IRIS-HEP Project Proposal: Improving Performance of Particle Tracking Algorithms in a Muon Collider

Chris Sellgren, UC Santa Barbara
Mentors: Simone Pagan Griso and Sergo Jindariani
Dates: June 26 - September 15, 2023

Introduction

Particle colliders are currently the most powerful tool for examining the fundamental forces and constituents of our universe. Examining the interactions of particles at high energies allows physicists to test our current predictions and search for new physics beyond the Standard Model. Collider discoveries can provide insight into cosmological mysteries and even support research in other industries, such as medicine and computer science.

The particle colliders currently in use primarily study proton-proton collisions (such as the LHC), or electron-positron collisions (such as the LEP). However, since the 1980s, physicists have discussed the possibility of developing an experiment that would collide muons. Unlike the hadrons of the LHC, muons are fundamental particles, but because they are much heavier than electrons, they lose over a million times less energy to synchrotron radiation. A muon collider would thus be able to reach TeV energy scales and probe fundamental physics with a compact and energy-efficient design. However, because muons have a lifetime of about 2 μs at rest, muon colliders face unprecedented challenges from particle decay creating a beam-induced background (BIB). The detection design of a muon collider must be able to isolate tracks from the collision products, which can involve a searching through a signal:background ratio on the order of 10^{-6}.

Project Proposal

This project will work to improve the performance of algorithms used for track reconstruction. The algorithms rely on a large set of parameters, which must be optimized to maximize the likelihood of producing a correct trajectory while suppressing the chances of producing a fake trajectory. Because the high multiplicity of measurements in a muon collider experiment creates a large number of fake measurements, the tracking parameters must be configured carefully to maximize efficiency of the algorithms. This project will involve a systematic study of algorithm performance, culminating in the creation of a standard set of requirements that will be used as a baseline for track reconstruction.

The current tracking algorithms can also be significantly improved in their computational efficiency. Currently, high RAM usage makes running simulations at scale difficult. Incorporating a multi-threading capacity will reduce the time required for simulation and analysis by a factor of at least 5. However, the current design of the tracking algorithms uses an iterative approach that would challenge multi-threading. Parallel processes must ensure to not find the same trajectory, so a global pool of measurements must be shared among the processes.

Designing the algorithms to run in parallel will require a logical redesign informed by user discretion and testing.

Incorporating multi-threading will involve transitioning to the new Key4HEP software stack. Depending on the status of Key4HEP by the time the project begins in the summer, the project will either focus on improving algorithm performance through parameter optimization, or improving computational efficiency through a threading-supported redesign. The timeline below reflects both possible project designs.

Timeline

Week	**Proposed Activity**
1-2	Familiarizing with software and tools, testing performance of current design and identifying regions of improvement, determining primary focus for output.
3-4	Run simulations to optimize parameters for track reconstruction OR Design a preliminary algorithm to incorporate multi-threading by dividing detector design into subcomponents to be handled by parallel processes.
5-6	Create tools and metrics to assess and compare algorithm performance OR Adjust algorithm design to handle boundary cases, such as overlap of detector components or particles moving between different subcomponents/processes.
7-8	Adjust and expand algorithm design based on performance needs, implement algorithms to filter output tracks OR Test, improve, or redesign threaded algorithm design for multiple cases.
9-10	Continue expanding algorithm design, begin compiling work in a consistent framework that can be shared with collaborators
11-12	Finalizing work. Debugging and cleaning code, optimizing efficiency, writing documentation for shared framework, preparing final presentation.

Software deliverables may include redesigned algorithms built for multi-threading capacity, or improved algorithms based on optimization of parameters that includes a standard of common tracks to be shared with collaboration. Other deliverables may include written reports of track requirement standards, metrics for comparing performance, and plots comparing different algorithm designs.