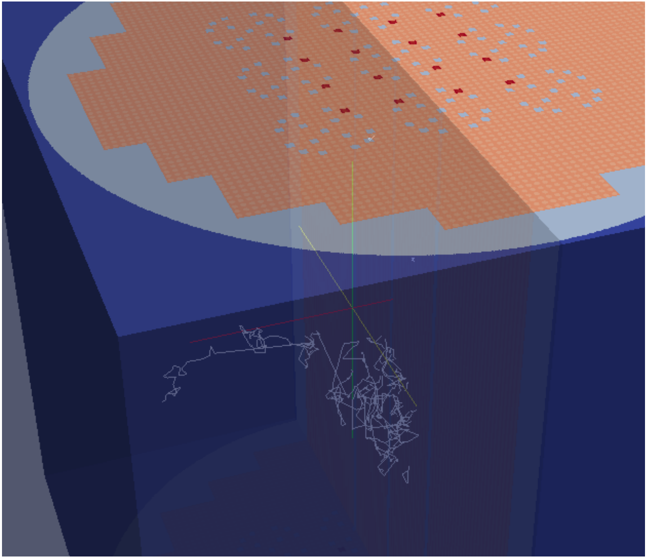
OpenMC

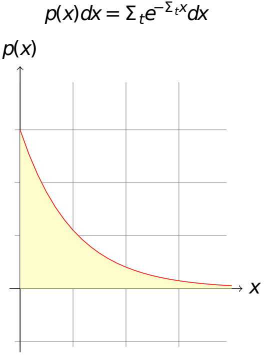
Workshop

Sterling Harper, Travis Labossiere-Hickman,
Luke Eure, Patrick White,
and Benoit Forget

April 6 2016

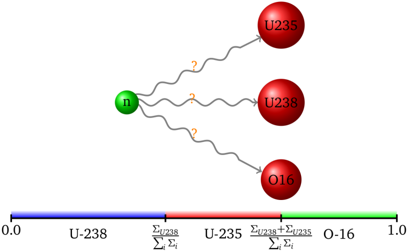
Monte Carlo simulates the movement and reactions of individual neutrons



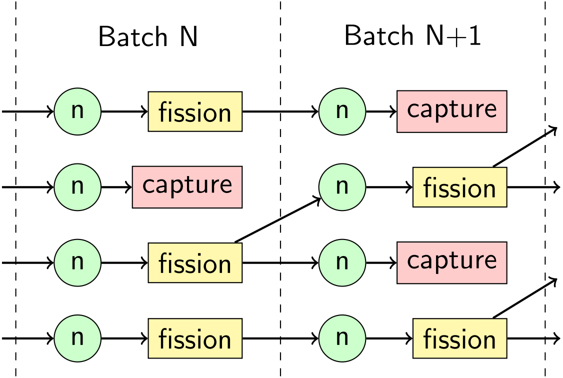


Monte Carlo randomly samples continuous distributions

For example, we sample an exponential to determine how far a neutron travels before the next collision



We compare cross sections to determine which nuclide a neutron collides with and which reaction happens

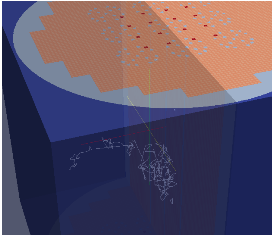


MC simulations are divided into batches for statistical reasons

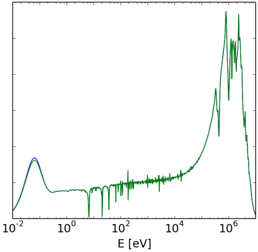
Neutrons are born from fission sites in the last batch

Monte Carlo is precise because it uses

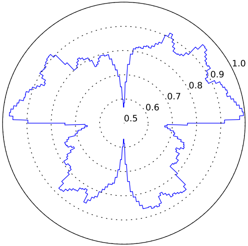
3D Space



Continuous Energy



Continuous Angle



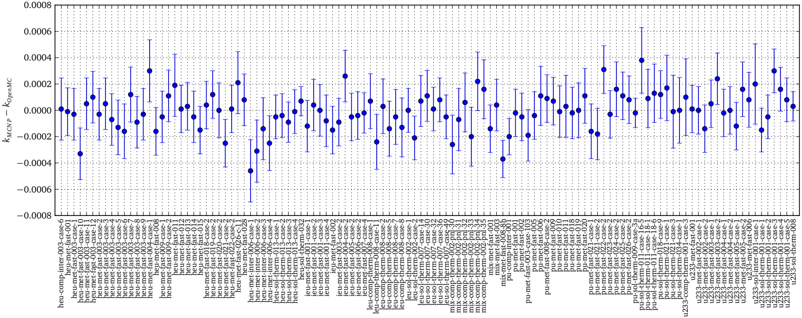
Jargon:

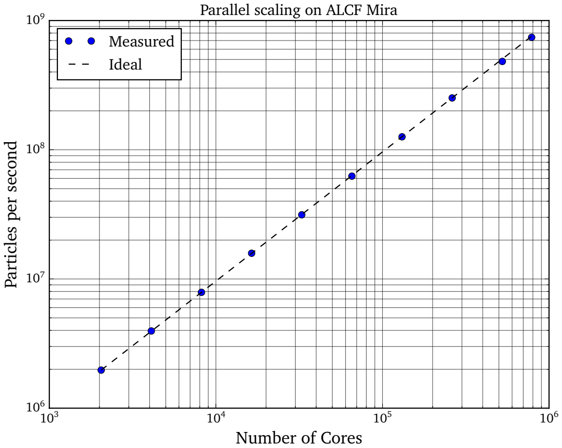
"Tallies" are results from an MC calculation

e.g. k_{eff}, flux, ^{238}U capture rate

OpenMC is validated against MCNP Criticality Benchmark Suite

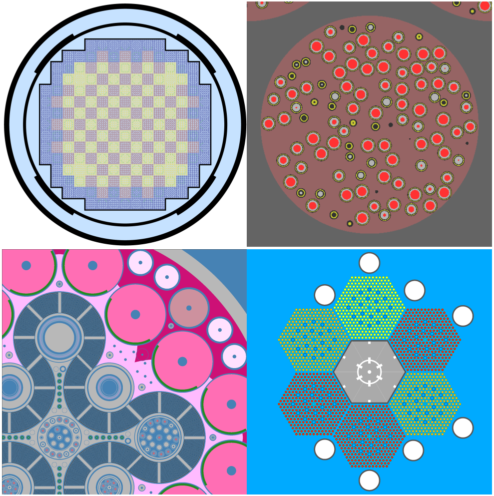
117 configurations with different spectra, materials, enrichment





OpenMC scales linearly up to ∞ processors

(786,000 cores, 3,150,000 threads on Mira supercomputer)



OpenMC can model a wide variety of geometries, for example:

Full-core PWR

TRISO particles

ATR

What makes OpenMC special is its Python-powered input generation and post-processing

```
>>> import openmc
```

OpenMC is 46,000 lines of F90 and 46,000 lines of Python

The OpenMC workflow:

1. Write Python code describing the problem.
2. Use `.export_to_xml()` to create XML files.
3. Run OpenMC (using Python or shell). This creates `tallies.out` and `statepoint.h5` output files.
4. Read `tallies.out` with a text editor or read `statepoint.h5` with Python.