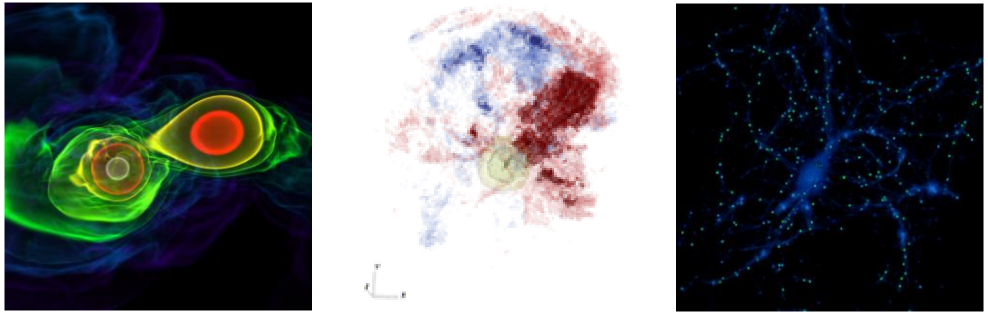
ADAPTIVE MESH REFINEMENT FOR EXASCALE

Max Katz, August 23, 2017

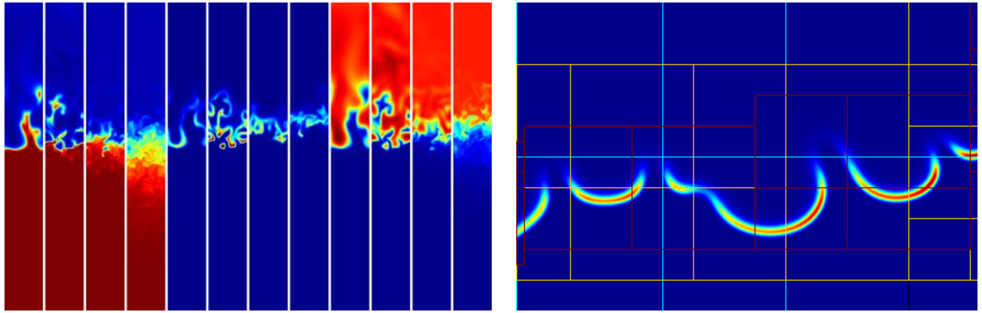
NVIDIA

AMR-ENABLED SCIENCE

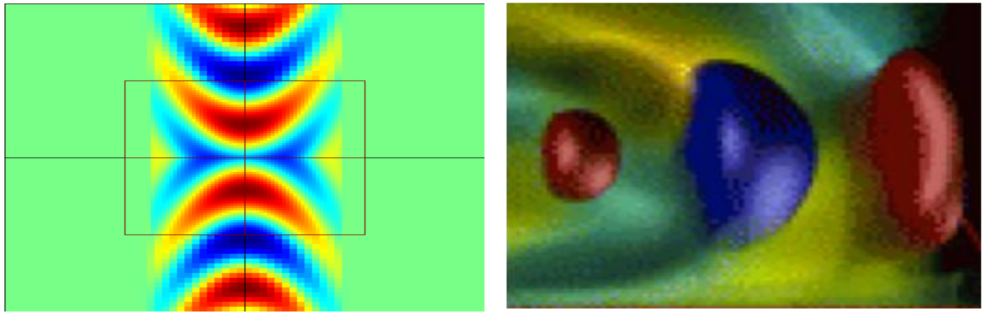
ASTROPHYSICS AND COSMOLOGY



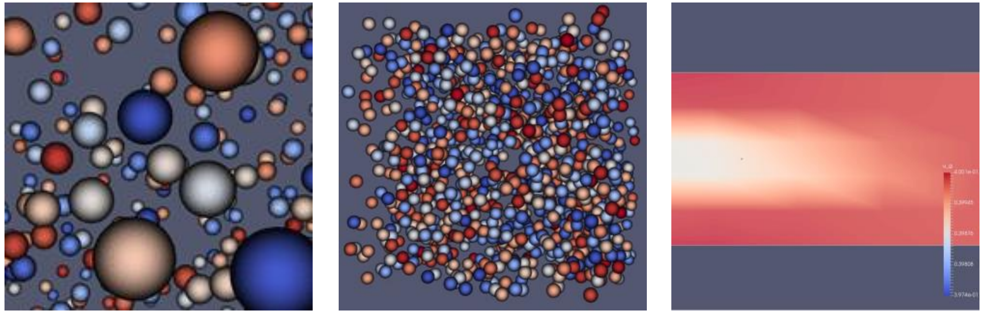
COMBUSTION



ACCELERATORS



MULTIPHASE FLOW



AMREX

Center for Computational Sciences and Engineering

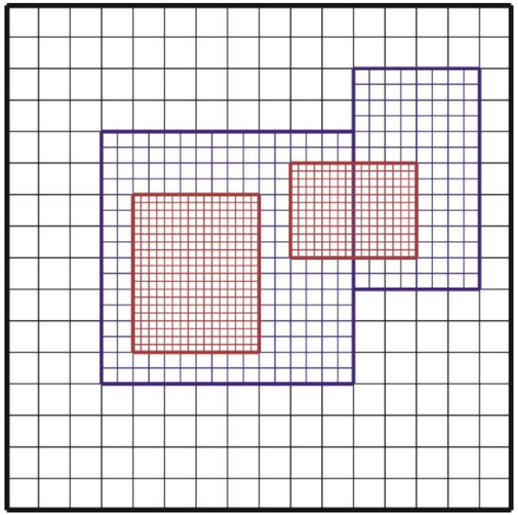
Berkeley Lab

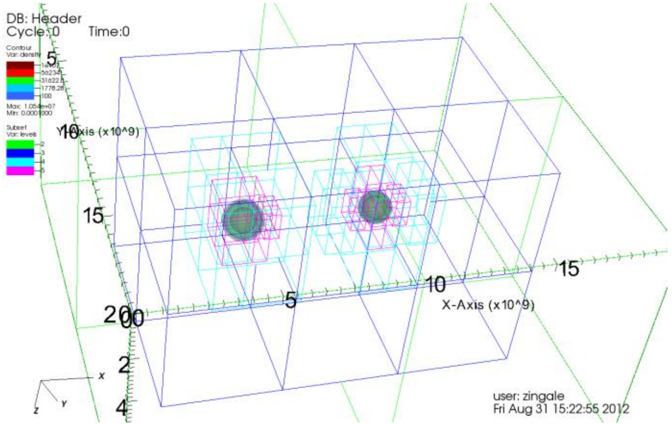
Block-structured AMR

Solution state defined on hierarchy of levels

Levels are unions of (logically) rectangular grids

Grids are dynamically adjusted



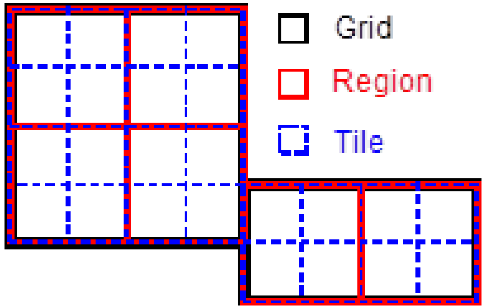


Data is in the form of

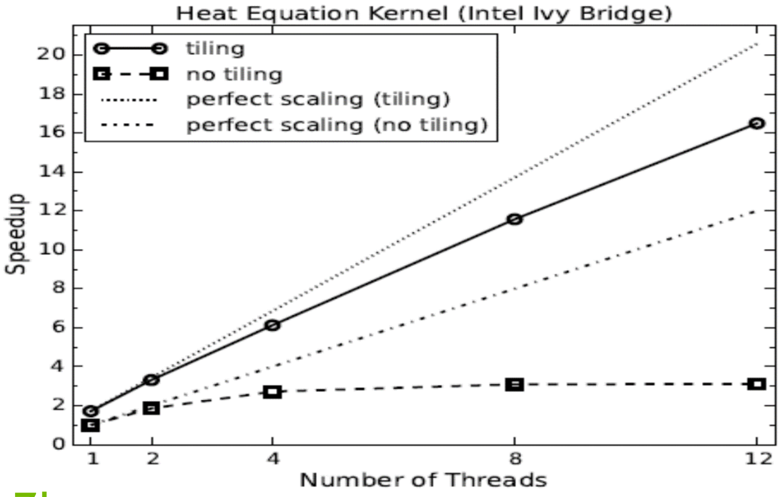
Zone center/edge/corner data

Particles

ON-NODE CPU PERFORMANCE



- Standard approaches for per-core performance
  - Vectorization, code transformation, autotuning
- Logical tiling used to improve serial, parallel performance
  - Data layout unchanged; different unit of work
  - Tiling is hidden in iterator over mesh patches
  - Leads to more efficient memory access



Data provided by Weiqun Zhang

AMREX GRID ITERATOR APPROACH

```
for (MFIter mfi(state); mfi.isValid(); ++mfi)
{
    box = mfi.tilebox();
    local_state = state[mfi];
    evolve(local_state, box);
}
```

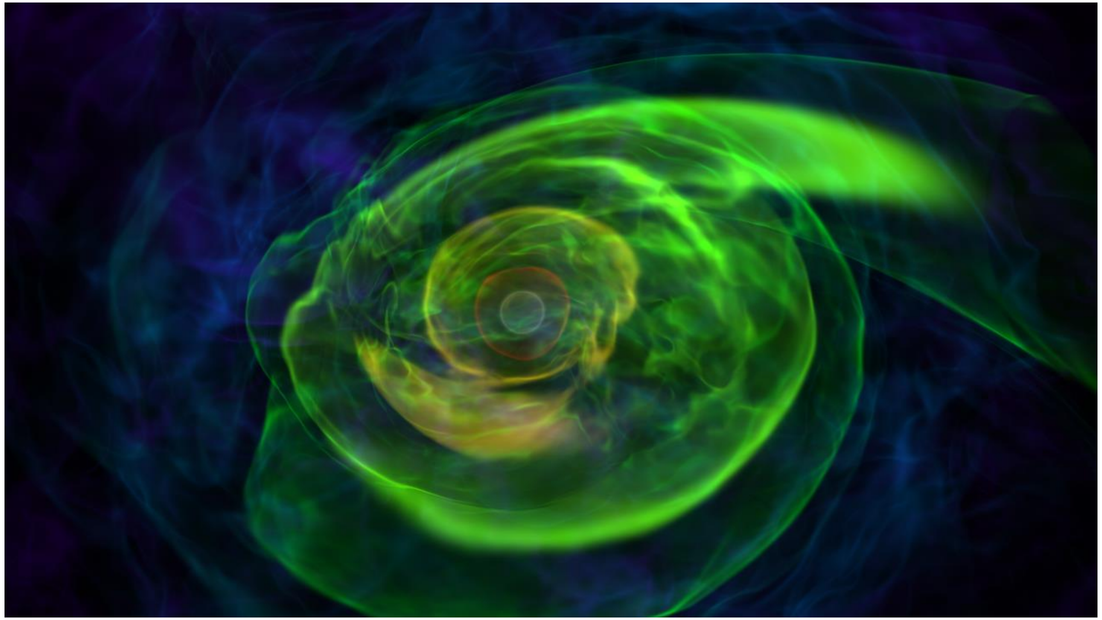
This iterator is configured to loop over grids or tiles, and could do particles instead

AMREX ASTROPHYSICS CODES

Castro: compressible hydrodynamics,
self-gravity, nuclear reactions

Maestro: low-Mach number hydrodynamics,
self-gravity, nuclear reactions

Nyx: cosmological hydrodynamics



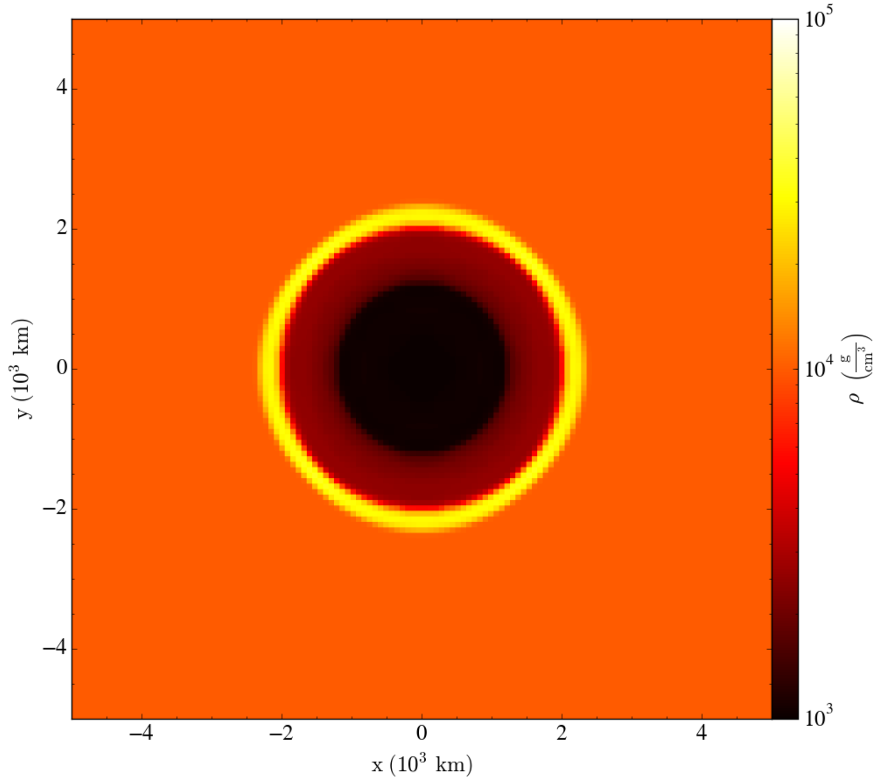
PERFORMANCE ANALYSIS TEST APPLICATION

StarLord: mini-app version of Castro hydrodynamics

- Sedov blast wave with an astrophysical twist
- 13 fluid species
- Realistic astrophysical equation of state
- Method of lines integration

Performance measured with standard Figure of Merit

- FOM == zones evolved per microsecond
- Usually measured with 128^3 to 256^3 zones on the grid



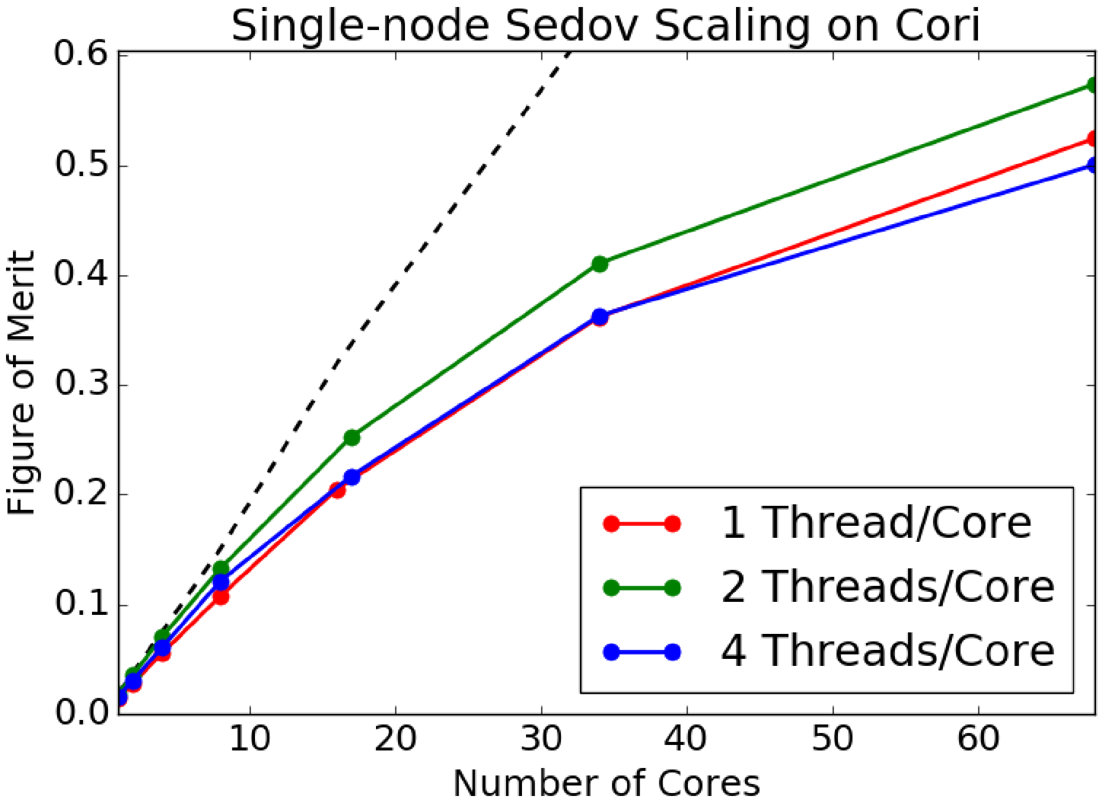
MINI-APP TEST RESULTS ON KNL

OpenMP only: Peak FOM = 0.6 zones/μsec

MPI + OpenMP (16 x 16): Peak FOM = 0.75

Affinity matters at high thread counts

Performance limited by lack of tiling

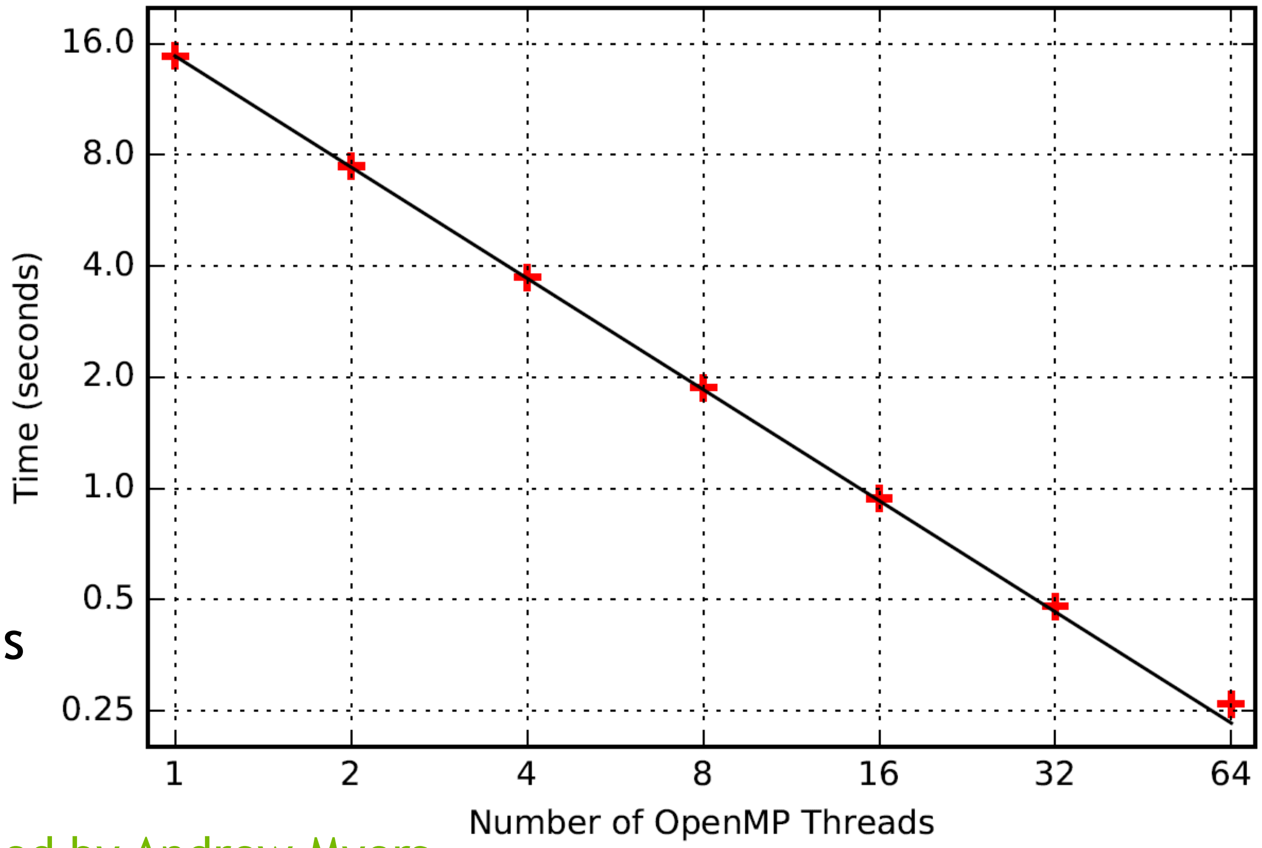


PARTICLE SCALING RESULTS ON KNL

WarpX electromagnetic PIC code is currently being ported to AMReX

Particles are sorted into local tiles for cache-friendliness

Reductions done on cell-by-cell basis for multithreading effectiveness



Data provided by Andrew Myers

EVALUATING A GPU STRATEGY

Use Unified Memory and C++ iterators to hide complexities of data motion

- Keep data resident as long as possible to avoid transfer costs

Initial MPI approach: rely on managed memory to do the right thing

- Evaluating approach with traditional (non-managed) memory

CUDA Fortran used to offload compute onto the device with minimal code markup

- We will also examine OpenMP and OpenACC for offloading

GRID LOOPING STRATEGY FOR THE GPU

```
for (MFIter mfi(state); mfi.isValid(); ++mfi)
{

    local_state = state[mfi];
    box = mfi.tilebox();
    evolve(local_state, box);

}
```

++mfi: asynchronously perform operations for each grid (e.g. reductions)

Grid loop is *not* tiled

state[mfi]: prefetch the data from this grid to the device

Function call: launch device kernel

~MFIter: perform device synchronize, optionally prefetch data back to host

MINI-APP TESTING ON SUMMITDEV

IBM Minsky nodes with 20 POWER 8 cores, 4 NVIDIA P100 GPUs at OLCF

Single POWER 8 core: FOM = 0.03 zones/μsec

20 POWER 8 cores: Peak FOM = 0.37 zones/μsec

Single P100 Peak FOM: 1.2 zones/μsec for single grid

Scaling to four GPUs: FOM = 3.1 zones/μsec (128^3), 3.8 zones/μsec (192^3)

AMREX COLLABORATORS

LBL: Ann Almgren, Vince Beckner, John Bell, Marc Day, Andrew Myers, Brian Friesen, Andy Nonaka, Weiqun Zhang

Stony Brook: Maria Barrios-Sazo, Don Willcox, Mike Zingale

MSU: Adam Jacobs

LANL: Chris Malone

https://github.com/AMReX-Codes

https://github.com/AMReX-Astro

NVIDIA