FleCSPH: A Parallel and Distributed Smoothed Particle Hydrodynamics Implementation Based on the FleCSI Framework

Julien Loiseau[1], Hyun Lim[2], Nichols D. Moss[3], Ben K. Bergen[3]

1. Laboratory of CReSTIC, University of Reims Champagne-Ardenne, 2. Department of Physics and Astronomy, Brigham Young University, 3. Applied Computer Science Group, Los Alamos National Laboratory

Introduction and Motivation

- **Project:** Provide a parallel and distributed toolkit for large scale multi-physics simulations based on smoothed particle hydrodynamics
- **Starting point:** FleCSI, a template C++14, framework for multi-physics purpose

Computer Science Goal:

- Provide parallel and distributed version of FleCSI tree and particle system
- Paraview support for output
- Fast multipole methods (FMM)

Physics Goal:

- Newtonian SPH formulations with different kernels. Add equations of state (EOS), both analytic and tabulated
- Test various problems in different dimensions
- Simulate compact binary mergers such as white dwarfs and neutron stars to obtain template of gravitational waves and accretion, ejecta

FleCSI

What? A compile time configurable framework designed to support multi-physics application development. Available at http://github.com/laristra/flecsi

Provide a general set of infrastructure design patterns

- Support for:
  - Mesh topology, mesh geometry, mesh adjacency information, **n-dimensional hashed-tree**, graph partitioning interfaces, dependency closures, …
- Functional programming model with control, execution, and data abstraction
- Several runtimes like MPI, Legion or Charm++

Why? Provide the base implementation for the tree topology used in SPH.

What? SPH implementation based on FleCSI framework. Available at http://github.com/laristra/flecsph

Why? Can use bulit-in functionalities in FleCSI

SPH is perfect method to test complicated problem for exascale computing

SPH is used widely in astrophysics and cosmology such as modeling supernova, white dwarfs, and the big bang cosmology

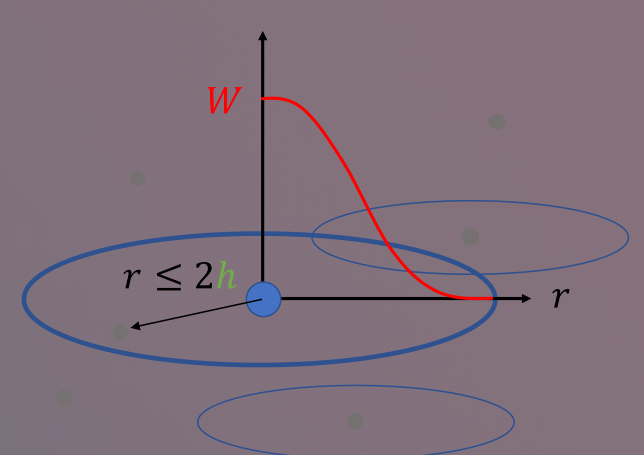
Smoothed Particle Hydrodynamics (SPH)

What? Explicit numerical *mesh-free* method, solve hydrodynamics PDE: Lagrangian, discretized in set of fluid elements

How? Their smooth field variables (density, velocity, internal energy, pressure) and derivatives interpolated via smoothing kernel W

$$\langle A\rangle(\vec{r}) \approx \sum_b \frac{m_b}{\rho_b} A(\vec{r_b}) W(|\vec{r}-\vec{r_b}|, h)$$

h smoothing length evolves for each particles (adaptive resolution)



- Equation of state (state of matter)

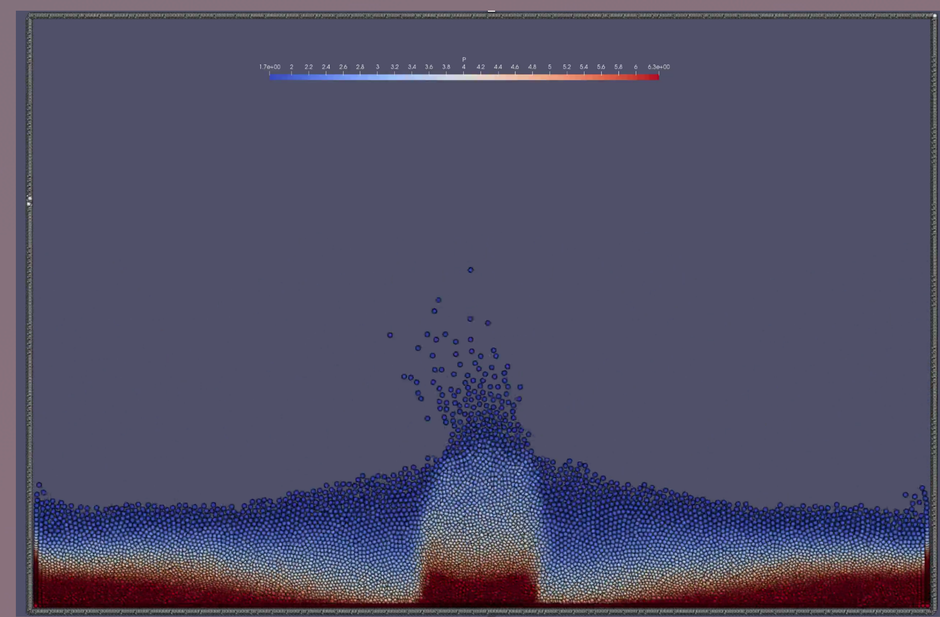
Why?:

- Discretized from exactly conserves mass, energy linear & angular momentum by constructions: independent of numerical resolutions
- Can handle deformations, low densities and vacuum
- Exactly advects fluid properties
- Easily combines with tree methods for solving Newtonian gravity via N-body
- Can be mapped into mesh grids that provides more extendable scheme

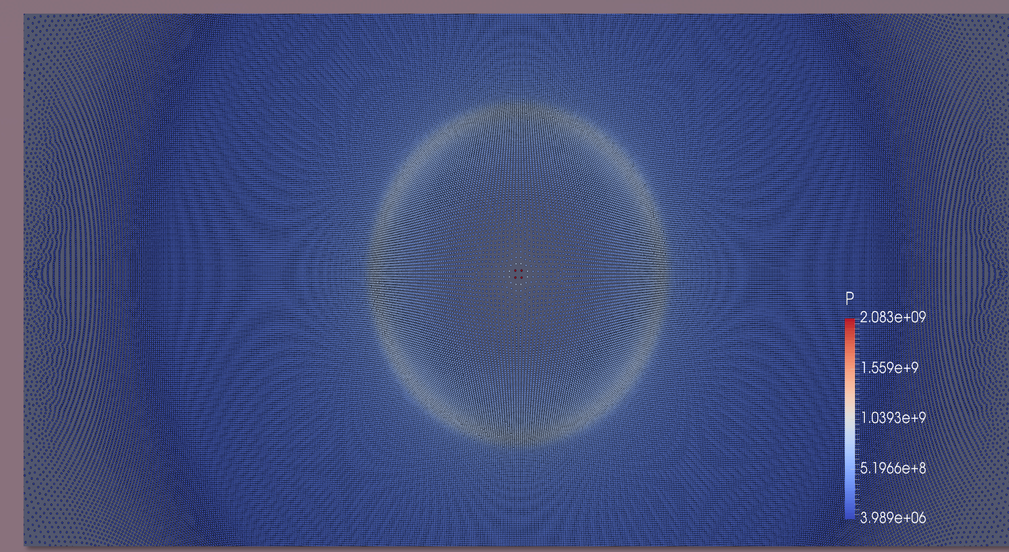
Test Problems Results

- SPH can solve different hydrodynamical problems that include discontinuities due to shock, high pressure etc. In this work, we present several different cases to validate the functionality of code

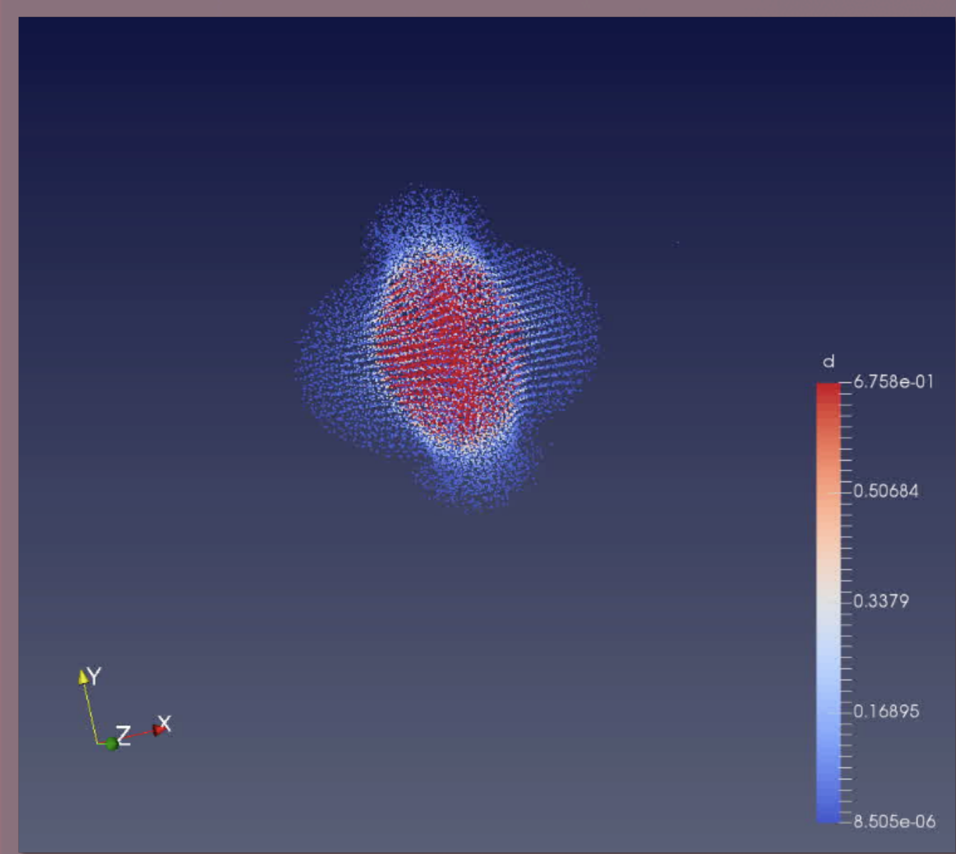
- 2D Fluid simulation with boundary



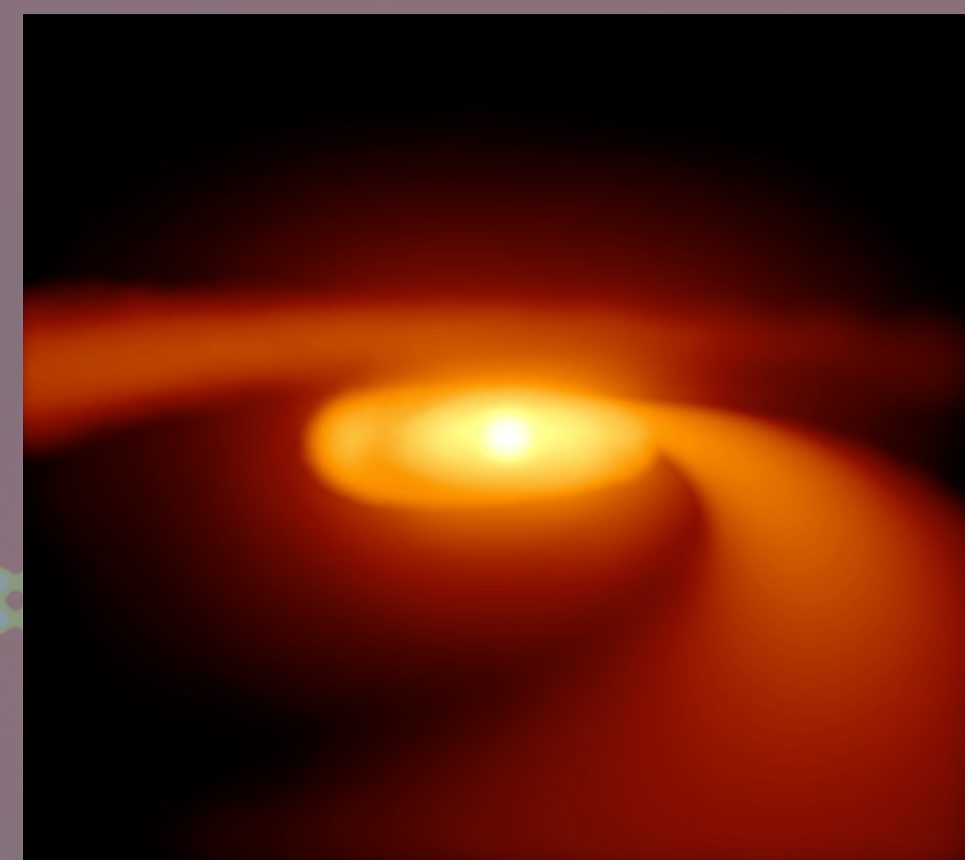
- 2D Sedov blast wave



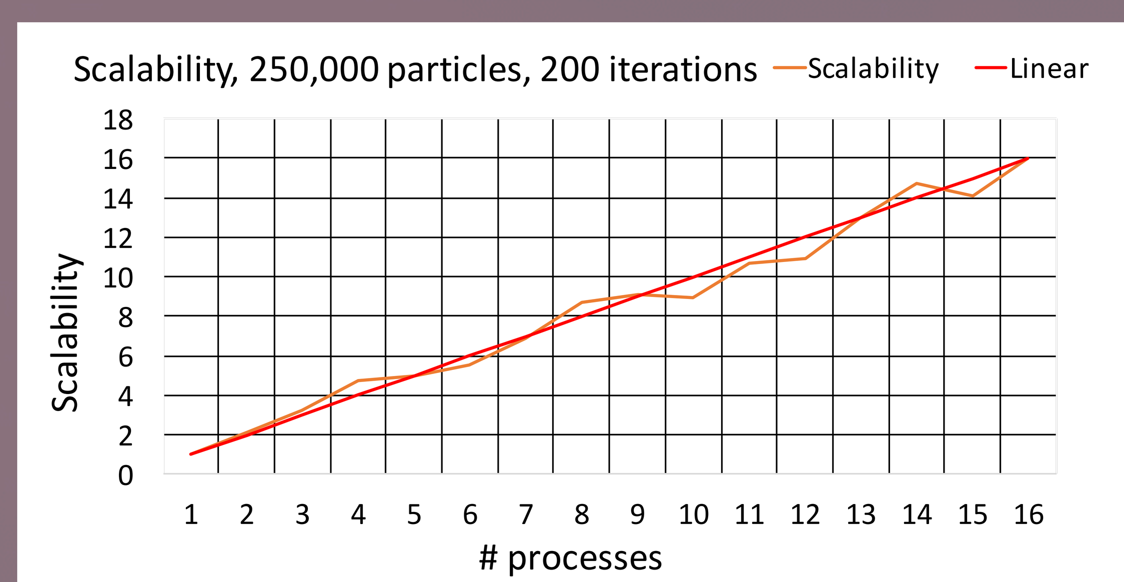
- Neutron stars head on collision



- Binary white dwarf merger



Validation and Performance Tests



Scalability tests done on DGX-1 Platform:

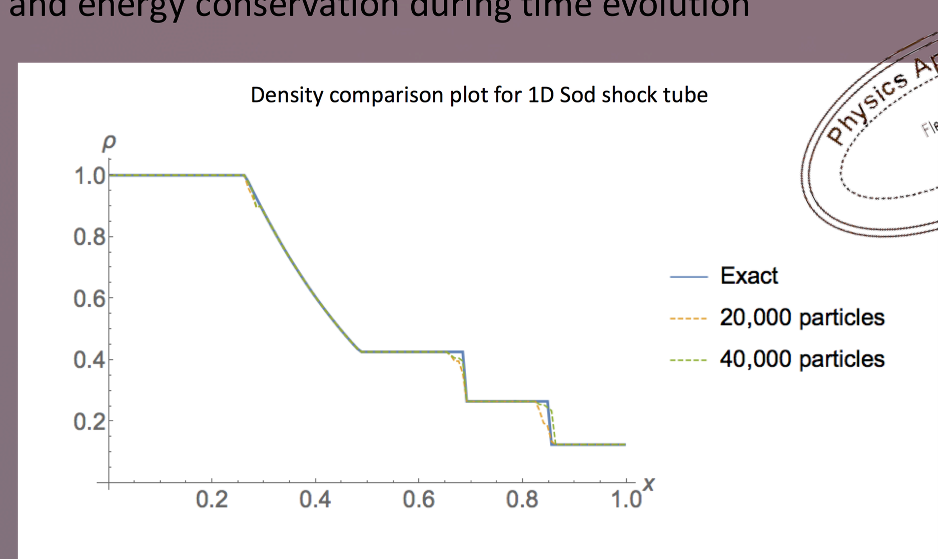
- 2 x CPU E5-2698 v4 2.2GHz

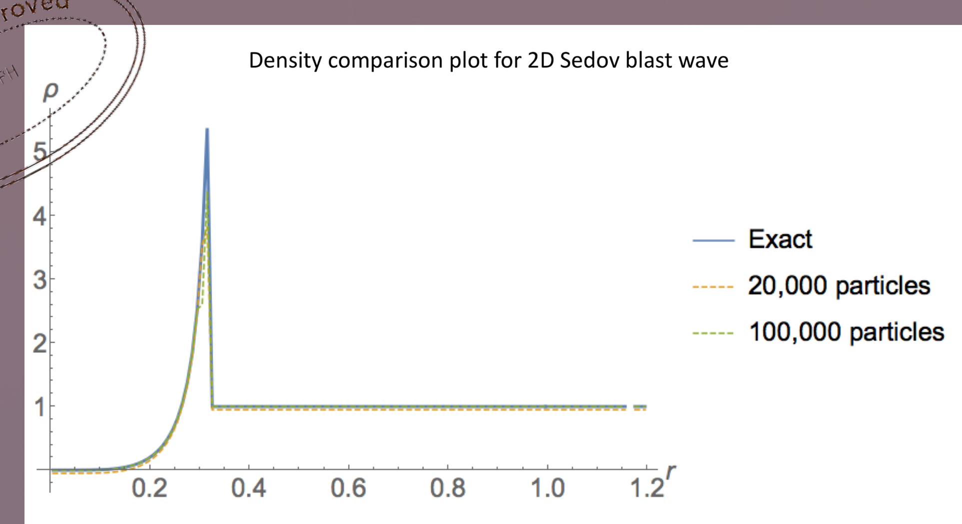
Good performances on that plaform, near linear scalability **for** Sedov in 2 dimensions.

Next scalability testing on a whole cluster: ROMEO Supercomputer center of Reims, France.

Working on Docker: Dockerfile available

Checking physics reliability: In-code assertions, Compare with well-known analytic solution, Check momentum, mass, and energy conservation during time evolution





Domain Decomposition

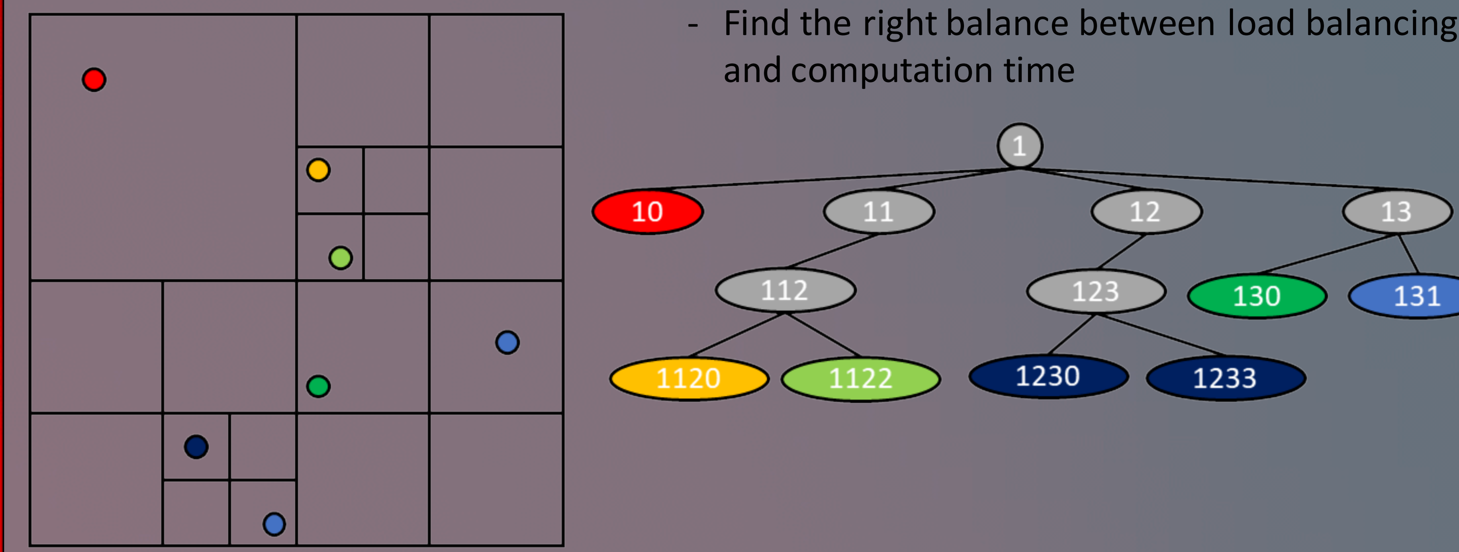
Morton Z-curve:

Give us a key assigned to each position:

- Sort particles
- Generate the tree
- Use for distribution

Further work:

- Implementing others space filling curves:
  - Peano
  - Hilbert
  - Gosper
- Find the right balance between load balancing and computation time



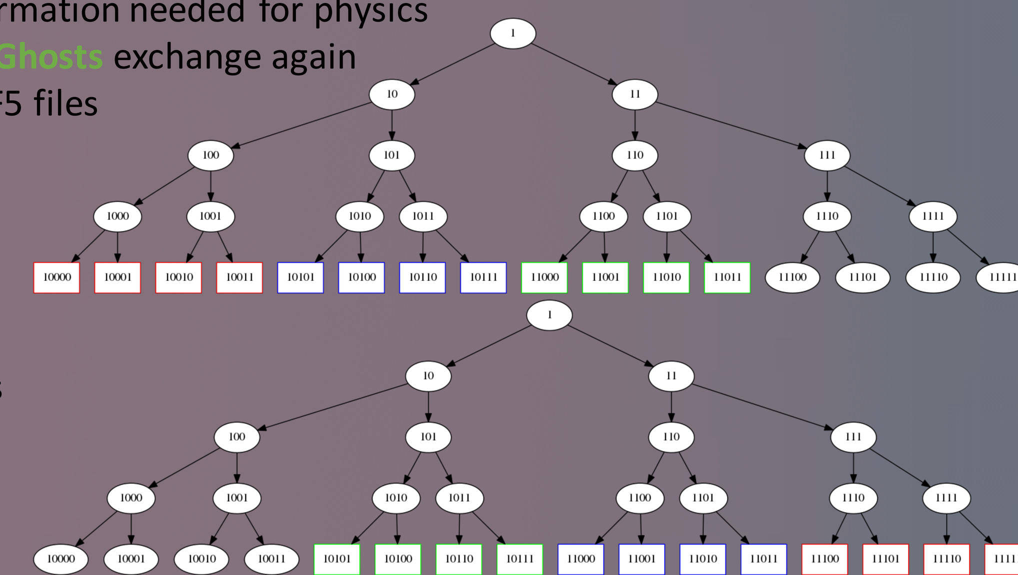
Distributed Tree Data Structure

Main algorithm:

- Read particles from HDF5 files
- Compute keys from positions and parallel sort the particles
- Generate the local tree from Exclusive and Shared particles
- Exchange interesting branches in the tree with neighbors
- Compute Ghosts from smoothing length
- Exchange Ghosts information needed for physics
- Physics and eventual Ghosts exchange again
- Write particles to HDF5 files

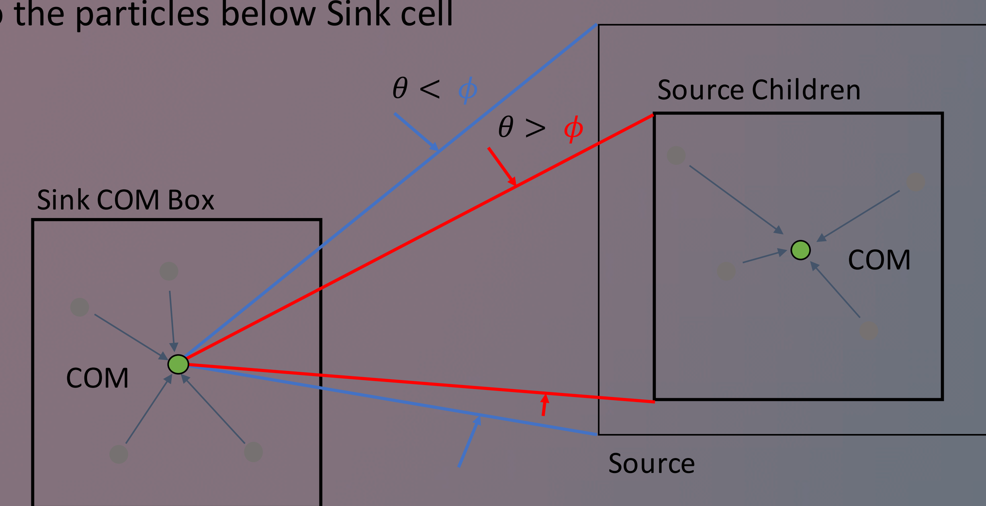
See:

Barnes and Hut trees



Tree topology also allows to efficiently compute long range forces like gravitation using **Fast Multipole Method**:

- Compute Center Of Mass (COM) by performing DFS in local tree
- Compute angle ϕ from Sink COM to Source bounding box
  - If more than Multipole Acceptance Criterion θ, open box
  - Else compute contribution of this source cell using Taylor expansion
- Propagate to the particles below Sink cell



Conclusions and Future Works

Computer Science:

- Include the tree distribution of the tree topology in FleCSI
- Provide accelerator support like GPU, Xeon Phi or FPGA.

Physics:

- Provide more realistic astrophysical data
- Tests different binary compact objects mergers that include neutron stars, white dwarfs, and black hole accretion

We acknowledge the support of **Christoph Junghans**, **Wesley Paul Even**, **Oleg Korobkin** and **Brett Okhuysen**.

We also thanks **Andre Vincent Pascal Grosset** for his contribution on the Parallel IO using HDF5 files and the **ROMEO Supercomputer Center of Reims, France and Stampede, Texas Advanced Computing Center**.