Snowmass2021 - Letter of Interest

IceCube and IceCube-Gen2 Simulation

Thematic Areas: (check all that apply □/■)
■ (CompF1) Experimental Algorithm Parallelization
■ (CompF2) Theoretical Calculations and Simulation
□ (CompF3) Machine Learning
□ (CompF4) Storage and processing resource access (Facility and Infrastructure R&D)
□ (CompF5) End user analysis
□ (CompF6) Quantum computing
□ (CompF7) Reinterpretation and long-term preservation of data and code

Contact Information:
Juan Carlos Díaz-Vélez (University of Wisconsin–Madison) [juancarlos@icecube.wisc.edu],

Authors (alphabetical):
Juan Carlos Díaz-Vélez (University of Wisconsin–Madison) [juancarlos@icecube.wisc.edu],
Kevin Meagher (University of Wisconsin–Madison) [kmeagher@icecube.wisc.edu],
Alex Olivas (University of Maryland) [aolivas@umd.edu],
David Schultz (University of Wisconsin–Madison) [dschultz@icecube.wisc.edu],
on behalf of the IceCube[1] and IceCube-Gen2[2] Collaboration [analysis@icecube.wisc.edu]

Abstract: (must fit on this page)
The IceCube Neutrino Observatory is a km^3 neutrino detector deployed at the South Pole. IceCube measures neutrinos by detecting the optical Cherenkov photons produced in neutrino-nucleon interactions. Monte Carlo simulations are essential for developing analysis methods, and for testing the performance of reconstruction algorithms. IceCube can detect events with energies ranging from 10 GeV to beyond 1 EeV. Simulating such a wide range of energies is computationally challenging. In order to address this, the IceCube Collaboration has explored novel methods and technologies including the use of GPGPUs in order to increase both the quality and throughput of Monte Carlo simulations. New plans for expansion of the detector will further challenge our capacity to efficiently produce Monte Carlo simulations and will require a concerted effort to explore new technologies and methods.

[1] Full author list available at https://icecube.wisc.edu/collaboration/authors/snowmass21_icecube
[2] Full author list available at https://icecube.wisc.edu/collaboration/authors/snowmass21_icecube-gen2

The IceCube Neutrino Observatory [1], located at the South Pole, instruments a cubic kilometer of Antarctic ice. IceCube uses 5160 digital optical modules (DOMs) arranged on 86 strings in a hexagonal array to detect Cherenkov radiation from relativistic charged particles emitted during neutrino interactions. This configuration can detect neutrinos as high as EeV energies while the center, more concentrated region of DOMs, called DeepCore, is optimized to extend the detection energy down to a GeV.

Simulation Overview

Monte Carlo simulations in IceCube are essential for developing analysis methods to identify signal from background, for testing the performance of reconstruction algorithms, and for determining the background contamination of data analysis samples. One of the main goals of the collaboration is to accelerate simulation workflow in order to generate Monte Carlo with statistics comparable to data. Direct photon propagation is currently done on dedicated GPU hardware at several IceCube Collaboration sites and through opportunistic grid computing.

IceCube's computational costs for simulation scale strongly with the number of optical modules and the volume of the detector. IceCube's simulation chain covers an extensive range of energies from neutrino oscillations at $O(10\,\text{GeV})$ to cosmogenic neutrinos at $O(10\,\text{PeV})$. IceCube has been at the forefront of using novel cyberinfrastructure, including the large-scale use of GPUs.

The IceCube Upgrade plans to improve on the resolution of GeV neutrinos by adding an additional seven strings concentrated around DeepCore with several DOM designs and additional calibration devices [2]. To improve the energy and angular resolution of TeV to EeV neutrino detection and increase the detection rate, Gen2 will add 120 strings to instrument a total volume of 7.9 km^3 [3]. Simulating these extensions will substantially increase computing requirements and software complexity. The increasing demand for resources from Monte Carlo simulation and individual analyzers will require a concerted effort across the collaboration and the addition of professional resources to develop improvements needed to support the Upgrade and IceCube-Gen2.

Simulation Software

The IceTray framework, written in C++ with Python bindings, is continually evolving to adapt to new standards. Maintenance of this code requires a high level of expertise in order to provide optimal modern C++ code and to support multi-core architectures in order to fully and efficiently utilize distributed resources. The IceCube simulation is composed of various IceTray modules and services necessary for generating both signal and background Monte Carlo. The code for both signal and background simulation needs to run in various distributed platforms and build against a wide range of third-party libraries shared with the astrophysics and particle physic community.
Generators:

Cosmic Ray Simulations - IceCube Simulation computing requirements are primarily dominated by background simulations given that there is roughly a factor of 10^6 cosmic-ray induced muons triggering the detector for each neutrino event. Along with atmospheric neutrinos, background cosmic-ray induced muons in the in-ice array are simulated with `CORSIKA` [4]. Producing the required statistics comparable to 10 years of triggered data requires roughly 30k years of CPU time and about 2.4k years of GPU time to produce, reconstruct, and filter. `CORSIKA` is also used to simulate full hadronic showers triggering the IceTop surface array. `CORSIKA` is widely used by the astrophysics community and members of the IceCube Collaboration actively participate in its development. There is currently a concerted effort to rewrite `CORSIKA` Fortran code in C++.

MuonGun - As an alternative to expensive background simulation, most of which does not trigger the detector, we can also simulate final-state muons that can be weighted according to a

parametrized flux calculated from CORSIKA simulations using the same approach of MUPAGE [5] which was developed by the ANTARES Collaboration [6]. These MuonGun simulations are significantly more efficient to produce, requiring about 6M CPU-hours and comparable GPU time to simulate in order to meet our goals. These simulations must be validated against CORSIKA before they are used, but this requires a significantly smaller simulated data set.

Neutrino Signal - Injection of neutrino signal is simulated with LeptonInjector, developed by the collaboration to be relatively fast and computationally inexpensive. GENIE [7] is also utilized along with GEANT4 [8] to simulate neutrinos with energies below 200 GeV.

Lepton Propagation: Propagation of leptons, which includes energy losses and stochastic generation, the main contributors to Cherenkov radiation and therefore detector signal is simulated using PROPOSAL [9], an open source C++ library developed and maintained by our collaborators at TU Dortmund University.

Photon Propagation: The the highly parallel nature of the simulation of photon propagation benefits with substantial acceleration of our simulation on GPGPUs [10]. All of the simulated photons go through the same steps before getting absorbed or hitting a sensor: photon propagation between the scattering points, calculation of the scattering angle and new direction, and evaluation of whether the current photon segment intersects with any of the optical sensors of the detector array. Each GPU performs the same computational operation on individual photons in parallel across multiple threads. Although a single thread runs slower than a typical modern computer CPU core, running thousands of them in parallel results in the much faster processing of photons on the GPU. Through the use of GPUs, we have achieved significant acceleration of the photon propagation by factors of 150 or more compared to running on a single CPU core. We currently maintain two separate and independent implementations of this code that run on OpenCL (PPC and CLSim) that can be used for cross-validation.

Detector Simulation: Simulation of the PMT, DOM Mainboard, and DAQ trigger. The IceCube detector simulation includes individually calibrated PMT waveforms, optimized event resampling for low-energy background simulation, and detailed models of the optical properties of the ice. The detector simulation is currently the second largest consumer of resources in the simulation chain. The ability to predict resource usage, to allow for efficient scheduling in a distributed environment, is of high-priority. This need will become even more so as the number, type, complexity, and density of optical modules increase during the Upgrade.

Future Improvements

IceCube's use of GPGPU/heterogeneous programming is currently isolated to photon propagation, but there is potential for other unexplored areas, such as detector simulation, reconstruction, and deep learning algorithms. The current propagators are written in OpenCL, whose future is questionable, with other technologies such as Sycl and OpenAcc as potential successors. Industry is rapidly evolving to support deep learning, and investigation in the new technology is going to be necessary over the next five years. Additional labor is required to support new compute platforms, e.g. Power9 or ARM, and for modifications to simulate IceCube-Gen2 and Upgrade hardware components.

We continually explore better and more efficient ways to meet the simulation needs of analyzers. New strategies are being developed for dynamic simulation of systematic uncertainties in our understanding of ice properties, hole-ice, and DOM sensitivity and for determining the impacts of these on physics analyses. A number of the software tools used for IceCube simulation are shared with other experiments in the astroparticle physics community and others have similar potential. Long-term development and maintenance of such tools is crucial for the community as is retaining the expertise of people with both computer science and physics backgrounds.

References

[1] M. G. Aartsen et al. The IceCube Neutrino Observatory: Instrumentation and Online Systems. *JINST*, 12(03):P03012, 2017.

[2] Aya Ishihara. The IceCube Upgrade – Design and Science Goals. *PoS*, ICRC2019:1031, 2020.

[3] M.G. Aartsen et al. IceCube-Gen2: The Window to the Extreme Universe. 8 2020.

[4] D. Heck, G. Schatz, T. Thouw, J. Knapp, and J. N. Capdevielle. CORSIKA: A Monte Carlo code to simulate extensive air showers. 1998. FZKA-6019.

[5] G. Carminati, M. Bazzotti, S. Biagi, S. Cecchini, T. Chiarusi, A. Margiotta, M. Sioli, and M. Spurio. MUPAGE: a fast atmospheric MUon GEnerator for neutrino telescopes based on PArametric formulas. 7 2009.

[6] M. Ageron et al. ANTARES: The first undersea neutrino telescope. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 656(1):11 – 38, 2011.

[7] Costas Andreopoulos, Christopher Barry, Steve Dytman, Hugh Gallagher, Tomasz Golan, Robert Hatcher, Gabriel Perdue, and Julia Yarba. The genie neutrino monte carlo generator: Physics and user manual, 2015.

[8] S. Agostinelli et al. Geant4—a simulation toolkit. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250 – 303, 2003.

[9] J.-H. Koehne, K. Frantzen, M. Schmitz, T. Fuchs, W. Rhode, D. Chirkin, and J. Becker Tjus. Proposal: A tool for propagation of charged leptons. *Computer Physics Communications*, 184(9):2070 – 2090, 2013.

[10] Martin Merck, Dmitry Chirkin, Juan Carlos Diaz Velez, and Heath Skarlupka. IceCubes GPGPU's cluster for extensive MC production. *J. Phys. Conf. Ser.*, 396:022046, 2012.