# Using oneAPI for simulation: the AdePT project

Technical Student: Daniel-Florin Dosaru

Supervised by: Predrag Buncic (CERN) and Prof. Dr. James Larus (EPFL)

CERN Openlab Technical Workshop, March 9th

Some slides from Stephan Hageboeck

## About me

dosarudaniel@gmail.com
Linkedin, Github

- Master student in Computer Science at EPFL

- Working on my master thesis as a Technical Student CERN in EP/SFT group
  - Started on February 1st 2021

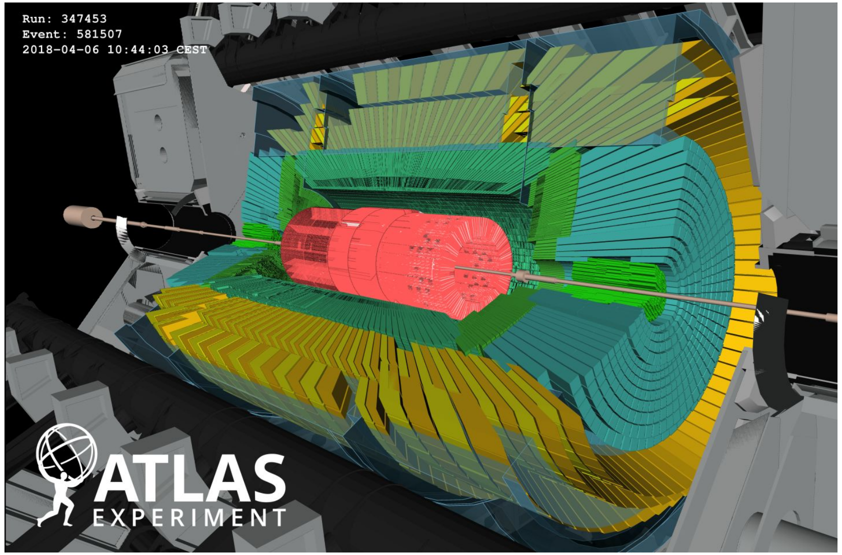
## What is AdePT

**A**ccelerated **d**emonstrator of **e**lectromagnetic **P**article **T**ransport

- Aims to demonstrate executing part of the detector simulation workflow on GPU
  - Builds on top of  its own containers (CopCore) that try to abstract CPU/GPU workflows
  - Evaluating solutions for abstracting the execution on CPU/GPU, including custom approaches
  - Targets NVIDIA GPUs using CUDA-C++ code
  - Uses VecGeom and VecCore libraries

# Detector Simulation on GPUs: Why?

- To understand the LHC data, have to simulate how particles interact with detectors
- Simulation takes ~ 40% of total CPU time used on WLCG
- Three main steps
  - Simulate particle collisions (~ fast)
  - Simulate what particles do in detector
  - Run reconstruction algorithms as if detector had been hit by real particles
- Most expensive steps:
  - Calorimeter simulation



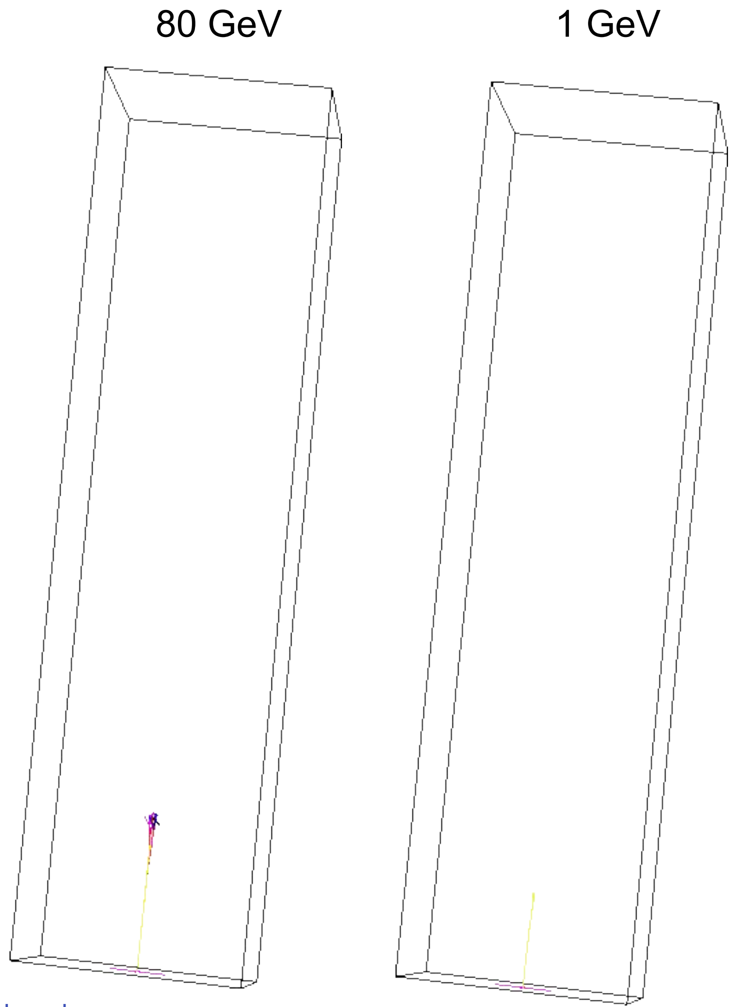
Slide from Stephan Hageboeck

## EM Showers are Expensive

- EM showers in lead glass
- Every thread takes a particle and processes it until it lost its energy
- Particle simulation is stochastic*
  - Cache problems
  - Branch prediction tricky
  - Really bad for CPUs
- Modern detectors can have very complicated layouts

* stochastic:

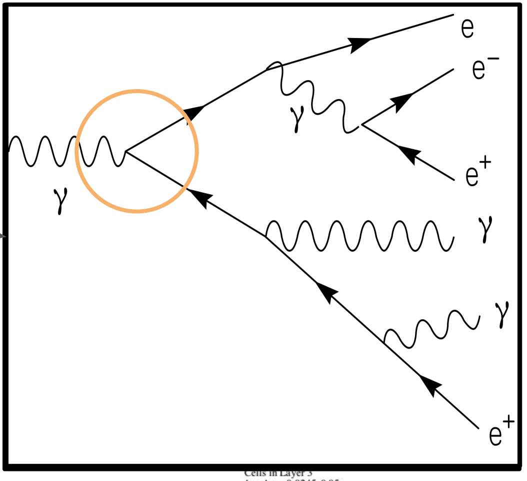
- Physics only predicts *probability* that something happens
- In every step, use a random number to decide *if* something happens

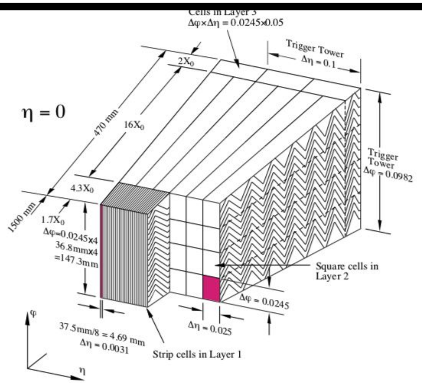


Slide from Stephan Hageboeck

## Don't Have to Start From Scratch

- Geant 4 has everything. Cannot use it on GPUs, though
- G4HepEM: Simulate EM physics
    - Mini library on top of Geant 4
    - Uses Geant 4 to initialise physics lookup tables
    - No calls to Geant 4 while simulation is running
- VecGeom: Detector description
    - Library for working with detector description
    - Developed for SIMD simulation prototype Geant V
- AdePT:
    - Some concepts/algorithms from the above can be (re-)used
    - Expect that both will need to undergo some refactoring/porting, though





Slide from Stephan Hageboeck

## AdePT roadmap

<u>March 2021</u>
Example demonstrating interfacing to G4HepEm library and tracking in constant magnetic field implemented (done)

**<u>April 2021</u>**
**Implemented example allowing physics validation between AdePT and Geant4 implemented (equivalent to TestEM3).**

<u>May 2021</u>
Geant4 example implemented allowing to call AdePT like a fast simulation process, delegating the simulation of the whole EM shower coming from a particle entering the calorimeter volume and then transferring back the energy depositions.

<u>June 2021</u>
First version magnetic field propagator for non-constant field implemented.
…...

# AdePT roadmap

March 2021
Example demonstrating interfacing to G4HepEm library and tracking in constant magnetic field implemented (done)

**April 2021**
**Implemented example allowing physics validation between AdePT and Geant4 implemented (equivalent to TestEM3).**

- **This version will be used as a basis for our oneAPI demonstrator**

May 2021
Geant4 example implemented allowing to call AdePT like a fast simulation process, delegating the simulation of the whole EM shower coming from a particle entering the calorimeter volume and then transferring back the energy depositions.

June 2021
First version magnetic field propagator for non-constant field implemented.
…...

## Our goals

Migrate the CUDA portion of AdePT code to oneAPI

1. Start by porting test code snippets to validate use of AdePT containers with oneAPI
2. Investigate how to incorporate dependencies (VecGeom, VecCore)
3. Focus on AdePT demonstrator planned for April ‘21 that includes parts of Geant4 code and port it to oneAPI
4. Demonstrate capability to run a single source on multiple devices
5. Compare CPU vs GPU performance, understand and document portability-related effort

## What has been done

- ✓ AdePT and CopCore interfaces (headers files) ported to oneAPI
  - adapted memory management, atomic data access and custom containers
- Currently working on the unit tests and simple examples
- The following have been completed:
  - ✓ **test_atomic.cu** - unit test for atomic operations.
  - ✓ **test_ranluxpp.cu** - unit test for RANLUX++ (random number generator)
  - ✓ **test_track_block.cu**
    → container for simulated tracks - unit test for BlockData concurrent container
  - ✗ some cuda examples still do not work as expected

Project on github: https://github.com/dosarudaniel/OneAdePT

## Summary

- Particle transport simulation is generally not well suited for running on hardware accelerators such as GPUs
  - However, some aspects of simulation, such as simulation of developing showers in the calorimeters , could make use of GPUs:
    - very time consuming
    - require only a subset of physics processes to be ported to GPU

- AdePT project is developing a demonstrator of calorimeter simulation CUDA-C++
  - targets NVIDIA GPUs
  - example allowing physics validation between AdePT and Geant4 should be ready in April

- Current work is focused on porting unit tests and simple examples to oneAPI

- Source code: https://github.com/dosarudaniel/OneAdePT
- AdePT: https://github.com/apt-sim/AdePT

## Thank you!

Technical Student: Daniel-Florin Dosaru

Supervised by: Predrag Buncic (CERN) and Prof. Dr. James Larus (EPFL)

## Current status

| Test number | Description | CPU | NVIDIA GPU | Intel GPU |
|---|---|---|---|---|
| test1.cpp | converted (unmodified) test_ranluxpp.cu | ✓ | ✓ | TBD |
| test2.cpp | converted (and modified) test_ranluxpp.cu | ✓ | ✓ | TBD |
| test3.cpp | converted (and modified) cufisher_price_v2.cu | ❌ | ❌ | TBD |
| test4.cpp | simplified cufisher_price_v2.cu example | ✓ | ✓ | TBD |
| test5.cpp | empty program, testing clang++ and the CMakeLists.txt | ✓ | ✓ | TBD |
| test6.cpp | converted (and modified) test_atomic.cu | ✓ | ✓ | TBD |
| test7.cpp | simplified converted (and modified) test_atomic.cu | ✓ | ✓ | TBD |
| test8.cpp | converted (and modified) test_atomic.cu using oneAtomic.h | ✓ | ✓ | TBD |
| test9.cpp | converted and modified  test_track_block.cu | ✓ | ✓ | TBD |