Second Look at Performance of TestEm3 in AdePT

Jonas Hahnfeld

April 20, 2021

Recap: Performance of TestEm3

- TestEm3:
  - Simplified sampling calorimeter, 50 layers (2.3 mm $PbWO_4$ + 5.7 mm lAr)
  - No magnetic field; 10,000 electrons of 10 GeV
- System: AMD Ryzen 9 3900 (12C/24T), GeForce RTX 2070 SUPER

- GEANT4-10.7.1 (1 thread): 497 seconds (G4HepEm: 489 s)

- AdePT (GPU): 115 seconds (default batch size of 26 particles)

- GEANT4-10.7.1 (24 threads): 43 seconds (G4HepEm: 43 s)

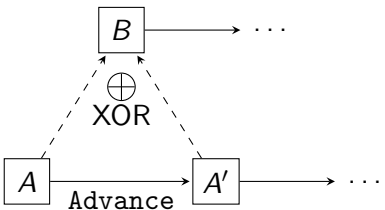
Random number generator

- Profiling points to RNG
  - More concretely: advancing RNG state for secondaries

Before apt-sim/AdePT#114

```
// Initialize a new PRNG state.
this->rngState = parent.rngState;
this->rngState.Skip(1 << 15);
```

Branching the RNG state



Input: state A
Output: state B for the secondary

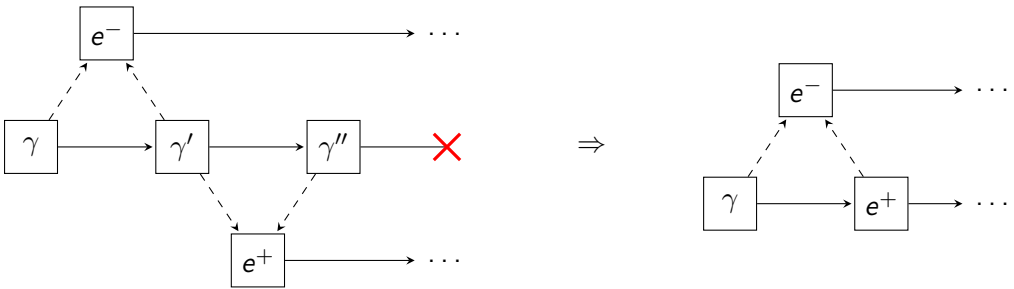
0. Remember the state A
1. `Advance` to state A'
2. XOR bits in A and A' to get B

Run time for 10,000 electrons: 115 s → 36.7 s (3.1x)

Internals of RANLUX++

- LCG with 576 bits of state
  - Expensive operation: advancing the state
  - Then: 11 `doubles` "for free"
- Before: threads advanced state when all bits used up
  - Could happen at various places during physics
  - A few threads would advance their state, all others had to wait
- Now: advanced original and branched state offer 11 `doubles` "for free"
  - Less (if any) expensive operations during physics

More Optimizations: Reuse state of killed primary particle



- Avoids one branching operation for annihilation and conversion
- Run time for 10,000 electrons: 36.7 s $\rightarrow$ 35.6 s

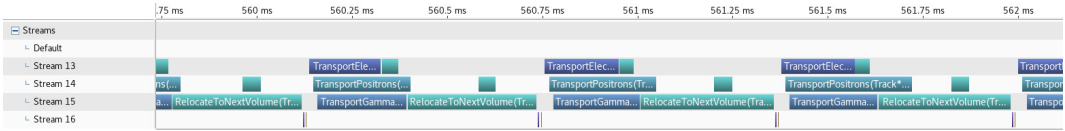
More Optimizations: Reduce thread divergence

- Result: Need at most one branched RNG state

$\Rightarrow$ Move operation before starting discrete interaction
  - Threads still synchronized before `switch` statement
  - Run time for 10,000 electrons: 35.6 s $\rightarrow$ 33.7 s

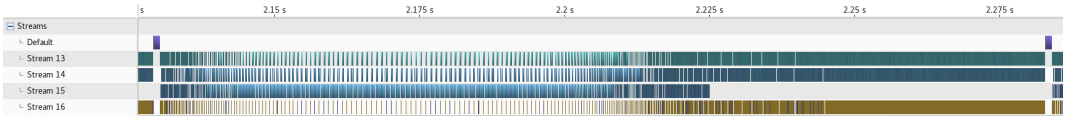
Taken together, another improvement of around 8 %
(via `apt-sim/AdePT#117`, compared to slide 4)

More Profiling: Geometry again



- `Transport` kernels are much faster compared to last presentation
- Now bound by relocation for gammas, even for this simple geometry
  1. Avoid virtual calls: 26.8 s (-20 %, see `apt-sim/AdePT#106`)
  2. Without separate kernel: 20.7 s (-38.5 %) – only for this particular case

Turning on a Magnetic Field



- Wait for looping particles, until hitting 1000 iterations
  - Need to detect these cases and deposit energy
  - Preferably not per particle on the GPU, but on the CPU
  - Example heuristic:
    1. No gammas, only charged particles (e^-/e^+)
    2. No change in number of tracked particles for 200 steps

Conclusion

- Optimized performance of TestEm3 on GPU by a factor of 3.4x
  - Now faster than GEANT4: 33.7 s vs 43 s !
  - But: GEANT4 computes safety, not included on the GPU (needed for MSC)

- Next steps:
  - Profile execution with realistic calorimeters (ATLAS / CMS)
  - Study energy distribution of particles entering the calorimeter