Concepts for fast large scale Monte Carlo production for the ATLAS experiment

Chiara Debenedetti

The University of Edinburgh

on behalf of the ATLAS Collaboration

Event Processing, Simulation and Analysis session

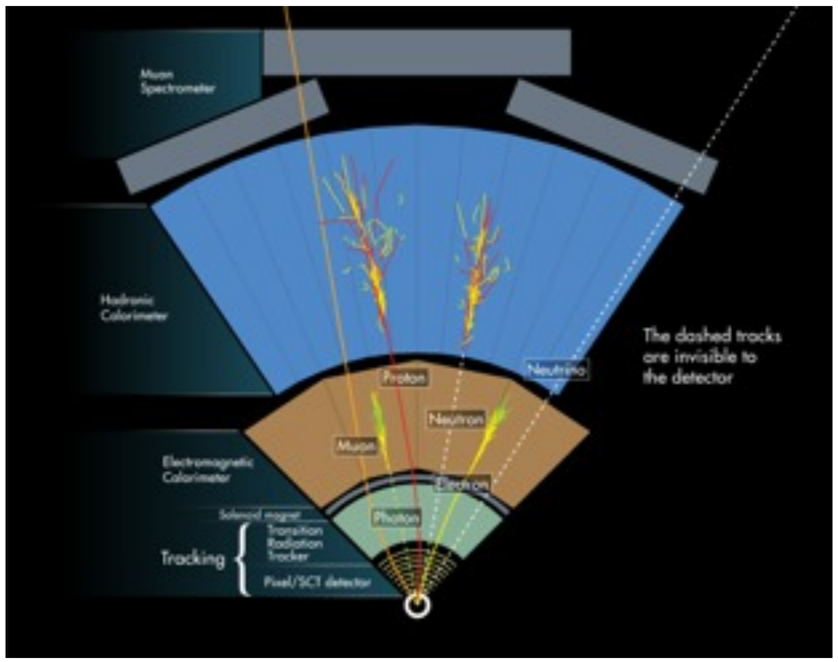
17.10.2013

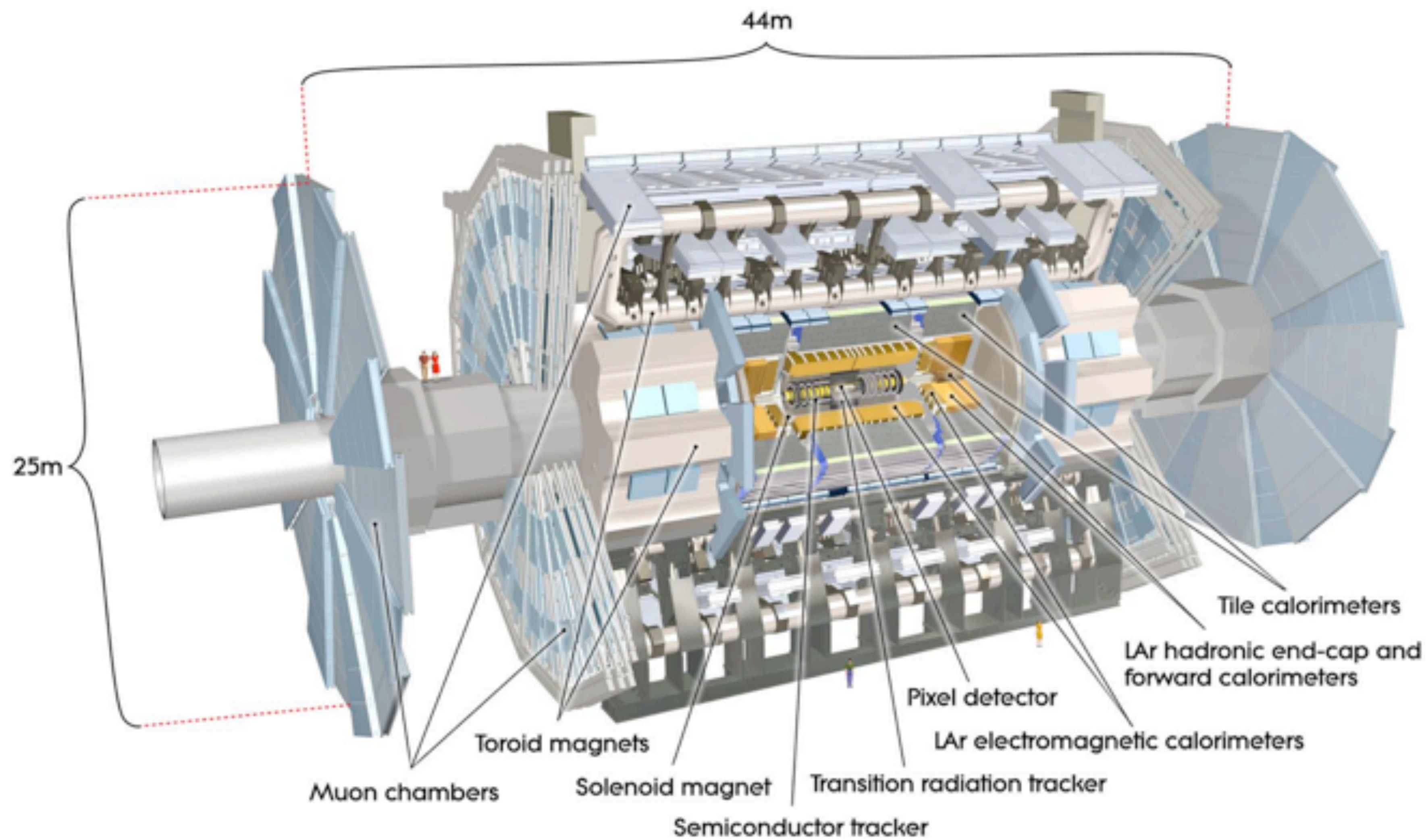
Overview

- ATLAS overview
- Why fast Monte Carlo production?
- State-of-the-art simulation
  - G4 simulation
  - alternative fast simulation methods
- Towards the future: Integrated Simulation Framework
  - implementation
  - performance
- Fast digitization
- Fast reconstruction
- Prospects for fast production chain
- Conclusions and outlook

The ATLAS detector

A quick glance at the ATLAS detector



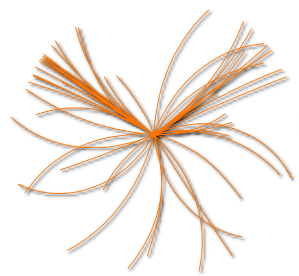


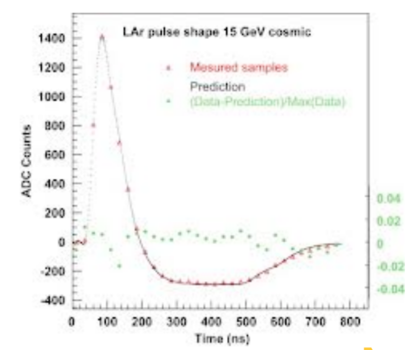
- Main subdetectors
  - **Inner Detector** => Silicon and transition radiation technologies, in solenoid magnetic field
  - **Calorimeters** => LAr EM calorimeter (in central and forward regions) and hadronic calorimeter (tile in central and LAr in forward region)
  - **Muon system** => spectrometer in toroidal magnetic field

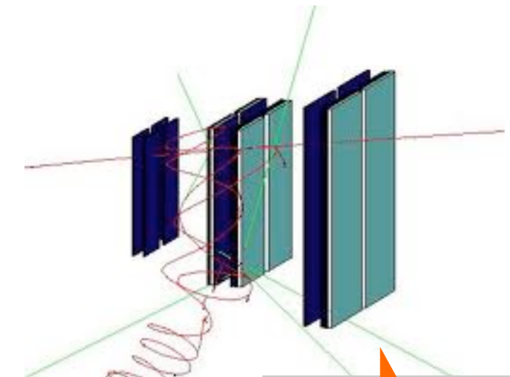
Why fast Monte Carlo production?

Current production chain

From 4-vectors to ROOT

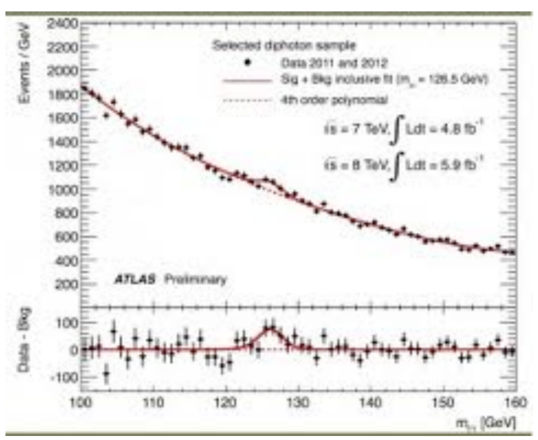


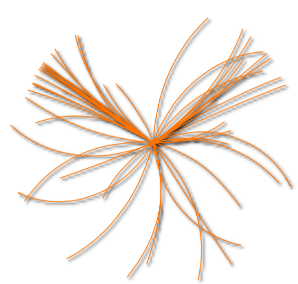





Event Generation → **Detector Simulation** → **Digitization** → **Reconstruction** → **Rootification**

Format: EVGEN HITS RDO ESD/AOD NTUPLE/HIST



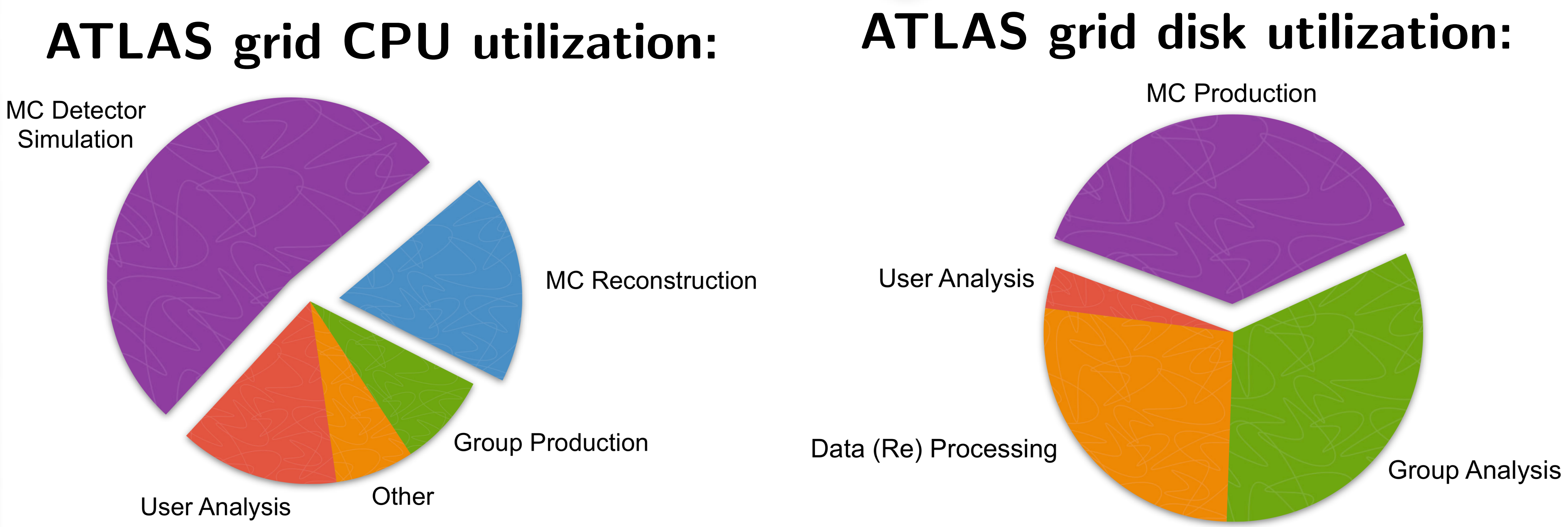



overlay up to 40 individual detector simulation events to emulate pileup

MC production on the Grid

- Grid CPU usage dominated by MC production
- MC production takes up large fraction of Grid disk usage => ***limitation***
- Precise detector simulation => highly CPU intensive
- Obstacle for physics analyses in need of large MC statistics => sensitivity limitation
- Higher luminosity and pileup => larger MC production needed

ATLAS Grid usage in 2012

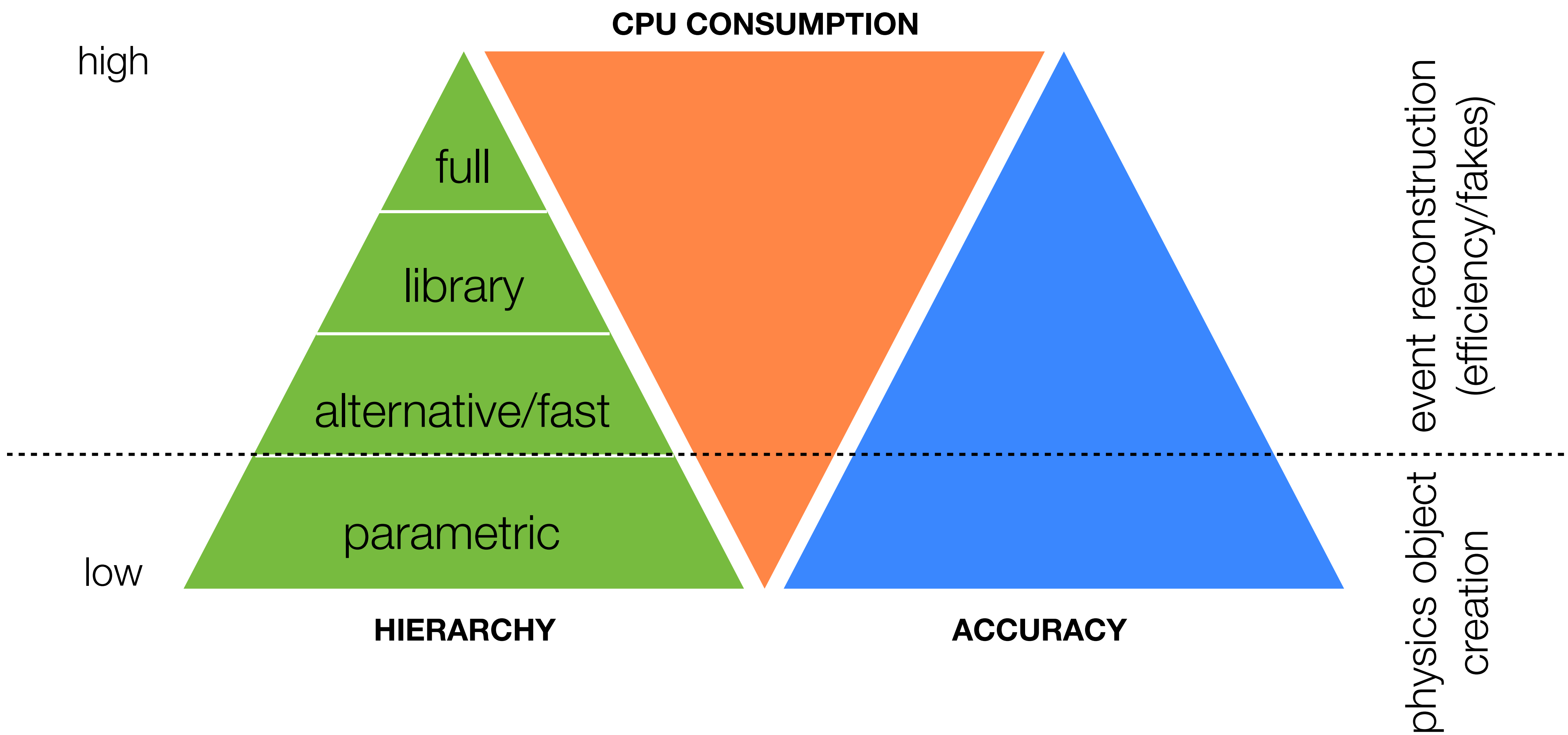


..that's why a faster chain is necessary for RunII!

ATLAS Simulation today

Simulation hierarchy

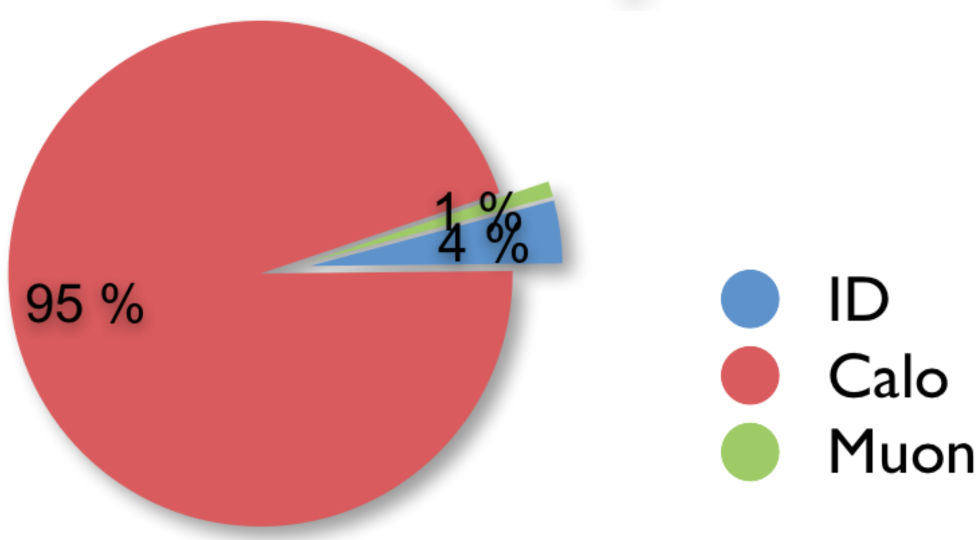
- Simulate interactions of particles with sensitive and non sensitive detector material
- Produce sensitive detector hits with position and deposited energy information => input to digitization
- More accurate simulation means slower simulation
  => *tradeoff between accuracy and speed*

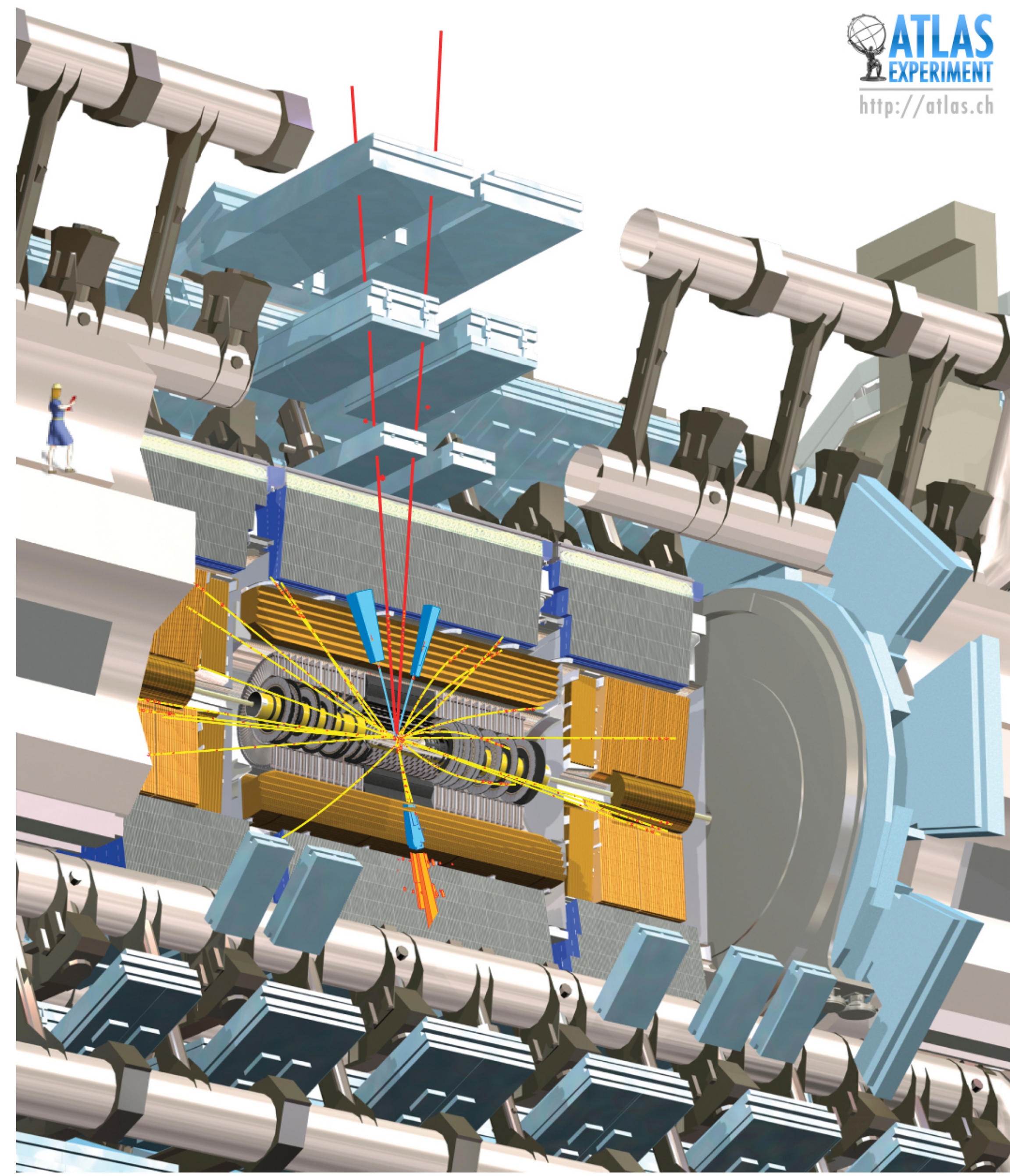


Standard ATLAS simulator: Geant4

- Close collaboration with Geant4 team
- Stable, fully validated and precise simulation
- Provides **full description of** the **ATLAS** detector
- High CPU consumption
  - mostly in **EM calorimeters**
  - every particle microscopic interaction with the material simulated
  - simulation of ~30M volumes

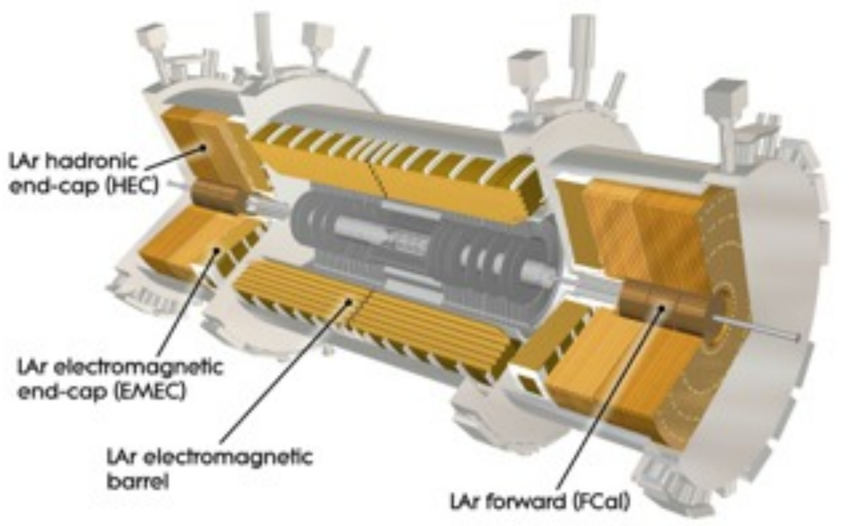
G4 simulation time per subdetector

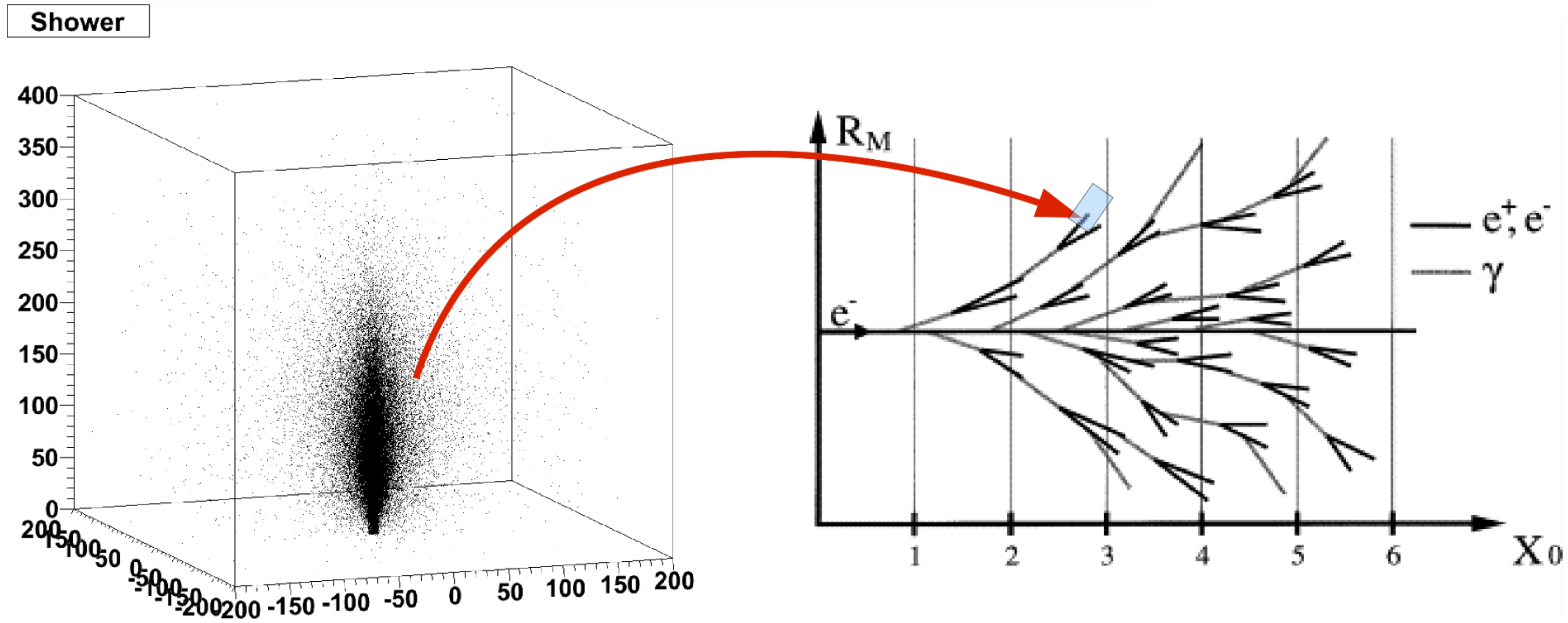




Speeding up: frozen showers

- Many high energetic particles in forward direction => **high CPU demand**
- Specific to forward EM calorimeters
- ***Idea***: replace low-energetic particles in developing particle showers with pre-simulated EM showers
  - libraries of frozen Geant4 showers assigned based on particle characteristics



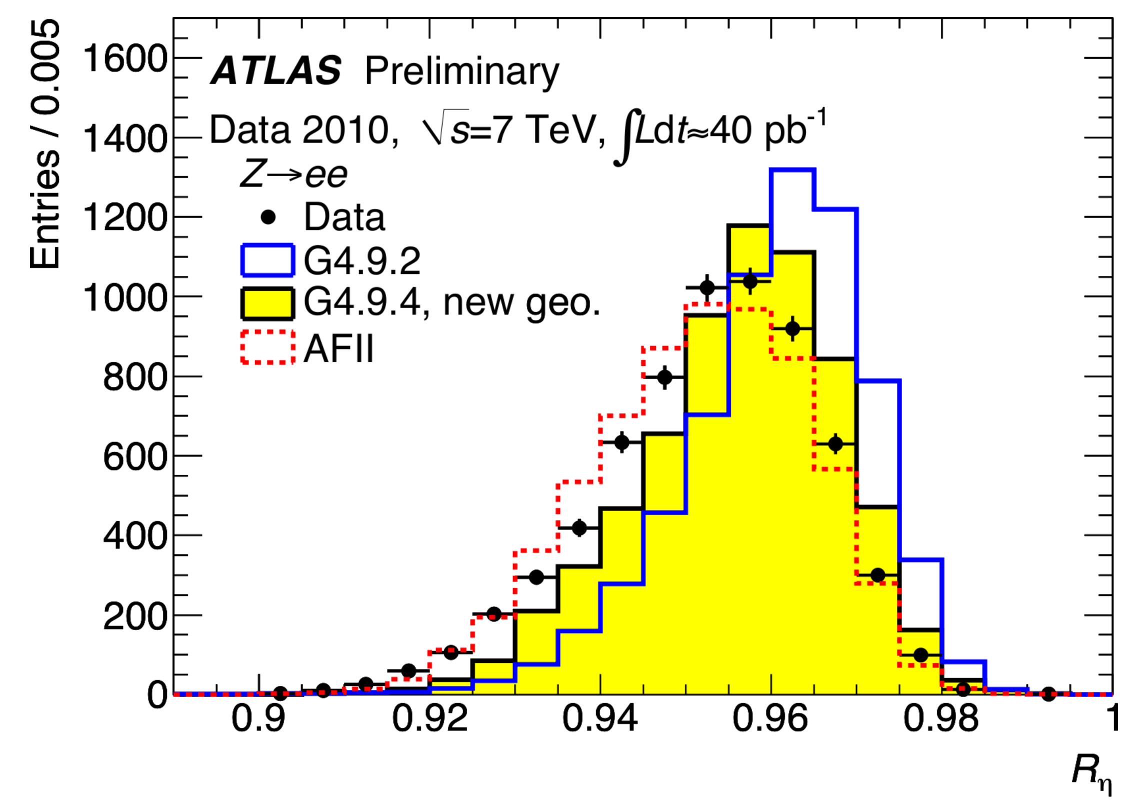


Default in ATLAS "Full simulation"
=> Geant4 + frozen showers for forward EM calorimeters

Even faster: FastCaloSim

- Parametrisation of calorimeter response based on Geant4 full simulation
- Use libraries based on incoming particles characteristics

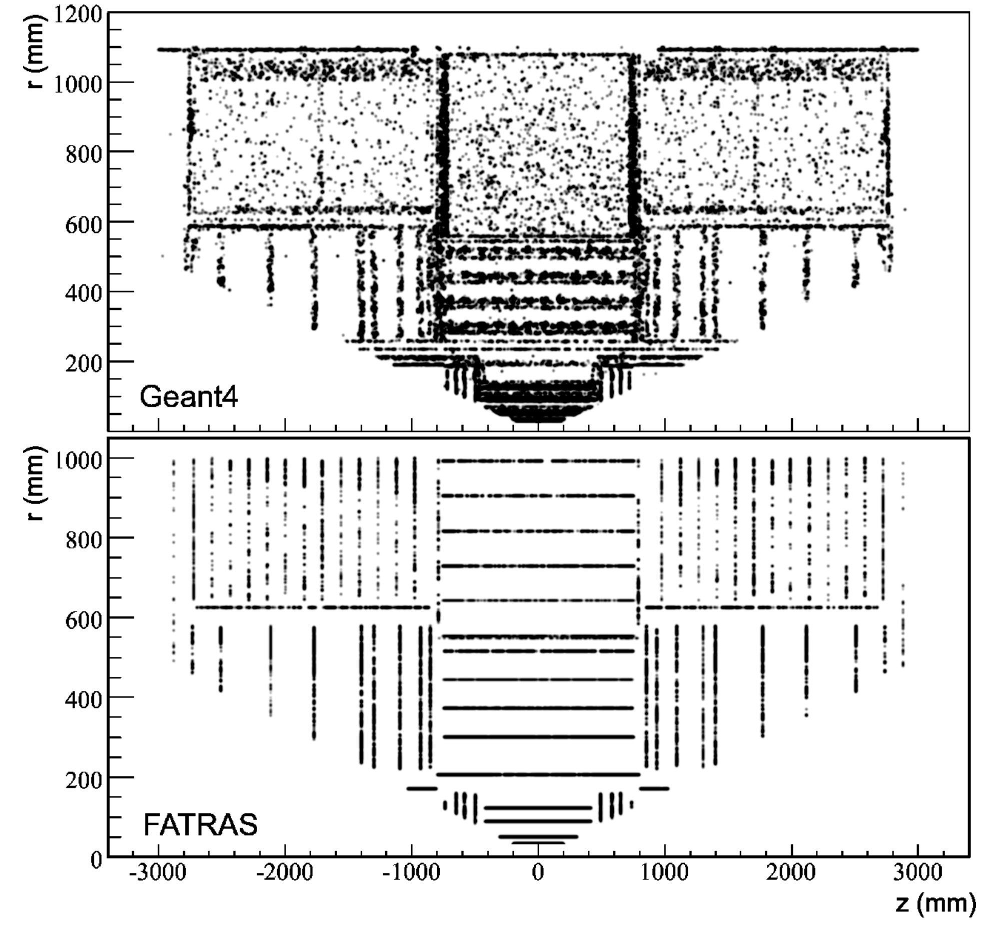
- EM processes well described
- **Tuned to data** for the EM shower shapes for latest production campaign (MC12)
- Currently **in use for ATLAS production** as an alternative for Full Simulation
- Used in physics analyses, requires dedicated calibration



Energy ratio R_η in a $\Delta\eta\Delta\varphi = 3\times7$ cell cluster with respect to a 7×7 cell cluster size in the bulk EM calorimeter layer 2

What about tracking? Fatras

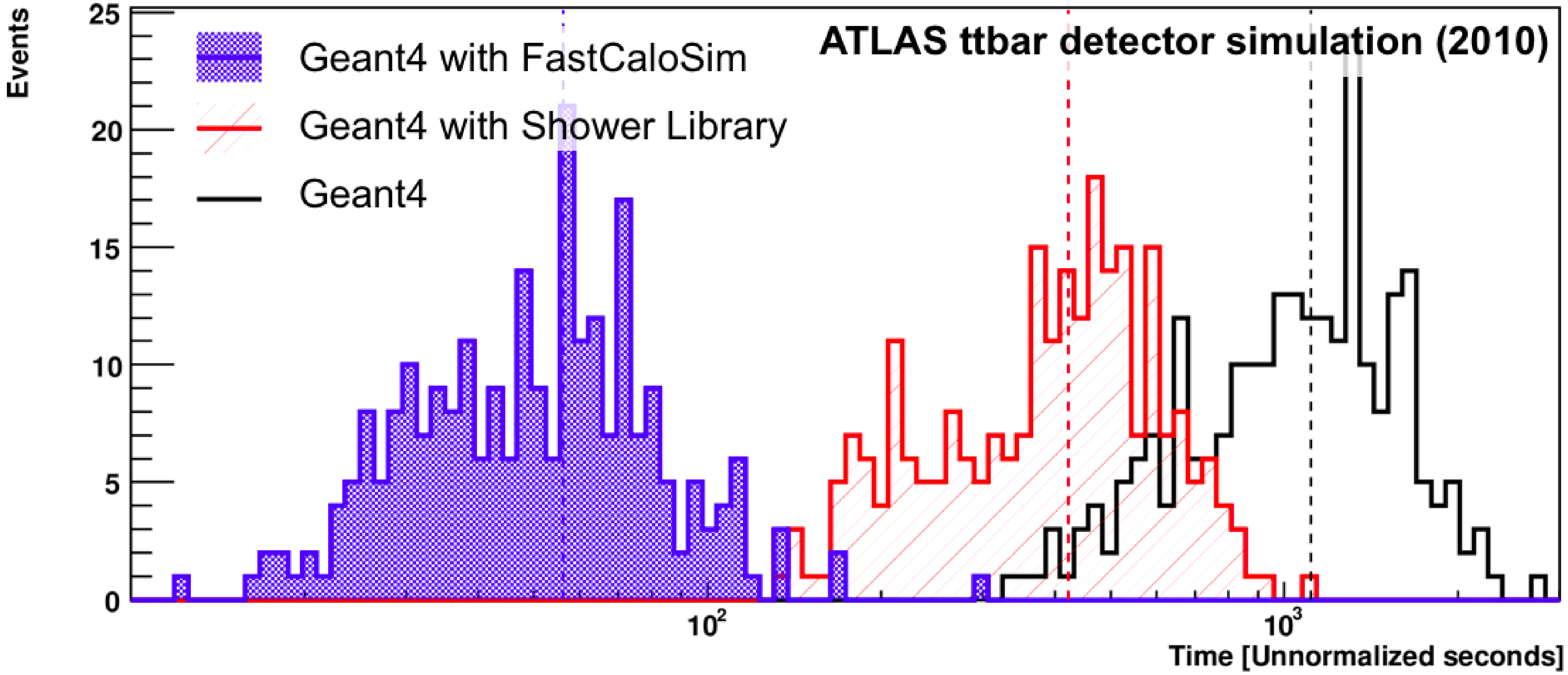
- Fast simulation for ATLAS tracking system
- Treats Inner Detector, muon spectrometer and muons interactions in the calorimeters
- ***Idea***: simplified detector geometry and interaction processes
- Uses geometry from track reconstruction
- Material projected on surfaces



Work in progress to access hadronic interaction modules from Geant4 (more accurate than using Fatras)

- Not yet used in official production, still **under validation**
- Factor of 100 **faster** than Geant4

Current simulation performances

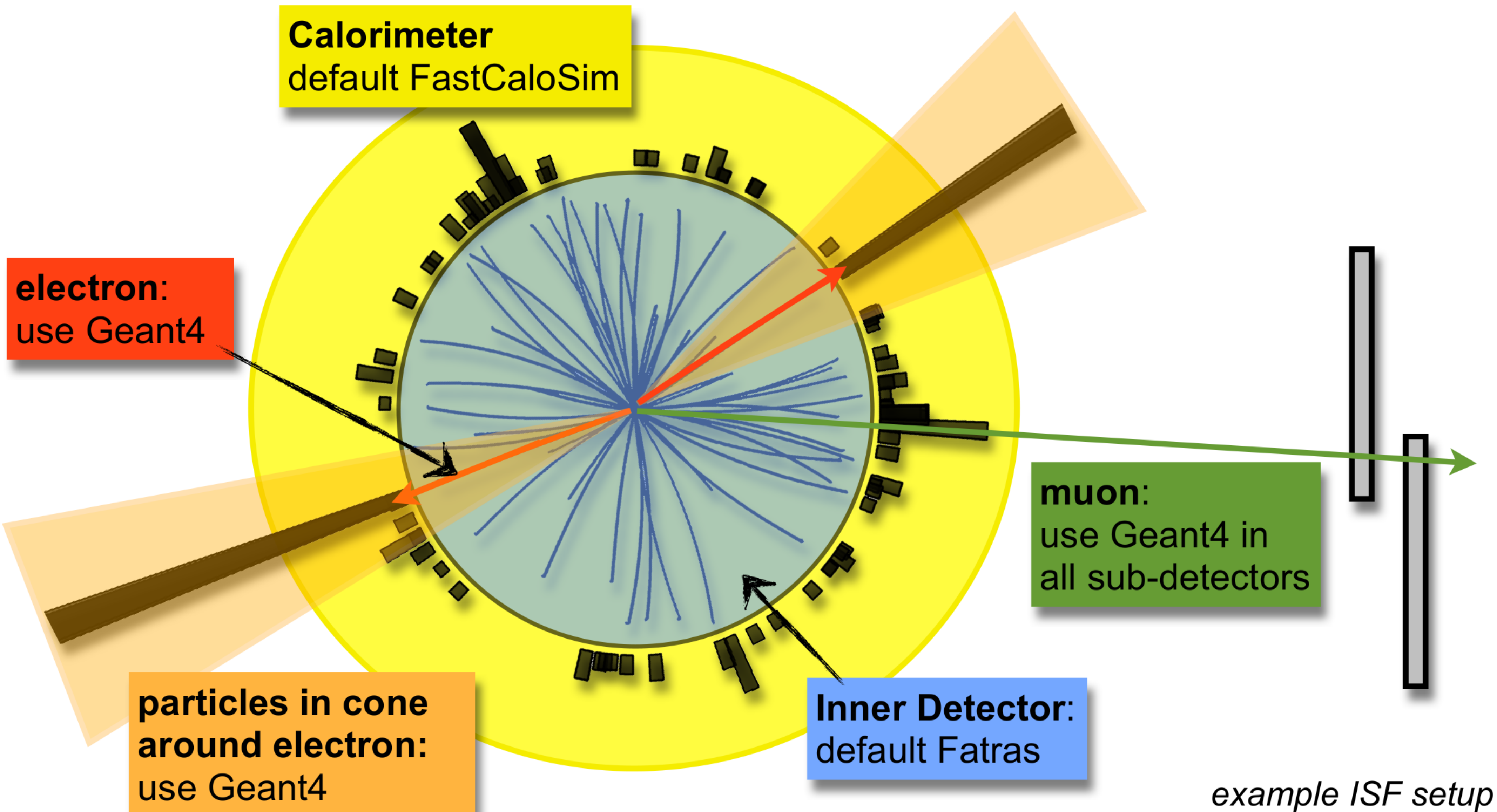


Number of events simulated in ATLAS for the MC12 campaign:

- 3.9 G => full simulation (Geant4 + frozen showers)
- 3.0 G => fast simulation (Geant4 + FastCaloSim)

Future developments: Integrated Simulation Framework

ISF scheme



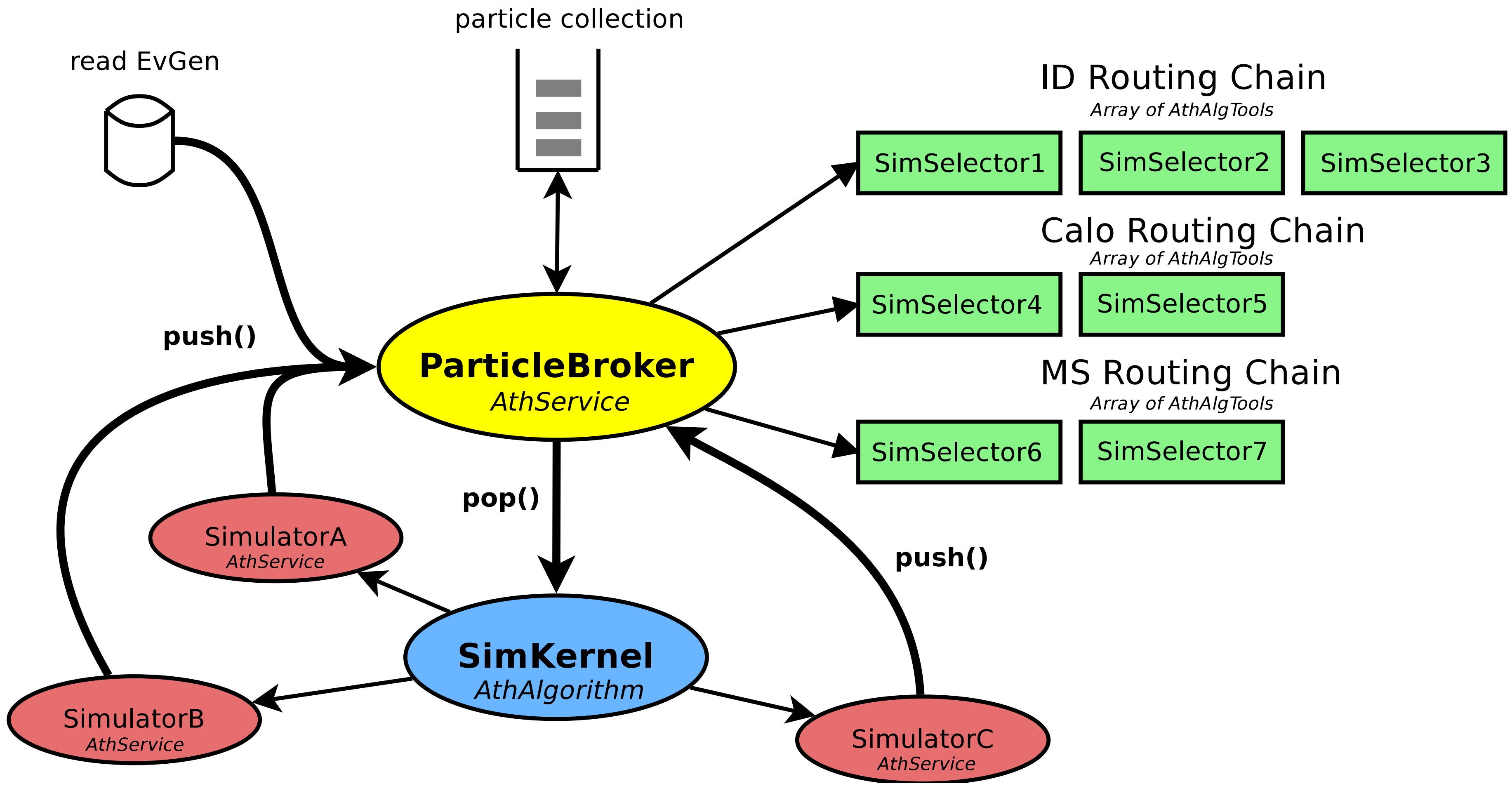
example ISF setup

- ***Idea***: use different simulation techniques for the same event, depending on region or particle type
- Main feature: **flexibility** with respect to particles => simulator assignment
- Designed to be compatible with multithreading and multiprocessing

ISF design

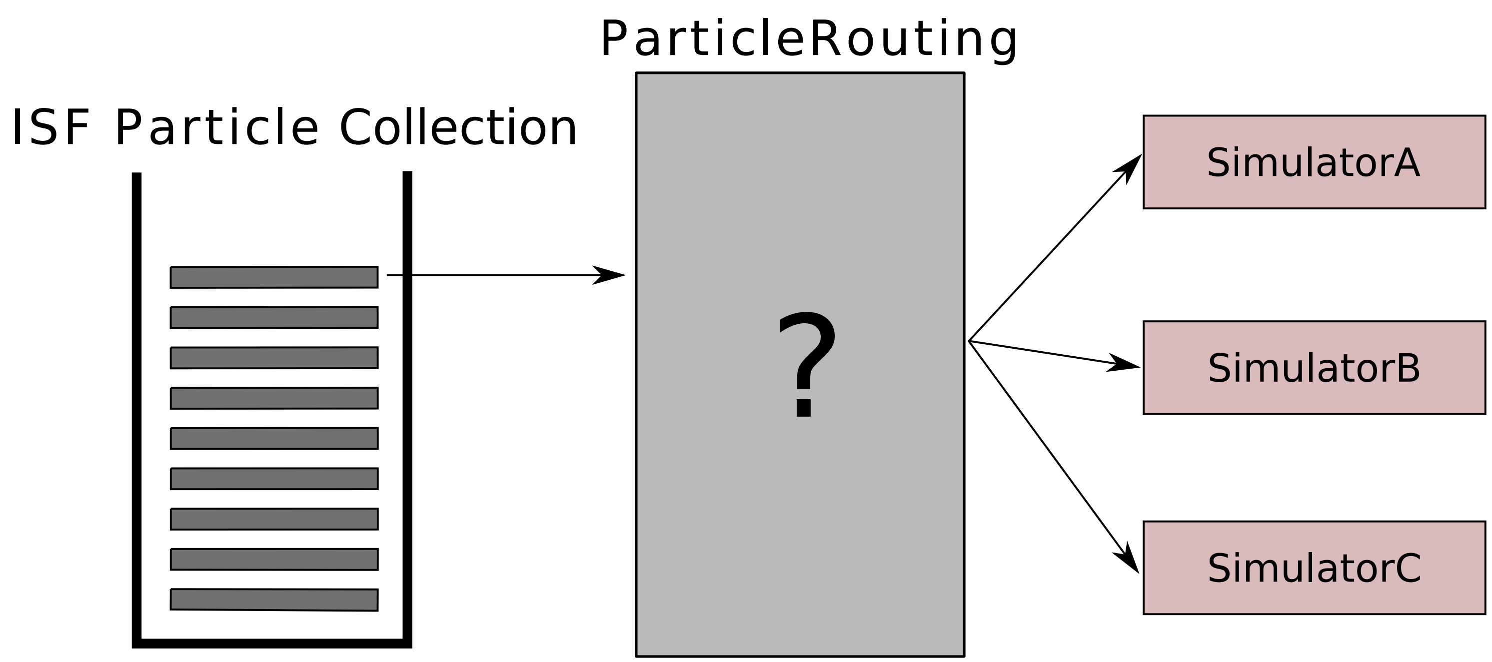
2 main components:

- **SimKernel** => implementation of main particle loop, sends particle to the different simulators
- **ParticleBroker** => functions as particle storage and determines which simulator is to be used
  - uses **RoutingChain** => one for each subdetector



More on particle routing

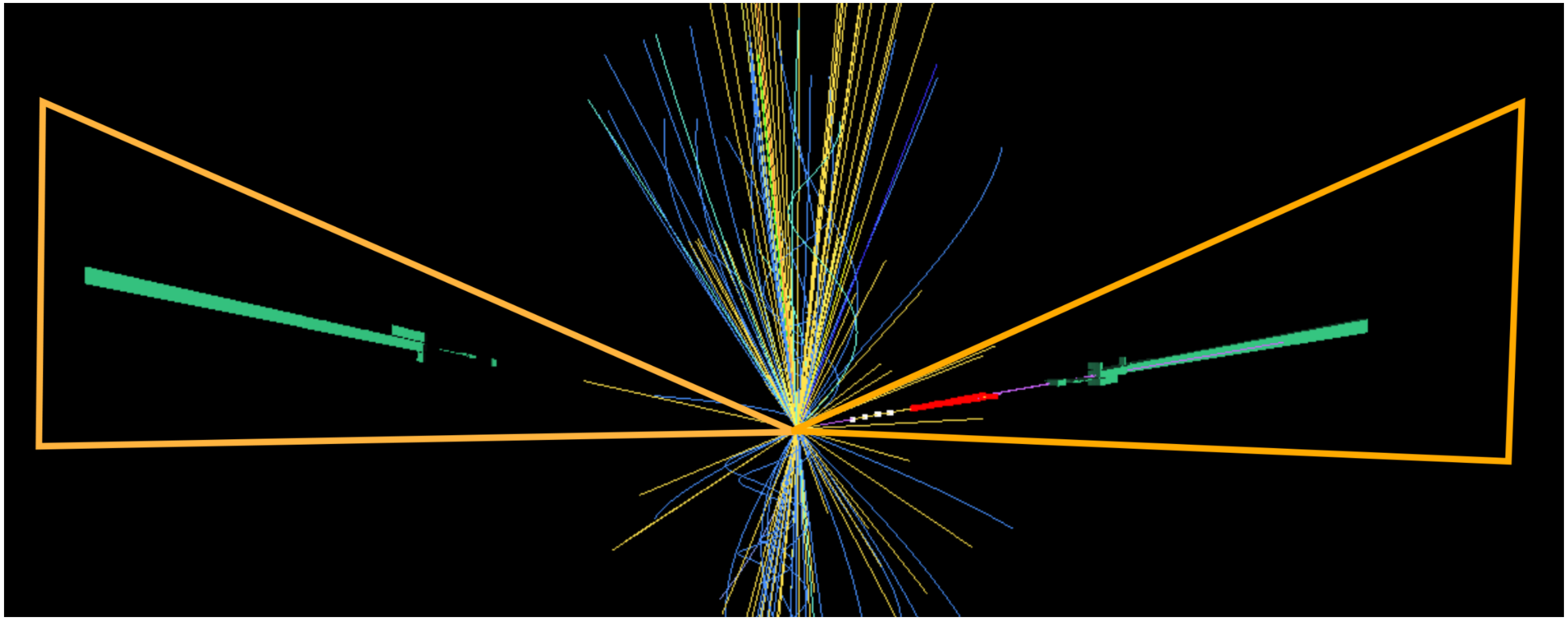
Aim: route particles through different simulators and subdetectors
=> need for bookkeeping



- Routing requirements
  - *Static rules*: use of particle type or kinematic properties
  - *Dynamic rules*: considering other particles in the event (cones)
  - Simple and intuitive configuration (no need to know ISF in depth)

ISF application: H→γγ

- H→γγ analysis needs high MC statistics for background shape evaluation
- Test gg→H→γγ events using ISF
- ***Method***: partial event simulation
  => only particles in cones around initial photons are simulated



Test of 4 different setups and comparison to understand time gain

ISF performance: H -> gamma gamma

ISF simulation setup	Speedup	Accuracy
Full Geant4	1	best possible
Geant4 with FastCaloSim	~25	approximated calorimeter
Fatras with FastCaloSim	~750	all subdetectors approximated
Fatras with FastCaloSim simulate only particles in cones around photons	~3000	all subdetectors approximated event simulated only partially

$gg \to H \to \gamma\gamma$ no pileup

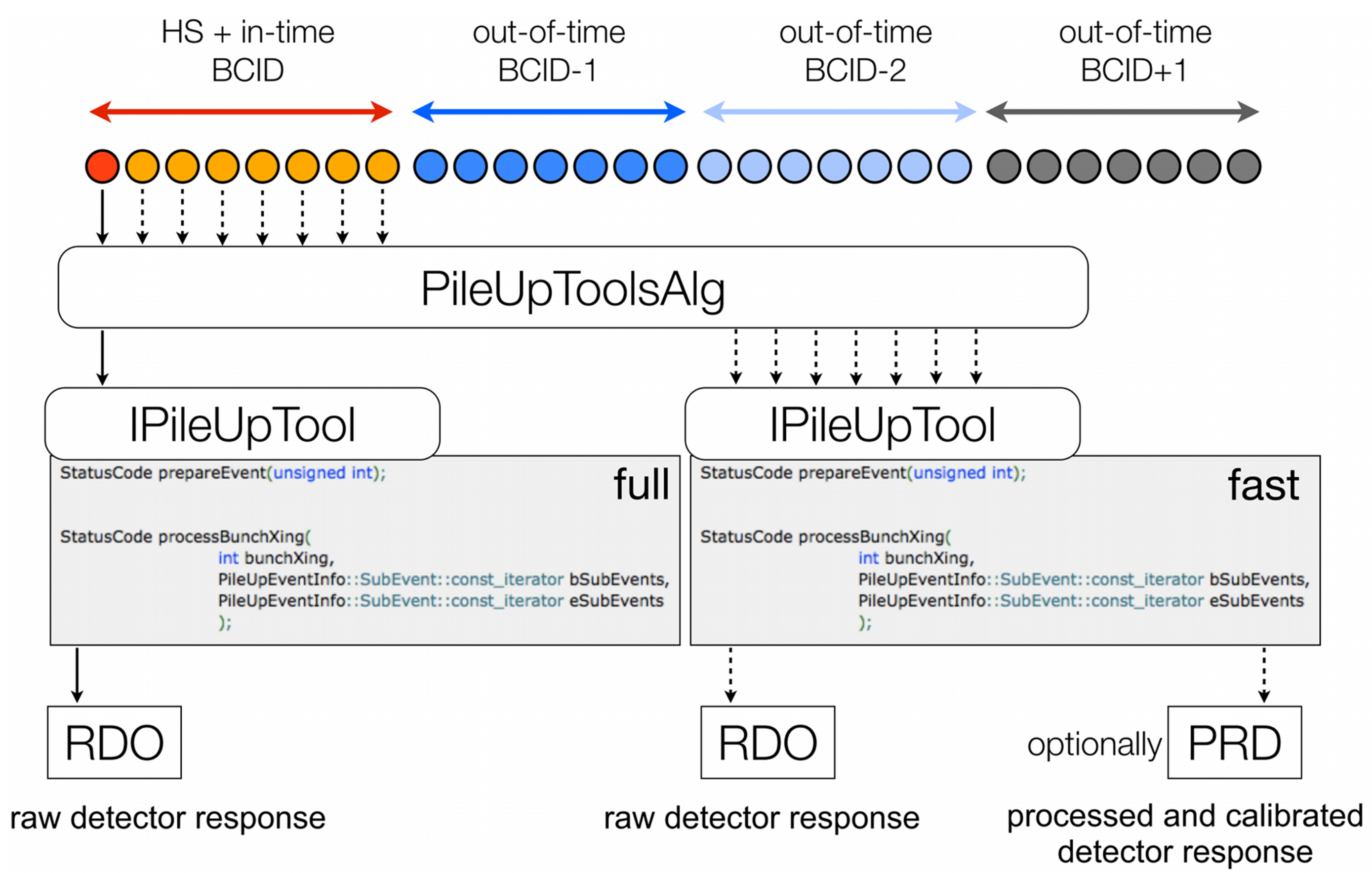
- Use of fast simulation => **significant speedup**
- Speed increased even further thanks to partial event simulation
  - helps in reducing output size

Fast Digitization and Reconstruction

Fast digitization

- Detector Simulation main bottleneck for current MC production
- Speedup of simulation up to ~3k times
  => *next bottleneck are Digitization and Reconstruction*
- Digitization time dominated by **Inner Detector** (~50%)
- Linear dependence on pileup

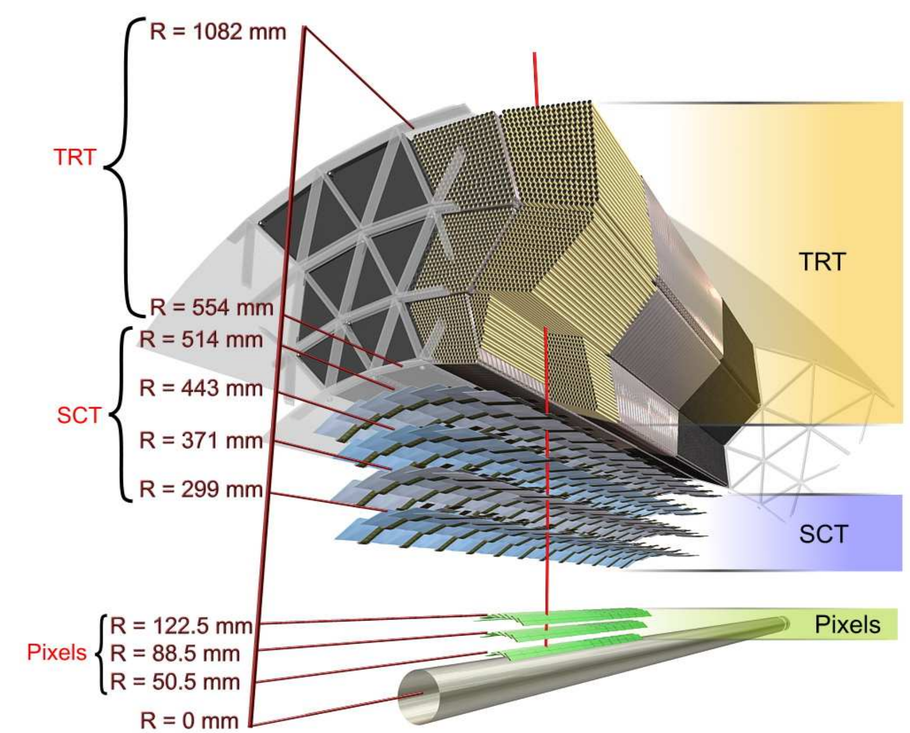
=> Fast Digitization and Reconstruction



HS: Hard Scatter (Event), BCID: Bunch Crossing Identifier, RDO: Raw Data Object, PRD: Prepared Raw Data

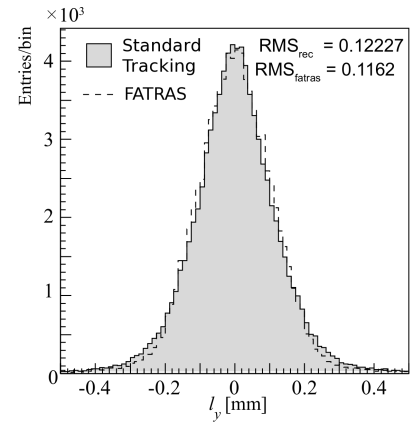
***Digitization**: from simulated hits to detector readout*
- *subdetector specific*
- *handles correct pileup treatment*

ID digitization => **2 technologies**:
- Silicon (Pixel and SCT)
- Transition radiation tracker

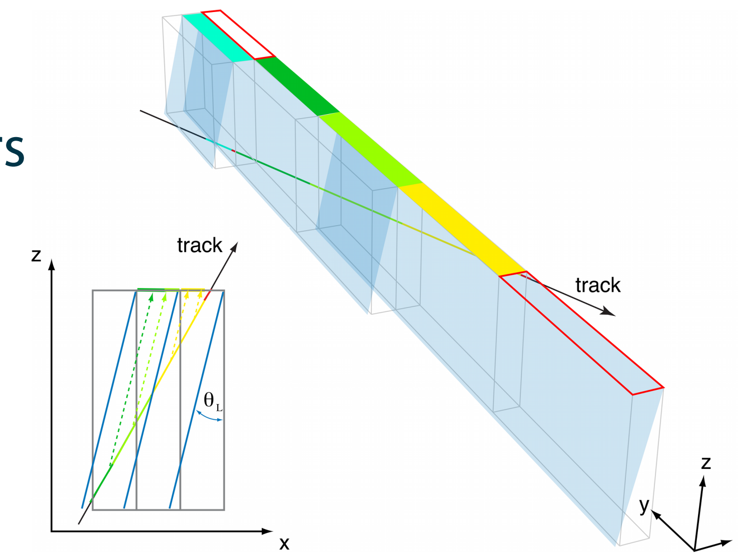


Silicon and TRT fast digitization

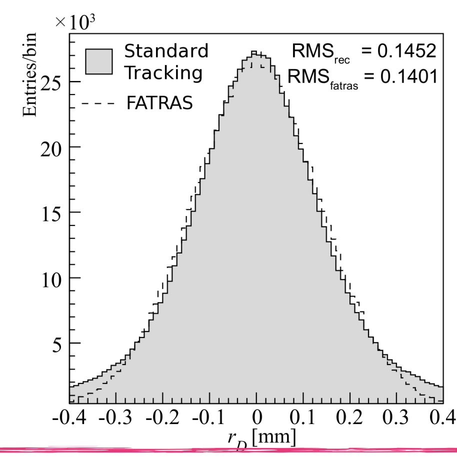
- Charge deposition estimated for each readout channel
- Simulated track length projected on readout surface
- Correct for Lorentz angle drift ($\vec{E}\times\vec{B}$)
- Smear to account for multiple scattering of drifting charge carriers



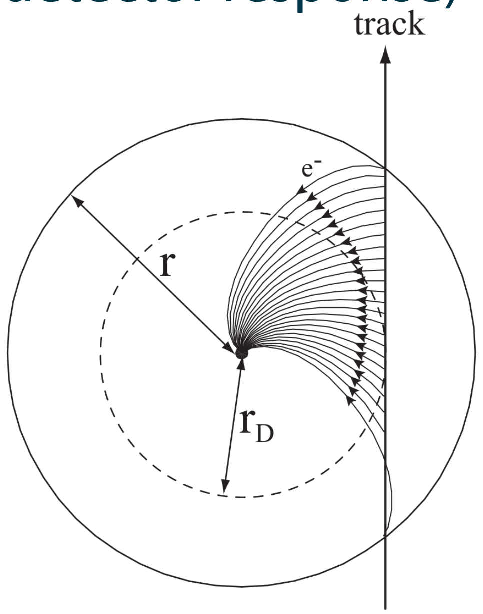
Comparison of pixel cluster position residuals l_y between historic FATRAS fast digitization and standard one



- From simulation hits => prepared raw data (PRD, processed and calibrated detector response)
  - evaluate closest approach radius
  - determine uncertainty of measurement
  - smear hit position with this information
- Parametrization of transition radiation response (allows for particle ID)



Comparison of TRT drift radius residuals r_D between historic FATRAS fast digitization and standard one

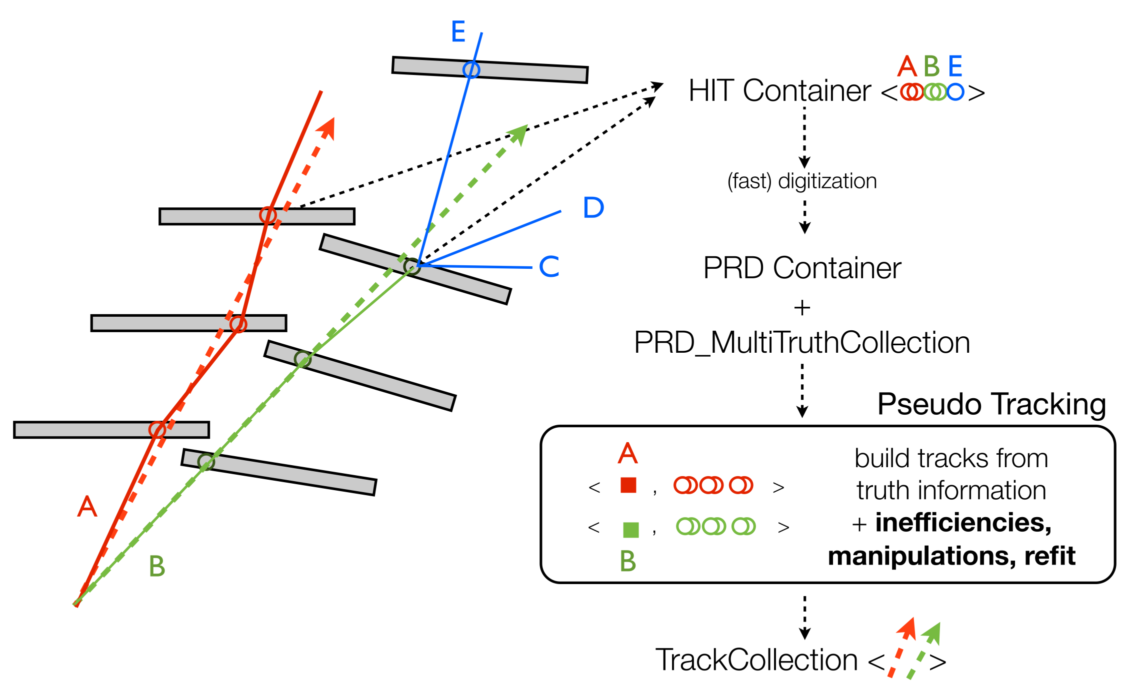


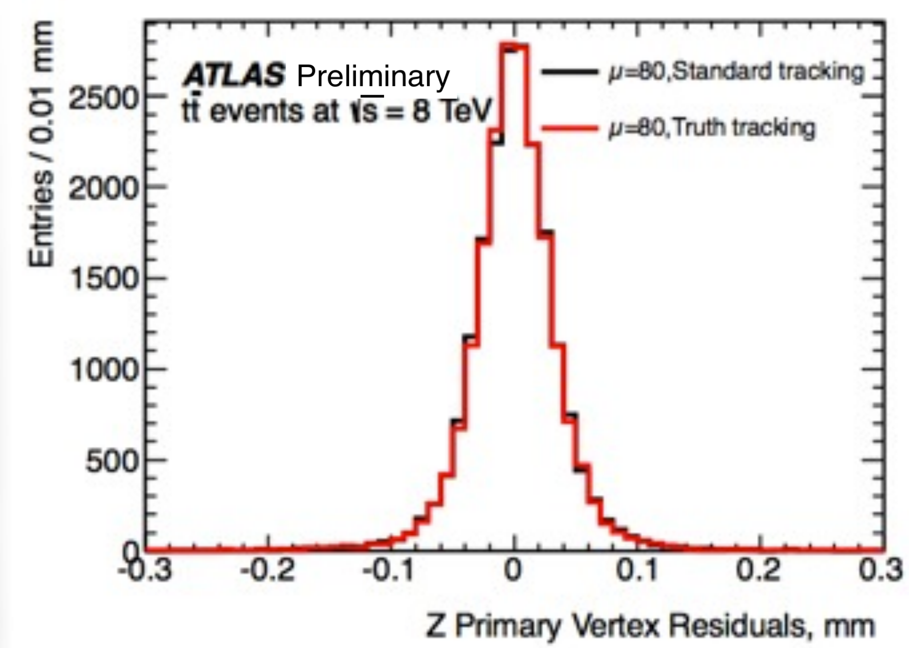
Fast reconstruction: description

ID reconstruction most time consuming because of combinatorics in pattern recognition

=> **Fast tracking**:

- Seed track from MC truth
- Skip most time consuming steps:
  - pattern recognition
  - track seeding
  - ambiguity treatment
- reconstructed track fit to hits from truth
- high efficiency at high pileup

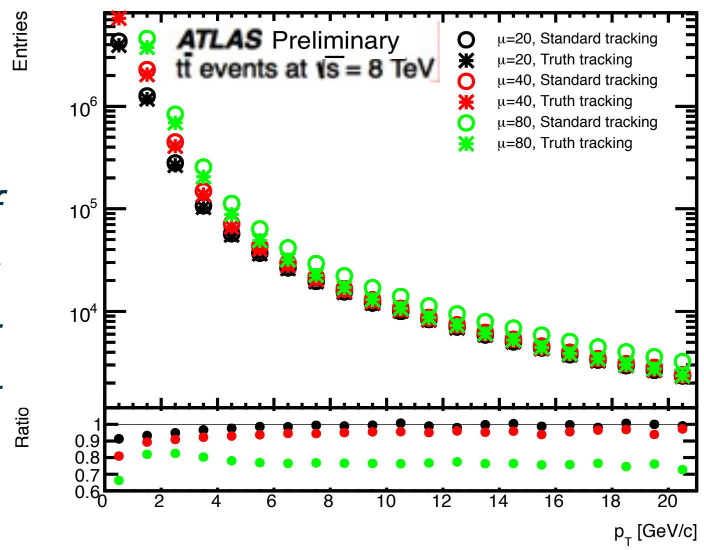


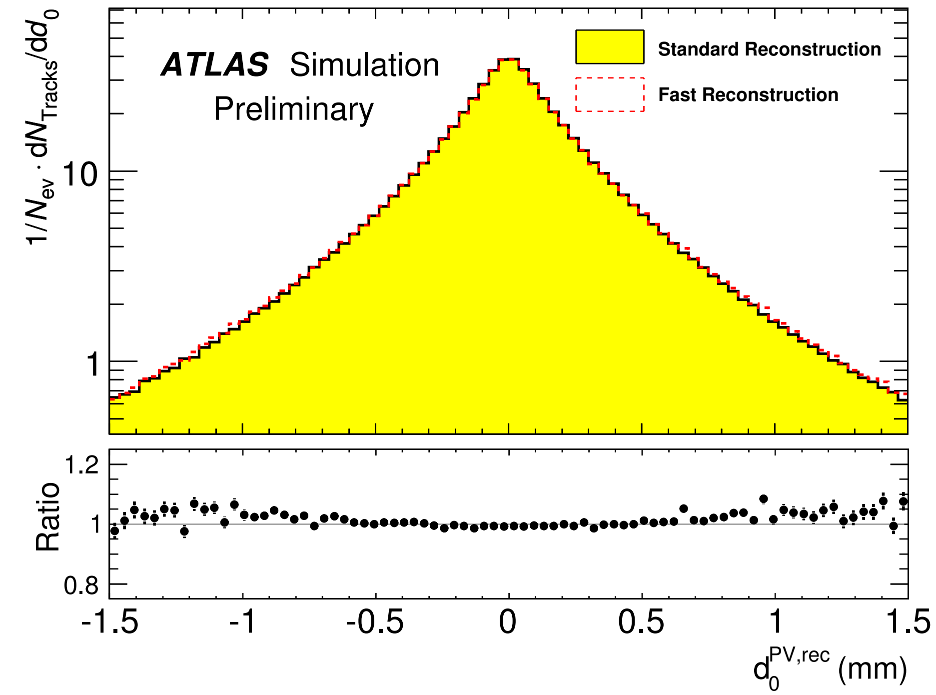


Comparison of the longitudinal primary vertex resolution using tracks found with the standard tracking and truth tracking at μ=80, where μ represents the average number of collisions per bunch crossing.

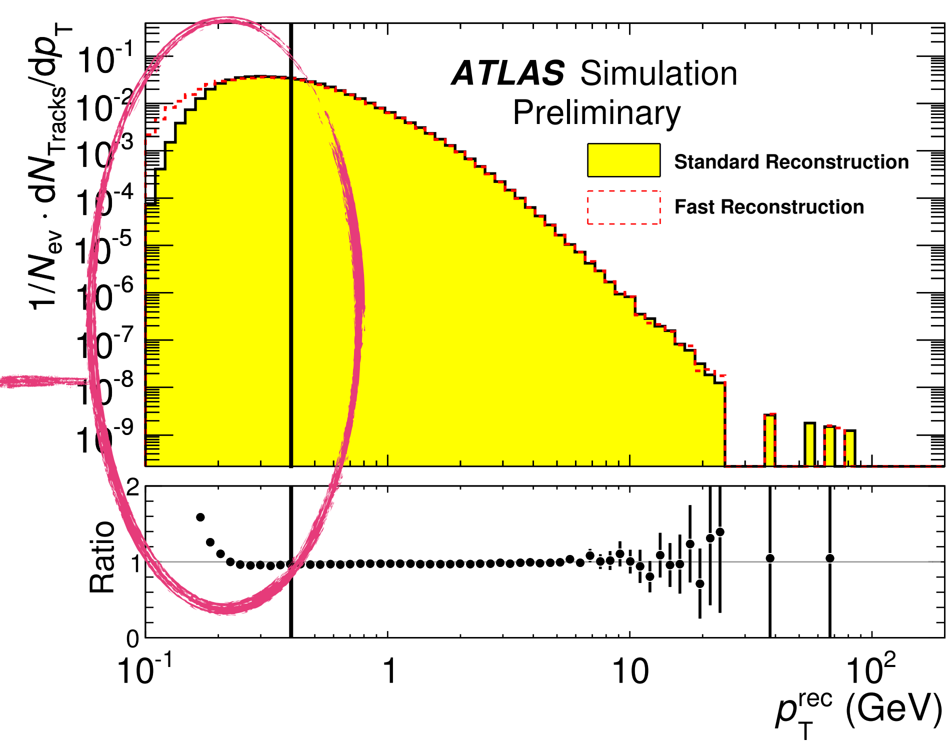
Fast reconstruction: performance

8 TeV tt Simulation Transverse momentum spectrum of reconstructed tracks with standard and fast tracking for different pileup scenarios





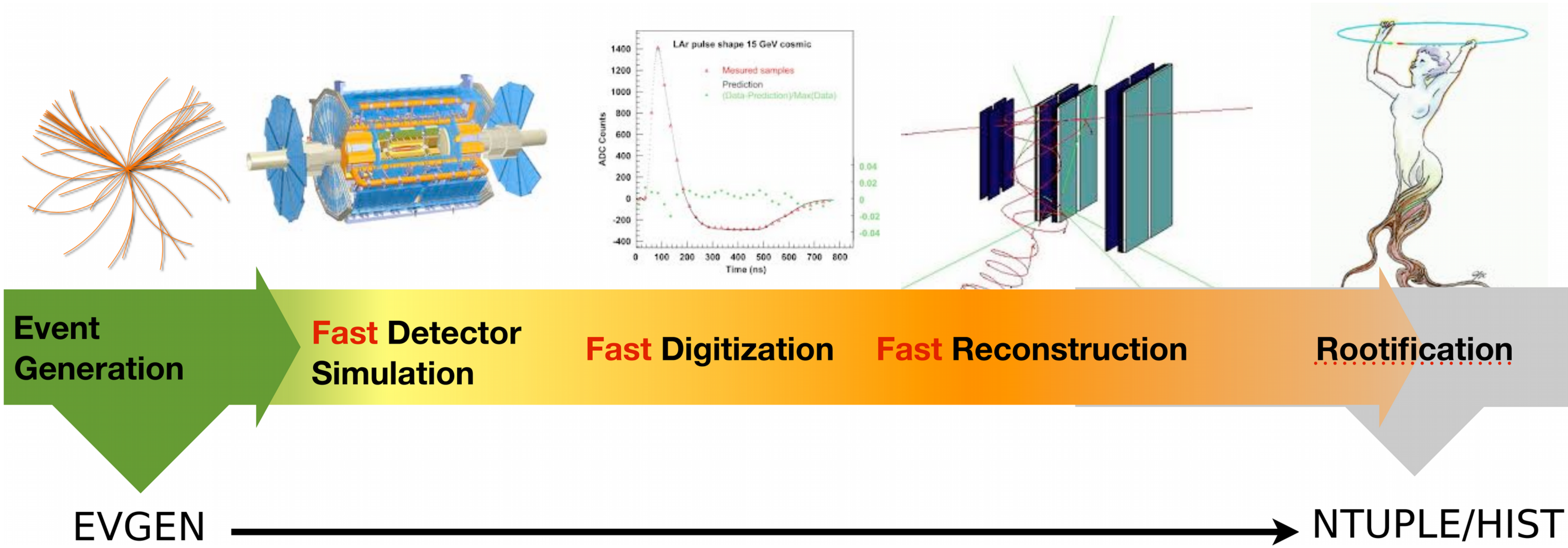
- Good agreement with standard Reconstruction
- Significant speedup
- Difference at low momentum not significant
  - p_T>400 MeV for standard ATLAS data and MC processing
- Fast reconstruction with better performance
  - inefficiency factor taken into account for low p_T particles



8 TeV minimum bias Simulation

Full Production chain, conclusions and outlook

Fast ATLAS MC production



- Evgen to ROOT in one go
  - I/O writing next bottleneck after Fast Sim/Digi/Reco
  - **no intermediate output** (minimisation of I/O overhead and storage disk space)
  - Fast Simulation + Fast Digitization + Fast Reconstruction
- Estimated time per event: **a few seconds**
  - possibility for large scale MC production with substantially lower resources

Conclusions and outlook

- ATLAS is developing and maintaining both full and fast simulation
- A lot of work ongoing on the **Integrated Simulation Framework**
  - dynamic use of different Simulation technologies based on event characteristics
- Fast digitization for Silicon and transition radiation tracking technologies
- Fast reconstruction => tracking based on seeding from MC truth

Fast MC production chain:

- combination of Fast Simulation, Digitization and Reconstruction
- 4-vectors → ROOT in one step
- only a few seconds necessary to process an event

BACKUP

Static routing

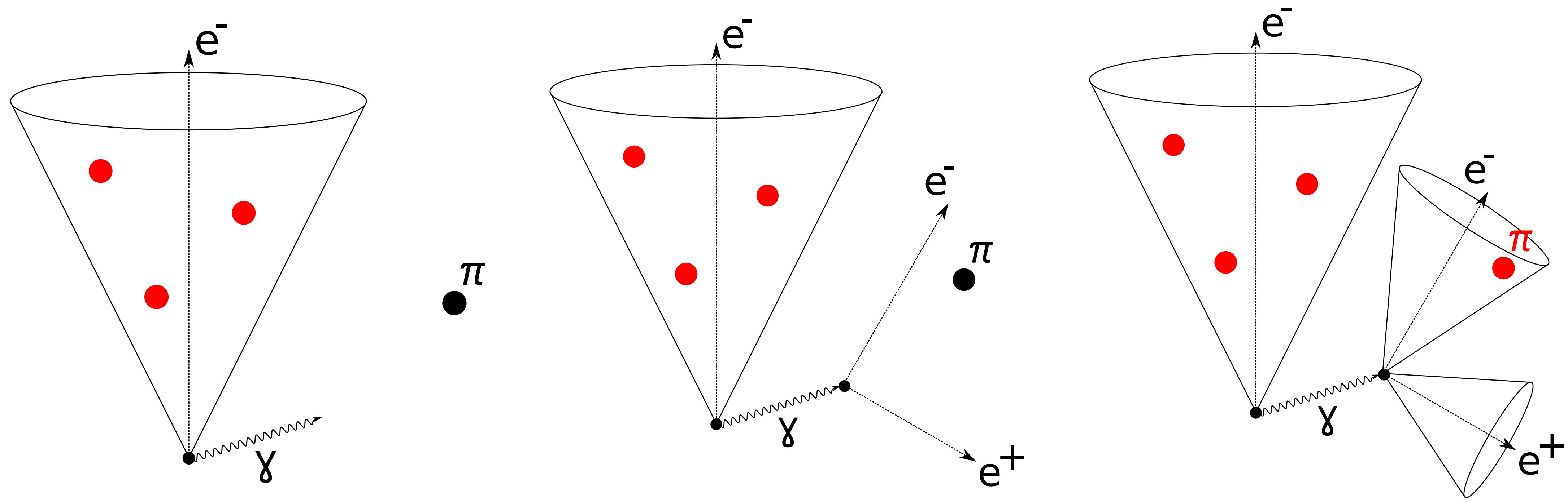
- Example:
  - selection based on **particle type**: all electrons to *simulator A*
  - selection based on **kinematic** characteristics: all p_T>30GeV particles to *simulator B*

- Advantages:
  - order independence
  - intuitive method
  - consistent through the full process
  - each particle simulated only once

MEMO: possible contradictions between selector decisions
=> need to define selectors in priority list

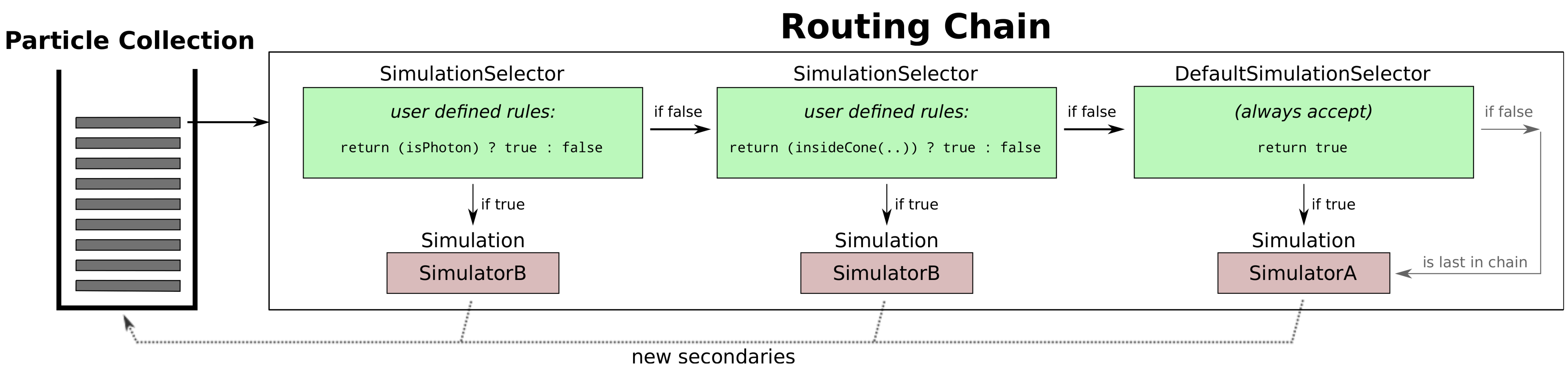
Dynamic routing

- Example:
  - dynamic cone selector:
    - a cone is registered for every new electron in the event
    - all particles inside the cones simulated with simulator A



- **Warning**:
  - decision on π dependent on simulation order
  - if simulated before conversion => selector decision is **inconsistent**

Routing chain



SimulationSelector dynamic rules can be dependent on initial generated particles

i. a particle is taken from particle collection
ii. the SimulationSelectors are asked to make a decision on the particle selection in a specific order
iii. if the particle is not taken by a SimulatorSelector, it is checked by the next in the chain
iv. the first simulation selector to accept the particle decides that is has to be simulated with the simulator attached to the specific selector