HACC: Fitting the Universe Inside a Supercomputer

Nicholas Frontiere
University of Chicago/Argonne National Laboratory

?

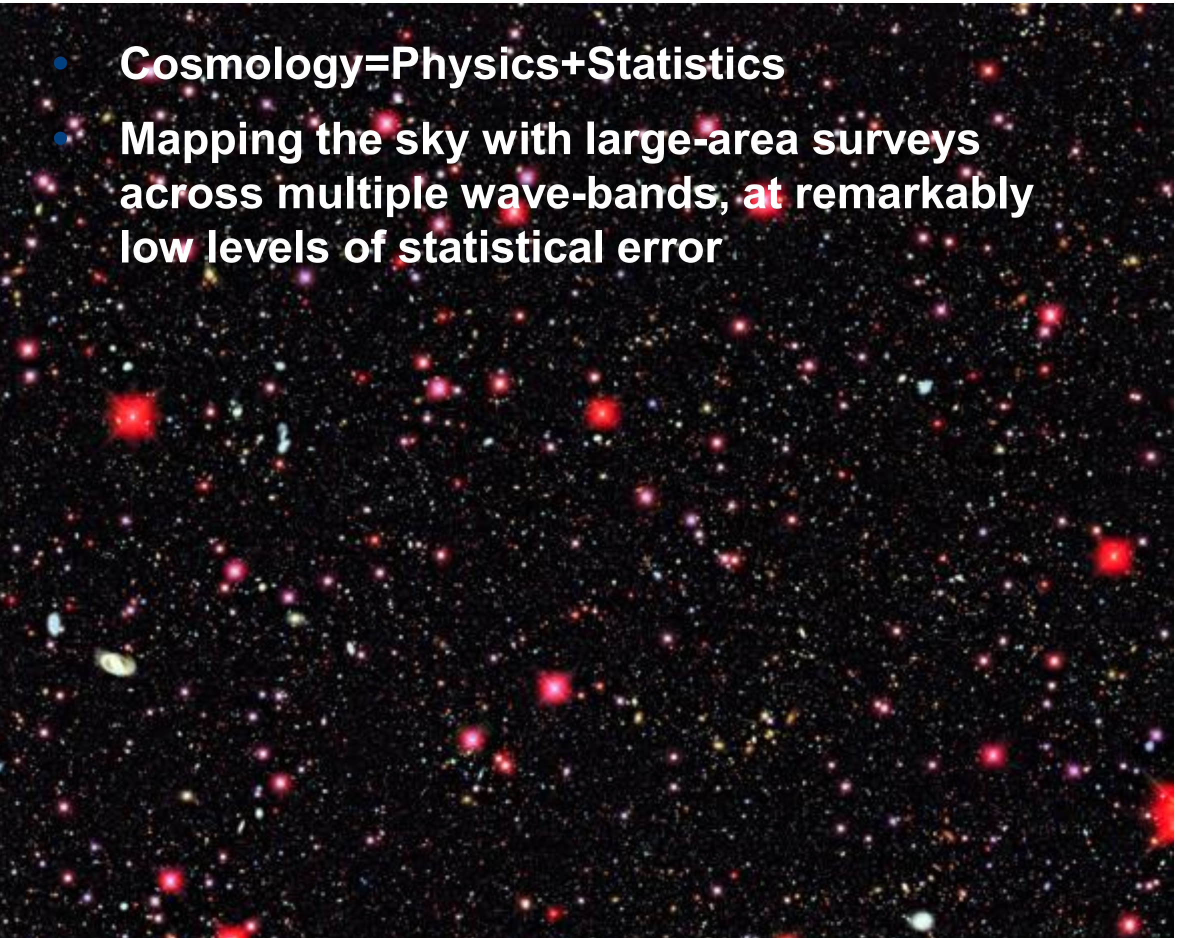
How Does Cosmology Fit in HPC?

- **General motivations for large HPC campaigns:**
  1) Quantitative predictions
  2) Scientific discovery, expose mechanisms
  3) System-scale simulations ('impossible experiments')
  4) Inverse problems and optimization

- **Driven by a wide variety of data sources, computational cosmology must address ALL of the above**

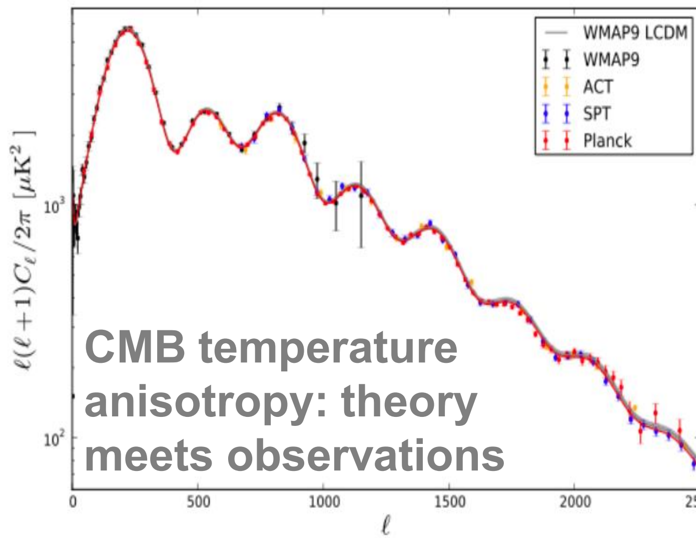
- **Role of scalability/performance:**
  1) Very large simulations necessary, but not just a matter of running a few large realizations
  2) High throughput essential
  3) Optimal design of simulation campaigns
  4) Analysis pipelines and associated infrastructure

Data ‘Overload’ Problem

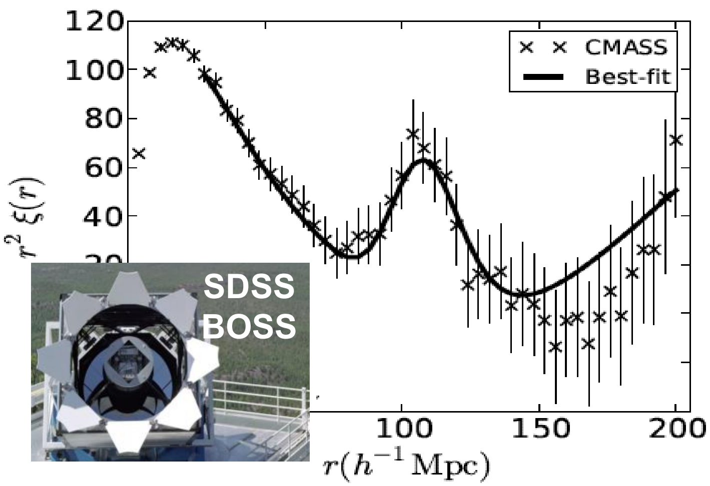


- Cosmology=Physics+Statistics
- Mapping the sky with large-area surveys across multiple wave-bands, at remarkably low levels of statistical error

Galaxies in a moon-sized patch (Deep Lens Survey). LSST will cover 50,000 times this size (~400PB of data)



CMB temperature anisotropy: theory meets observations

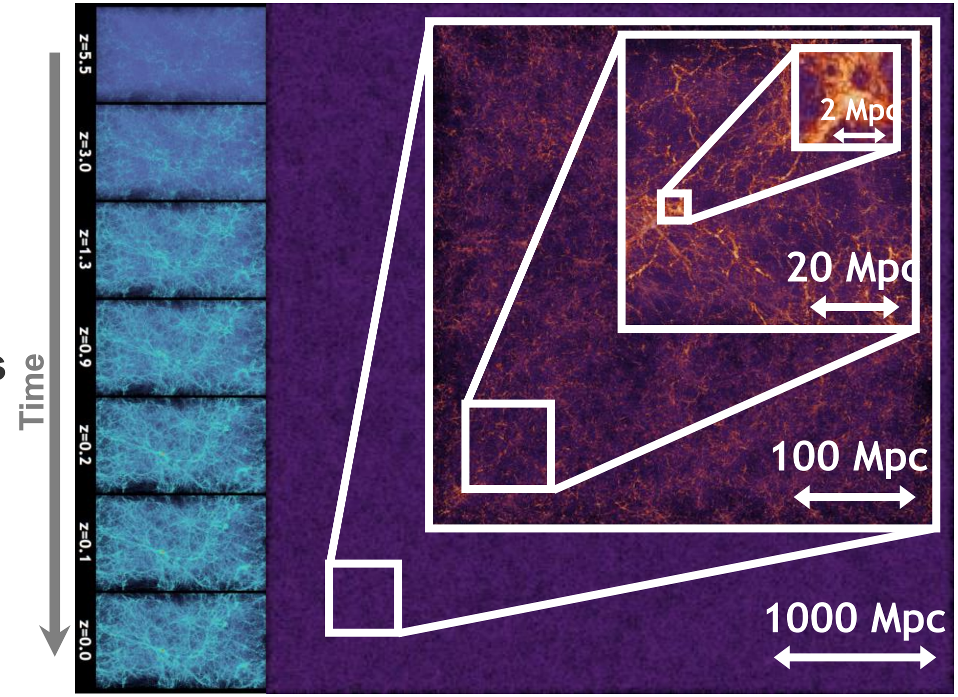


The same signal in the galaxy distribution

Large Scale Structure Simulation Requirements

Force and Mass Resolution:

- Galaxy halos ~100kpc, hence force resolution has to be ~kpc; with Gpc box-sizes, a **dynamic range of a million to one**
- Ratio of largest object mass to lightest is **~10000:1**



Gravitational Jeans Instablity

Physics:

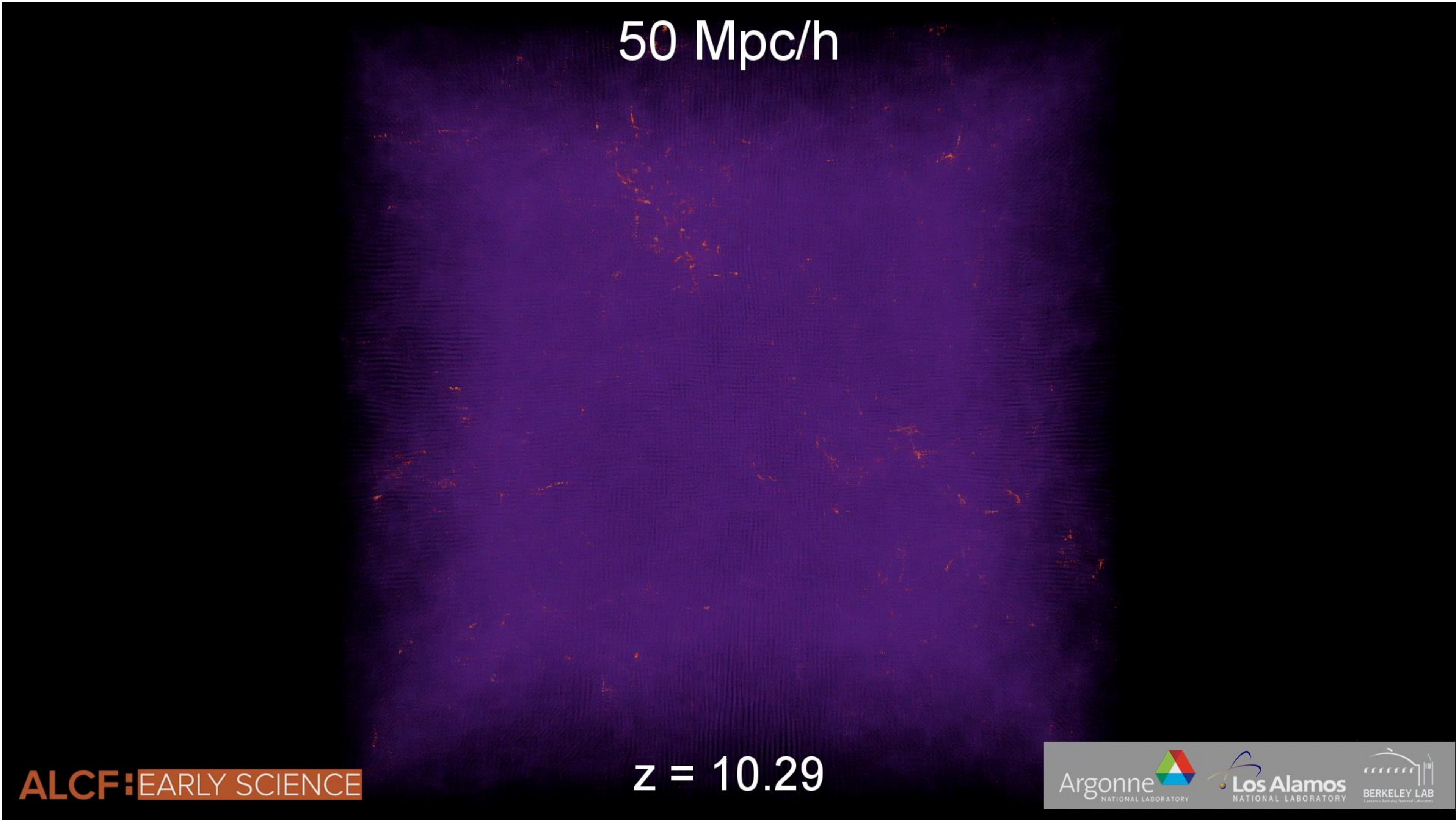
- Gravity dominates at scales greater than ~Mpc
- Small scales: galaxy (subgrid) modeling, semi-analytic methods to incorporate gas physics/feedback/star formation

Computing 'Boundary Conditions':

- Total memory in the PB+ class
- Performance in the 10 PFlops+ class
- Wall-clock of ~days/week, in situ analysis

Can the Universe be run as a short computational 'experiment'?

Dynamic Range in the Outer Rim Simulation



The Outer Rim Run on Mira: 1.1 trillion particles, 4.2 Gpc box

Large Scale Structure: Vlasov-Poisson Equation

$$\frac{\partial f_i}{\partial t} + \dot{\mathbf{x}}\frac{\partial f_i}{\partial \mathbf{x}} - \nabla\phi\frac{\partial f_i}{\partial \mathbf{p}} = 0, \qquad \mathbf{p} = a^2\dot{\mathbf{x}},$$

$$\nabla^2\phi = 4\pi G a^2(\rho(\mathbf{x},t) - \langle\rho_{\rm dm}(t)\rangle) = 4\pi G a^2\Omega_{\rm dm}\delta_{\rm dm}\rho_{\rm cr},$$

$$\delta_{\rm dm}(\mathbf{x},t) = (\rho_{\rm dm} - \langle\rho_{\rm dm}\rangle)/\langle\rho_{\rm dm}\rangle),$$

$$\rho_{\rm dm}(\mathbf{x},t) = a^{-3}\sum_i m_i \int d^3\mathbf{p} f_i(\mathbf{x},\dot{\mathbf{x}},t).$$

Cosmological Vlasov-Poisson Equation

- **Properties of the Cosmological Vlasov-Poisson Equation:**
- **6-D PDE with long-range interactions, no shielding, all scales matter, models gravity-only, collisionless evolution**
- **Extreme dynamic range in space and mass (in many applications, million to one, ‘everywhere’)**
- **Jeans instability drives structure formation at all scales from smooth Gaussian random field initial conditions**

Separation of Scales

Particle-Mesh Method:

- The fluid elements (particles) are interpolated to a grid.
- Solve VP Eqn for potential using FFTs: $e.g.\ \nabla^2\varphi(x) = g(x) \Longrightarrow -k^2\tilde{\varphi}(k) = \tilde{g}(\mathrm{k})$
- Interpolate resulting force ($\nabla\varphi$) back to the particles and evolve them.

Problem:

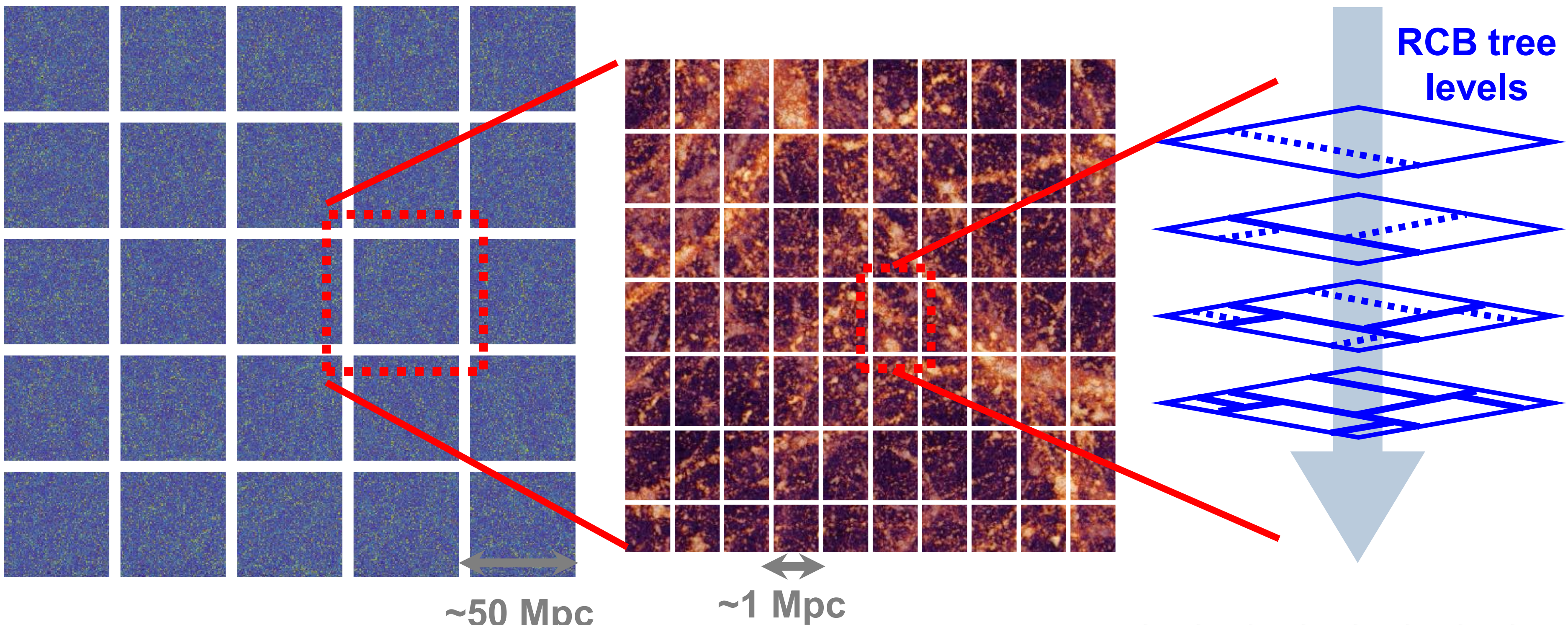
- Although using a PM technique is the most computationally efficient, we'd need a $\approx (10^6)^3$ grid to capture the full dynamic range of the simulation!

Separation of Scales Solution:

- Use the FFT for as much as possible and use some less-memory hungry technique for smaller scales.
- Longer spatial scales have longer characteristic time scales so we can "subcycle" the smaller scale computations relative to the longer ones.
- The small scale computations are rank-local, and can be offloaded to accelerators

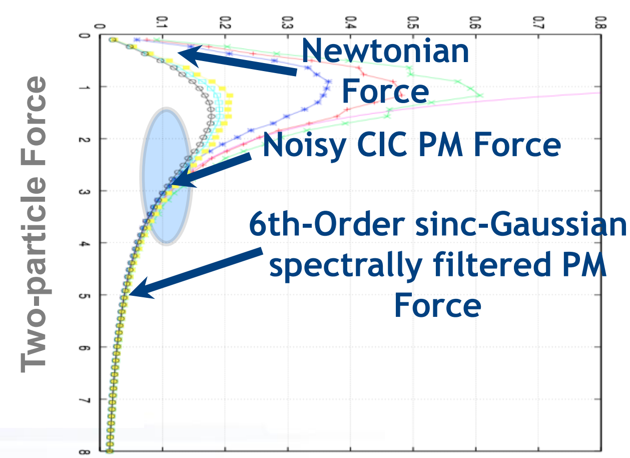
$$force\colon\ f(x) = f_{long}(x) + f_{short}(x)$$

'HACC In Pictures'



HACC Top Layer:
3-D domain decomposition with particle replication at boundaries ('overloading') for Spectral PM algorithm (long-range force)

HACC 'Nodal' Layer:
Short-range solvers employing combination of flexible chaining mesh and RCB tree-based force evaluations



ADDITIONAL PHYSICS

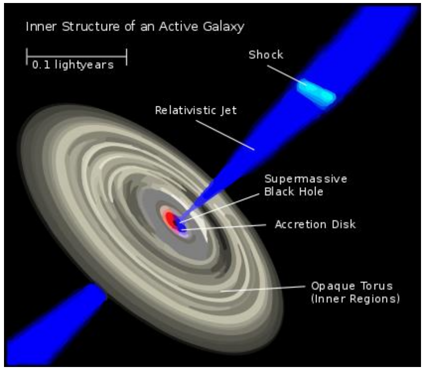
- Gravity dominates the physics at scales greater than ~Mpc. For smaller scales it is important to capture of the impact of baryon physics on structure formation
- Adiabatic Hydro

$$\frac{d}{dt} = \frac{\partial}{\partial t} + \vec{v} \cdot \nabla \qquad \frac{du}{dt} = -\frac{P}{\rho}\nabla \cdot \boldsymbol{v}$$

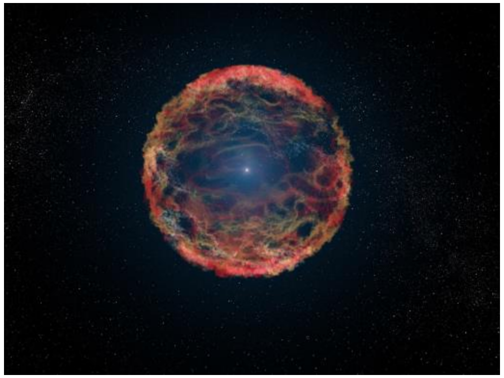
$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = -\rho\nabla \cdot \boldsymbol{v} \qquad \frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t} = -\frac{1}{\rho}\nabla P$$

- Subgrid (i.e. everything else!)

<u>Active Galactic Nuclei</u> <u>Star Formation and Supernova Feedback</u>



What happens in a black hole stays in a black hole.



…

SPH

- Smoothed Particle Hydrodynamics
  - Particles serve as interpolation points for calculating fluid properties.
  - Fluid elements are represented with a smoothing function (kernel) W.
  - Smoothing scale *h*, defines the support of the kernel.

$$\psi(\mathbf{r}) = \int \psi(\mathbf{r}')W(\mathbf{r}-\mathbf{r}', h)d\mathbf{r}'$$

$$\psi_i = \sum_j \psi_j W(|r_i - r_j|, h)V_j = \sum_j m_j \frac{\psi_j}{\rho_j} W(|r_i - r_j|, h)$$

$$\frac{\mathrm{d}\boldsymbol{v}_a}{\mathrm{d}t} = -\sum_b m_b \left(\frac{P_b}{\rho_b^2} + \frac{P_a}{\rho_a^2} \right) \nabla_a W_{ab} \qquad \frac{\mathrm{d}\hat{e}_a}{\mathrm{d}t} = -\sum_b m_b \left(\frac{P_a \boldsymbol{v}_b}{\rho_a^2} + \frac{P_b \boldsymbol{v}_a}{\rho_b^2} \right) \cdot \nabla_a W_{ab}$$

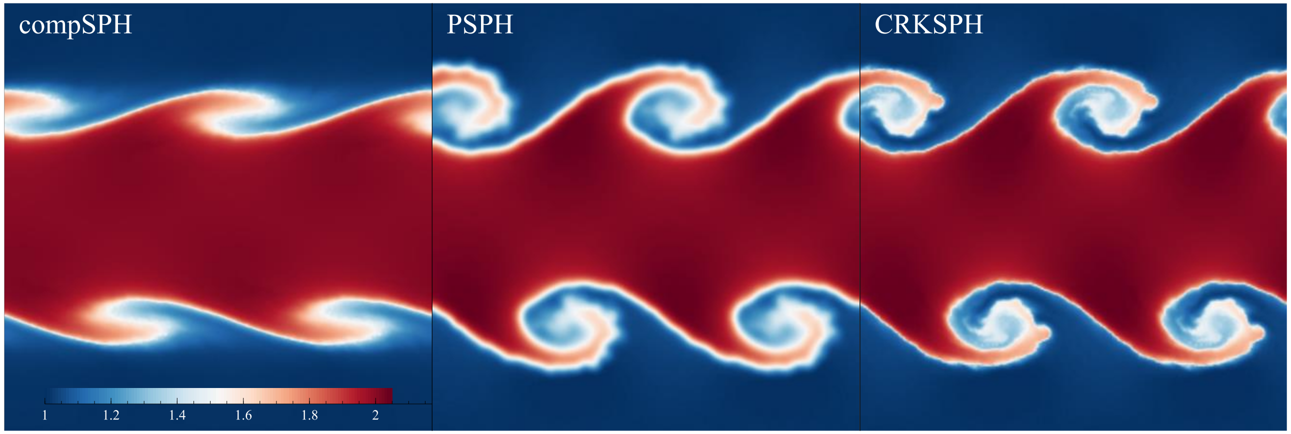
CRKSPH

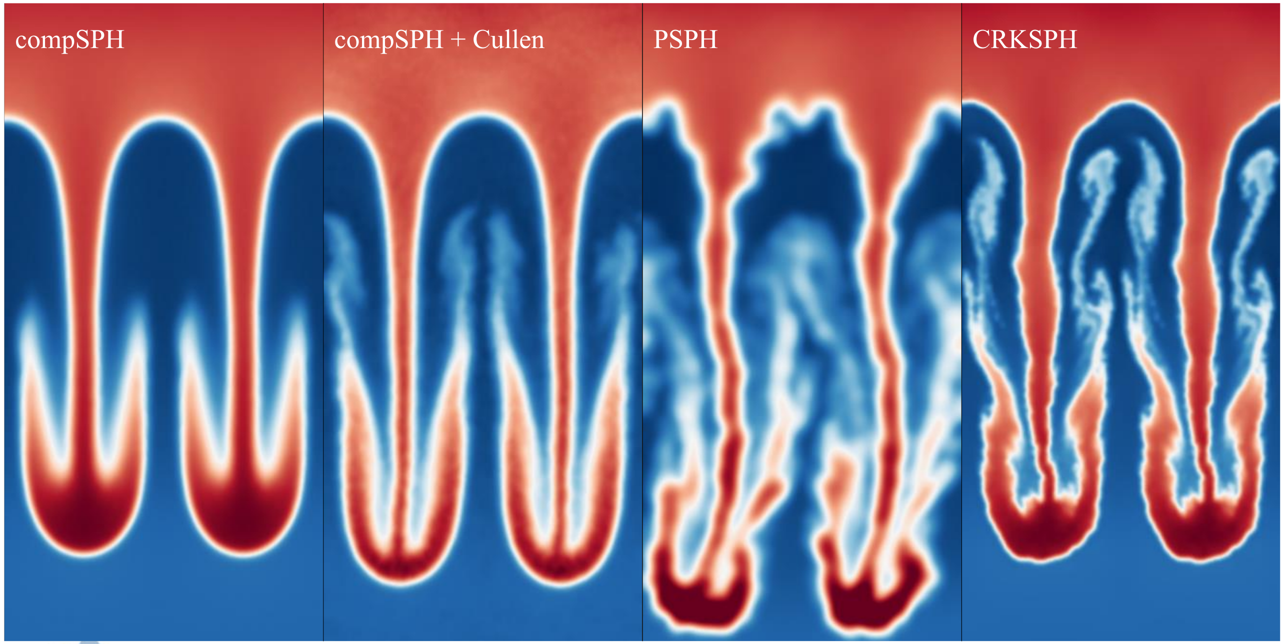
- Conservative Reproducing Kernel SPH*
  - An improved Smoothed Particle Hydrodynamic (SPH) solver
  - Higher order reproducing kernels
    - Exactly reproduce constant and linear order fields
  - Conservative reformulation of the dynamic equations that maintain machine precision energy and momentum conservation
  - Uses a new artificial viscosity form that capitalizes on the increased accuracy calculation of the velocity gradients. Improves the excessive diffusion normally encountered in SPH
  - Developed and piloted during CSGF practicum!

* Frontiere, Nicholas, Cody D. Raskin, and J. Michael Owen. "CRKSPH–A Conservative Reproducing Kernel Smoothed Particle Hydrodynamics Scheme." *Journal of Computational Physics* 332 (2017): 160-209.

Example Comparisons

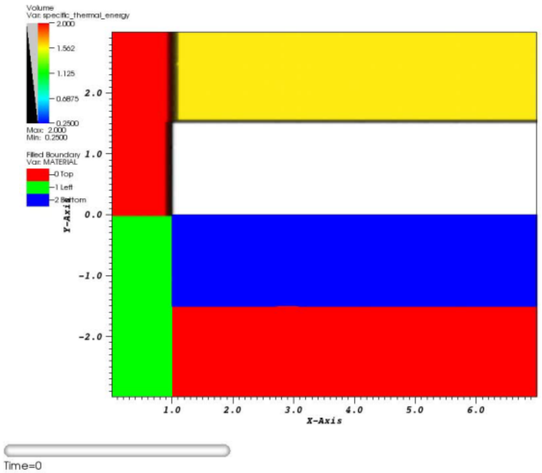
Kelvin-Helmholtz

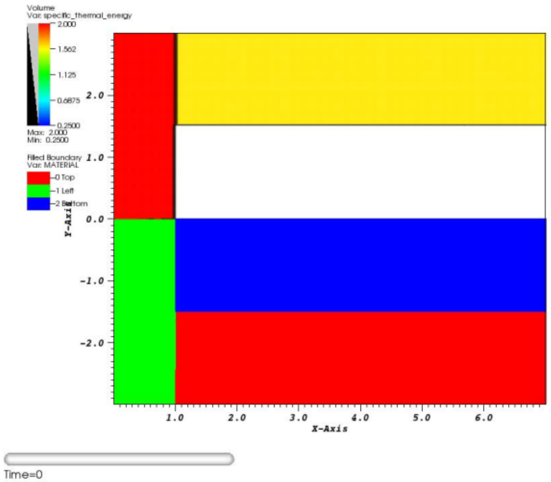




Rayleigh-Taylor

SPH



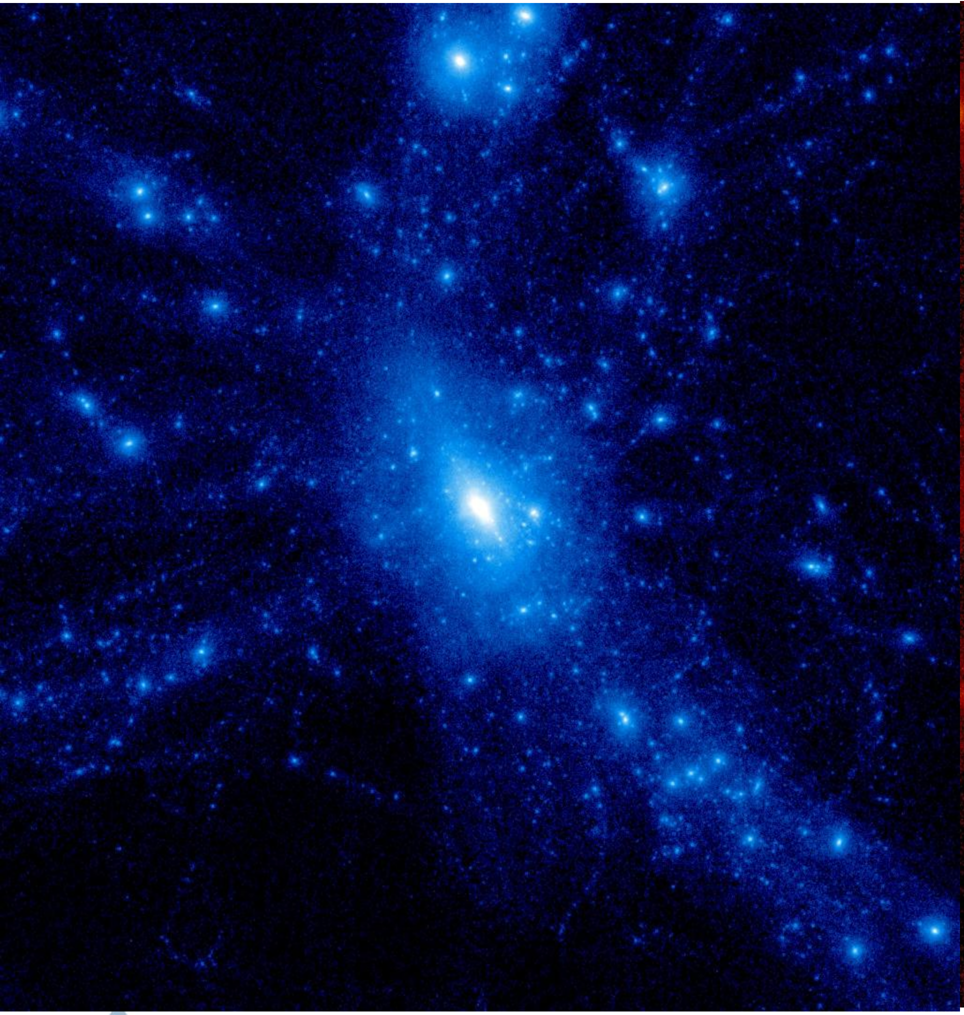


CRKSPH

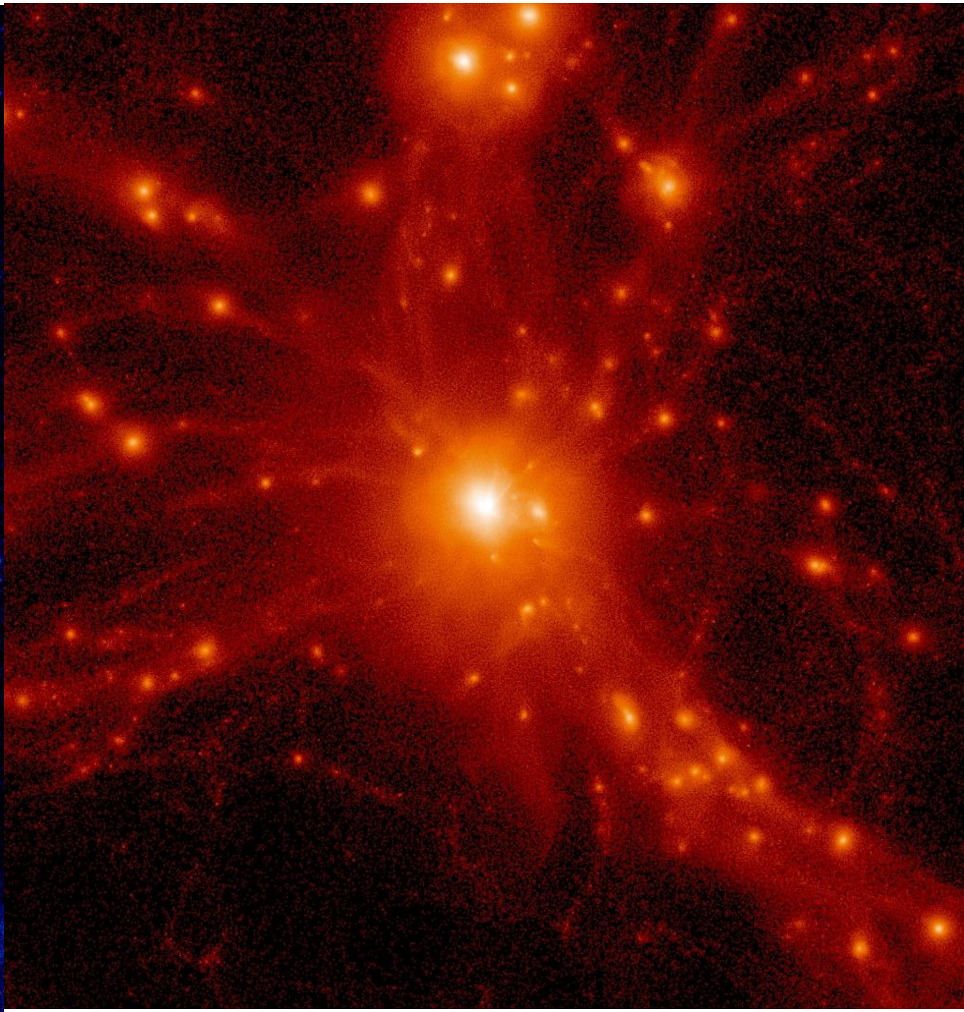
Triplepoint Shock

N-body: Gravity + Hydro

CDM



CDM + Baryons



Architectural Challenges

Architectural 'Features'

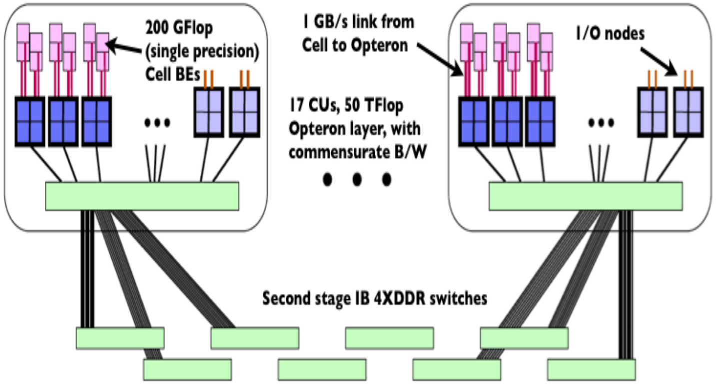
•**Complex heterogeneous nodes**

•**Simpler cores, lower memory/core (will weak scaling continue?)**

•**Skewed compute/communication balance**

•**Programming models?**

•**I/O? File systems?**



Roadrunner exemplar still relevant!

Combating Architectural Diversity with HACC

•**Architecture-independent performance/scalability:** 'Universal' top layer + 'plug in' node-level components; minimize data structure complexity and data motion

•**Programming model:** 'C++/MPI + X' where X = OpenMP, Cell SDK, OpenCL, CUDA, --

•**Algorithm Co-Design:** Multiple algorithm options, stresses accuracy, low memory overhead, no external libraries in simulation path

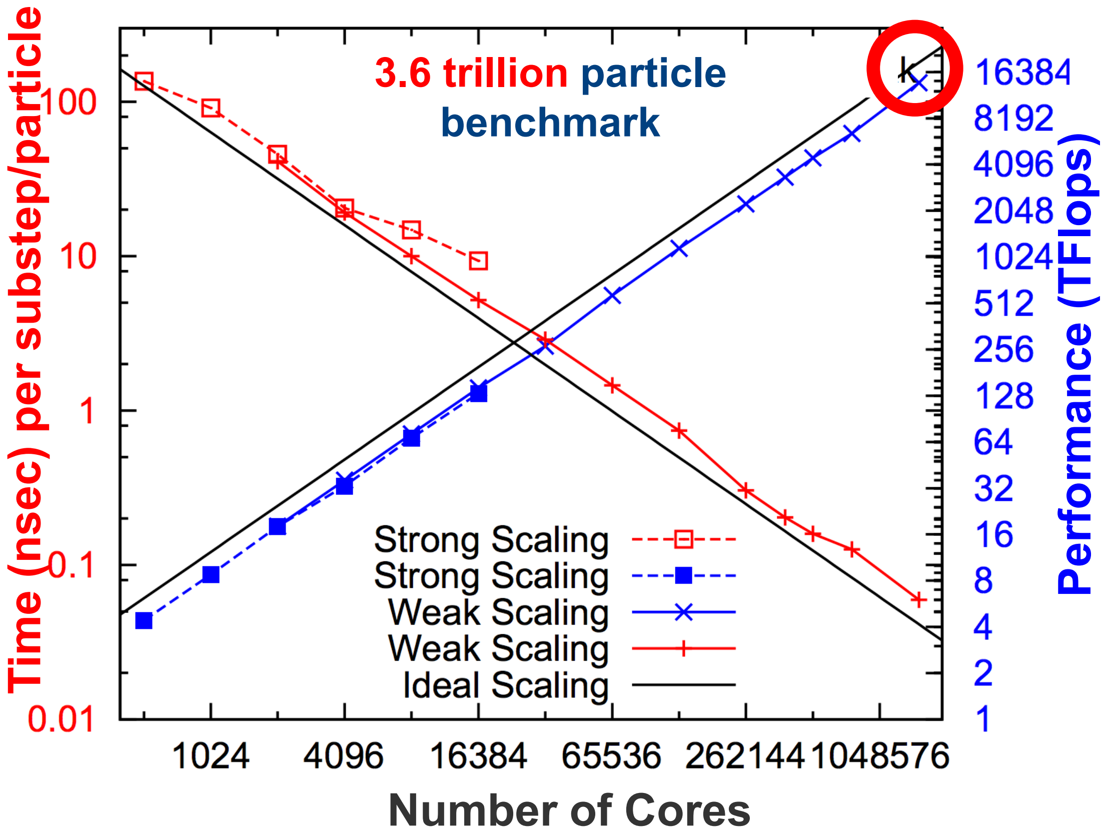
•**Analysis tools:** Major analysis framework, tools deployed in stand-alone and in situ modes

HACC on the BG/Q

Scalability and Performance:

- Kernel performance is 80% of peak, vs. theoretical maximum of 81%, sustained performance can reach 69% of peak
- Improved load-balancing (with ‘hyper-local’ trees) and time-stepping (4X)
- I/O with compression
- Excellent strong scaling; performance is very good even at memory footprints of 100MB/core
- In full production status on BG/Q

13.94 PFlops, 69.2% peak, 90% parallel efficiency on 1,572,864 cores/MPI ranks, 6.3M-way concurrency



HACC weak scaling on the IBM BG/Q (MPI/OpenMP)

HACC on Titan: GPU Implementation (Schematic)

P3M Implementation:

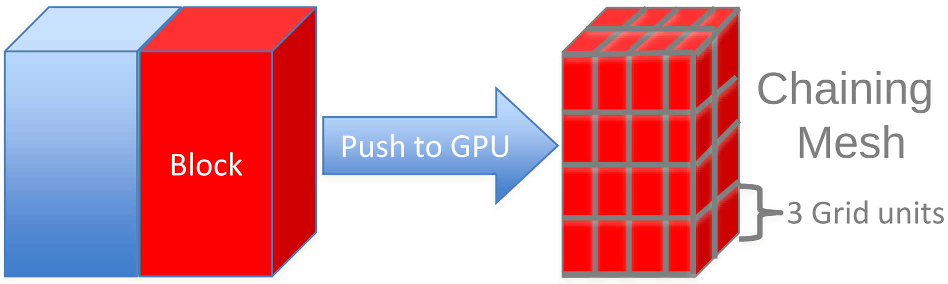
•Spatial data pushed to GPU in large blocks, data is sub-partitioned into chaining mesh cubes

•Compute forces between particles in a cube and neighboring cubes

•Natural parallelism and simplicity leads to high performance

•Typical push size ~2GB; large push size ensures computation time exceeds memory transfer latency by a large factor

•More MPI tasks/node preferred over threaded single MPI tasks (better host code performance)



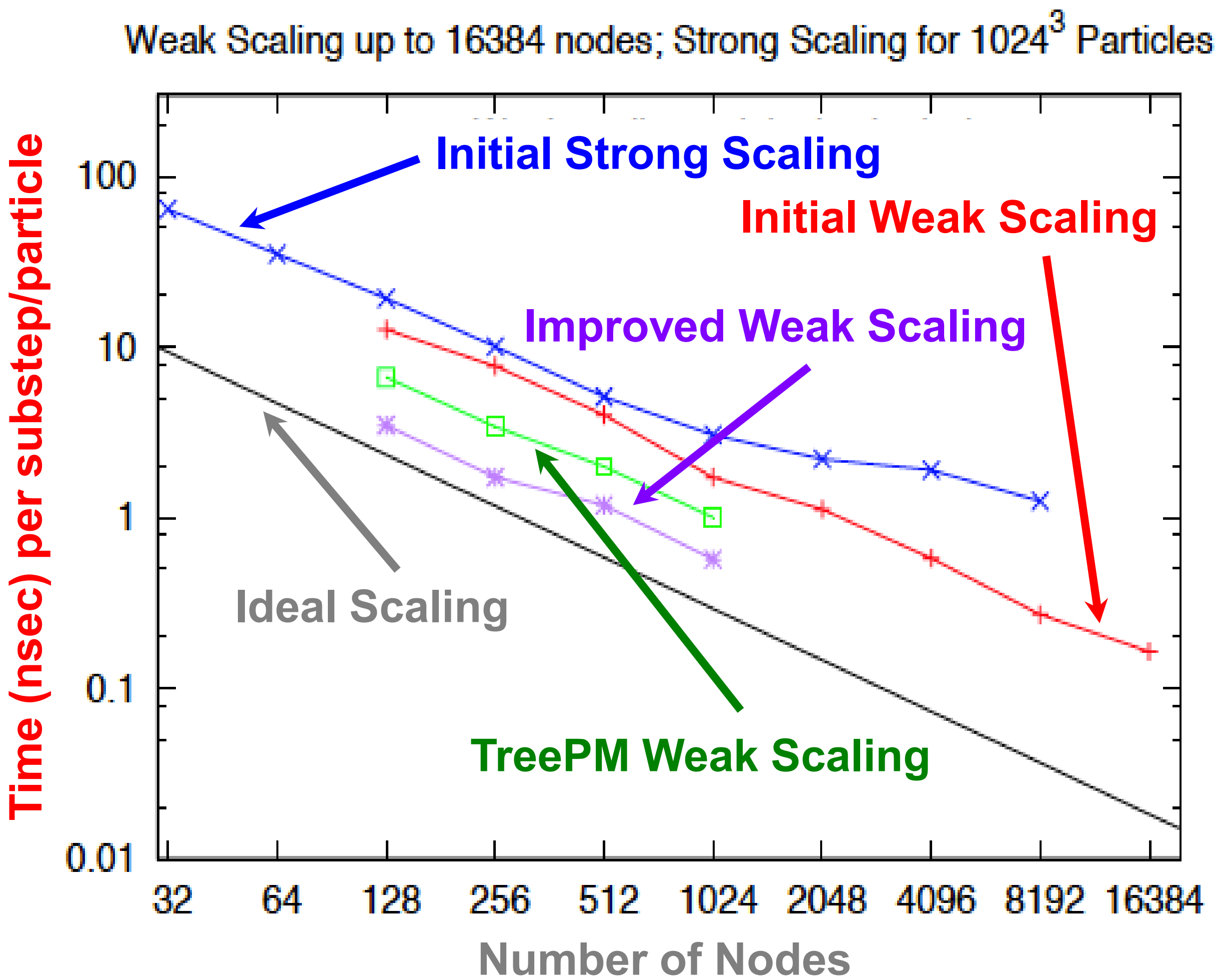
New Implementations:

•P3M with data pushed only once per long time-step, completely eliminating memory transfer latencies (orders of magnitude less); uses 'soft boundary' chaining mesh, rather than rebuilding every sub-cycle

•TreePM analog of BG/Q code written in CUDA, also produces high performance

HACC on Titan: GPU Implementation Performance

- P3M kernel runs at **1.6TFlops/node** at 40.3% of peak (73% of algorithmic peak)
- TreePM kernel was run on 77% of Titan at **20.54 PFlops** at almost identical performance on the card
- Because of less overhead, P3M code is (currently) faster by factor of two in time to solution



99.2% Parallel Efficiency

Acknowledgements

DOE CSGF

THANKS FOR THE MONEY!