Detector Simulation in SWIFT-HEP

Ben Morgan

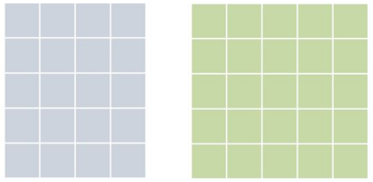
Detector Simulation R&D @ November 2022

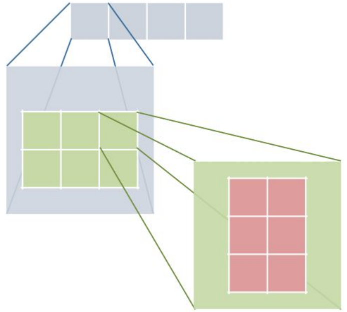
- **AdePT Project (CERN-SFT)**
  - *https://github.com/apt-sim*
- **Celeritas Project (ECP: ORNL, FNAL, Argonne, LBL)**
  - *https://github.com/celeritas-project*
- Regular working and strategy meetings between projects, plus wider engagement through Geant4 and HSF workshops/meetings this year:
  - *VecGeom Evolution Workshop*
  - *HEP Community Workshop on Simulation on GPUs*
  - *AdePT/Celeritas working meeting @ CERN (June/July 2022)*
  - *Geant4 Workshop*
- Can only give a shorter overview today, follow the above links for full details, together with the credits linked in the slides.

Challenges for Monte Carlo on GPUs

- **Execution**: divergence and load balancing
  - *GPUs want every thread doing the same thing*
  - *MC: every particle is doing something (somewhat) different

- **Memory**: data structures and access patterns
  - *GPUs want direct, uniform, contiguous access*
  - *MC: hierarchy and indirection; random access*
  - *Memory allocation is a particular problem*


Structured grid data


Monte Carlo data

Credit: Seth Johnson (ORNL)

Geant4-like Simulation on GPU: the Good

- **Particle transport is embarrassingly parallel**
    - *Tracks are simulated independently → good for GPU simulation*
    - *However, leads to very different tracking than Geant4 (stack based)*
    - *Secondary and stopped tracks need to be handled (changing population)*

- **Many computations and mathematical functions**
    - *Logarithms, square roots, exponential, sin & cos*
    - *GPUs can provide higher throughput for these*

Credit: Jonas Hahnfeld (CERN)

Geant4-like Simulation on GPU: the Bad

- **Monte Carlo simulations governed by random numbers**
    - *Many interactions require rejection-based sampling*
    - *$\longrightarrow$ Thread divergence, bad for performance on GPUs*

- **Geant4 simulates many different particle types**
    - *Many different physics processes and models*
    - *$\longrightarrow$ Thread divergence, bad for performance on GPUs*

- **Divergence also comes from geometry and field propagation**

Credit: Jonas Hahnfeld (CERN)

Geant4-like Simulation on GPU: the Other

- **Cross sections require data lookup by kinetic energy**
  - *Depends on simulation history, which is random*
  - *$\longrightarrow$ No memory coalescing, bad for performance on GPUs*

- **Geant4 almost exclusively uses doubles**
  - *Required in some places – a unit vector must be unit!*
  - *Care must be taken when reducing precision...*

Credit: *Jonas Hahnfeld (CERN)*

Objectives of AdePT and Celeritas

- ***Understand usability of GPUs for general particle transport simulation***
  - *Prototype e+/e−/γ EM shower simulation on GPU, evolve to realistic use-cases*
  - *Focus on EM physics given computational cost in HEP workflows, prior knowledge of applicability of physics models on GPU*
- ***Implement GPU-targeted components for physics, geometry, field, with data models and workflow***
  - *Integrate components in a hybrid CPU-GPU Geant4 workflow ("Fast Sim" approach)*
  - *Off/onload tracks, data to GPU/CPU for specific geometric regions*
  - *Most realistic short-term objective to allow testing/use in existing experiment code*
- **Ensure correctness and reproducibility**
  - *Validate GPU-only, CPU+GPU off/onload against pure CPU Geant4*
- **Understand bottlenecks and blockers limiting performance**
  - *Feasibility and future effort required for efficient simulation workflows on GPU*
- **Celeritas also have a longer term objective to include full hadronic physics**

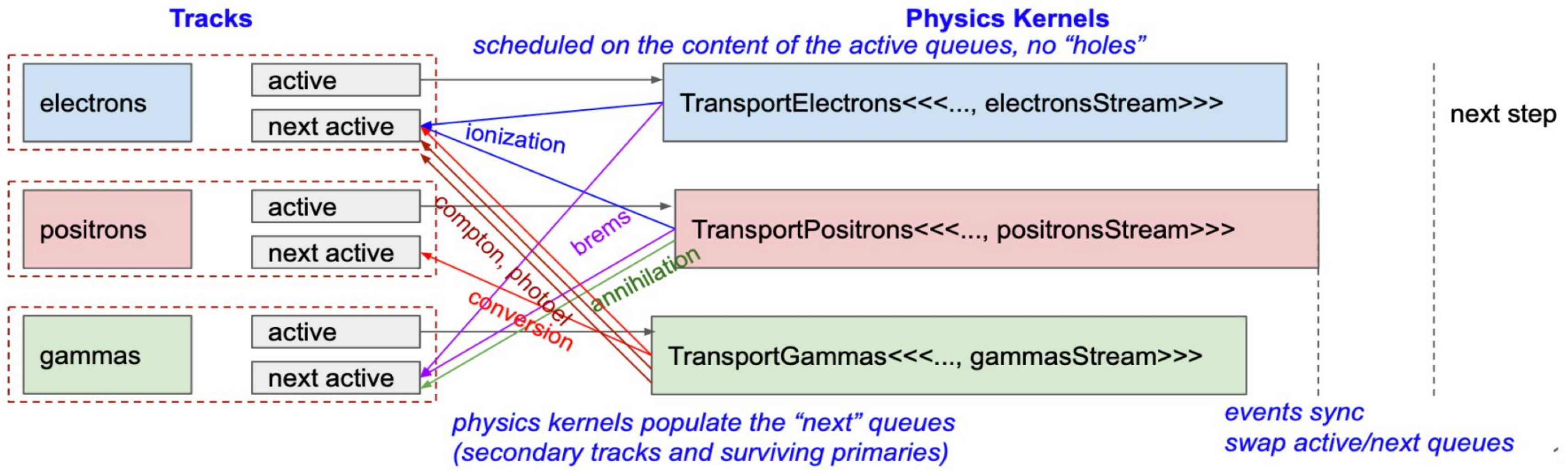
Development Approaches

- **AdePT:**
- Implement features as new examples
  - *Flexibility to explore different directions, optimizations*
- Common functionality built up from successful/existing features
  - *Core types, helper functions*
  - *Geometry: VecGeom library*
  - *Physics: G4HepEM library*
- Portability (non-CUDA) not a major priority in current phase
  - *VecGeom a blocker here*

- **Celeritas:**
- Ground-up approach, core principles
  - *Data-oriented programming*
  - *Composition-based objects*
  - *Revisit older design/impl choices*
- Common functionality:
  - *Geometry: VecGeom or ORANGE*
  - *Physics: Data imported from Geant4, models reimplemented for GPU*
- C++-only execution code, allowing portability to HIP and others
  - *Only for ORANGE geometry*

- **Different approaches with active collaboration allows broad range of designs to be explored in R&D phase - with eventual aim to converge on common solutions**

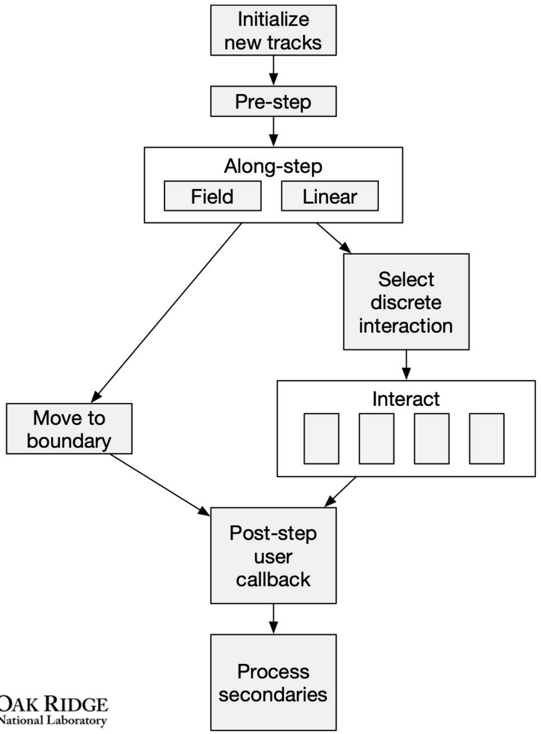
AdePT: Stepping Workflow


Tracks
Physics Kernels
scheduled on the content of the active queues, no "holes"
electrons
active
next active
positrons
active
next active
gammas
active
next active
TransportElectrons<<<..., electronsStream>>>
TransportPositrons<<<..., positronsStream>>>
TransportGammas<<<..., gammasStream>>>
ionization
brems
annihilation
compton, photoel
conversion
next step
physics kernels populate the "next" queues
(secondary tracks and surviving primaries)
events sync
swap active/next queues

- Can start kernels for particle types in parallel streams (transport is independent)
- Synchronization means overhead
  - *Synchronize with host once at the end of the step (stepping loop control)*
- Main optimization playground
  - *Better work balancing between warps, reducing impact of tails, better device occupancy*
  - *Experimenting with smaller kernels (separating discrete and continuous interactions)*

Credit: Witek Pokorski (CERN)

Celeritas: Stepping Workflow

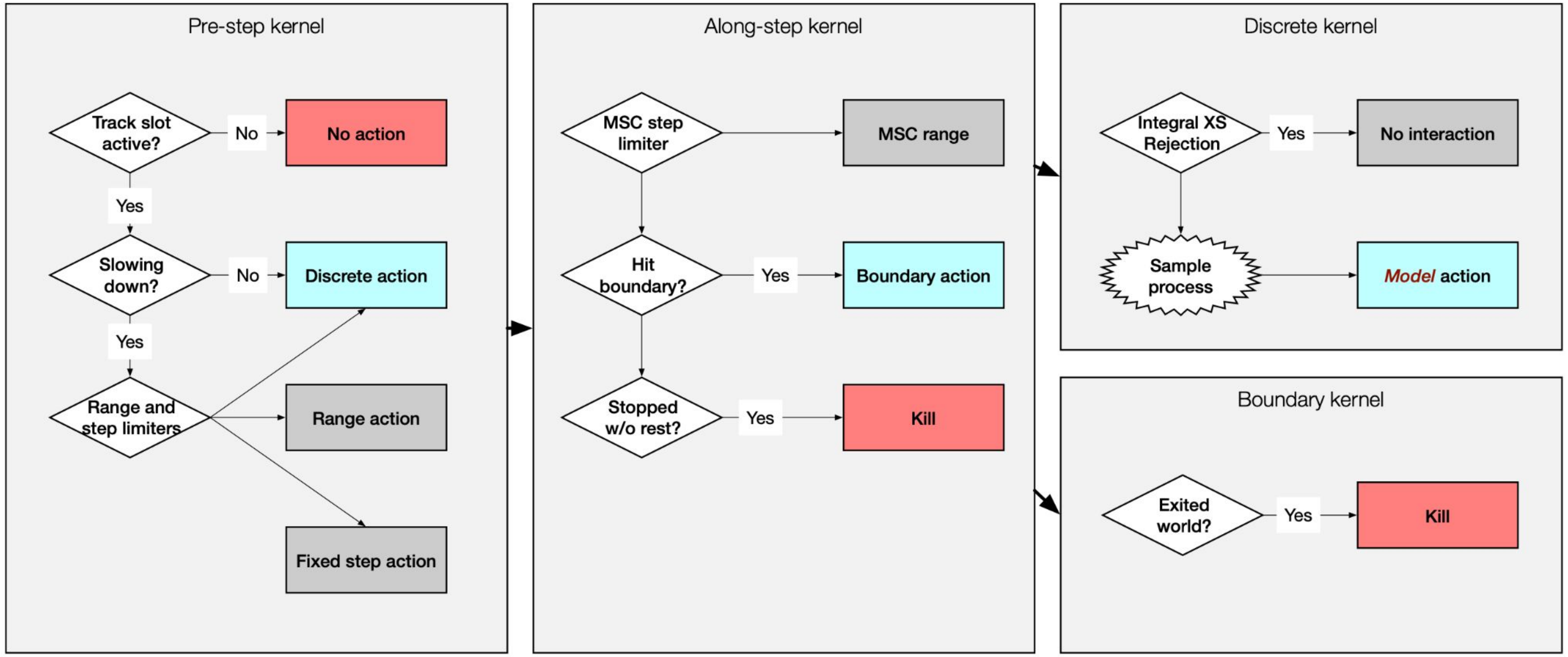


```
extend_from_primaries                 ▷ Copy primaries to device, create track initializers
while Tracks are alive do
    initialize_tracks                                   ▷ Create new tracks in empty slots
    pre_step                               ▷ Sample mean free path, calculate step limits
    along_step                                           ▷ Propagation, slowing down
    boundary                                              ▷ Cross a geometry boundary
    discrete_select                                       ▷ Discrete model selection
    launch_models                      ▷ Launch interaction kernels for applicable models
    extend_from_secondaries               ▷ Create track initializers from secondaries
end while
```

- Action/Event based control flow
- Smaller kernels, each determining next Action, or performing Interaction

Credit: Seth Johnson (ORNL)

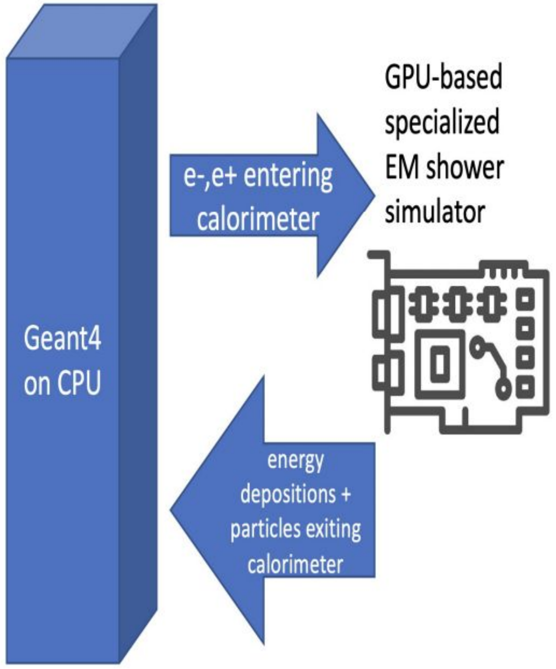
Celeritas: Inside Kernels



Credit: [*Seth Johnson (ORNL)*]

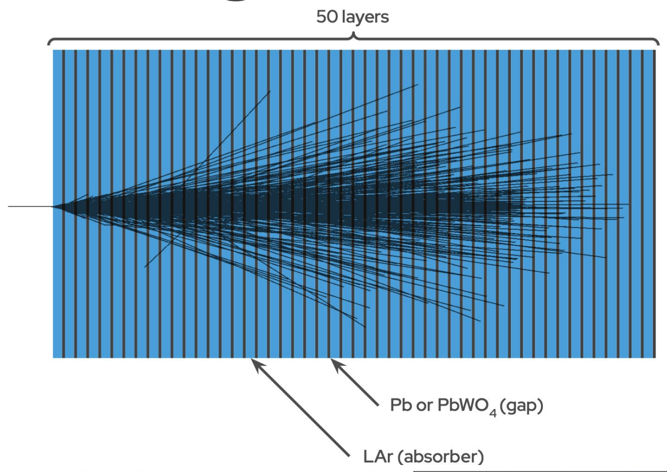
Strategies for integration with Geant4 applications

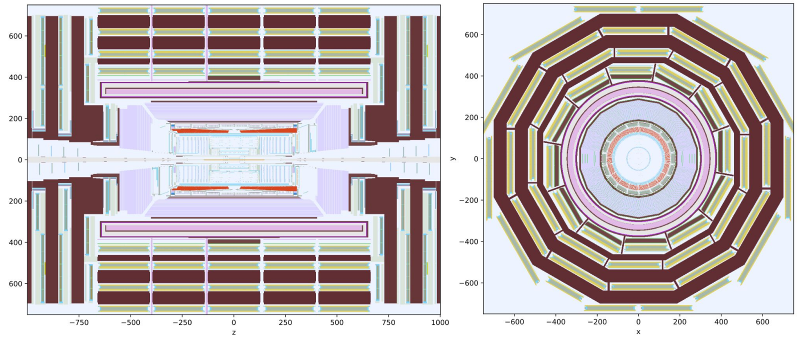
- AdePT and Celeritas only model e-/e+/g physics at present, so cannot be used standalone for simulating a full experiment
- Instead, use them as a “service” to offload those particles to GPU according to preconditions, e.g. when in a specific Region like a calorimeter
  - *Basically the same as “Fast Simulation” methods*
- Several workflows using standard Geant4 hooks under investigation, all with same basic challenges:
  - *Minimizing number/size of on/offload actions*
  - *Allowing user-defined actions, such as scoring/hits, on GPU*
  - *Handing back particles (e.g. exiting particles, hadrons from photonuclear processes) from GPU to CPU*


Geant4 on CPU
e-,e+ entering calorimeter
GPU-based specialized EM shower simulator
energy depositions + particles exiting calorimeter

Progression Problems for Benchmarking

- Both AdePT and Celeritas have adopted two primary test cases for benchmarking and validation
  - *Run on GPU and in CPU+GPU hybrid modes*
- "TestEM3" taken from Geant4 examples as a core test case
  - *50 layer Pb (or PbWO4) / LAr sampling calorimeter*
  - *1-10GeV e- primaries in beam*
  - *Validation, basic scoring and performance measurements*
- CMS 2018 GDML geometry
  - *Same primaries, also HepMC3 input*
  - *Use of more complex workflows, scoring*

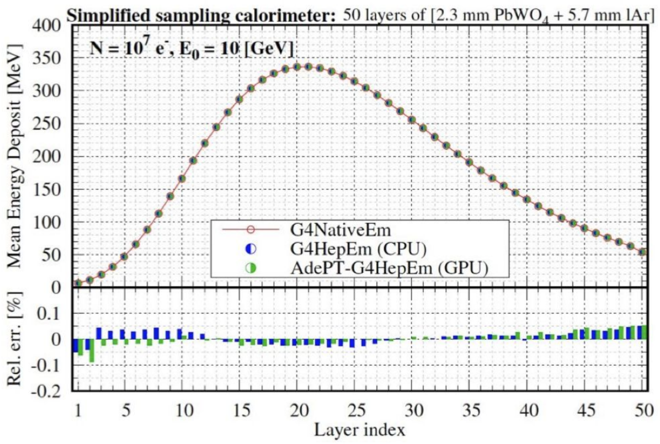




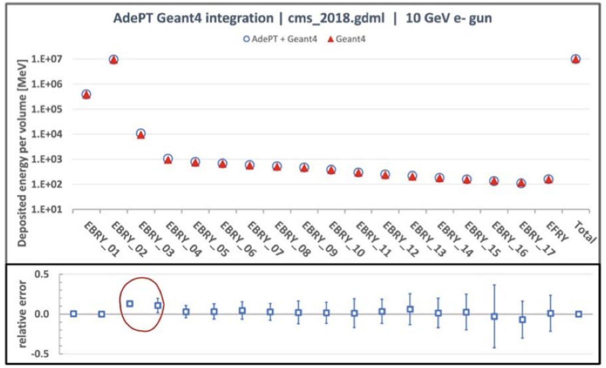
AdePT: Physics Validation

- Validation of G4HepEM against Geant4 standalone is essential
  - *Comparisons to CPU references (in general Geant4-based) done for each added item of functionality*
  - *Both for standalone and Geant4 integration examples*
- EM physics now fully validated
  - *At ‰ level in the sampling calorimeter test case*
  - *Still working on the last bugs/features in the hybrid workflow*

Sampling calorimeter example



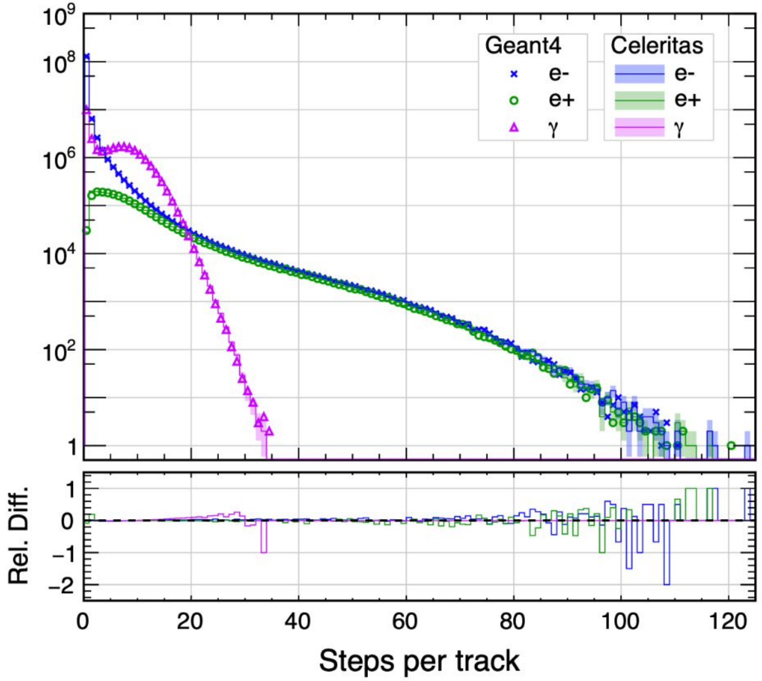
AdePT integration with Geant4



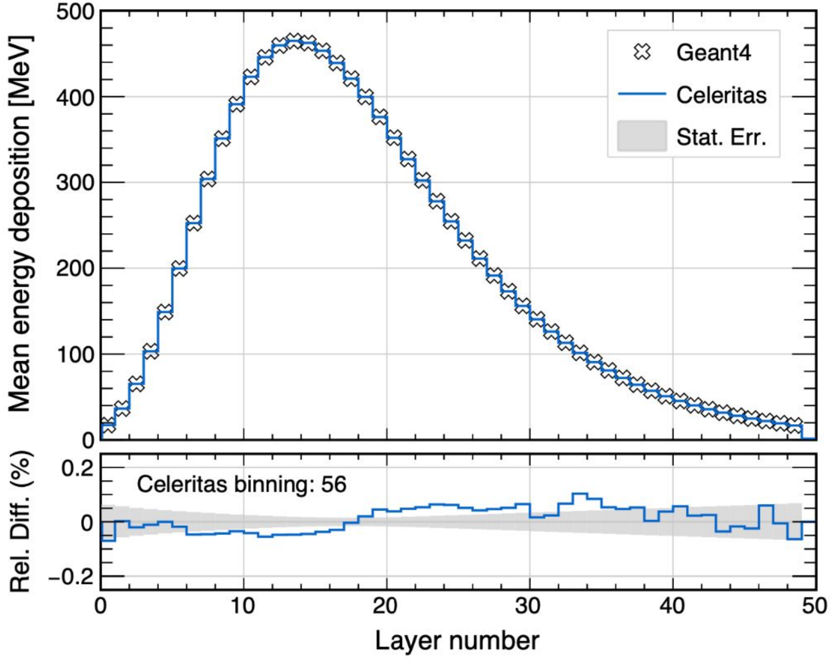
Credit: *Witek Pokorski (CERN)*

Celeritas: Physics Validation

- *No field*
- *No Multiple Scattering*



"Simple CMS" step counts
Isotropic 1GeV gammas

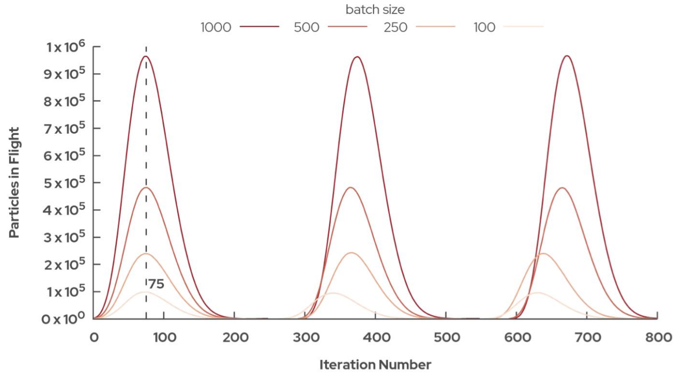


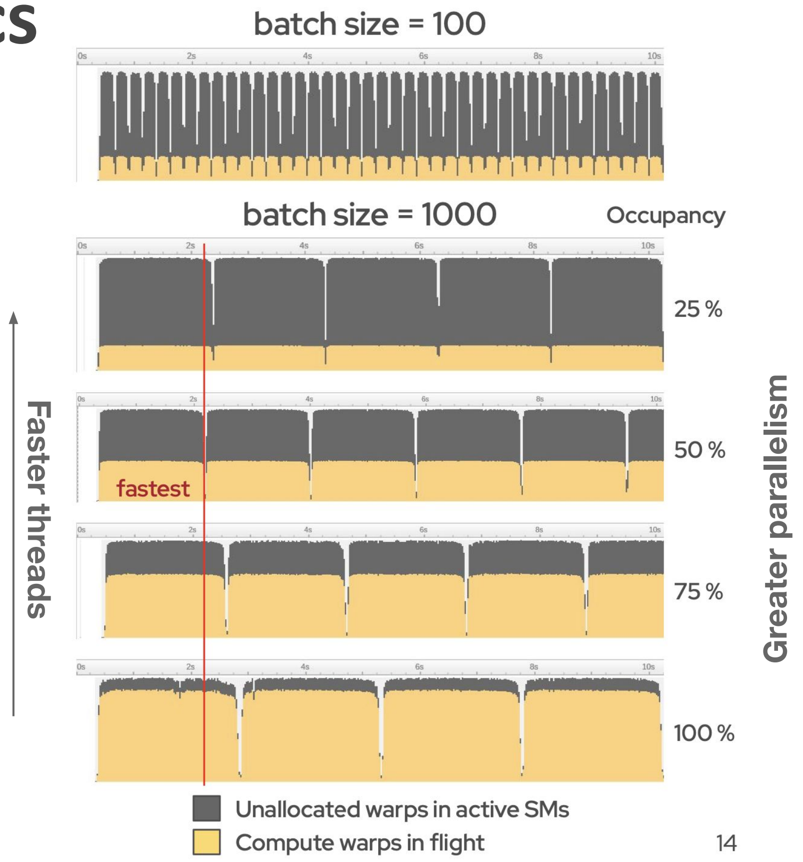
Sampling Calorimeter energy deposition
Monodirectional 10GeV electrons

Credit: [Seth Johnson (ORNL)]

AdePT: Runtime Characteristics

- Configuration space for GPU runs
  - *Number of input particles per batch*
  - *Number of registers per thread*
  - *Number of threads per block*
- Higher batch size => more work per N steps
  - *Limited by available memory and tracks*
- Hints of strategies to fill “work gaps”
  - *More CPU threads, i.e. concurrent events*

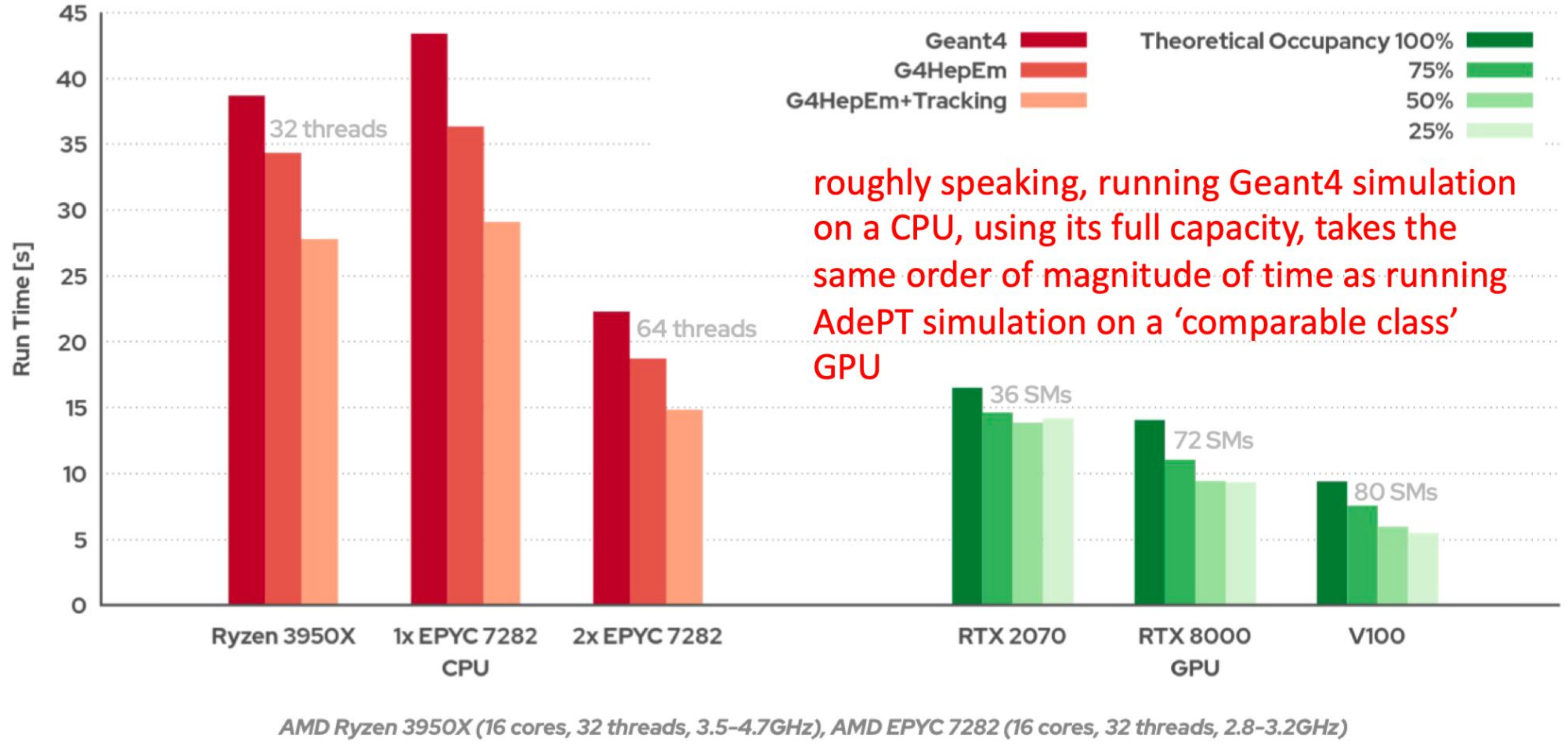




Credit: AdePT Developers

"TestEM3" 50 layer Pb/Ar sampling calorimeter as in Validation plots.

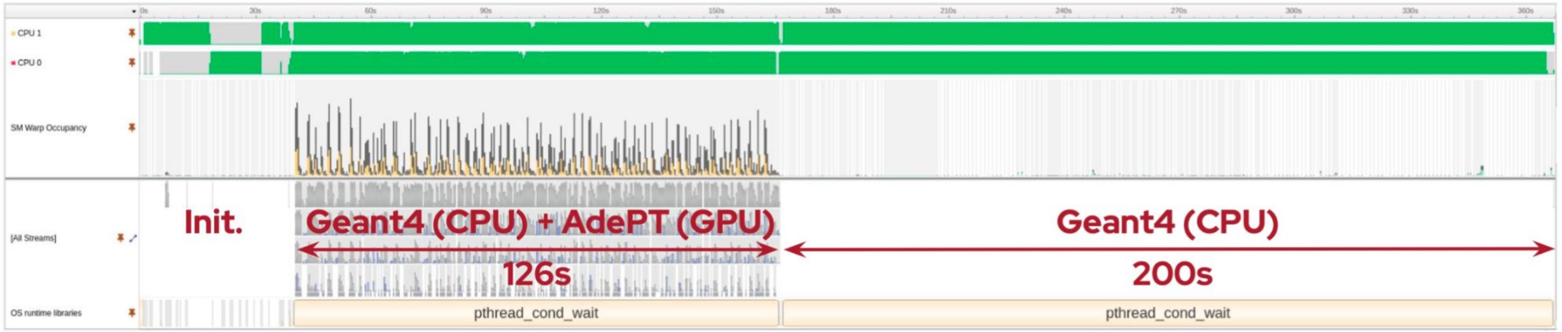
AdePT: Performance Measurements



roughly speaking, running Geant4 simulation on a CPU, using its full capacity, takes the same order of magnitude of time as running AdePT simulation on a 'comparable class' GPU

Credit: [Witek Pokorski (CERN)]

AdePT: Performance of CMS Integrated/Standalone

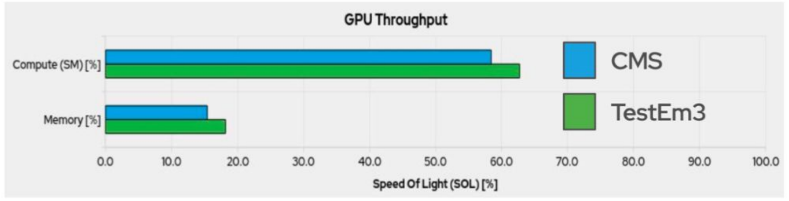


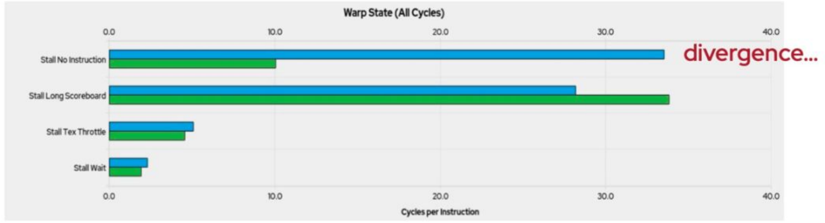
Comparison to Geant4

Above is the timeline of CMS simulation comparing AdePT integrated into Geant4 to Geant4 (Ryzen 3950X, RTX2070), with a speedup of 37% when using 2 CPU threads + 1 GPU vs only 2 CPU threads.

Impact of detector geometry

On the right, 10^6 electrons at 10GeV on Nvidia Tesla V100 with TestEm3 geometry vs the CMS geometry. The total simulation run time for the simplified calorimeter (TestEm3) setup is 549s vs 1455s for the CMS geometry



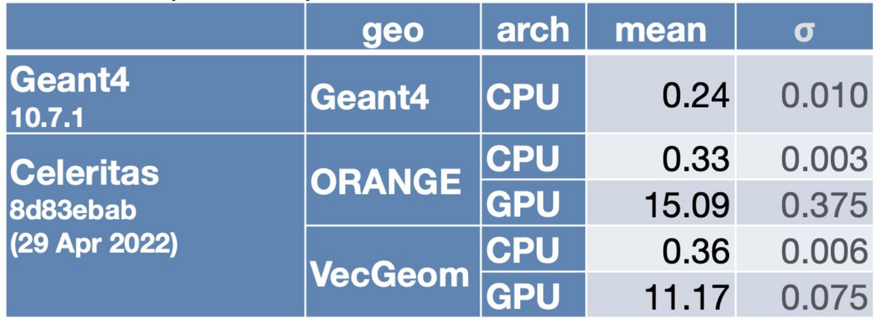


Credit: Witek Pokorski (CERN)

Celeritas: Early Performance Measurements

- TestEm3 — simplified calorimeter
  - *50 alternating layers of Pb and lAr*
  - *10k 10 GeV electron primaries split between 7 events*
- Equivalent configurations of Celeritas/Geant4
  - *No magnetic field*
  - *Disabled multiple scattering, energy loss fluctuations, Rayleigh scattering*
  - *Excludes initialization time*
- No spline interpolation in Celeritas (for now)
  - *~3% performance penalty for Geant4 with spline*
  - *Compensate by using 8× cross section grid points: <2% slower*

Work rate (events/s)



	geo	arch	mean	σ
Geant4 10.7.1	Geant4	CPU	0.24	0.010
Celeritas 8d83ebab (29 Apr 2022)	ORANGE	CPU	0.33	0.003
		GPU	15.09	0.375
	VecGeom	CPU	0.36	0.006
		GPU	11.17	0.075

- *Apples-to-Apples: Celeritas CPU vs GPU*
  - *Power9 CPU, 7 cores*
  - *1xPower9 CPU + 1xV100 GPU*
- *Celeritas 30-45x faster on GPU vs CPU*

Credit: Seth Johnson (ORNL)

Celeritas: New Performance Measurements

- Suite of progression problems - sampling calorimeter, “simple” CMS, CMS 2018 geometries
- 1300 10GeV e- per event, 7 events
- Also testing on AMD, showing similar trends
- Not currently optimized
  - *MSC slows GPU tracking by 2x*
  - *Occasional tracking failures in field*
- Several areas for improvement in workflow identified

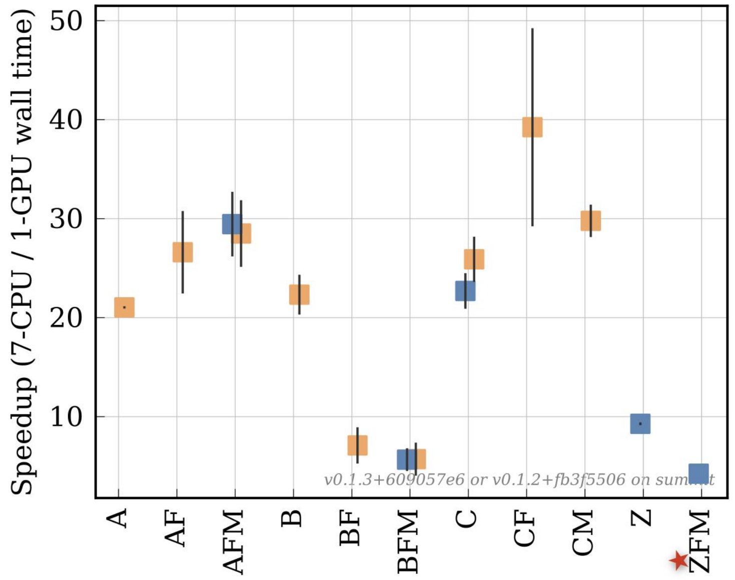
ORANGE
VecGeom

Problem definition

A	testem15
B	simple-cms
C	testem3
Z	cms2018

Modifier

F	+field
M	+msc



Credit: Seth Johnson (ORNL)

Celeritas: Early Performance of Geant4 Integration

- Acceleritas example - uses Geant4 tracking, instead of fast sim, hooks to select/queue tracks for offload to Celeritas
- Basic geometry, physics, primaries setup
    - *SimpleCMS geometry, no magnetic field*
    - *Physics on Host/Device using Geant4 11 FTFP/Standard EM, MSC on/off*
    - *20x Higgs->ZZ events*
- Runs in Intel E5-2650 plus Tesla V100 w/32GB using 1 CPU core, walltime in seconds
    - *Note that measurements are from earlier this year*

PhysicsList	Geant4	G4+Celeritas	Lower Bound	Gain	Max. Gain
Default	206.9 ± 0.9	119.5 ± 0.2	88.1 ± 0.2	1.73 ± 0.01	2.35 ± 0.01
No MSC	177.0 ± 0.8	95.7 ± 0.1	87.7 ± 0.2	1.85 ± 0.01	2.02 ± 0.01

Credit: Soon Yung Jun (FNAL)

Further Progression Problems for Benchmarking

- Ongoing design discussions on additional use cases/setups for validation and performance both on pure GPU and hybrid Geant4+GPU workflows:
  - *Physics models and parameters*
  - *Detector geometries, regions for GPU offload*
  - *Inputs (primaries), Outputs (scoring, hits)*
  - *CPU/GPU hardware, workflow parameters (e.g. CPU threads, GPU tracks in flight, Host/Device memory) - important to measure in realistic setups!*
  - *Performance metrics to measure*
- Need to consider both simple (~“TestEM3” calo) and complex (e.g. “LHC experiment”), and to isolate different areas (e.g. geometry, offload)
- ***Already established links with ATLAS/CMS, further collaboration/ideas from other experiments and community very welcome here!***

Summary

- 2022 has seen major progress in Simulation R&D for EM physics on GPU
- AdePT and Celeritas have demonstrated feasibility of approach
  - *Near full EM physics validated*
  - *Initial examples of Geant4 CPU offloading EM particles to GPU implemented/profiled*
- **Many avenues for optimization identified, most significant being GPU friendly geometry modeling and navigation**
- **Feedback and contributions on GitHub welcome!**

WARWICK
THE UNIVERSITY OF WARWICK