IceCube Computing Slide Deck

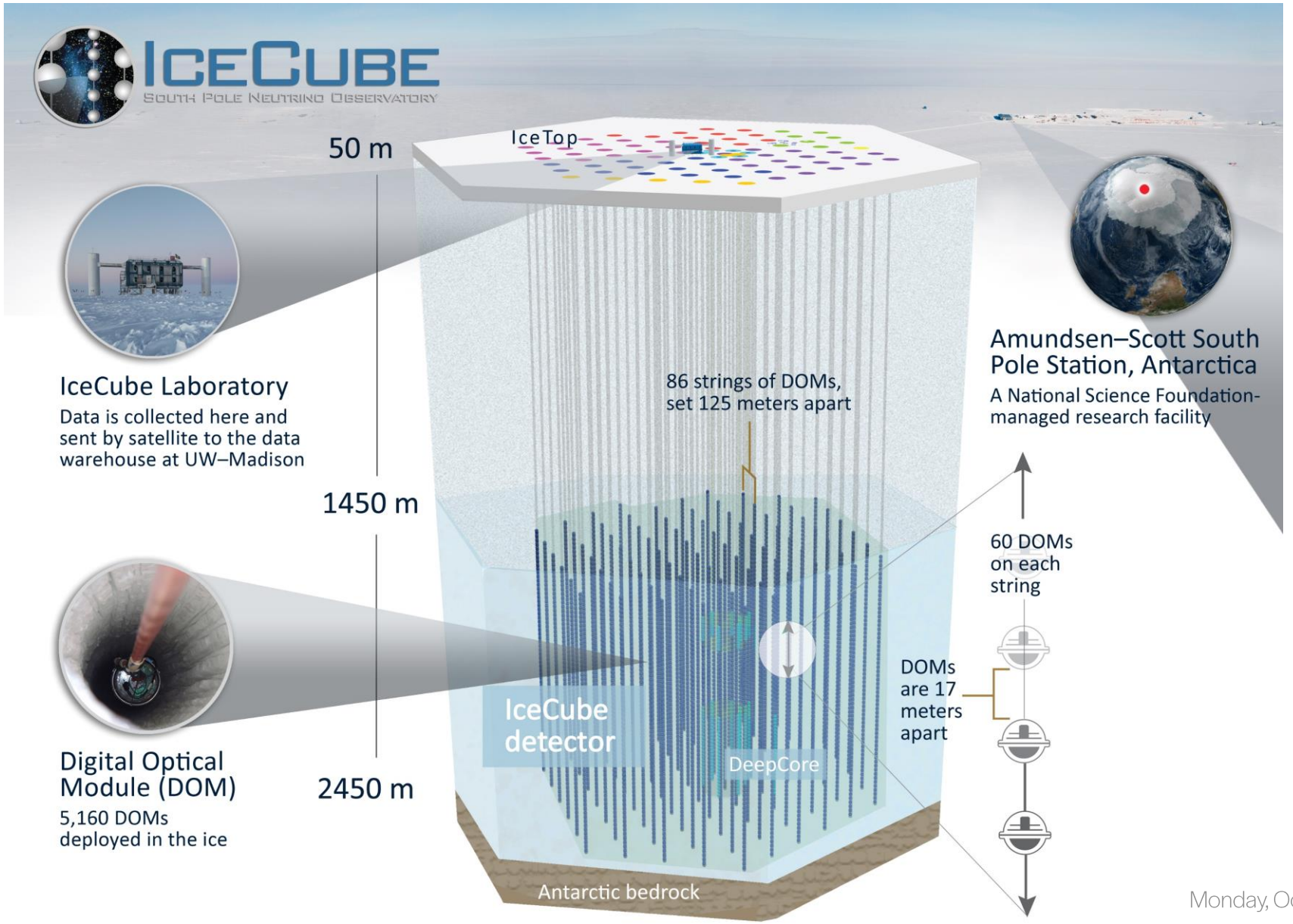
Benedikt Riedel
UW-Madison

Conference
10 October 2022

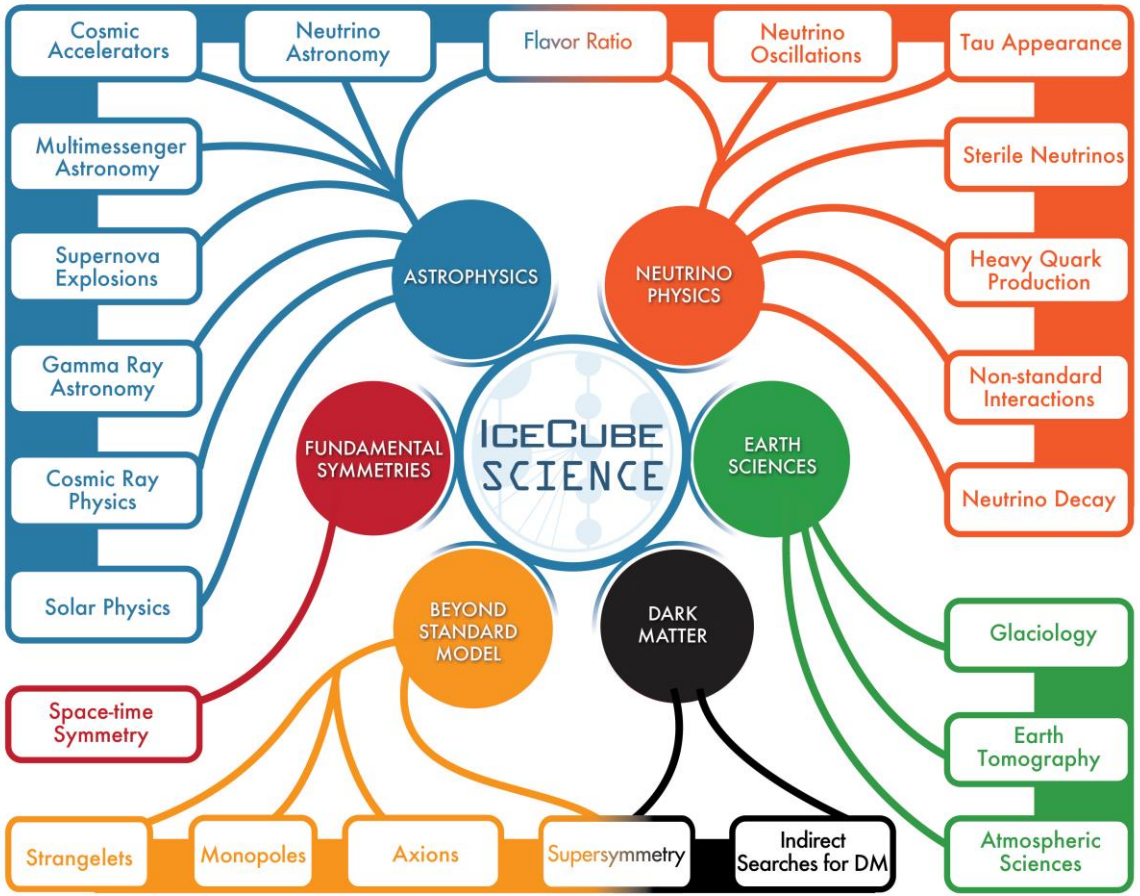
IceCube
South Pole Neutrino Observatory

Science

IceCube

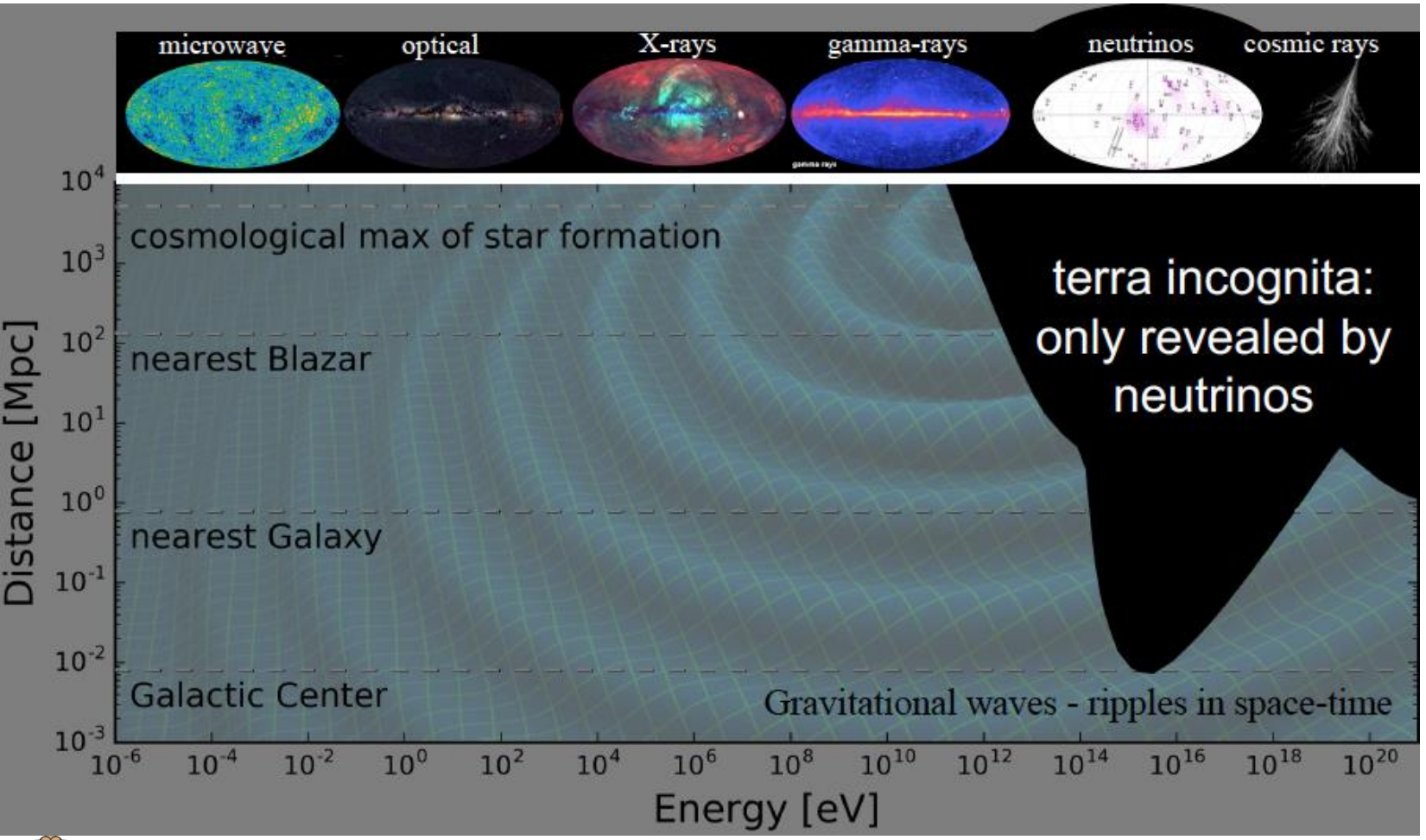

IceCube
South Pole Neutrino Observatory
50 m
IceTop
IceCube Laboratory
Data is collected here and sent by satellite to the data warehouse at UW–Madison
1450 m
86 strings of DOMs, set 125 meters apart
Amundsen–Scott South Pole Station, Antarctica
A National Science Foundation-managed research facility
60 DOMs on each string
DOMs are 17 meters apart
IceCube detector
DeepCore
Digital Optical Module (DOM)
5,160 DOMs deployed in the ice
2450 m
Antarctic bedrock

IceCube Science



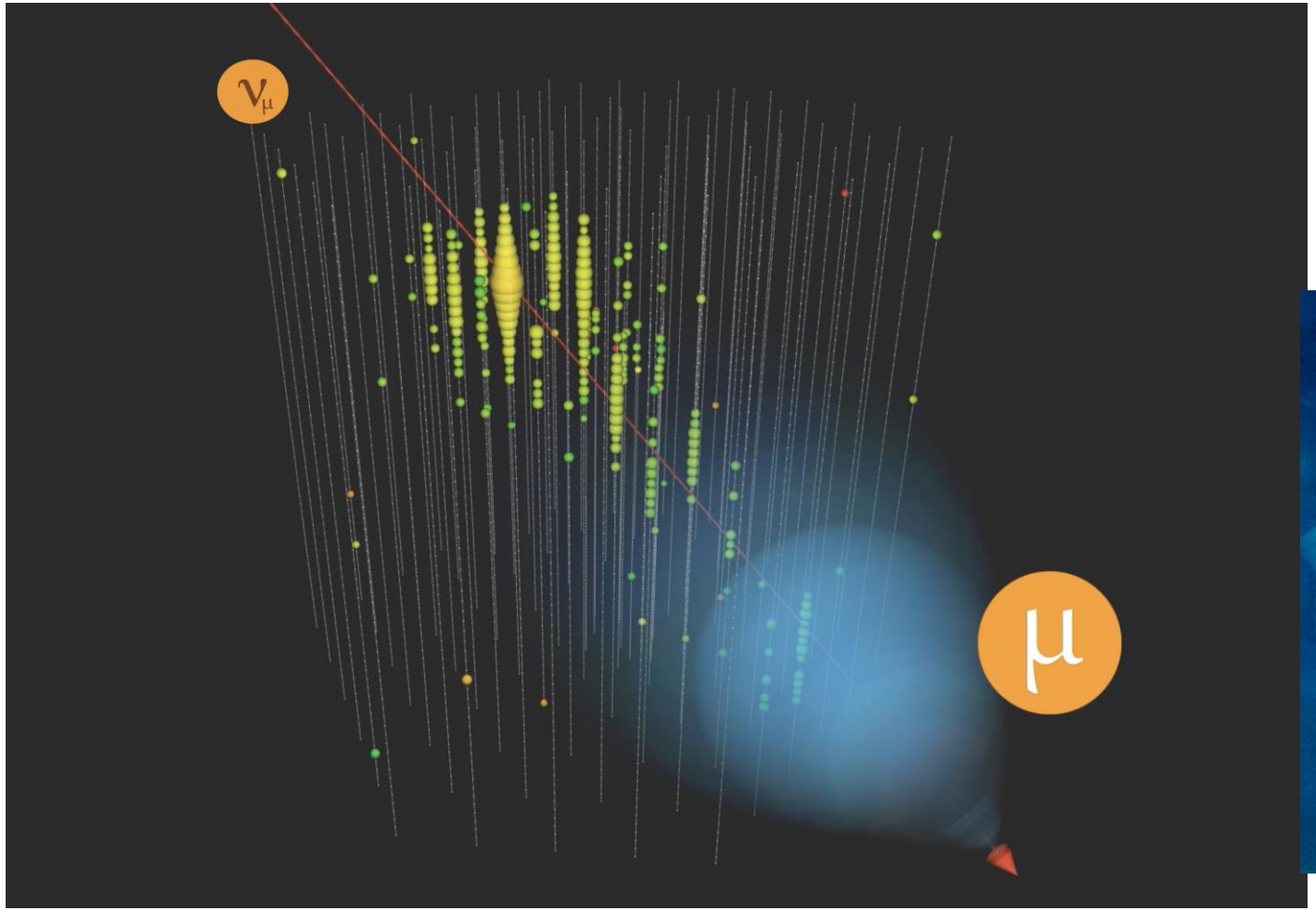
- Novel instrument in multiple fields
- Broad science abilities, e.g. astrophysics, particle physics, and earth sciences
- Lots of data that needs to be processed in different ways
- Lots of simulation that needs to be generated

IceCube Science – Why neutrinos?

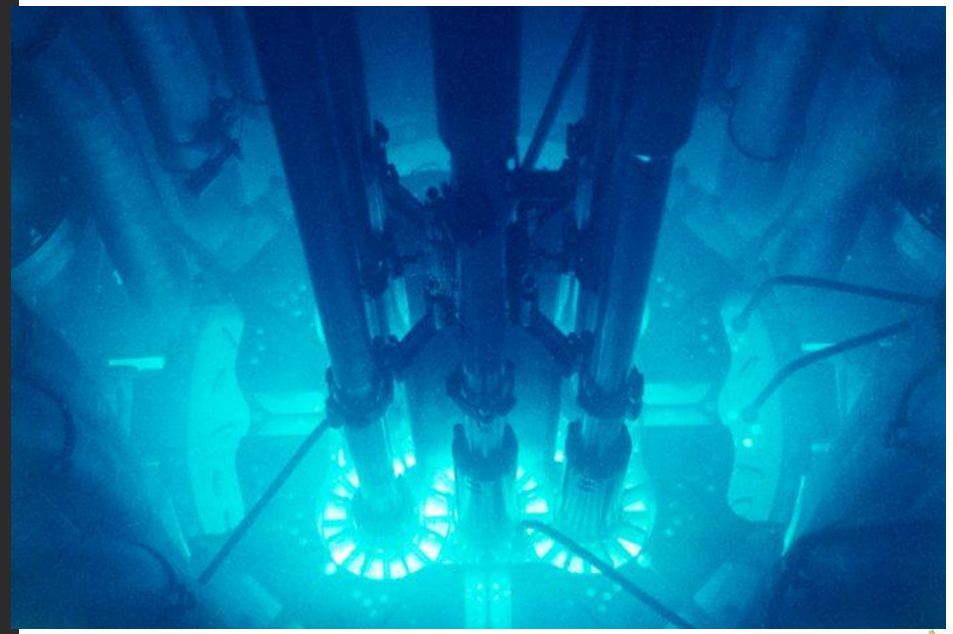


- 20% of universe is dark to "traditional" astronomy, i.e. using electromagnetic waves/light
- Need a new set of "messengers" – Gravitational Waves and Neutrinos

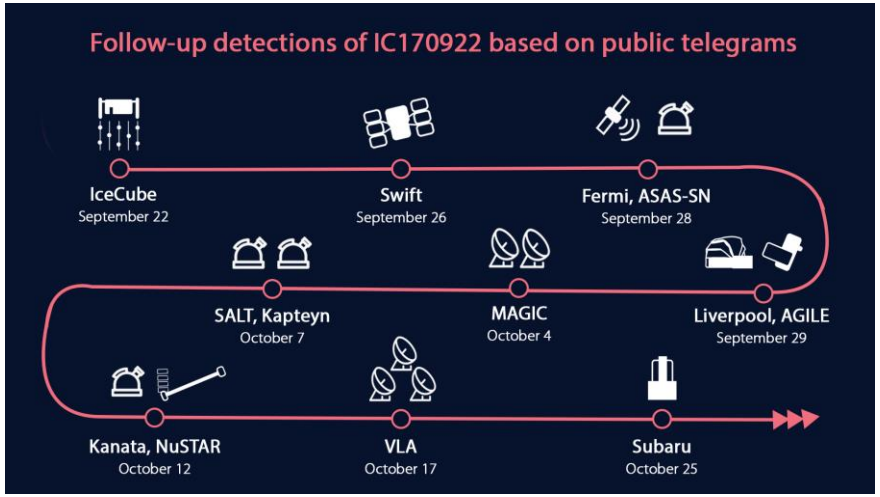
IceCube Science – How does it work?

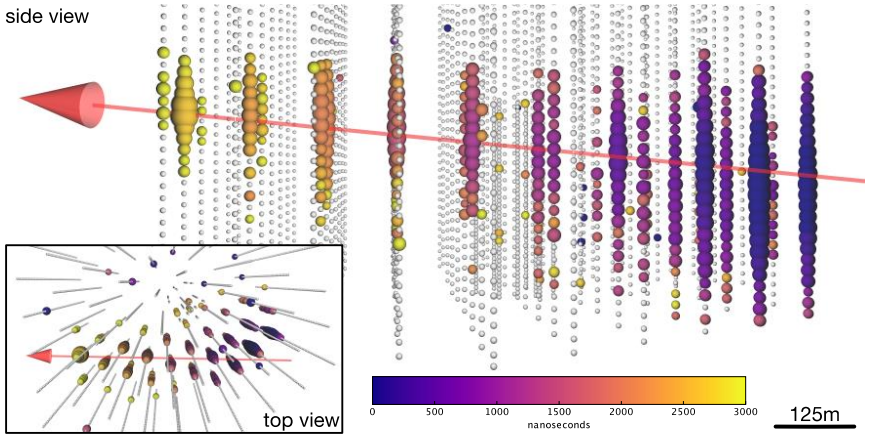


- Cherenkov light - Sonic boom with light
- Cherenkov light appears when a charged particle travels through matter faster than light can



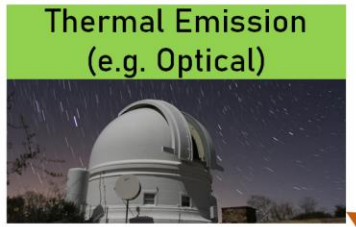
IceCube Science – Multi-Messenger Astrophysics

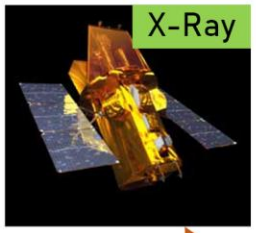


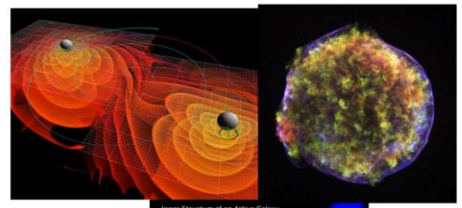


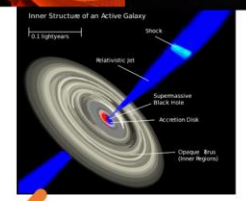
- Multi-Messenger Astrophysics (MMA)
  - Observing astrophysical phenomena with more than one "messenger" (gravitational waves, neutrinos, EM)
  - One of NSF's Big 10 Ideas
- IceCube detected an event that came from Blazar TXS 0506+056
  - Follow up observations by several observatories/telescopes showed signal
  - Back catalog showed access for this source
- **Fast response to alerts requires significant cyberinfrastructure**
- NS+NS merger would be ideal IceCube+LIGO observation
- Core collapse supernova would be ideal for IceCube+DUNE observation

Multi-Messenger Astrophysics









UW-Madison

Raw Data

Iridium

TDRSS

Ship

Alert Data

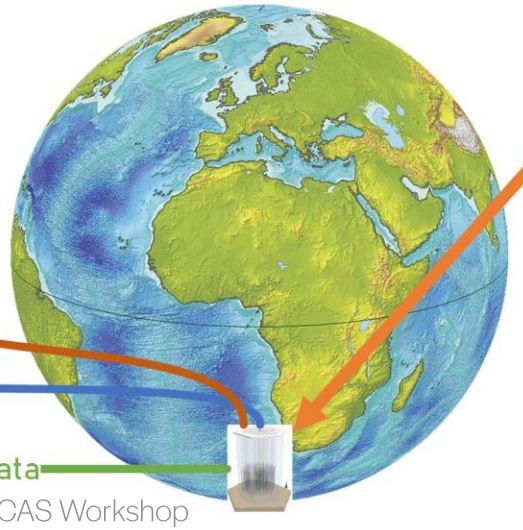
Reduced Data

Disk

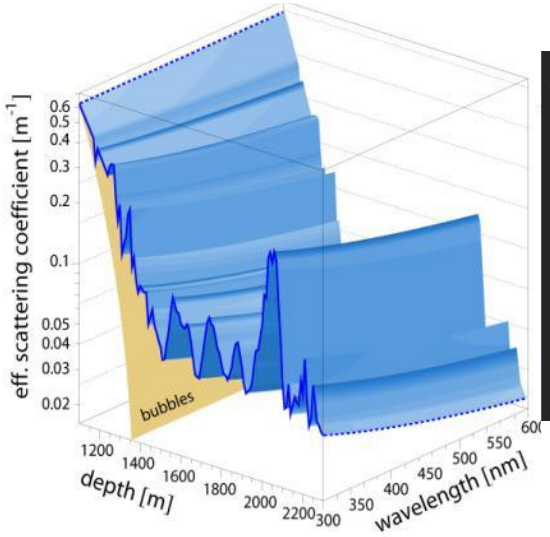
Raw Data

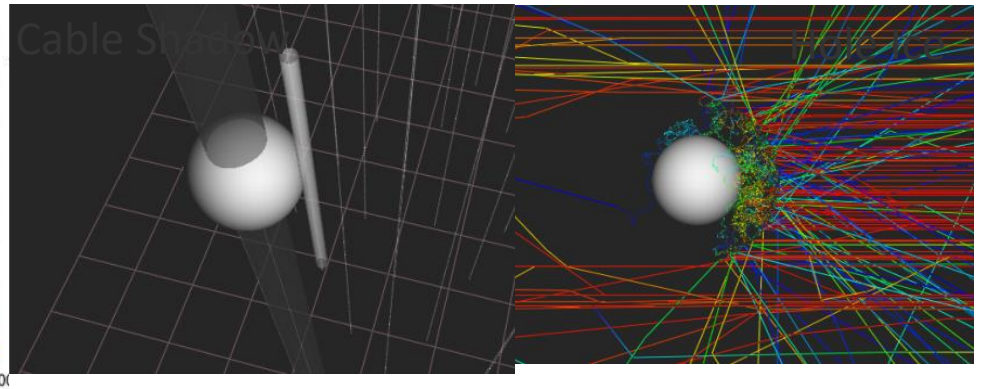
Raw Data

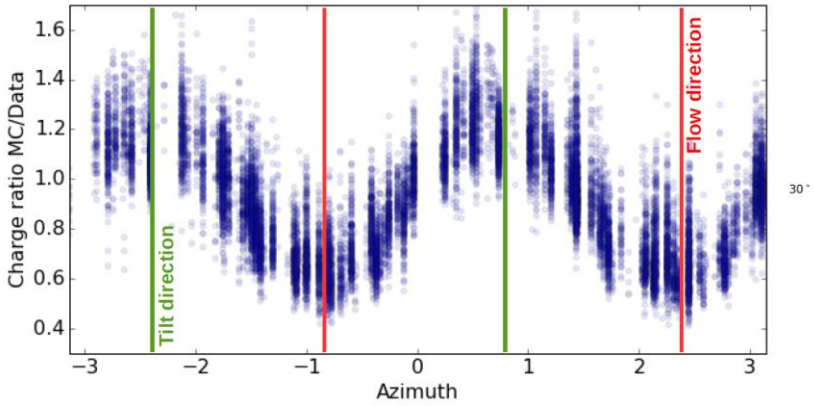
ν

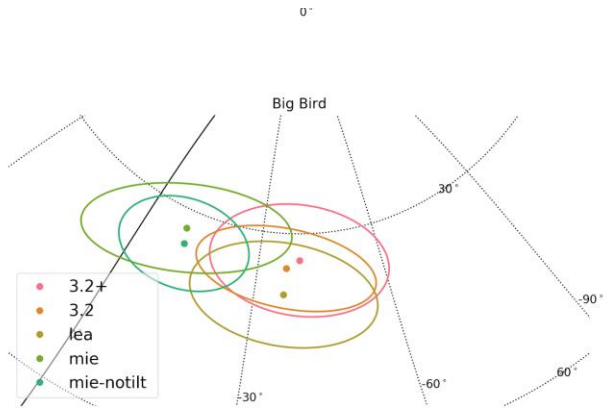


Ice Model



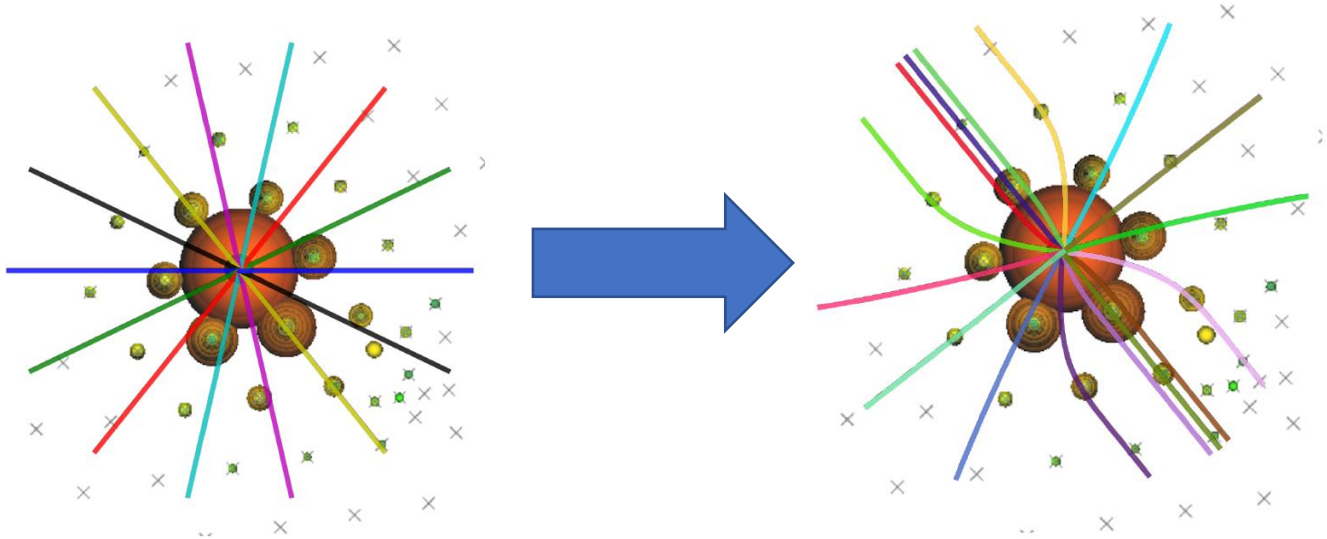






- Natural medium – Hard to calibrate properly
- Dropped a detector into a grey box
  - The ice is very clear, but…
  - Is it uniform?
  - How has construction changed the ice?
- Ray-Tracing with GPUs
  - Too complex for a parametrized approach
  - Needs brute force approach
- Drastic changes in reconstructed position with different ice model

Ice Model



Cartoon considering flow of ice

- New calibration data is available every year – More data being data, new ice cores being taken, etc.
- Want to have most up-to-date information in detector
- “Fitting” the model is compute intensive – Up to 400 GPU years
- Newer analyses want the most up-to-date simulation
- Redoing it all? - Lots of systematic error space between models

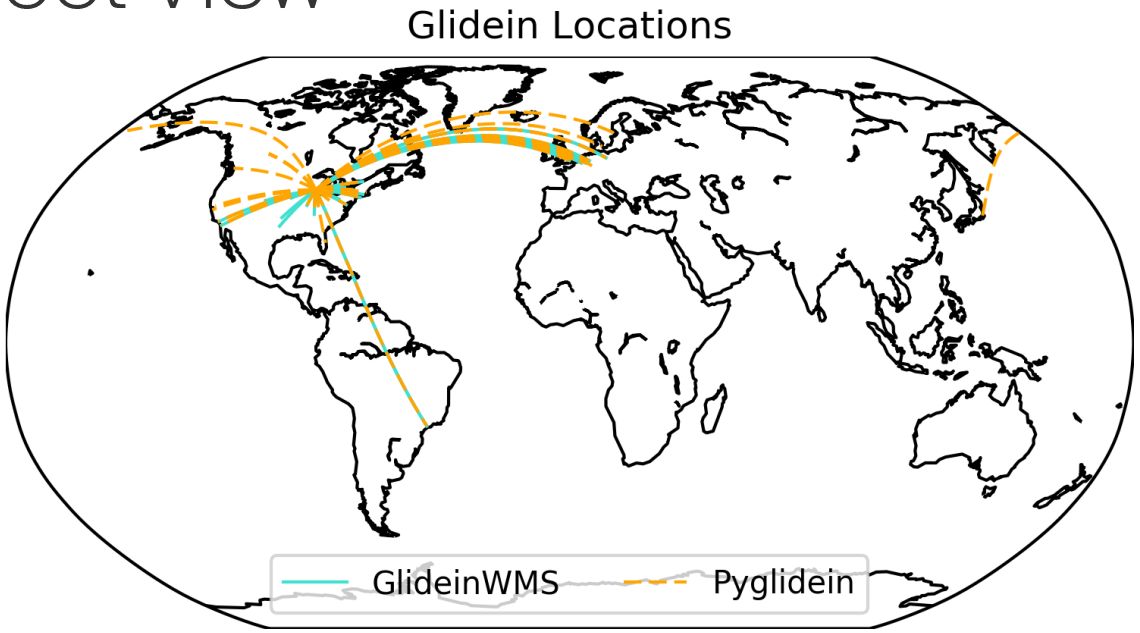
Computing

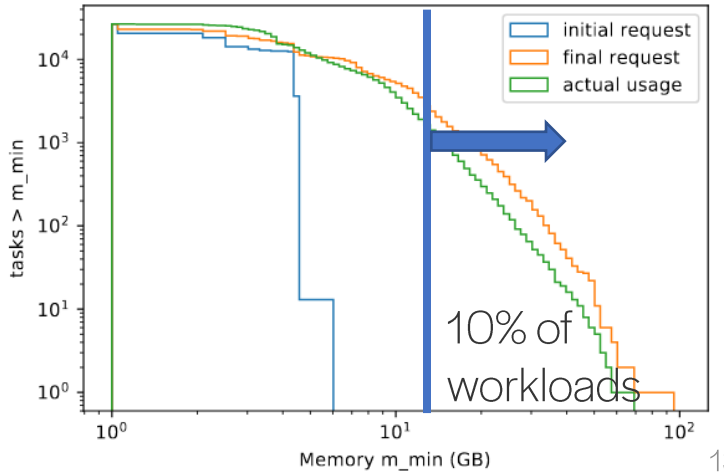
IceCube Computing – 30,000 Foot View

- Classical Particle Physics Computing
  - Ingeniously parallelizable – Grid Computing!
  - "Events" - Time period of interest
  - Number of channels varies between events
  - Ideally would compute on a per event-basis
- Several caveats
  - No direct and continuous network link to experiment
  - Extreme conditions at experiment (-40 C is warm, desert)
  - Simulations require "specialized" hardware (GPUs)
  - In-house developed and specialized software required
  - Large energy range cause scheduling difficulties – Predict resource needs, run time, etc.

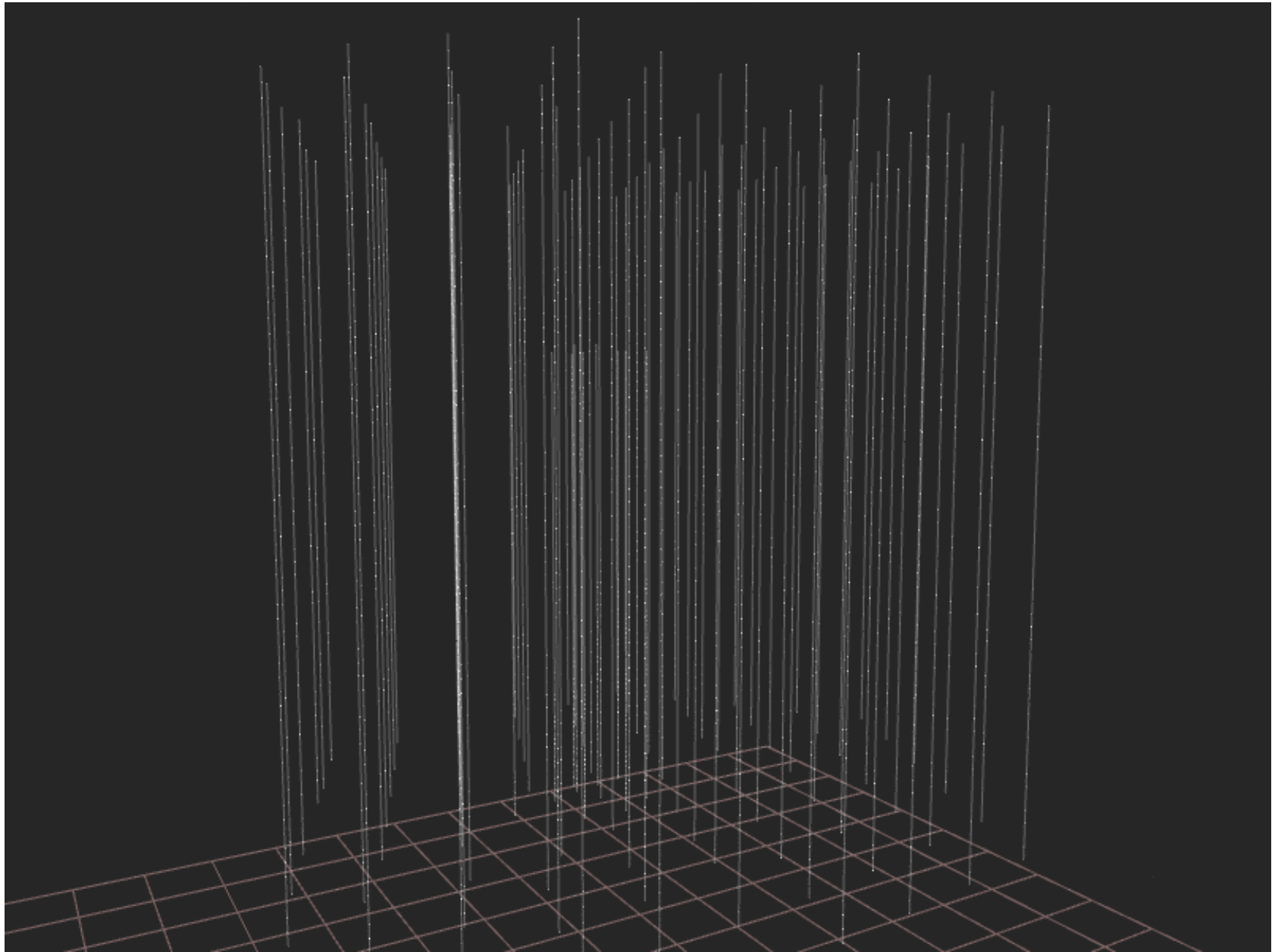
IceCube Computing – 10,000 Foot View

- Global heterogeneous resources pool
- Mostly shared and opportunistic resources
- Atypical resources requirements and software stack
  - Accelerators (GPUs)
  - Broad physics reach with high uptime- Lots to simulate
  - “Analysis” software is produced in-house
    - “Standard” packages, e.g. GEANT4, don’t support everything or don’t exist
    - Niche dependencies, e.g. CORSIKA (air showers)
- Significant changes of requirements over the course of experiment - Accelerators, Multi-messenger Astrophysics, alerting, etc.

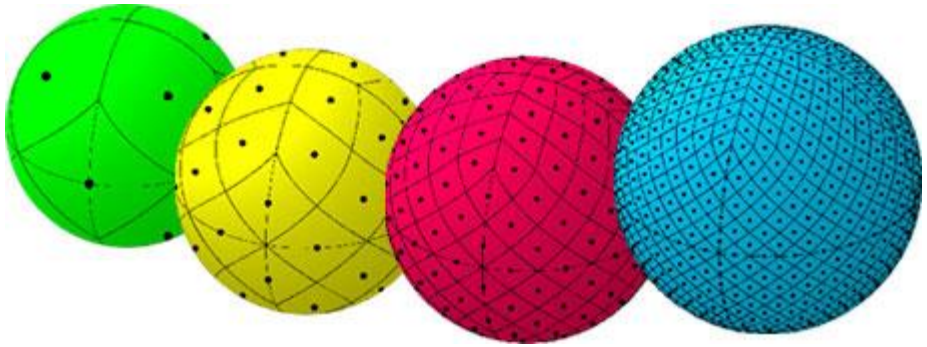


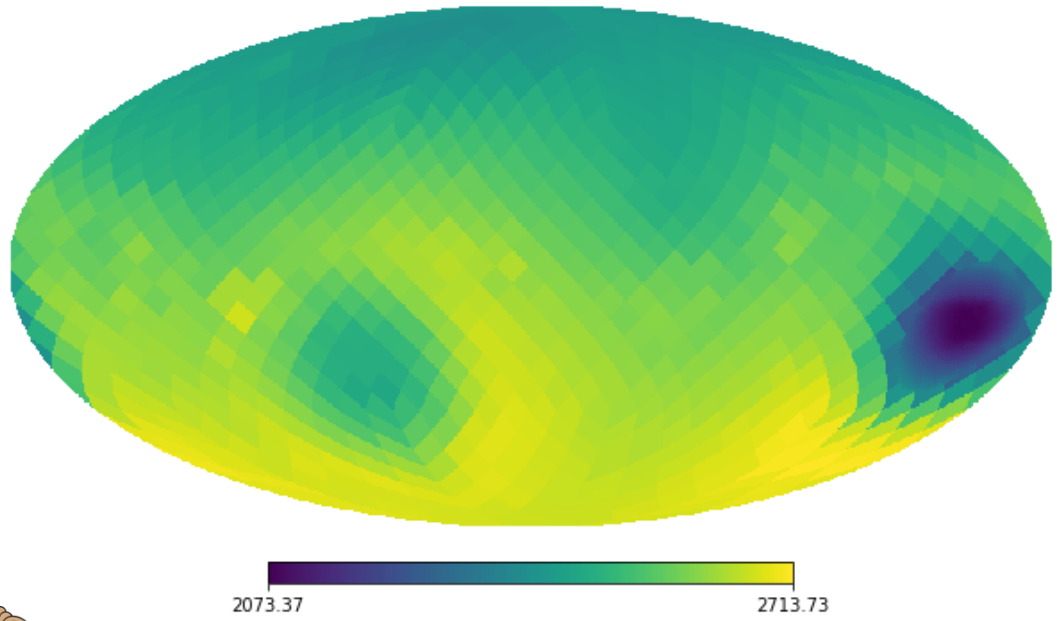


Why GPUs?



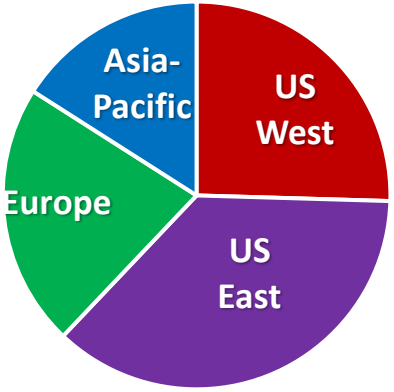
Multi-Messenger Astrophysics – Reconstruction



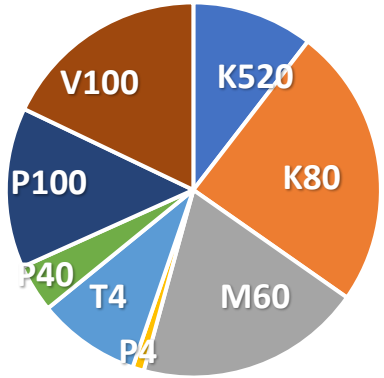


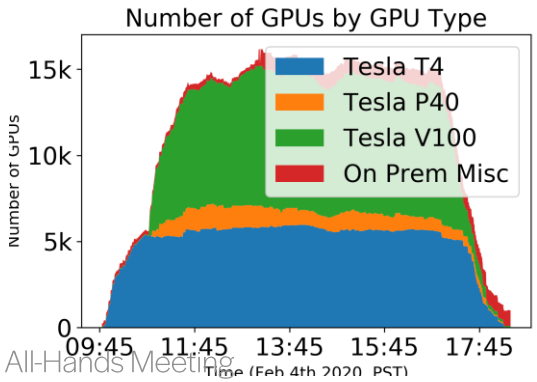
- Most accurate directional reconstruction comes by scanning across the sky
  - Split sky into constant surface area pieces
  - Test each directional hypothesis against likelihood
  - Create directional likelihood map
  - Gives most probable direction and error
- Each hypothesis calculation is independent – Easy to split up workload across O(1000[000]) or cores

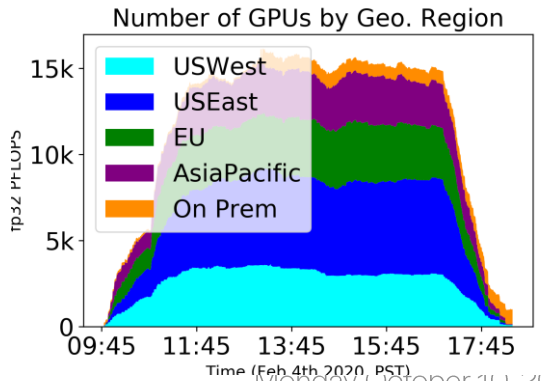
GPU Cloudburst Experiments



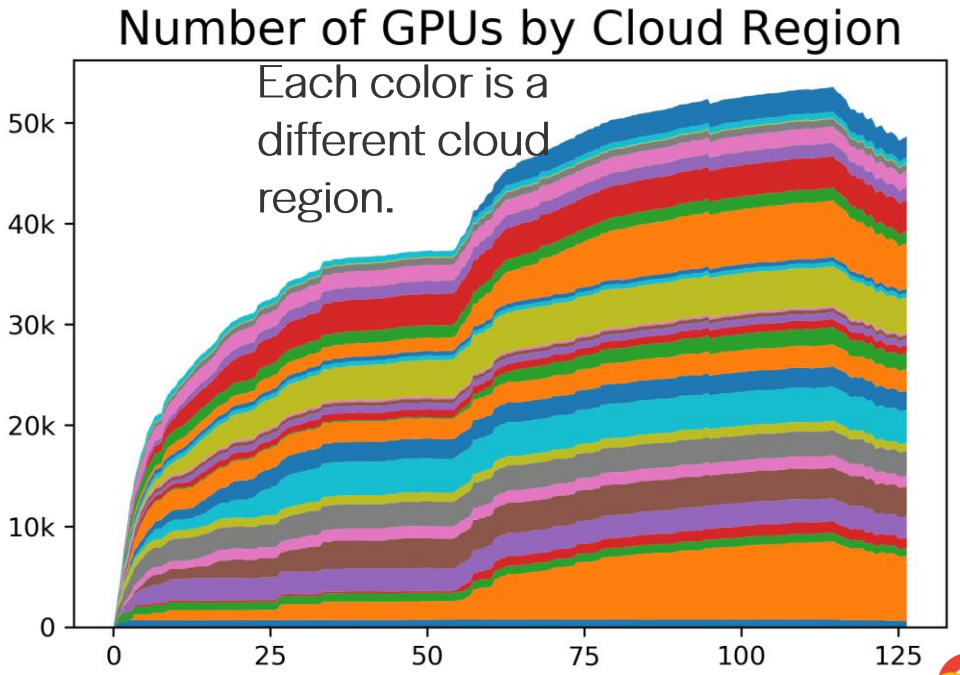
- Original Goal: Create an ExaFLOP compute pool in the cloud (80,000 NVIDIA V100) and address review panel recommendations
- Cloud provider(s) do not have those resources available – We were promised they do
  - Pre-allocated resources
  - Single cloud provider does not have those resources
- First Experiment – On Nov 16 2019 we bought all GPU capacity that was for sale in Amazon Web Services, Microsoft Azure, and Google Cloud Platform worldwide - **Creating The Largest GPU Cloud Pool in History**
  - 51k NVIDIA GPUs in the Cloud
  - 380 Petaflops for 2 hours (90% of DOE's Summit, No. 1 in Top 500)
  - Distributed across, US, EU, and Asia-Pacific
  - Cost: $50-150k (under NDA)
- Second Experiment – More realistic test
  - Most cost-efficient GPUs for 8 hours
  - Achieve 1 ExaFLOP-hour of compute
  - Distributed across, US, EU, and Asia-Pacific
  - Cost: ~$60k







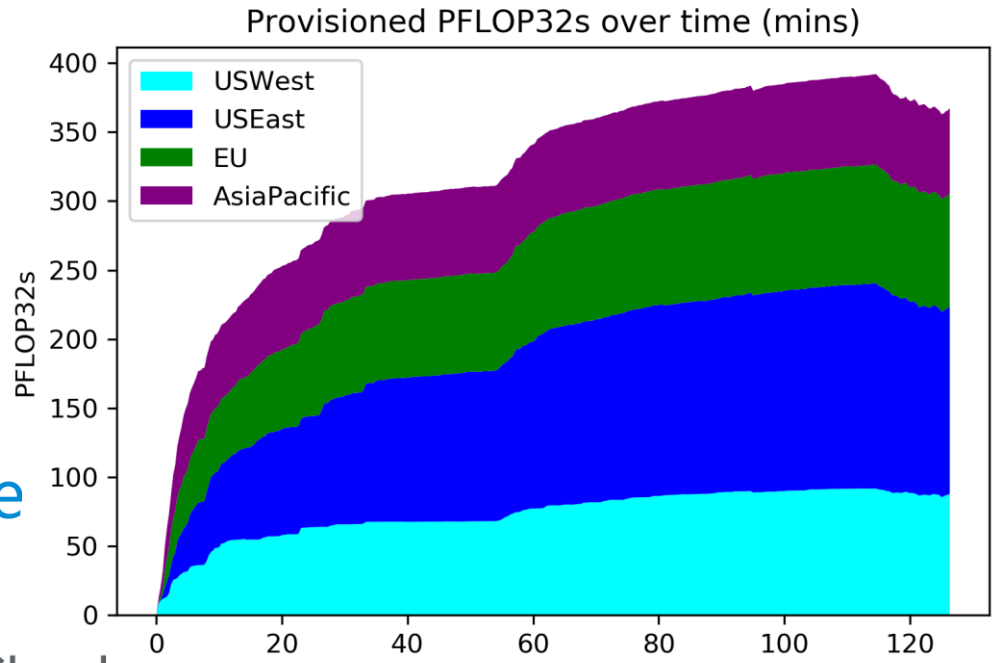
GPU Cloudburst – 1st Experiment



aws

Azure

Google Cloud

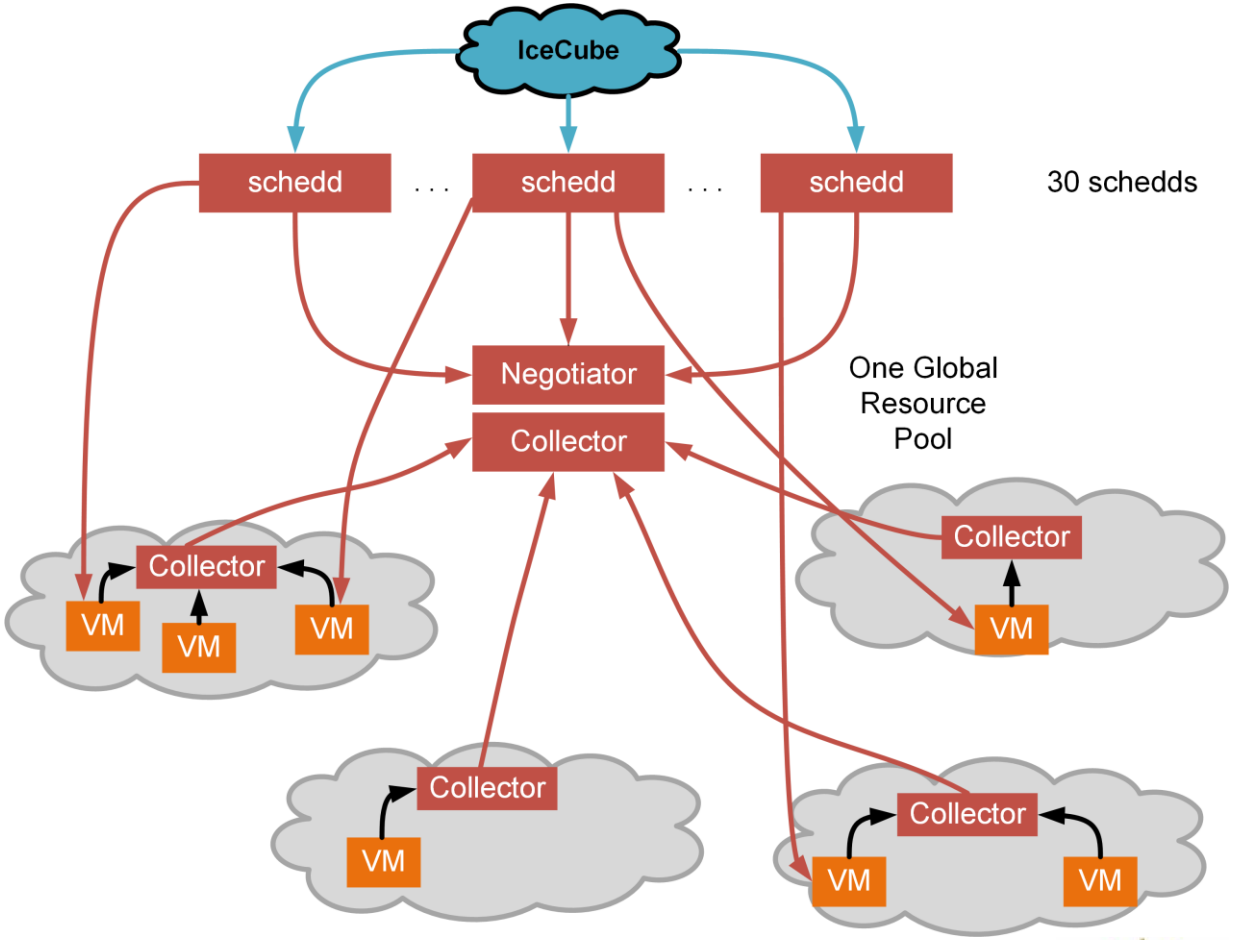


Peaked at 51,500 GPUs

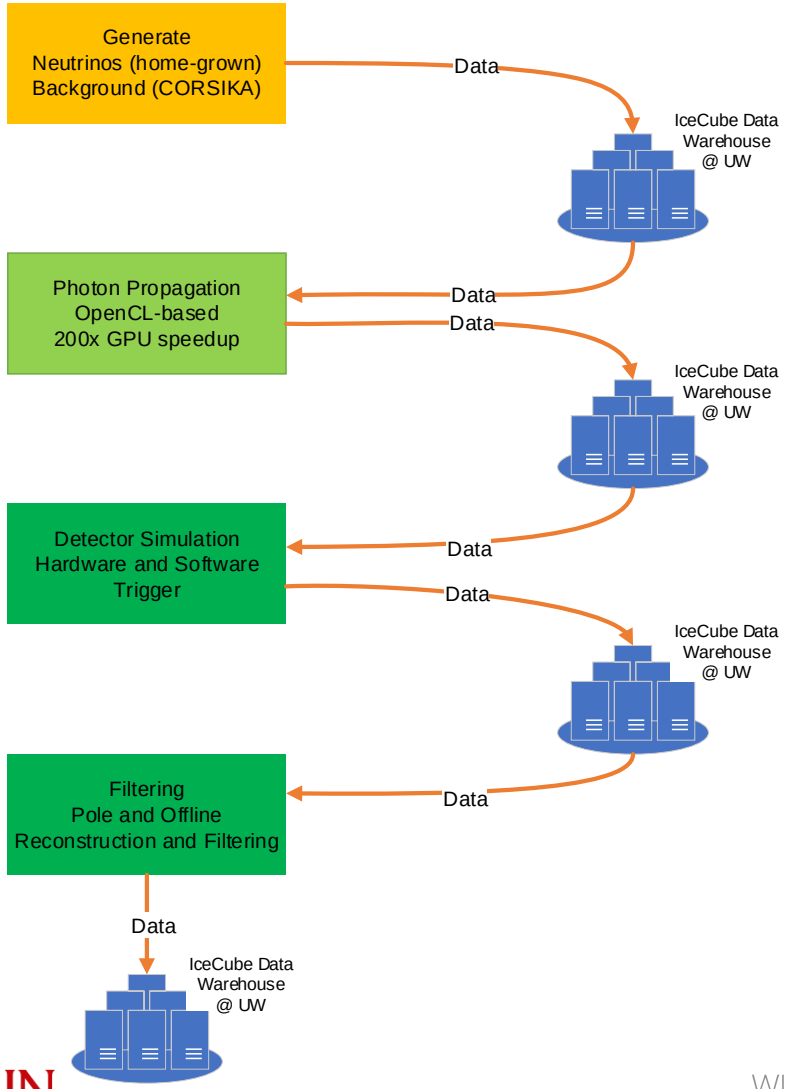
Total of 28 Regions in use.

GPU Cloudburst Technology

- Multi-collector HTCondor setup – Already well-established
- Collector in each cloud region to reduce load on start-up – No idea where resources would be
- Workload is computing heavy compared to typical IceCube load – Reduce potential networking cost
- 1st demo: In and output data stored in cloud
- 2nd demo: Input came from UW, output stored in cloud

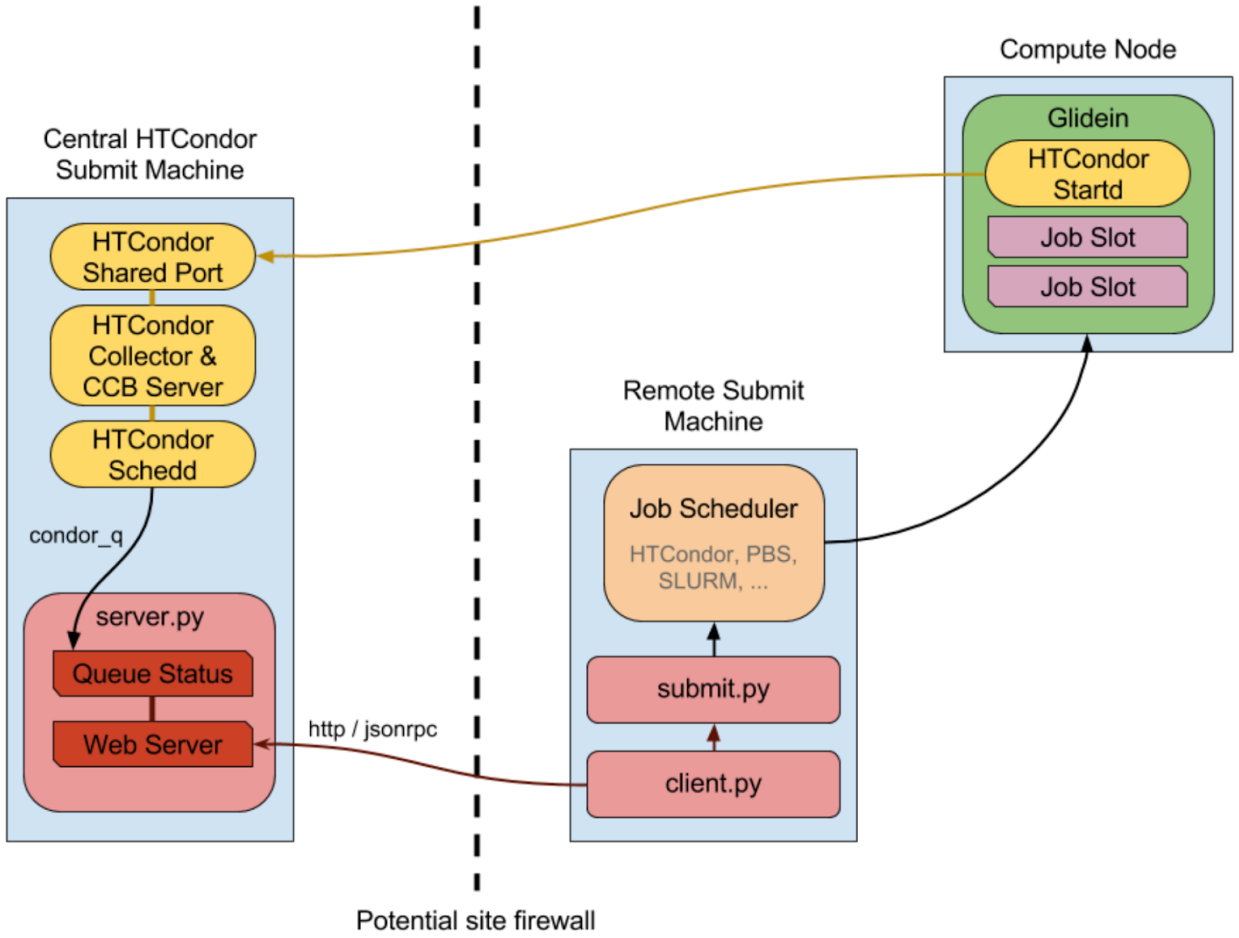


IceCube Grid – Simulation Workflow



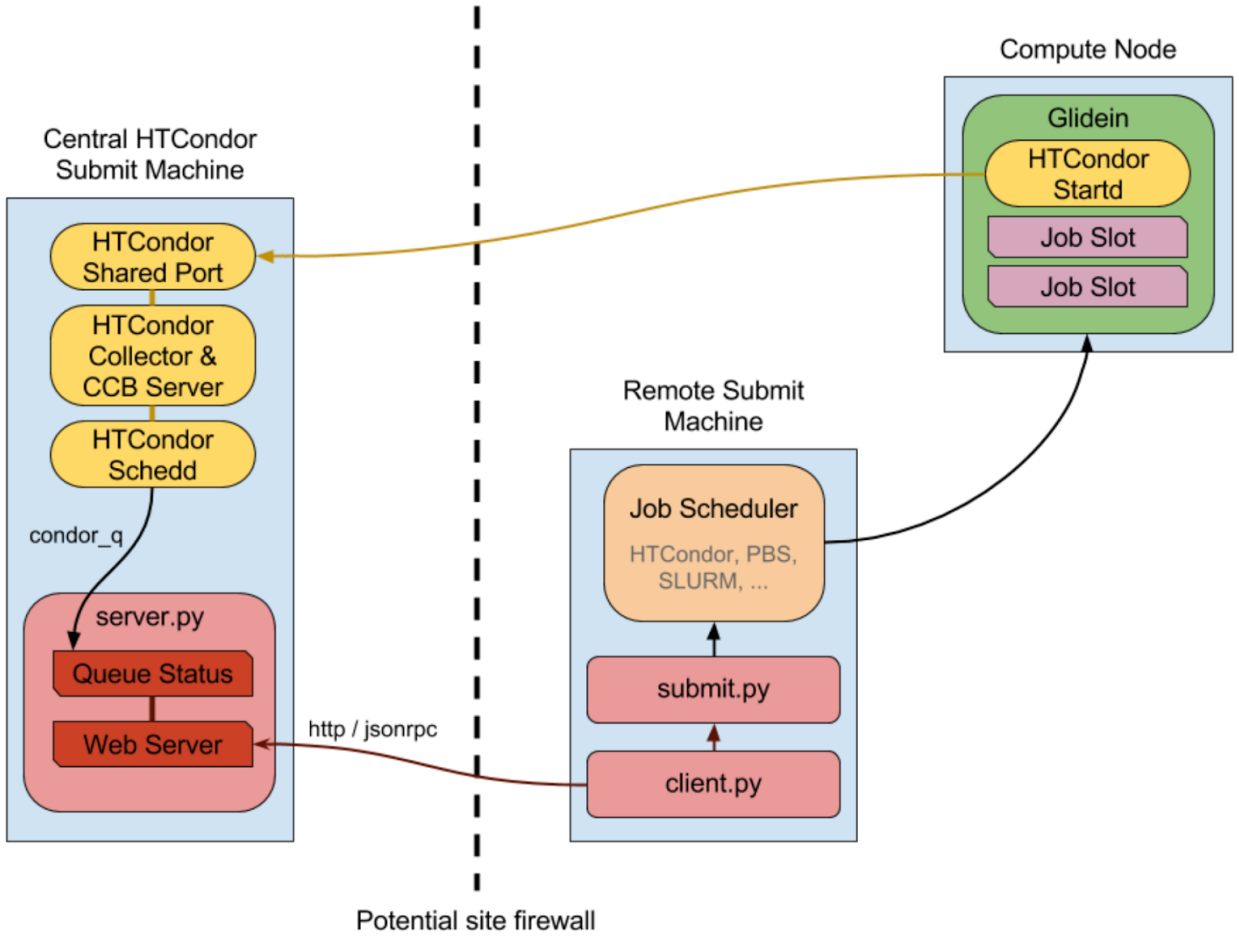
- Fairly straightforward particle physics-like workflow
- Big constraint is lack of dedicated resources
  - No data aware scheduling
  - Lots of data movement – Lots of time wasted to move data
- Different steps can have drastically different requirements

IceCube Grid – PyGlidein - I



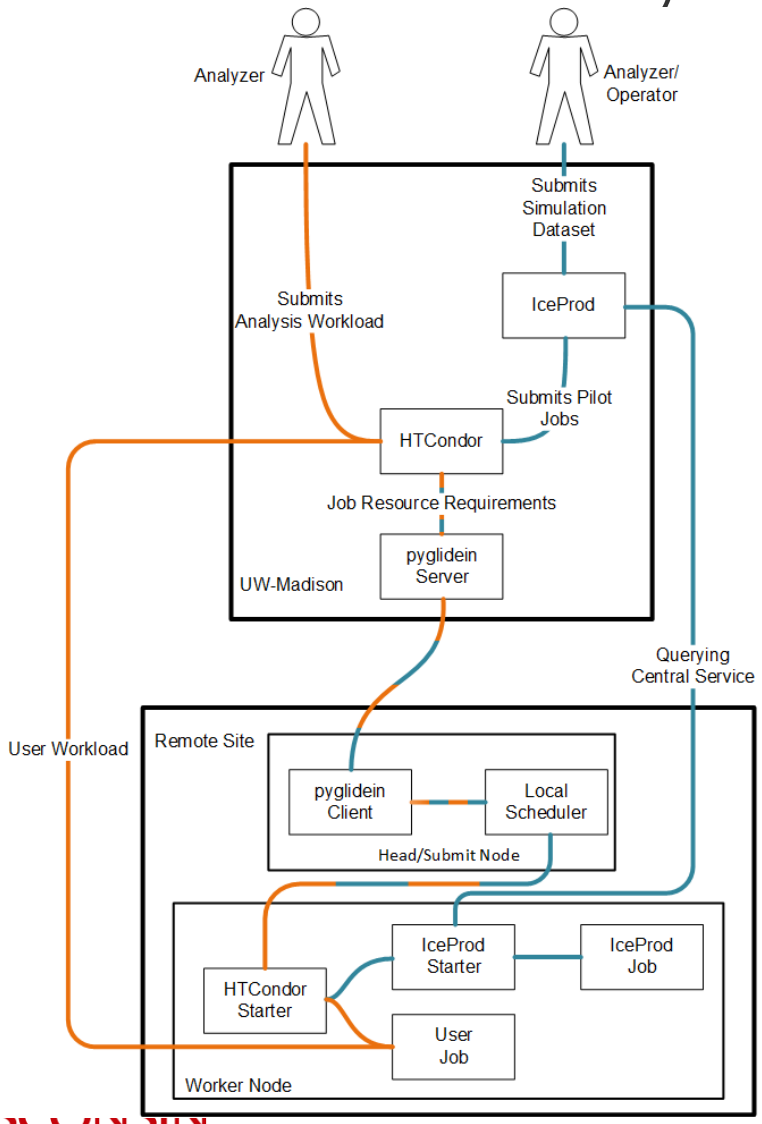
- Separate job submission from workflow management
- Lightweight design as possible
- Only difference between sites is a config file
- Why separate system?
  - Performance issues – Maximum ~3500 jobs
  - Experts needed for deployment, operation, and monitoring
  - Individual users could not use distributed resources

IceCube Grid – PyGlidein - II



- Exposed the job resource requirements - CPU, Memory, Disk, GPU – via HTTP
- Remote client queries for job requirements and submit **`HTCondor startd`** jobs accordingly within local resource constraints
- When **`HTCondor startd`** executes connects back to central pool
- Multiple jobs are submitted per single job in pool – Assuming other jobs will be able to use slots, otherwise dies within set amount of time

IceCube Grid – PyGlidein - III



- User perspective
  - HTCondor + Data Management
  - "Just an HTCondor pool"
- Operator perspective
  - Little overhead to add cluster to pool
  - Fairly easy to monitor, e.g. `condor_status`
  - No need for a CE - Use `SSH` or `cron` for submission
  - Local container support
- Future improvements
  - Code needs clean-up – Organic growth to support multiple schedulers
  - User container support

Thank you!

Questions?