Dynamic Stack CORSIKA
Or Multiprocess Server

Kevin Meagher
Global Fit Workshop
Tokyo, Japan
14 September 2019

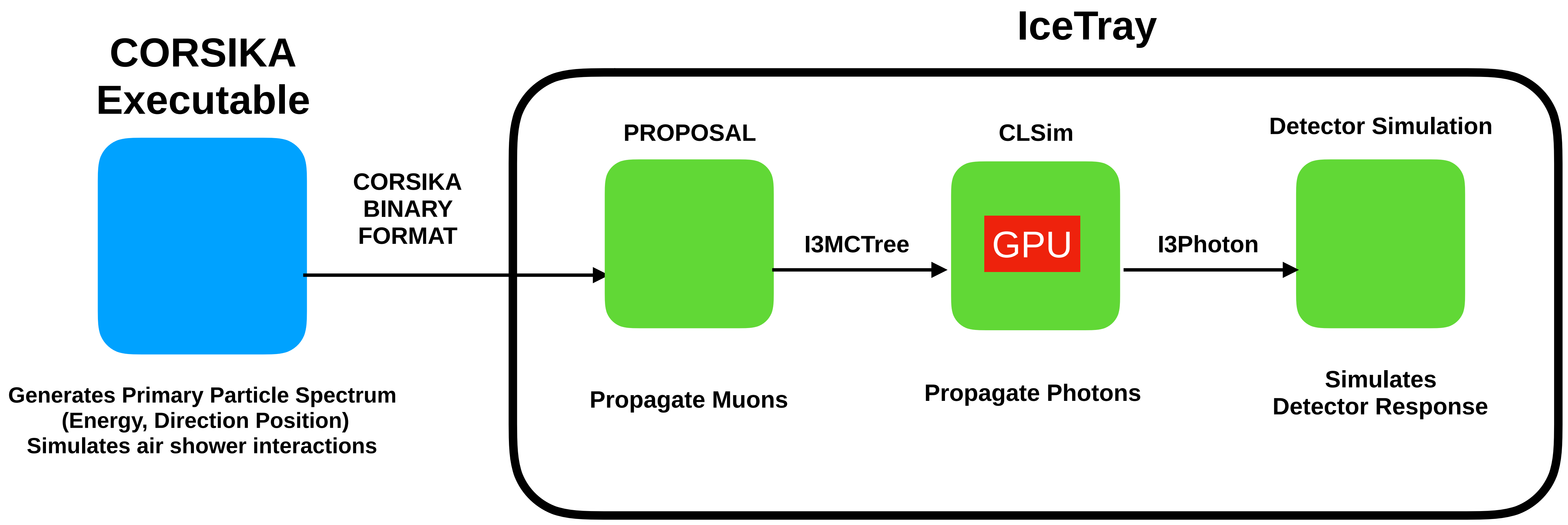
WIPAC
WISCONSIN ICECUBE
PARTICLE ASTROPHYSICS CENTER

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Overview

- Current state of IceCube Simulation with CORSIKA
- Advantages of Dynamic Stack and Server CLSim
- Current status of software development
- Topics of Hackathon/Discussion

How CORSIKA currently works



I3Photon = Lots of Memory
I3MCPE = Uses less memory

CORSIKA files are generated by a separate CORSIKA binary
IceTray then processes in a linear fashion by each module
First Muons are propagated by by PROPOSAL,
Then Photons are processed by CLSim using the GPU
The entire I3Photon sequence is stored in memory before being converted to the binned I3MCPE format

Why is CORSIKA so hard to produce?

Low Energy:

- CORSIKA shower files are created separately and transferred at the start of the job which saturates IO

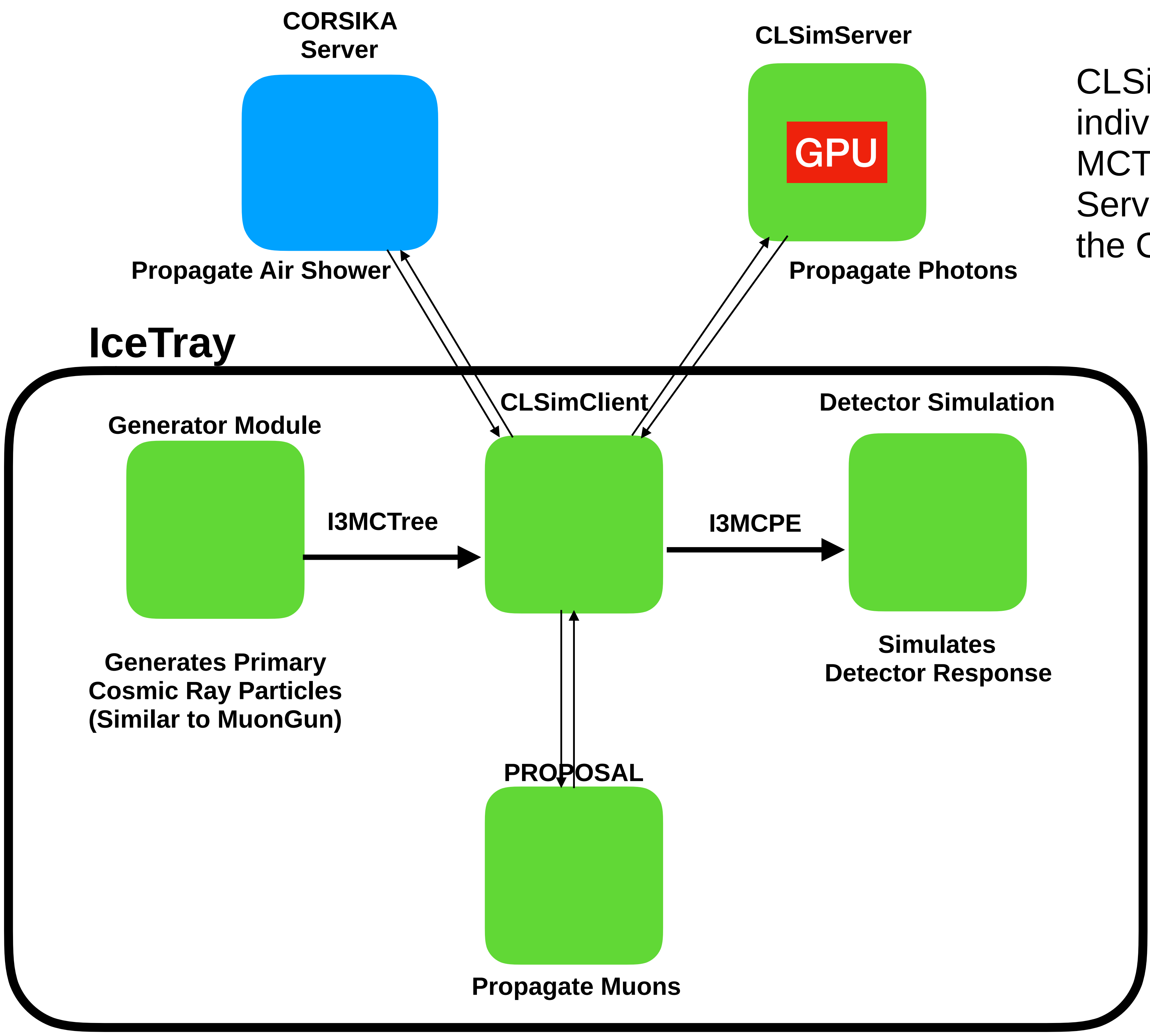
Medium Energy:

- CORSKIA showers need more CPU than GPU which is an inefficient use of our cluster resources (CPUs sit idle while waiting for GPUs to finish)

High Energy:

- Large showers push the memory limits on nodes. CLSim needs to store the entire event (both the MCTree and I3Photons) in memory. Power-law statistics require that we have to allocate memory for very rare events.

How Multiprocess Server CLSim works



CLSimClient passes individual particles from the MCTree to the CORSIKA Server, to PROPOSAL to the CLSimServer

I3MCPE are created directly from the output of each individual CLSim propagation Saving memory

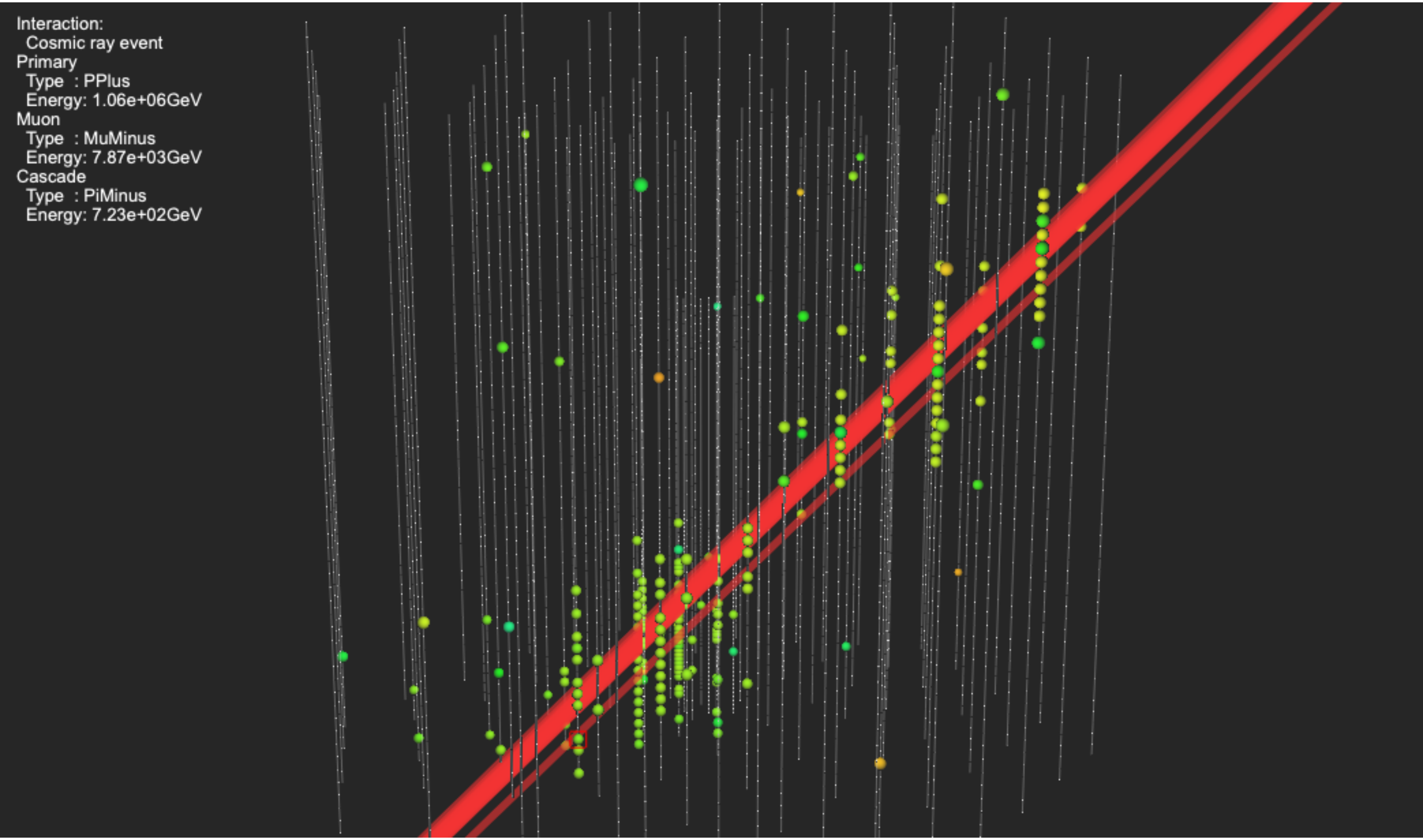
What do we gain from this?

No Need to break up CORSIKA jobs by energy:

- Low Energy:
  - No need to generate CORSIKA files separately (prevents IO bottleneck)
- Medium Energy:
  - Multiple instances of IceTray can share a CLSimServer resulting in better CPU utilization
- High Energy:
  - Individual particles are passed from CORSIKA to PROPOSAL to CLSim and binned I3MCPE are made for each particle from CLSim rather than the entire event. This significantly reduces the memory footprint
  - Multiple instance of IceTray can run in the same cluster job

Event More Benefits: Oversampling and Other Tricks

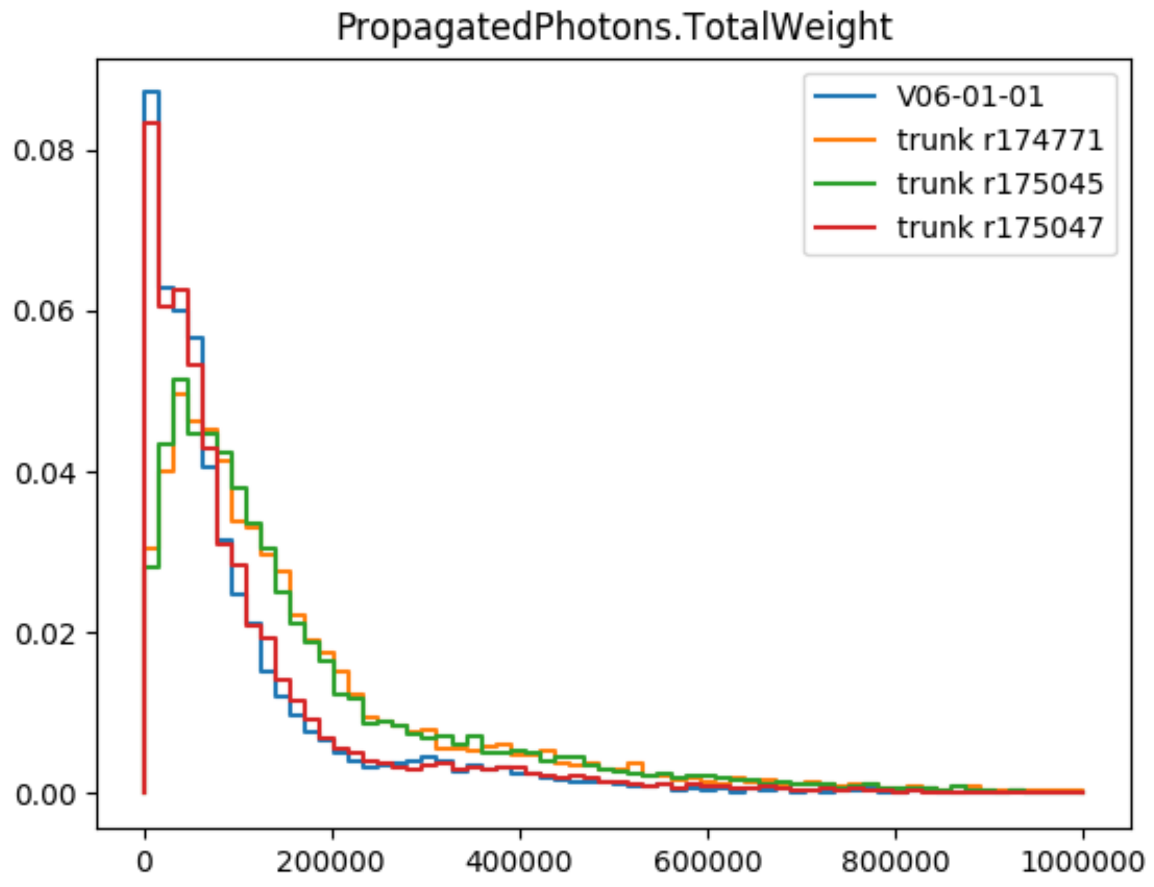
- Cosmic ray air shower primary can be generated according to arbitrary spectral and spatial distributions
  - Generate primaries directly on the detector cylinder before cosmic ray propagation (similar to MuonGun)
  - Oversample IceCube muon “Lanes”
- Different CORSIKA configuration cards can be sent to different events
  - Only populate EM component if the shower hits IceTop.
  - Set muon energy threshold based on the inclination of the shower
- CORSIKA propagation of a shower can be under-sampled based on shower development
  - Kill events with low leading energy muon
- Oversample coincident events
  - e.g. 1 for coincident showers, 10^{-4} for single showers



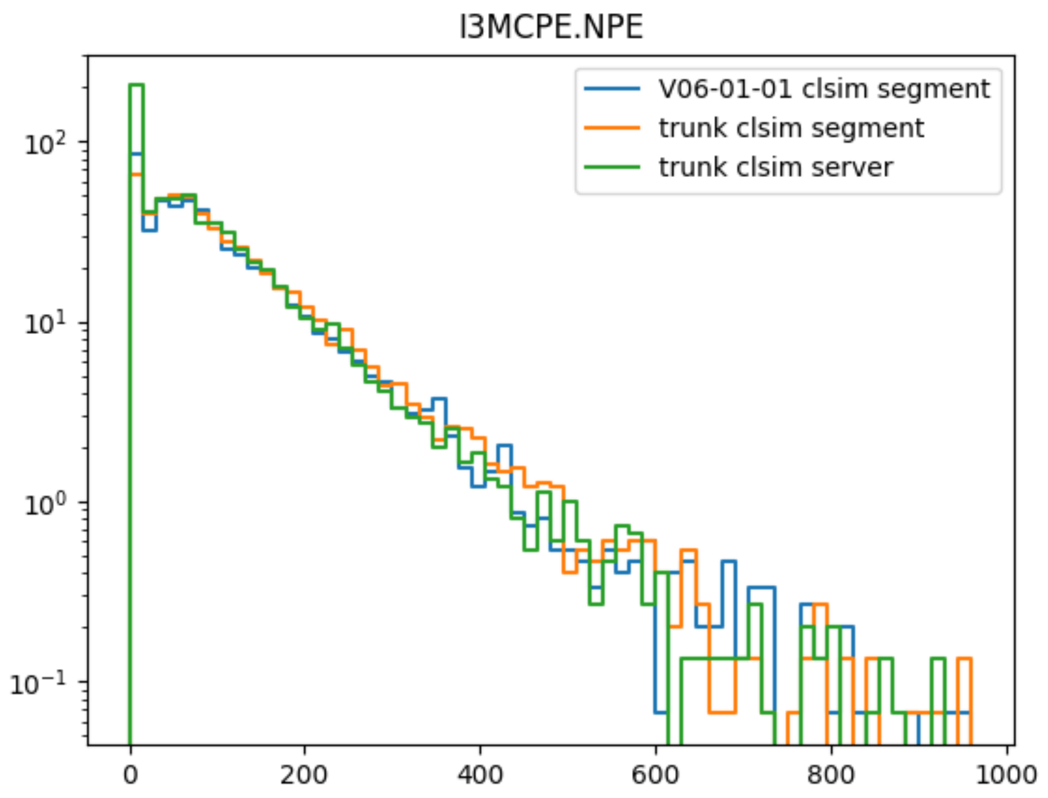
Interaction:
  Cosmic ray event
Primary
  Type : PPlus
  Energy: 1.06e+06GeV
Muon
  Type : MuMinus
  Energy: 7.87e+03GeV
Cascade
  Type : PiMinus
  Energy: 7.23e+02GeV

Fixed bug in PROPOSAL

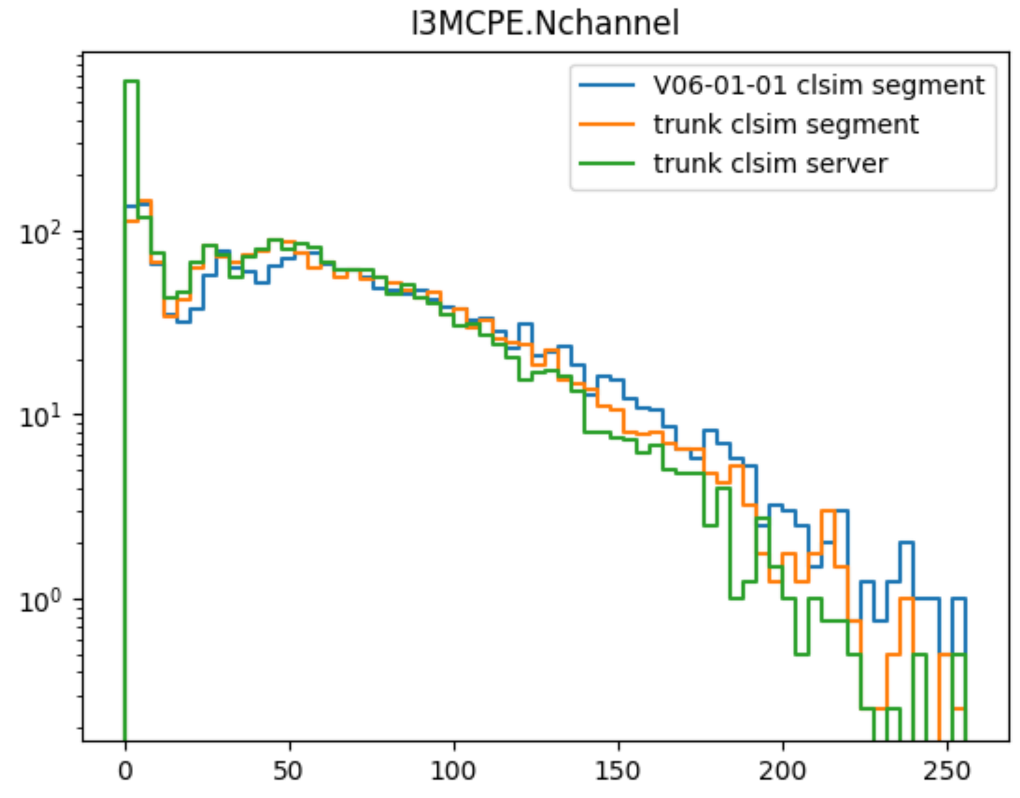
- In the past, PROPOSAL identified stopping muons with negative energy but new version stored rest mass
- This was caused stopping muons to be regenerated, effectively doubling the amount of light in the detector
- Bug was tracked down by Jakob van Santen
- Fixed by Jan Soedingrekso in r175047

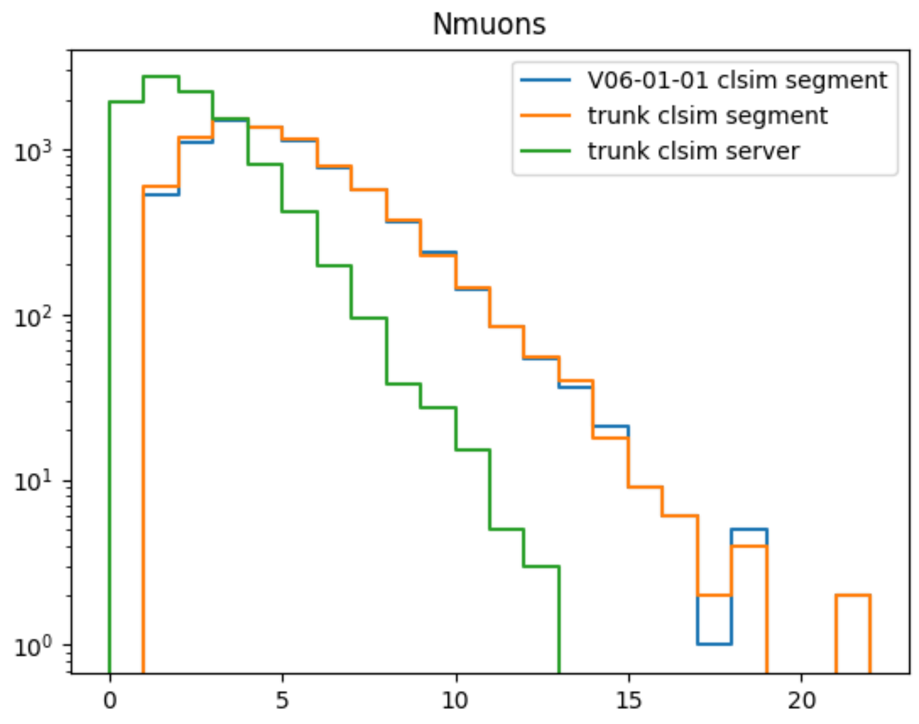


Monoenergetic Single Direction Cosmic Rays



Pulses observed in DOMs match last release of simulation software





This was accomplished by simulating less muons

Current Status

- Monoenergetic showers shows excellent agreement with previous versions of simulation
- Full spectrum 2π job just finished on cluster: results soon
- To Do: implement and optimize multiple processes in same cluster job
- Develop appropriate weighing: put all information in “S” frame and save it in hdf5 file
- Determine oversampling/biasing scheme

Topic for Hackathon:

How to oversample simulation to maximize usefulness in analyses?

Currently proposed:

- Oversample events with high leading energy muon
- Oversample DeepCore “Lanes”
- Oversample Coincident showers
- Whatever you can think of

Open Souce nuflux

- Neutrino-flux and NewNuFlux are still being used in a substantial number of analyses
- Should be replaced with MCEq for future analyses
- I was asked to prepare it for release as an IceCube open source project
- Renamed to “nuflux” to comply with PEP8
- No changes to any numerical results
- Improvements: build system, documentation, remove segfaults, unit tests, example scripts, etc…
- Add a repository of IceCube spectra text files
- Talk to me at hackathond or see talk in Software II on Monday at 14:45

Backup

PROPOSAL Bug

Earlier versions of PROPOSAL/MMC used the sign of the final energy as a flag to designate stopping muon track

```
[I3MMCTrack = [
 (xi, yi, zi, ti, Ei) = (404.223 ,-0.996252 ,800 ,4522.09 ,0.338673)
 (xc, yc, zc, tc, Ec) = (-44.3509 ,-0.849819 ,25.6933 ,5404.34 ,0)
 (xf, yf, zf, tf, Ef) = (-522.694 ,-0.693668 ,-800 ,5404.34 ,-1330.06)
 Elost = 0.338673
```

Negative Sign

Sept 2018 PROPOSAL received a major update, at that update the final state for stopping muons was changed to the rest mass

```
[I3MMCTrack = [
 (xi, yi, zi, ti, Ei) = (452.997 ,64.3785 ,800 ,4435.44 ,50.5399)
 (xc, yc, zc, tc, Ec) = (343.927 ,64.9432 ,611.144 ,6928.48 ,0.105658)
 (xf, yf, zf, tf, Ef) = (343.927 ,64.9432 ,611.144 ,6928.48 ,0.105658)
 Elost = 50.4343
```

Muon Rest Mass

See Bug #2340 for details

Current Status

- CORSIKA binary compiled with dyadic stack c++ extensions from Dominic Baack and Jakob van Santen was successfully compiled on cluster with py3-v4 cvmfs (requires gcc>5.2 and boost>=1.64)
- Server CLSim branch was merged into trunk (Jan 2019)
- Custom Primary generator passed to CORSIKA finished
- Bug in PROPOSAL Fixed 14 August 2019
- Cluster is currently producing Dynamic Stack/multi process CORSIKA for evaluation