MFIX-DEM Enhancement for Industry-Relevant Flows

Presenter name: Hari Sitaraman

Date: Aug 24, 2020

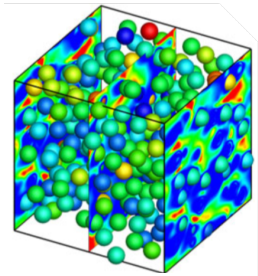
Project leads

Dr. Ray Cocco (PSRI, co-PI)
Dr. Hari Sitaraman (NREL, co-PI)
Prof. Thomas Hauser (Univ. CO, co-PI)
Prof. Christine Hrenya (Univ. CO, PI)

Funding: Office of Fossil Energy

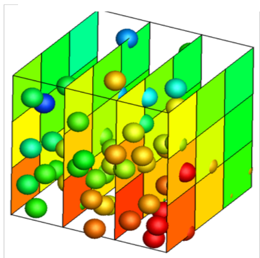
Background and motivation

Tenneti & Subramaniam (2016)

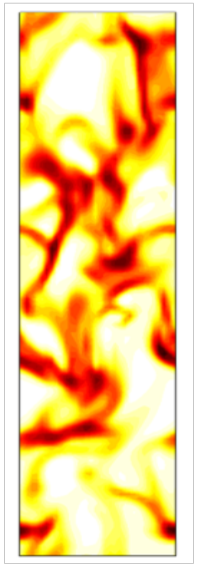


DNS

CFD-DEM

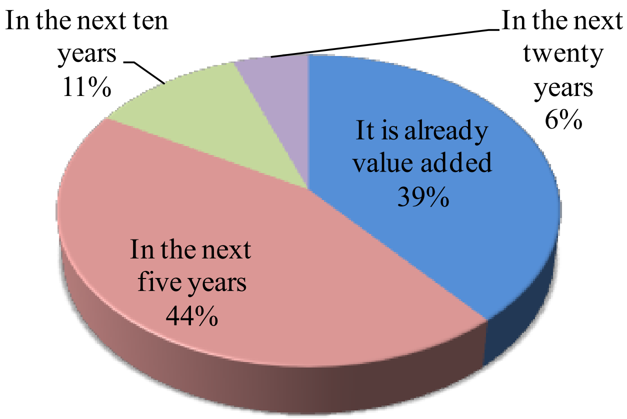


Two-Fluid Model (TFM)



Fullmer and Hrenya
(*Ann. Rev. Fluid Mech.*, 2017)

Expected value added through DEM



PSRI Industrial Survey
(Cocco et al., *Chem. Eng. Prog.*, 2017)

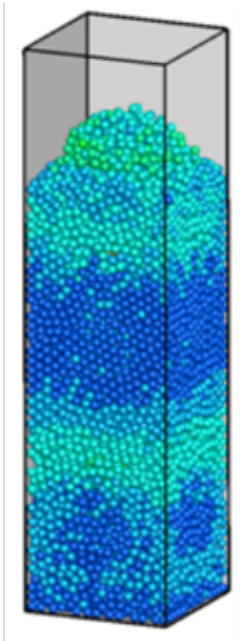
More detail, fewer closures ←

Less CPU time →

DEM:

a balance between computational overhead and sources of uncertainty

Background and motivation



CFD-DEM
$N_p \sim 10^5 - 10^7$

Laboratory
$N_p \sim 10^9 - 10^{10}$

Industry
$N_p \sim 10^{14}$

Goal: DEM application toward industrially relevant flows

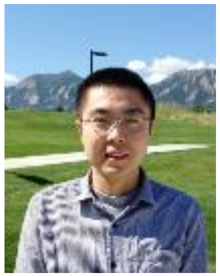
Challenges

- Speed $\Rightarrow$ Optimization & Algorithms (this talk)
- Results reliability $\Rightarrow$ Validation & Uncertainty Quantification

Team

University of Colorado
Chemical & Biological Engineering
DEM modeling of granular and gas-solid flows, MFIX


Christine Hrenya


Peiyuan Liu


Steven Dahl

University of Colorado
Research Computing
High-performance computing, CFD


Thomas Hauser


Shandong Lao


Aaron Holt

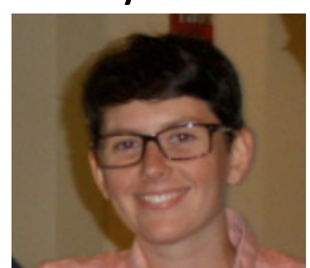
NREL
Computational Science
High-performance computing, CFD


Hari Sitaraman


Deepthi Vaidhynathan

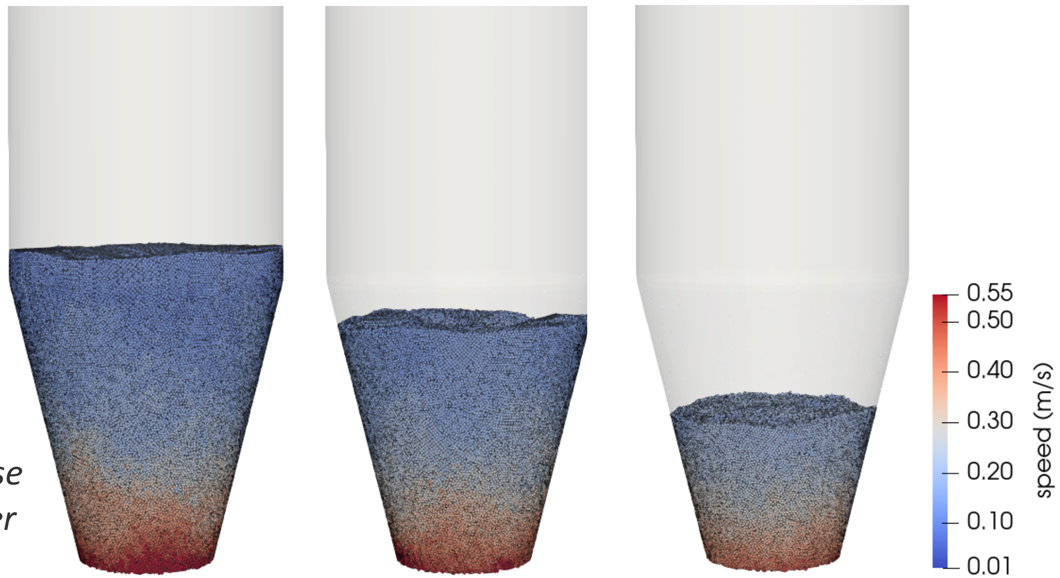

Ray Grout


Ray Cocco


Wyatt LaMarche

PSRI
Industrial Application and Experiments of Particle Flows

Recent accomplishments



1-million particle case shown here for easier visualiation

Hopper discharge simulation with 1 billion particles

- 889 nodes (32000 MPI ranks) on *Eagle*, NREL
- 17.5 hours
- 30 cm tall, 6 cm diameter, 100 μm size particles

- **~ 5X speed-up achieved with two important algorithmic changes**
  - Adaptive time integration
  - Spatial reordering of particles

Adaptive time integration for particles

Adaptive time integration for particles

- Time step controlled by collisional time scale
  - Depends on spring and damping constants
  - CFL is set at 0.02 in MFiX

$$\tau_p = \pi \left(\frac{k_n}{m_{\text{eff}}} - \frac{\eta_n^2}{4 m_{\text{eff}}^2} \right)^{-1/2}$$

$$\Delta t_p = CFL \; \tau_p$$

- Challenge with constant particle time-step
  - Fluid residence time scales with system dimensions
  - Particle time scale is intrinsic to phase properties

- For large-scale systems computational cost increases
  - More number of particles
  - More number of time steps

- **Can we adapt particle time-steps over the course of simulations?**
  - **Dilute systems may be able to advance with bigger steps**
  - **Reduction in collisions as particles slow down**

Adaptive time integration for particles

Default MFIX scheme (locally 2nd order)

$$\mathbf{f_i}^n = \mathbf{f_i}(\mathbf{v_i}^n, \mathbf{x_i}^n, t^n)$$

$$\mathbf{v_i}^{n+1} = \mathbf{v_i}^* = \mathbf{v_i}^n + \mathbf{f_i}^n \Delta t_p$$

$$\mathbf{x_i}^{n+1} = \mathbf{x_i}^n + \mathbf{v_i}^* \Delta t_p$$

Interaction forces

Velocity update

position update

Higher-order embedded scheme (locally 3rd order)

$$\left.\begin{aligned} \mathbf{v_i}^* &= \mathbf{v_i}^n + \mathbf{f_i}^n \Delta t_p \\ \mathbf{x_i}^* &= \mathbf{x_i}^n + \mathbf{v_i}^* \Delta t_p \end{aligned}\right\}$$

Same as default scheme

$$\mathbf{f_i}^* = \mathbf{f_i}(\mathbf{x_i}^*, \mathbf{v_i}^*, t + \Delta t_p)$$

A predicted force!

$$\left.\begin{aligned} {}^{\dagger}\mathbf{v_i}^{n+1} &= \mathbf{v_i}^n + \frac{1}{2}(\mathbf{f_i}^n + \mathbf{f_i}^*)\Delta t_p \\ {}^{\dagger}\mathbf{x_i}^{n+1} &= \mathbf{x_i}^n + \frac{1}{2}(\mathbf{v_i}^n + \mathbf{v_i}^*)\Delta t_p \end{aligned}\right\}$$

2nd order RK/trapezoidal scheme

Adaptive time integration for particles

Higher-order embedded scheme (locally 3rd order)

$$\mathbf{v_i}^* = \mathbf{v_i}^n + \mathbf{f_i}^n \Delta t_p$$

$$\mathbf{x_i}^* = \mathbf{x_i}^n + \mathbf{v_i}^* \Delta t_p$$

Original 2nd order scheme

$$\mathbf{f_i}^* = \mathbf{f_i}(\mathbf{x_i}^*, \mathbf{v_i}^*, t + \Delta t_p)$$

$${}^\dagger\mathbf{v_i}^{n+1} = \mathbf{v_i}^n + \frac{1}{2}(\mathbf{f_i}^n + \mathbf{f_i}^*)\Delta t_p$$

$${}^\dagger\mathbf{x_i}^{n+1} = \mathbf{x_i}^n + \frac{1}{2}(\mathbf{v_i}^n + \mathbf{v_i}^*)\Delta t_p$$

3rd order update

$$\epsilon^{n+1} = \max_i(|{}^\dagger\mathbf{v_i^{n+1}} - \mathbf{v_i^*}|^2)$$

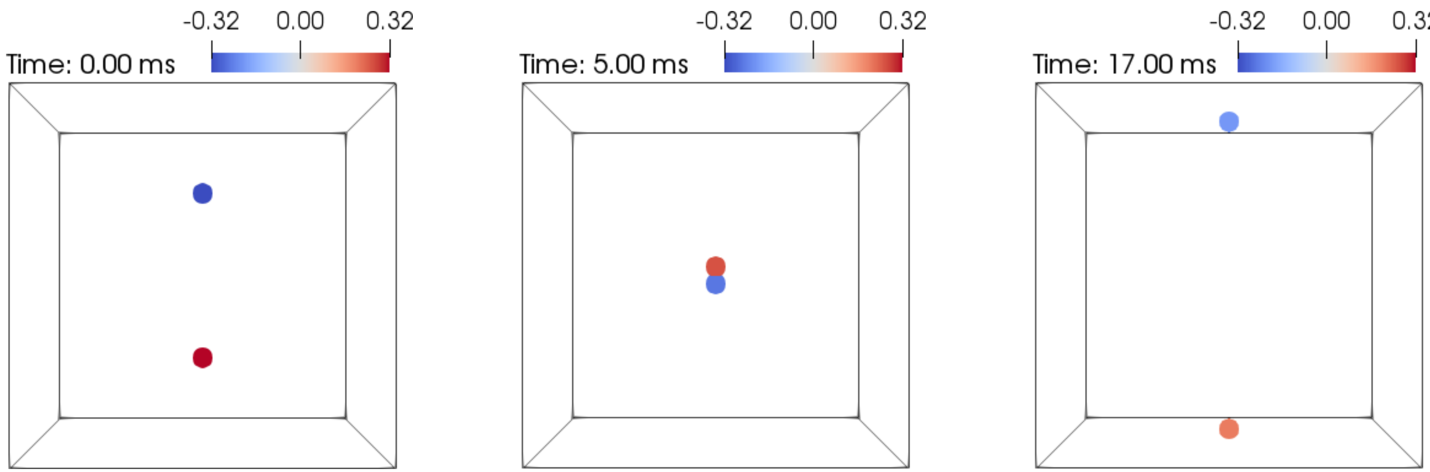
Error between solutions

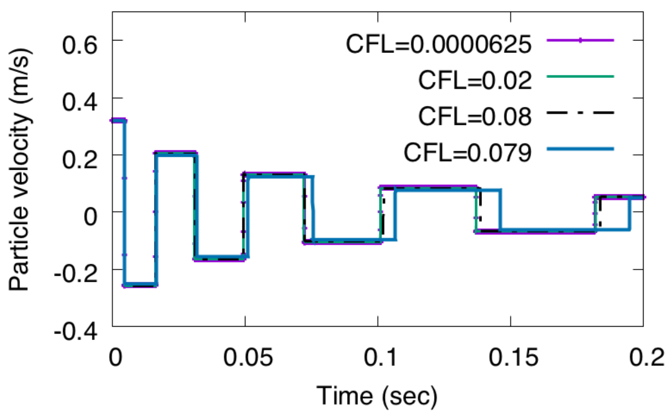
$$\Delta t_p^{n+1} = \max\left(\min\left(\frac{\epsilon_0}{\epsilon}\Delta t_p^n, \Delta t_p^{\max}\right), \Delta t_p^{\min}\right)$$

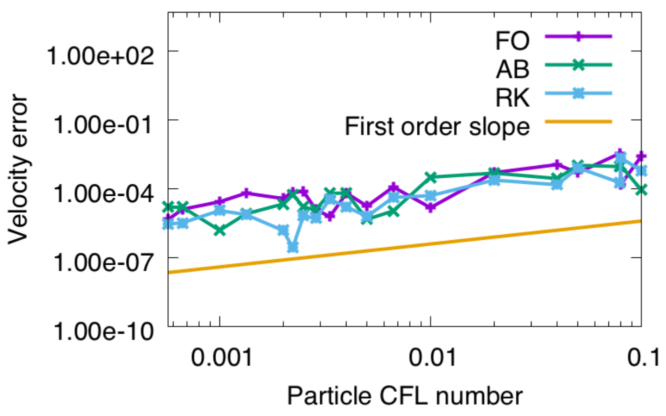
Time-step adaption

- Double force calculation can be avoided by reusing from previous time-step
- Facility to redo a timestep if the error jumps are high

Binary collision with default scheme

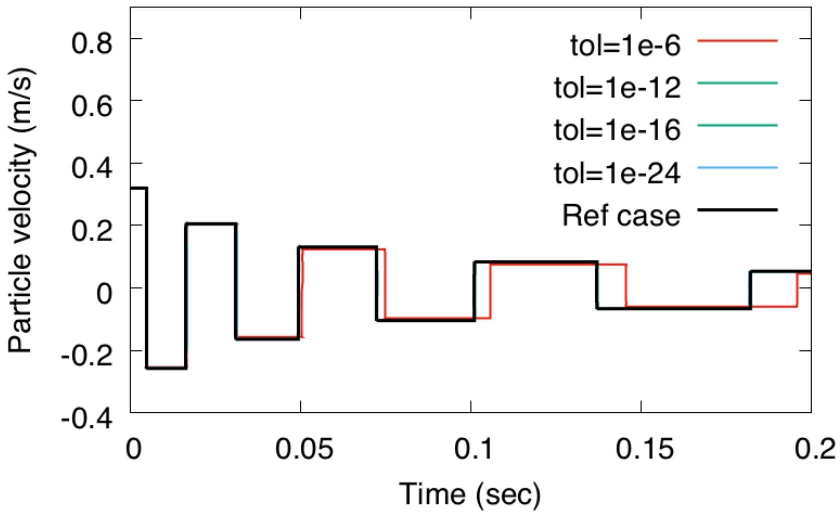


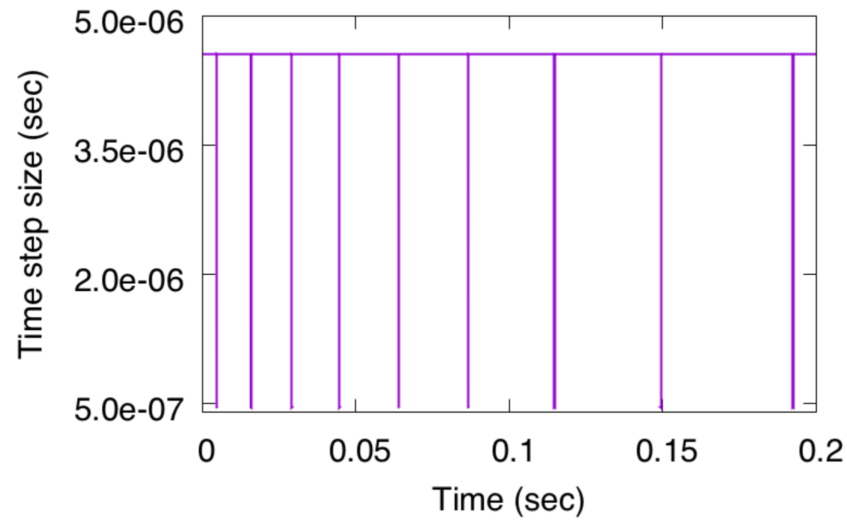




- Slight perturbation in CFL results in large deviations
- Euler-explicit, Adams-Bashforth, 3rd order RK give 1st order convergence
- Lower time-step does not guarantee lower error

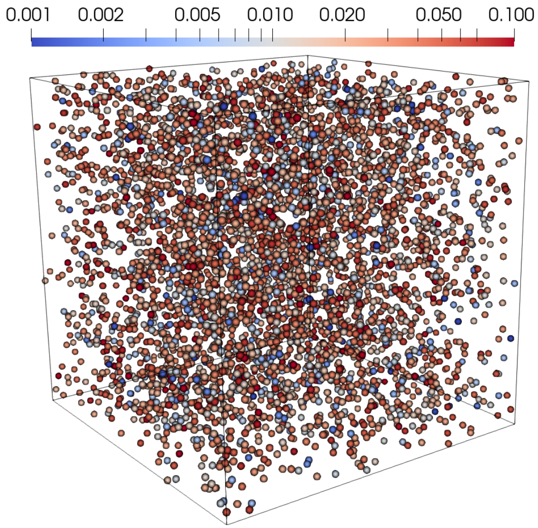
Binary collision with adaptive time-stepping

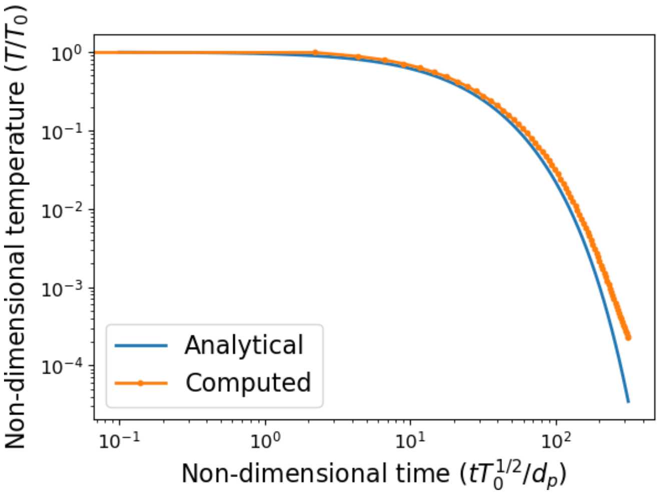


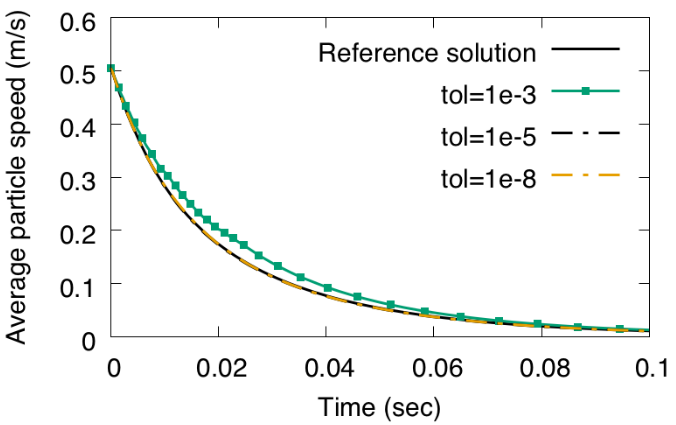


- Adaptive time-stepping match reference solution well
    - Deviations only at high error tolerances
- Time step size is adapted only during collision events
- Adaptive time-stepping was about 6X faster than default scheme

Homogenous cooling system (HCS)



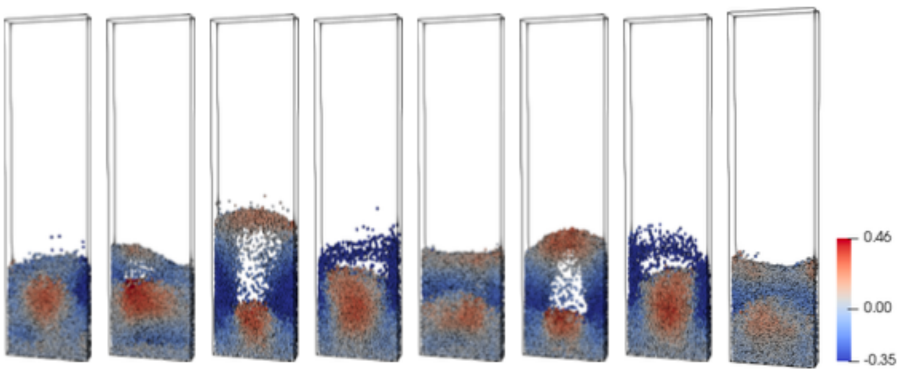


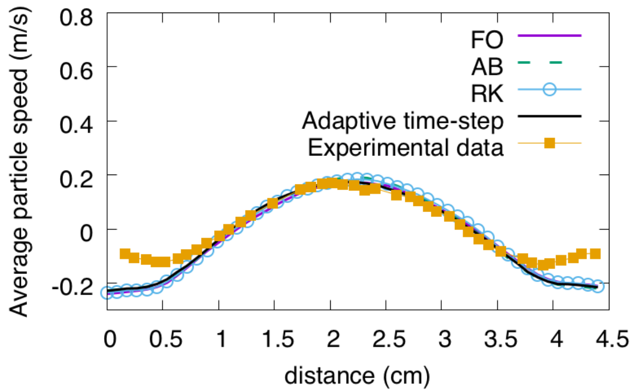


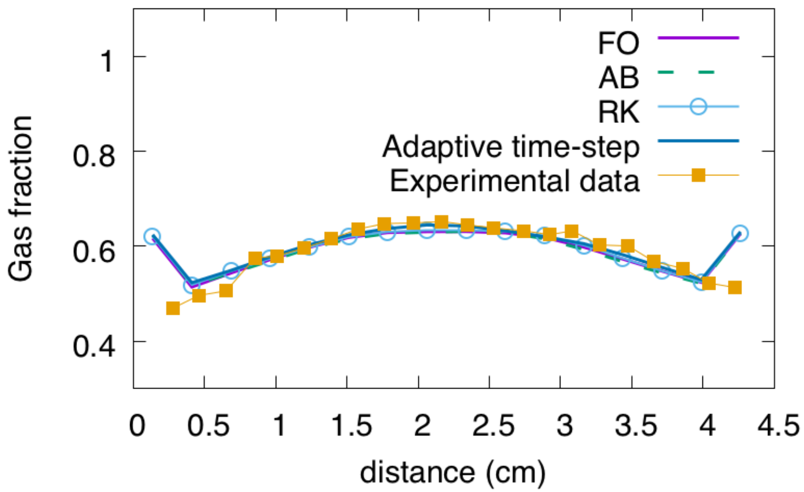
Case	Wall clock time (s)
Default scheme (CFL=0.01)	38.0
Default scheme (CFL=0.02)	19
Adaptive scheme (tol=1e-5)	6.5
Adaptive scheme (tol=1e-8)	16.2

- Adaptive time-stepping gave an improvement of ~ 3X while reproducing the reference solution

Fluidized bed



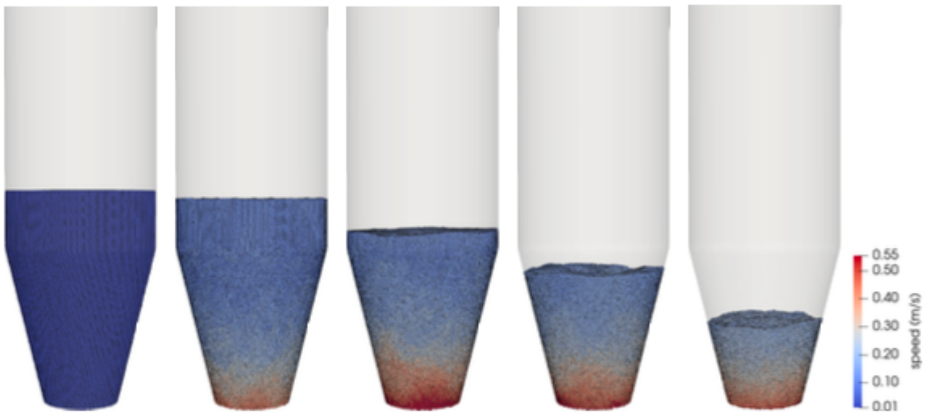


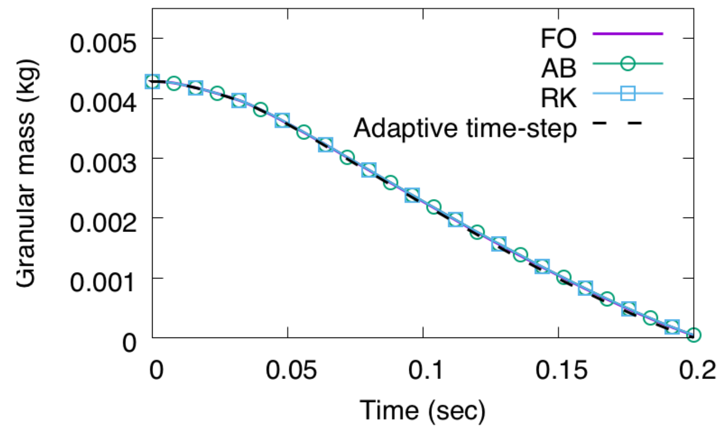


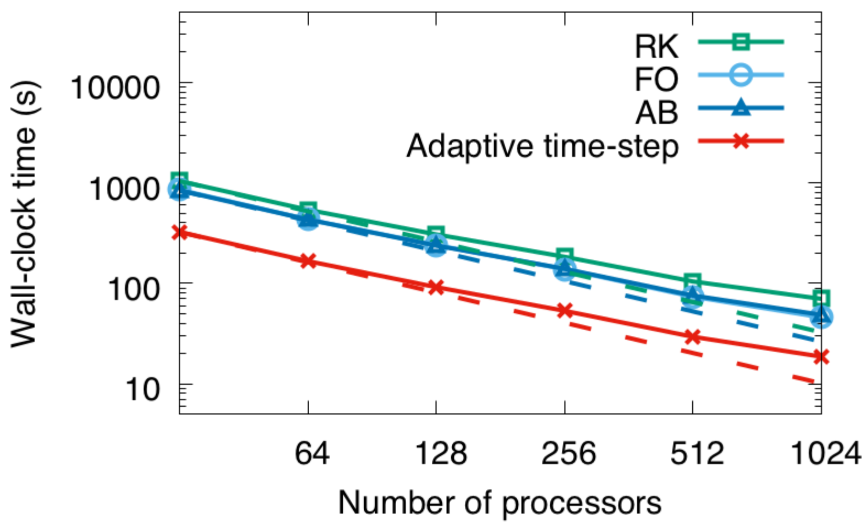
- Fluidized bed validation case with experiments by Muller et al.*
- Adaptive time-stepping gave an
    - improvement of ~ 1.7X while being validated with experiments
    - Lower tolerance values did not give improvements
    - Setting tolerances is important

*C. R. Muller, S. A. Scott, D. J. Holland, B. C. Clarke, A. J. Sederman, J. S. Dennis, L. F. Gladden, Validation of a discrete element model using magnetic resonance measurements, Particuology 7 (4) (2009) 297–306.

Hopper discharge



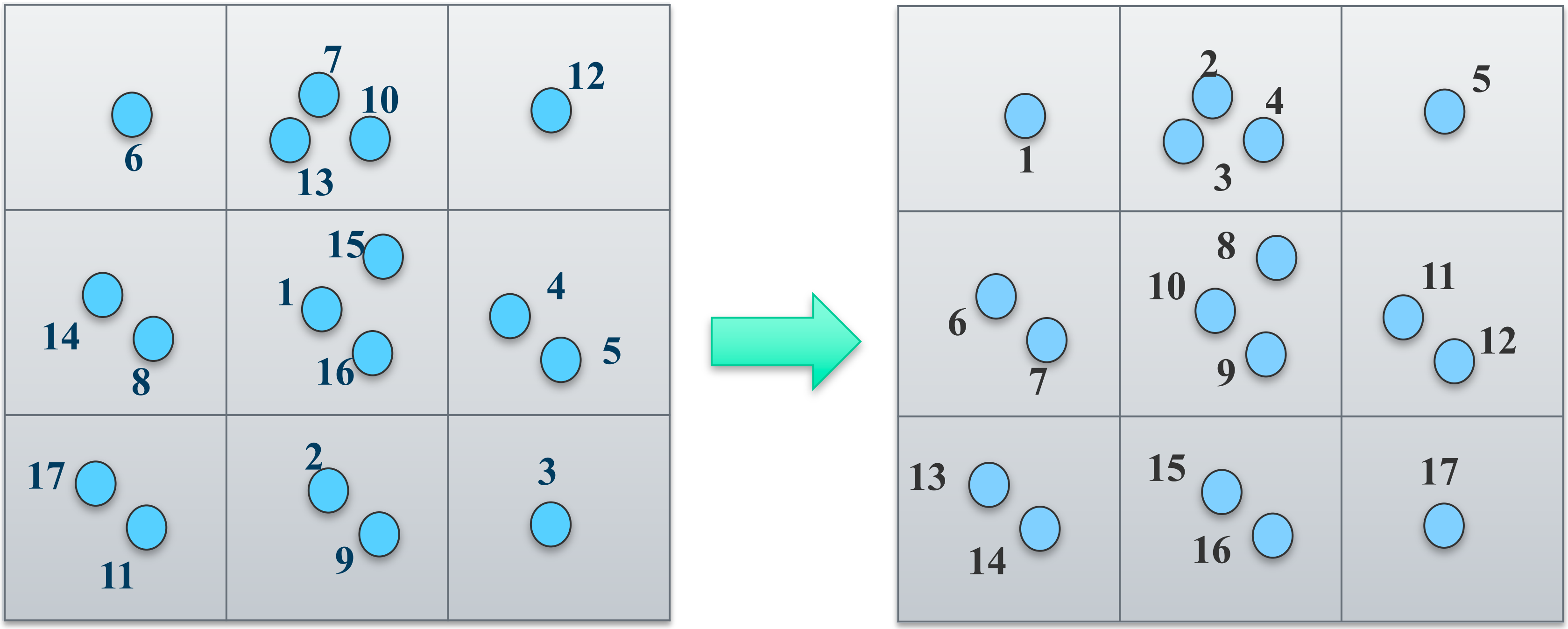




- Pure DEM case with 1 mill. particles
- Adaptive time-stepping gave an
  - improvement of ~ 3X while reproducing identical solutions
  - Lower tolerance values did not give improvements
  - Exhibits good parallel scaling

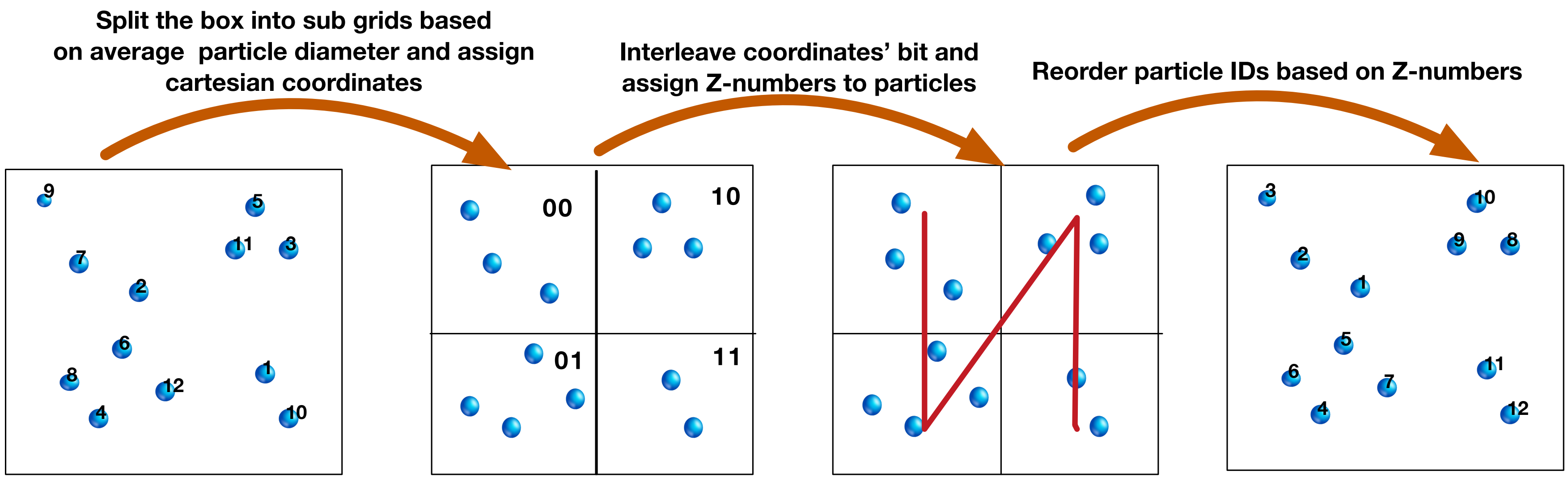
Spatial reordering of particles

Spatial reordering - main idea



- Particles are stored in memory agnostic to their location in space
- Spatial reordering will increase spatial data locality in memory while
  - ➢ Building neighbor lists
  - ➢ Calculating inter particle forces – collisions
  - ➢ Creating send/receive buffers
- **Two strategies – Morton ordering of uniform/kD-tree partitions**

Morton ordering of uniform partitions



1	2	3	4	5	6	7	8	9	10	11	12

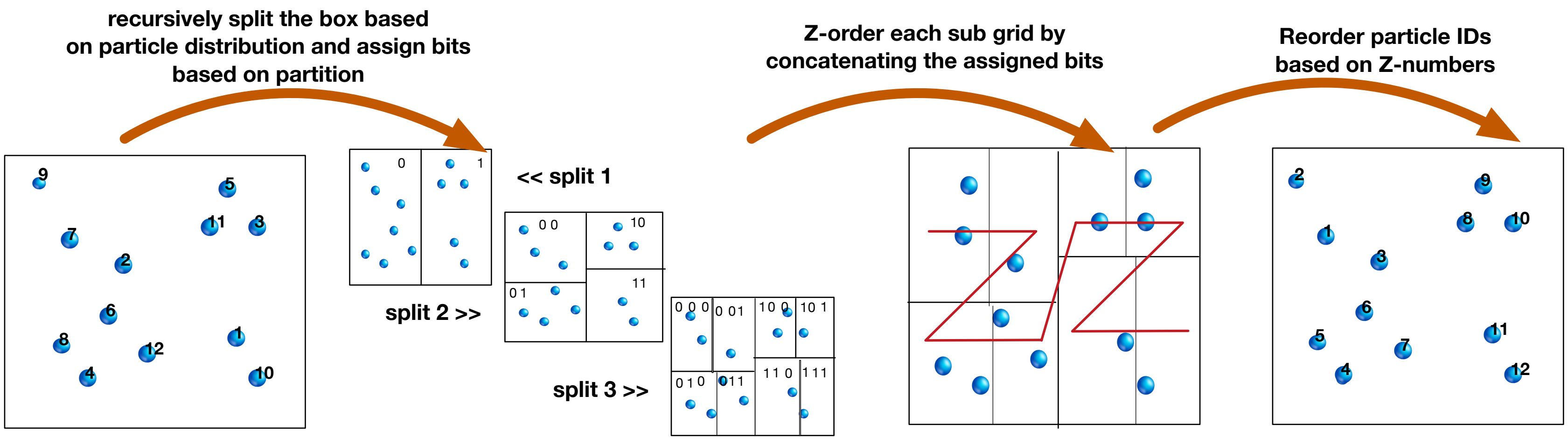
Default particle data structure

2	7	9	4	6	8	12	3	5	11	1	10

Spatially reordered particle data structure

- Each box of co-located particles is divided into uniform partitions
- Particles in each partition assigned a Z-number
    - For example, coordinate of (3,4) in an 8 x 8 grid is 011010 = 26
- Particles are sorted and reordered based on their Z number

Spatial reordering of particles



1	2	3	4	5	6	7	8	9	10	11	12

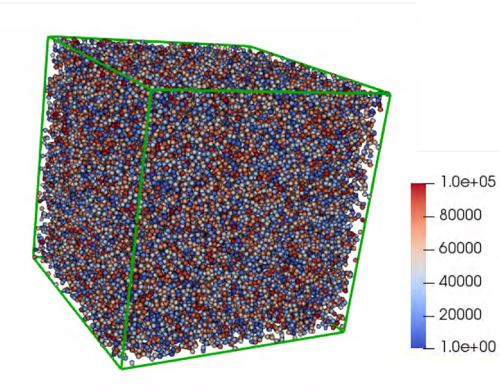
Default particle data structure

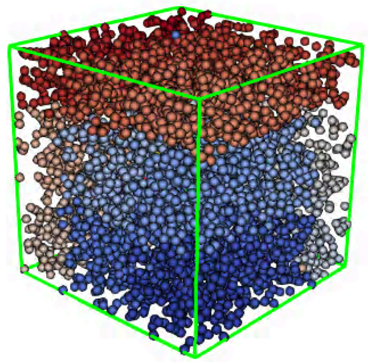
7	9	2	4	8	6	12	11	3	5	1	10

Spatially reordered particle data structure

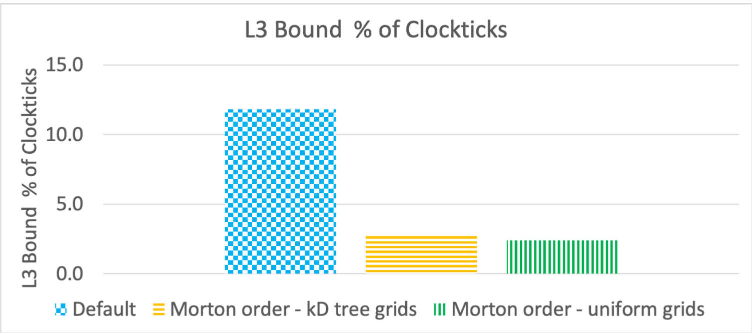
- Each box of co-located particles is divided into kD-tree based partitions
- Concatenated binary coordinates gives the Z number
- Particles are sorted and reordered based on their Z number
- Useful when there is non-uniform distribution of particles in space

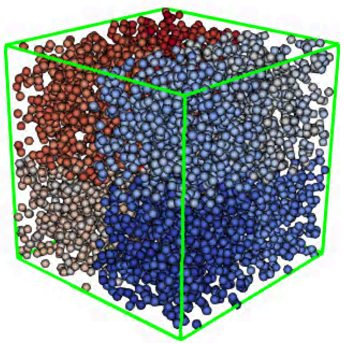
Homogenous cooling system case



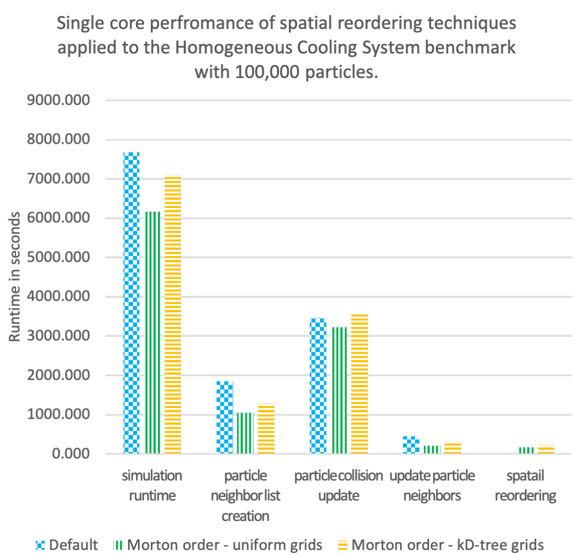


kD-tree partitions



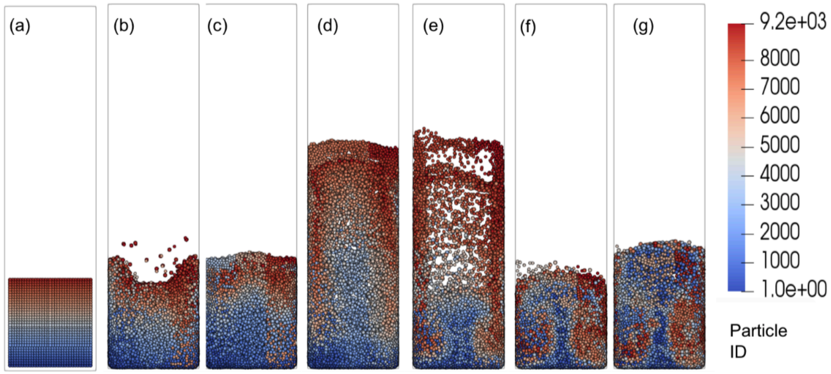


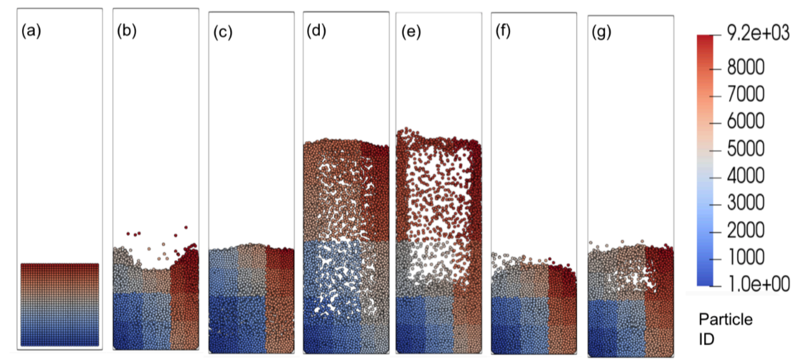
Uniform partitions

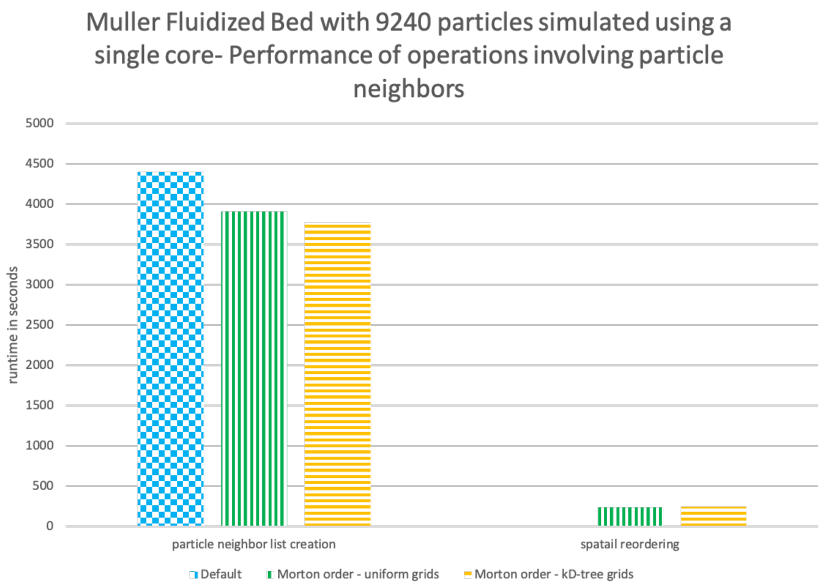


- Overall improvement of ~ 20% with reordering
- Reordering routines take less than 2% of total run time
- Neighbor list creation time is reduced by 50%

Fluidized bed case





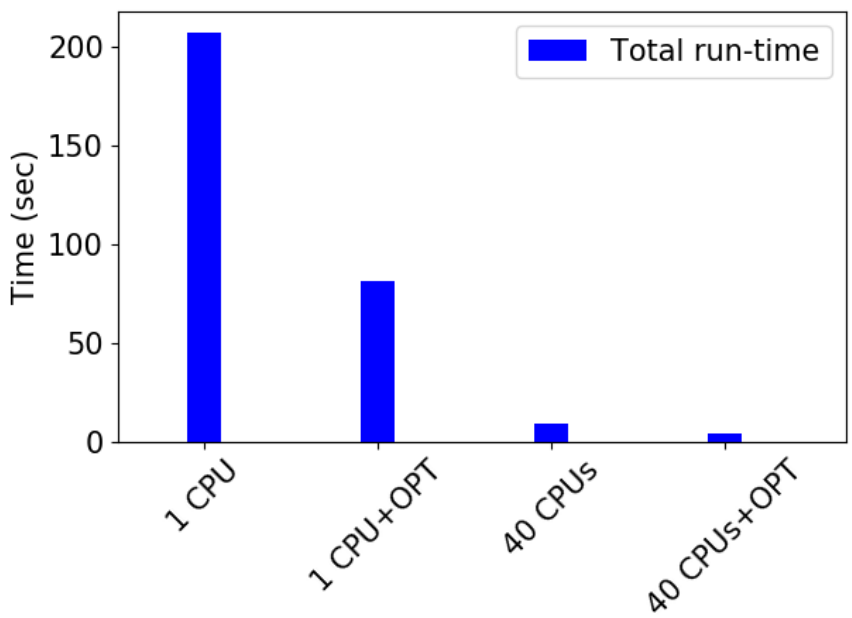


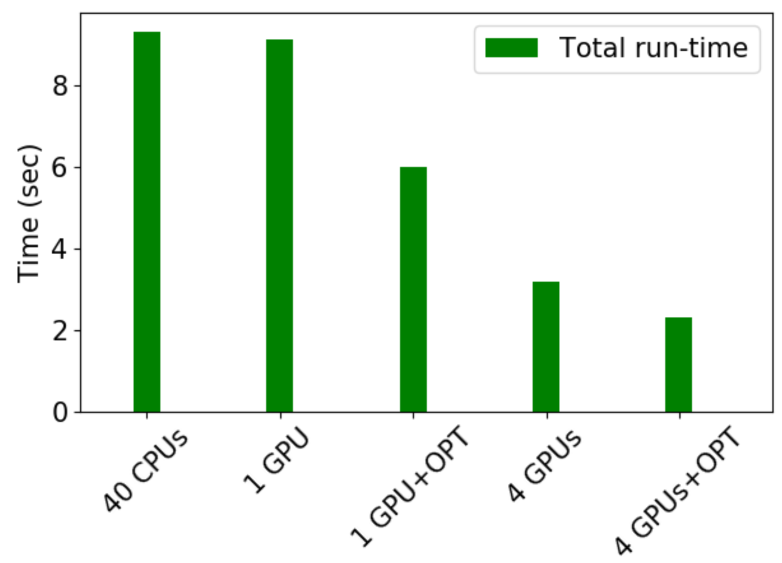
- Non-uniform distribution of particles during bubbling
- kD-tree based partitioning gives better performance
- 20% improvement in building neighbor lists
- Improvements are greater with larger number of particles.

GPU performance improvements

GPU performance

- Next milestone - Riser simulations with 1 billion particles
- Testing Riser simulations with 2 million particles on Frontera (TACC)





- The use of 1 GPU is equivalent to CPU run with 40 MPI ranks
- Use of 4 GPUs on a single node gives about 3X performance improvement over 40 CPUs
- Use of our optimizations gives 4X improvement

Summary

- Two important algorithmic changes
  - **Adaptive time integration of particles**
    - Use of an embedded higher/lower order scheme
    - Time step is adapted based on error
      - Increased performance, detect/correct failures
    - Sitaraman et al.," An error-controlled adaptive time-stepping method for particle advancement in coupled CFD-DEM simulation", Powder Tech., (in review)
    - Future work: explore space of tolerance and accuracy
  - **Spatial reordering of particles**
    - Reordering particles to achieve better memory performance
    - Two approaches using Morton ordering
      - Uniform and kD tree partitions
    - 10-20% speed-up for cases with ~ 100k particles
    - Vaidhynathan et al.," Memory optimization for particle access in CFD-DEM Simulations", (under internal review)
    - Future work: dynamic selection of method at run-time

Acknowledgements

- Funding from DOE, Office of Fossil Energy
- Ann Almgren from LBNL and Jordan Musser, William Fullmer from NETL for MFIX-Exa code, funded by ASCR Exascale Computing Project (ECP)

Questions/Comments?