Impact of Advanced Computing Architectures on Nuclear Data Needs

Paul K. Romano
Computational Scientist, Argonne National Laboratory

Workshop for Applied Nuclear Data Activities (WANDA 2023)
March 1, 2023

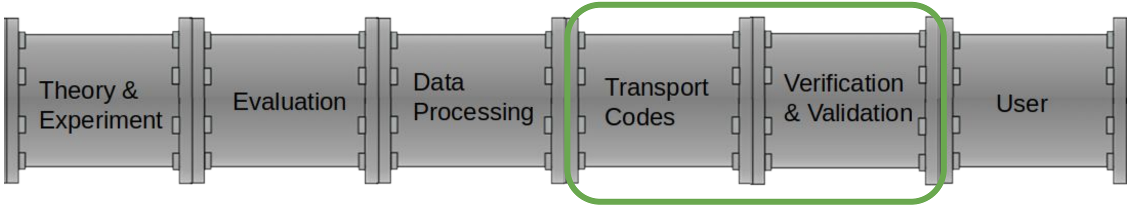
Argonne NATIONAL LABORATORY

My Background

- Joint appointment in Computational Science (CPS) and Nuclear Science and Engineering (NSE) divisions at ANL

- Focus area lead for computational particle transport: multiphysics, reactor design/analysis, fusion neutronics, high-energy physics, and nuclear data

- **Nuclear data tie-in**: Project lead for the OpenMC Monte Carlo code



- **Computing tie-in**: Working under Exascale Computing Project on porting high-fidelity coupled neutronics (OpenMC) and CFD (Nek5000)

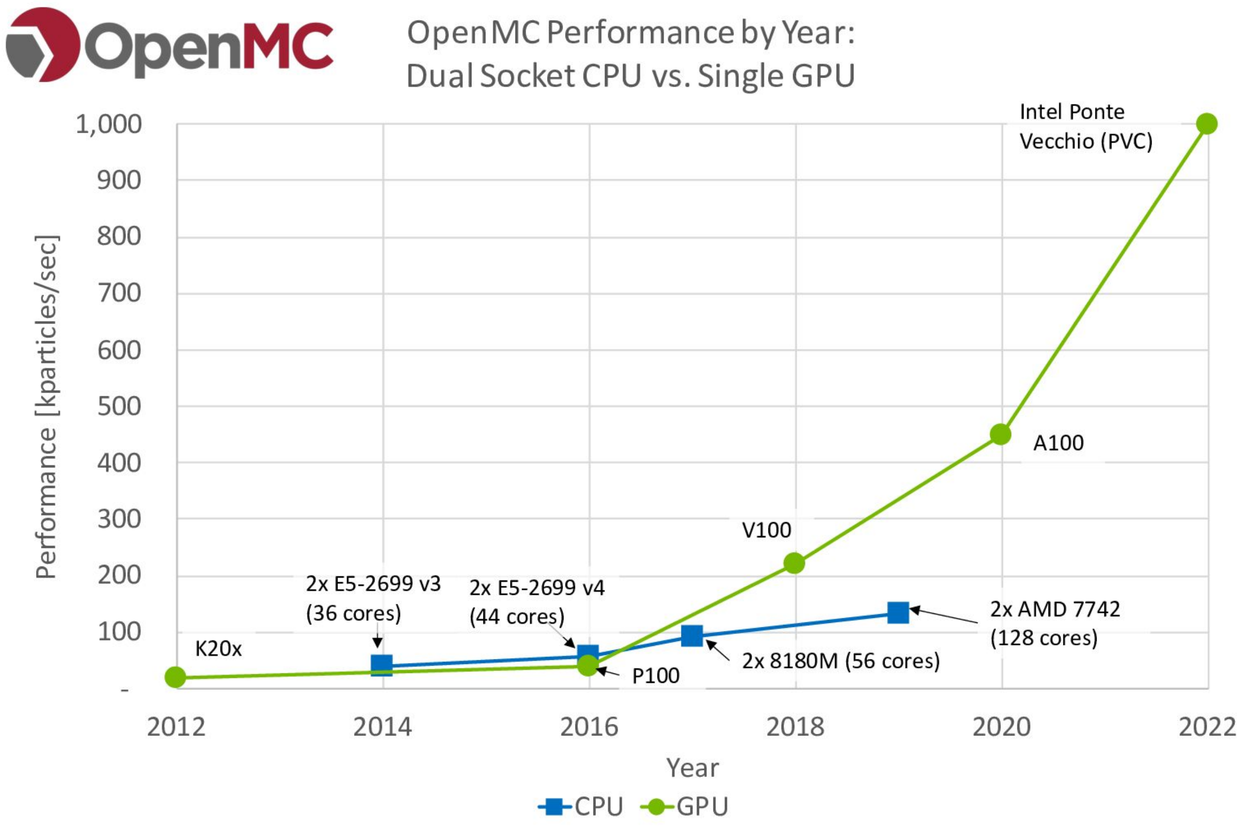
Computing Architectures

- While the outlook on next-generation architectures was unclear 5–10 years ago, we are now very clearly living in the age of GPUs, aided in the US by the efforts under the Exascale Computing Project

- 7 of the current top 10 supercomputers are based on GPUs
  - Applications that are not able to take advantage of GPUs are missing out on most of the performance potential

- While use of GPUs for scientific simulation has been driven by large machines, benefits filter down to smaller GPU-based systems as well

OpenMC Performance: CPU vs GPU



Argonne NATIONAL LABORATORY

Inactive batch performance on HM-Large reactor w/depleted fuel

EXASCALE COMPUTING PROJECT

1

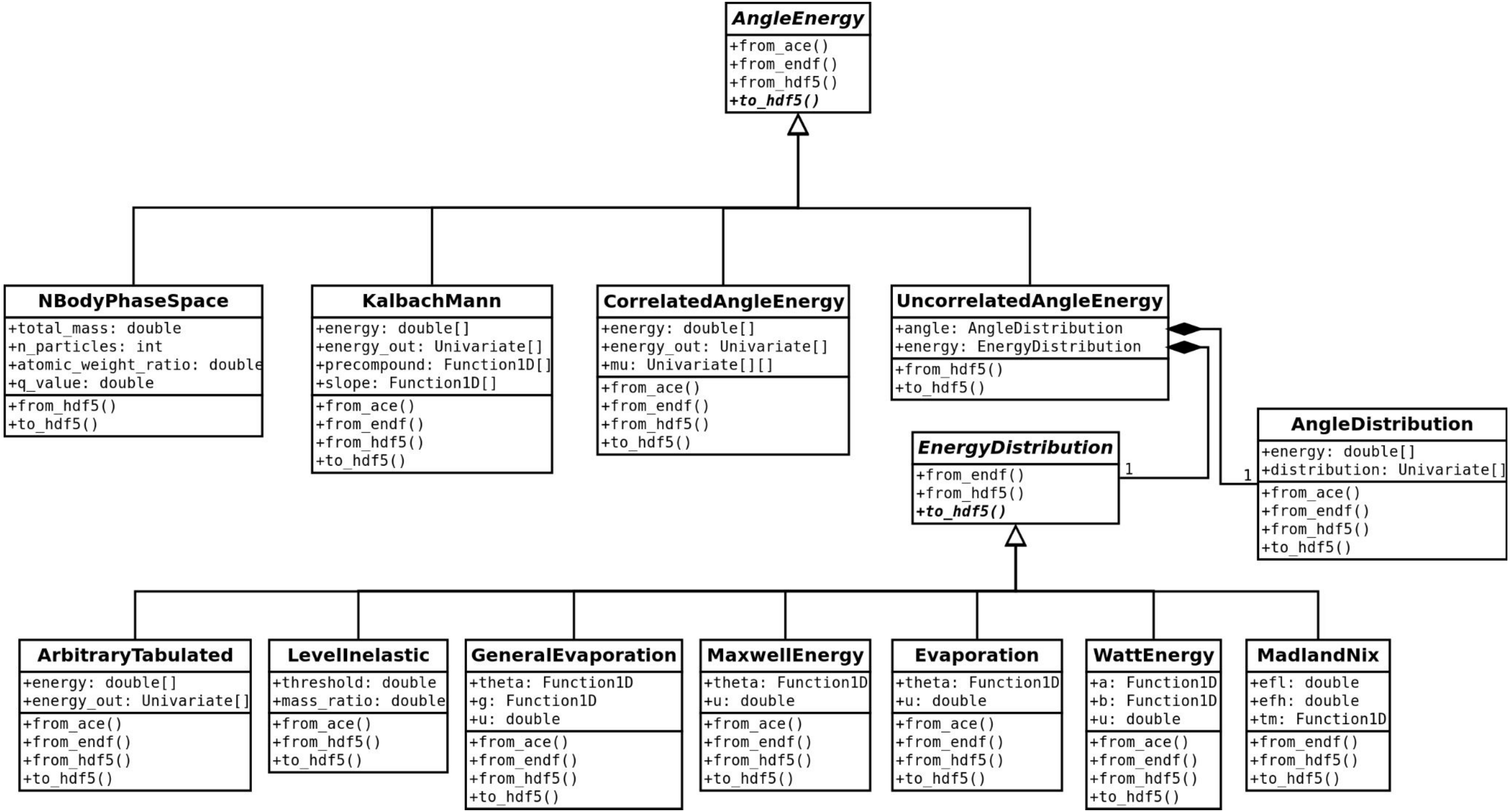
What have we learned?

- Programming for GPUs is not easy: managing multiple memory spaces, immature programming models, immature tools, immature hardware, debugging/performance, lack of virtual tables (polymorphism)

- GPUs are inherently not well-suited for Monte Carlo (heavy branching logic, random memory access)
  - However, CPUs are equally bad!

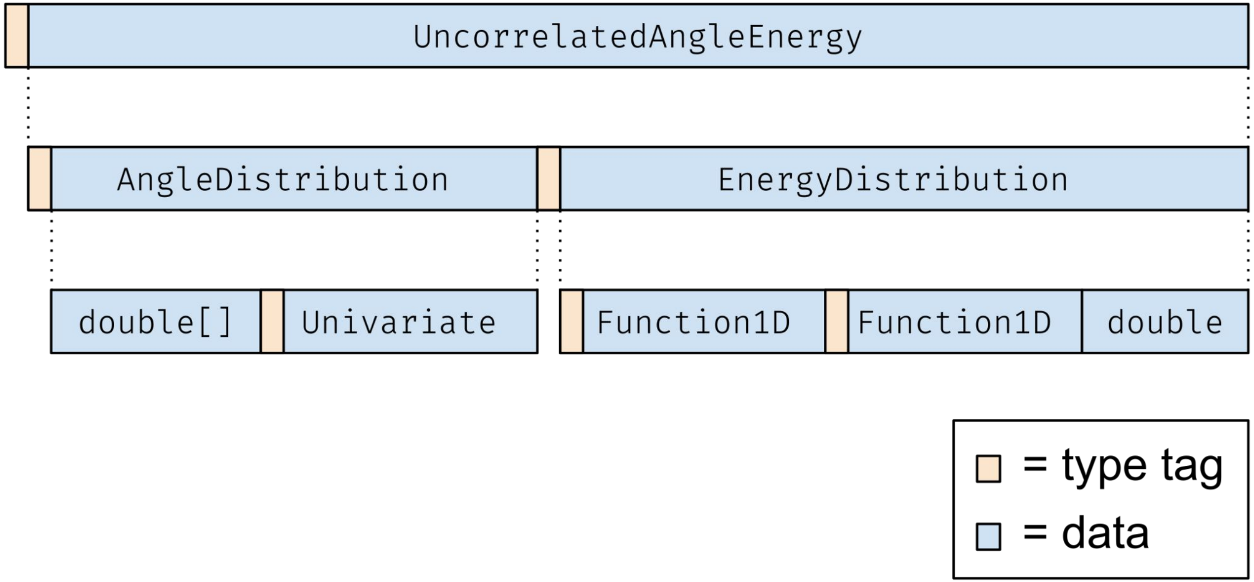
- With respect to nuclear data, the primary practical difficulty is the use of <u>complex, nested data hierarchies</u>

Data hierarchy: polymorphism



Data hierarchy: flattened classes

- Normally, nested classes handled by "pointer chasing"
- On GPUs, all data ends up being flattened into single opaque array
- Virtual tables replaced by switch statements



Data hierarchy: “better” solutions

- Just wait? — compiler vendors and programming models may eventually handle runtime polymorphism

- Nuclear data processing codes could provide “uniform” outputs
  - Simplifies end use at cost of higher memory

- Move virtual dispatch to the host, finer-grained GPU kernels (Celeritas)

- Machine learning models could provide parameterized forms of distributions?

Other future directions

- On GPUs and other data-parallel architectures, strong incentive to use less memory and more FLOPs

- Model-based physics is attractive for Monte Carlo transport simulations
  - For example, windowed multipole data for resolved resonance range
  - Fission event generators (FREYA, CGMF, GEF, etc.)
  - Ideally, better physics *and* better performance

- Integration of libraries in GPU-enabled code adds more complexity

Thank you!