Fermilab U.S. DEPARTMENT OF ENERGY Office of Science

# Achieving Perlmutter Readiness for the Synergia Accelerator Modeling Framework with Kokkos

Eric G. Stern, Qiming Lu, Marc Paterno, James Amundson

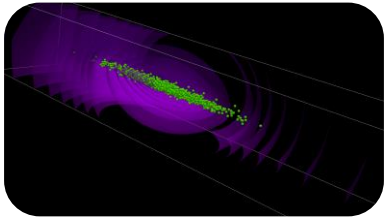
NUG2020 Lightning Session

17 Aug 2020



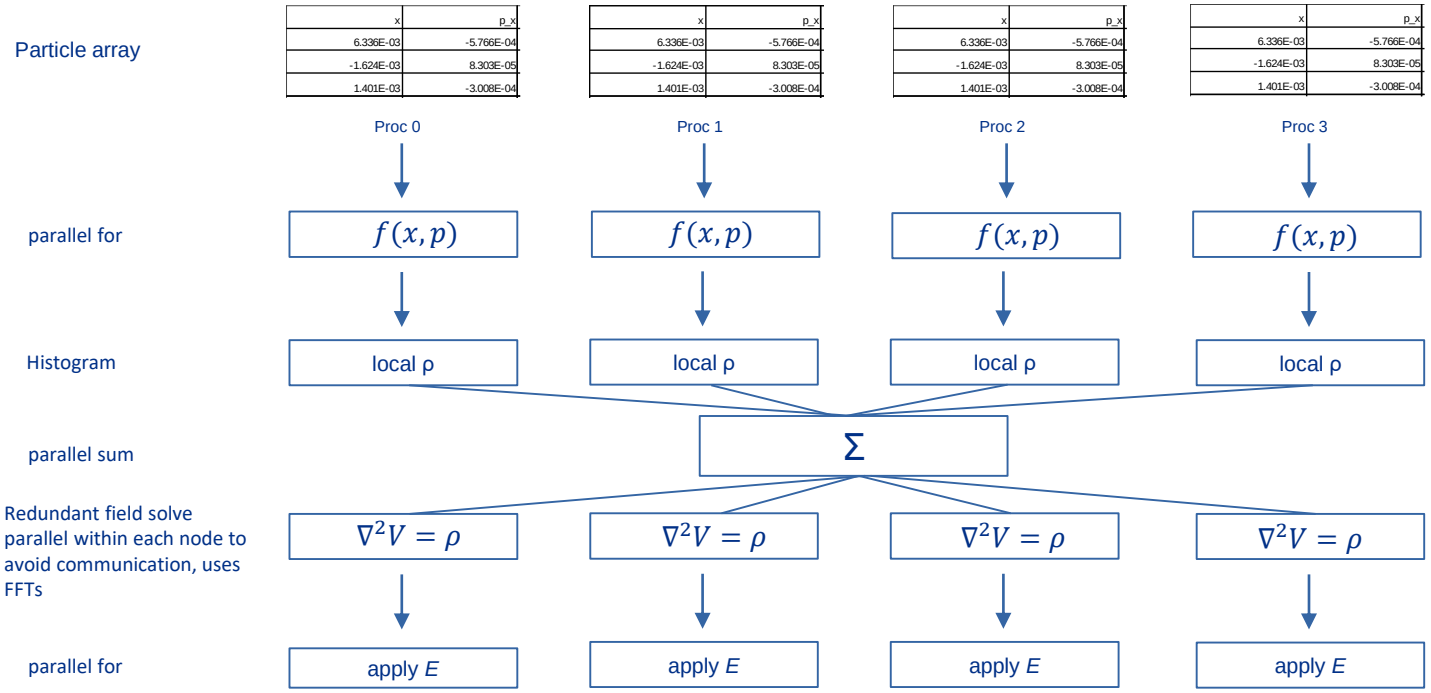
FERMILAB-SLIDES-20-108-SCD

# Synergia Modeling Framework



- C++ library with Python wrappers to perform particle-in-cell accelerator simulations.
  - Most simulations are written in Python and import modules to perform the heavy calculation. Main processing loop is in C++.
- Uses MPI/OpenMP parallel processing to scale to large problems.
- Runs on Desktop/laptop, small/medium clusters, supercomputers including NERSC/Cori.
  - Small problems can be run on small systems (number of particles, size of accelerator, etc.)
  - Code scales well for large problems on large systems.

# Synergia computational ingredients






Fermilab

# New systems will not work like the old ones

The bulk of computing power at the new large facilities will be provided by some kind of GPU or other co-processor. Synergia must adapt!

What do we want for an upgraded Synergia?

- Keep broad accessibility across computing platforms.
- Use “standard” languages and programming techniques as much as possible.
- Avoid architecture lock-in for code maintainability and execute-everywhere capability.
- Minimize architecture specific code and algorithms.
  - (a previous CUDA specific version was unmaintainable and rotted into uselessness)

# kokkos

- Part of the Exascale Computing Project (`https://kokkos.org`).
- Supports both regular CPUs and CPUs with attached co-processors (GPU, etc.) employing back-ends, e.g. OpenMP or CUDA.
- Hardware agnostic: supports NVIDIA (now), AMD and Intel GPUs (promised)
  - Perlmutter nodes will use the NVIDIA A100 GPU
- Templated C++ library maintained as an open-source repository on Github.
- Abstracts array data as a *View* that may reside either in host or device memory.
- Application invokes parallel dispatchers that connect to the device specific backend to implement parallel operations.
- For Synergia, after initial setup that requires CPU resources, data resides in device memory.

# Benchmark accelerator simulation results

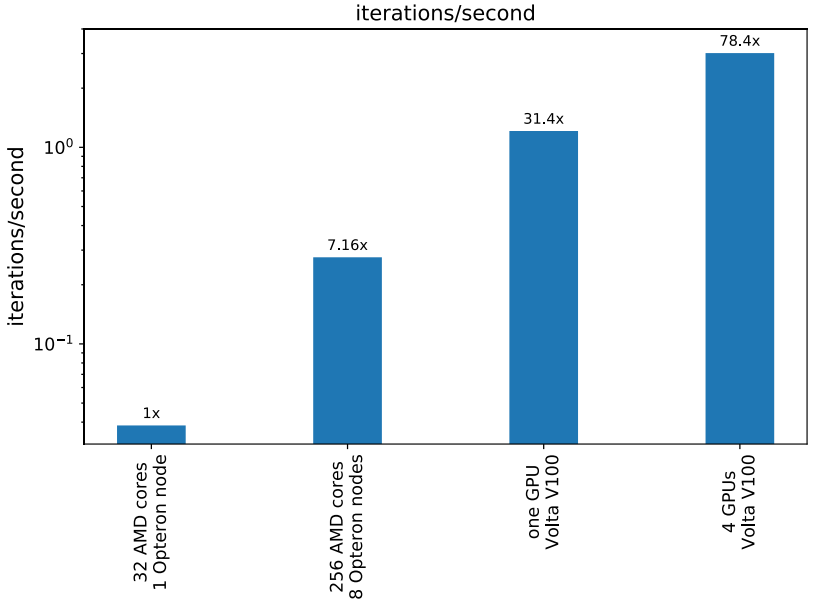
After about a year…

- Comparison between PowerPC + one or four 5120 thread NVIDIA V100 GPUs (similar to Summit nodes)

  and

  - 1 or 8 AMD 32 core Opteron nodes

- Perlmutter will have next generation A100 GPUs



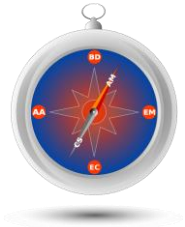



Fermilab

# Caveats

- It took a year of work.
- Some algorithms and operations would not operate efficiently on attached processor memory and had to be redesigned for the GPU.
- Some device-specific code was necessary:
  - We had to rely on a CUDA specific FFTW implementation for GPUs.
  - The Kokkos primitive used for charge deposition is not efficient on CPUs.

# We are ready for Perlmutter!!



Synergia development was supported by the DOE SciDAC-4 ComPASS project.

Work supported by the Fermi National Accelerator Laboratory, managed and operated by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy.