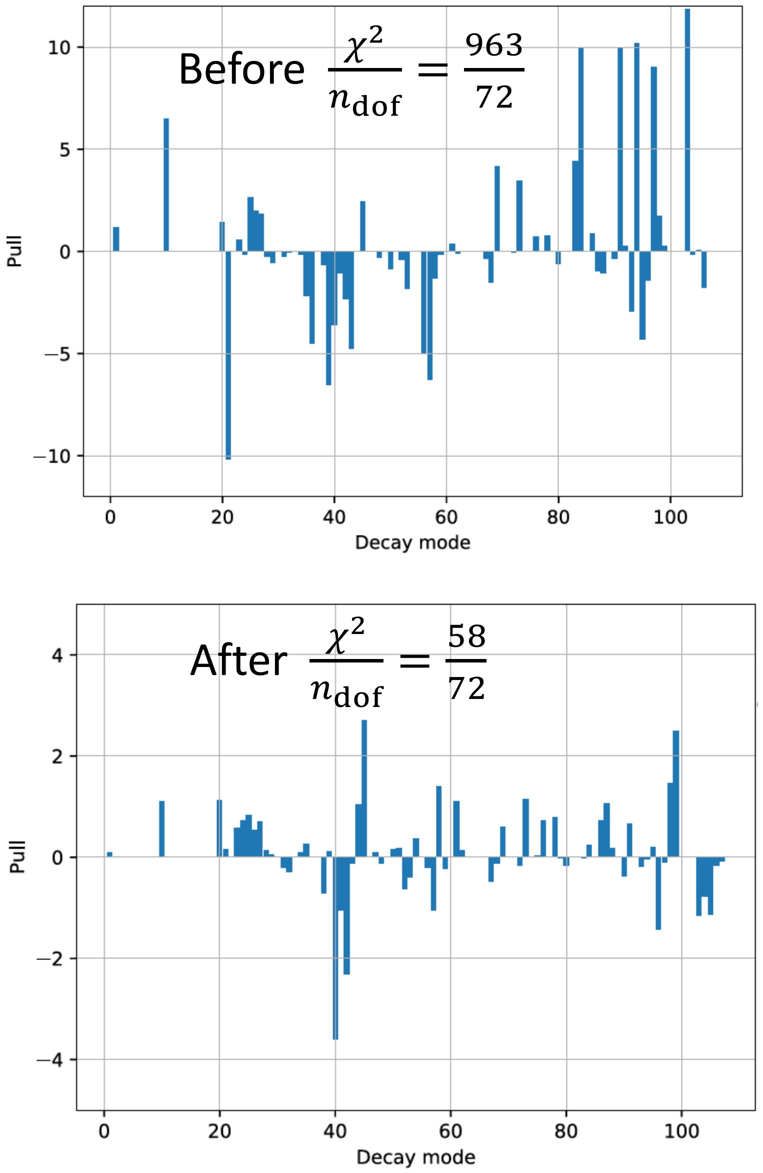
- PDG collects all measurements, but limited metadata
- Will probably improve with upcoming PDG API

⇒ Still will need checking actual papers to avoid ambiguities, for example double counting of resonant decay modes

- Explored recently possibility to update table content by generating decays and comparing fractions of generated decays with world averages of branching fractions
  - Tune worst offenders until χ^2 becomes reasonable or does not improve due to conflicting information
  - Ignore inclusive branching fractions in the tuning, but check them at the end.

⇒ Promising, but needs testing with larger number of decays

Test with D_s^+ decays



Be aware of different y scale

Summary and outlook

- Physics-wise code is kept stable
- Working on code consolidation (modernization, removing duplications, improving docu)
- Finalized common testing framework for validation
- Currently making EvtGen threadsafe

$\Rightarrow$ Converged on preliminary set of solutions to enable thread-safety of generator
(full exploitation of multithreading will require further structural changes)

$\Rightarrow$ Performance limited by external dependencies (especially PHOTOS)

$\Rightarrow$ τ decays: plan to iron out basis conversion for Pythia8 (interesting also for TAUOLA)

$\Rightarrow$ FSR: exploring use of Sherpa’s PHOTONS++ as alternative to PHOTOS

- Working on other improvements: handling and update procedure of decay table