OpenMP parallelism for fluid-particulate system

Amit R. Amritkar

Prof. Danesh Tafti

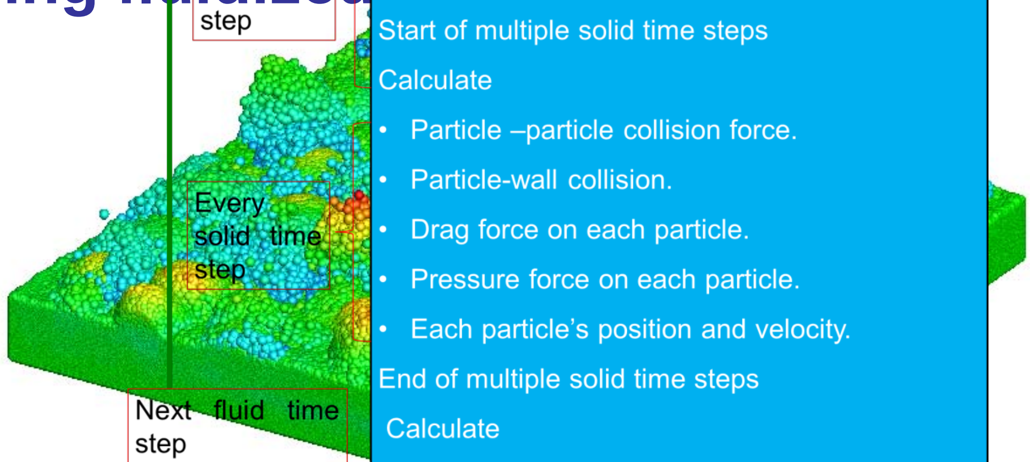
Handan Liu

Outline

- CFD-DEM and parallelization
- Algorithm
- Results
  - Validation
  - Scaling study
- Future work

Coupled fluid-particulate system

- **Procedure of Fluid(CFD) + Particulate(DEM) coupling**
- **Example**
  - **Circulating fluidize**
  - **Bubbling fluidized bed**



CFD SOLVER (GENIDLEST+DEM subroutine)

Calculate drag on solids due to gas (F_{drag})

Drag on fluid due to solid is ($-F_{drag}$)

Calculate fluid velocity

Calculate fluid pressure etc.

Every fluid time step

Start of multiple solid time steps

Calculate

- Particle –particle collision force.
- Particle-wall collision.
- Drag force on each particle.
- Pressure force on each particle.
- Each particle's position and velocity.

Every solid time step

End of multiple solid time steps

Calculate

- Volume fraction in each fluid cell.
- Volume averaged solid velocity.

Next fluid time step

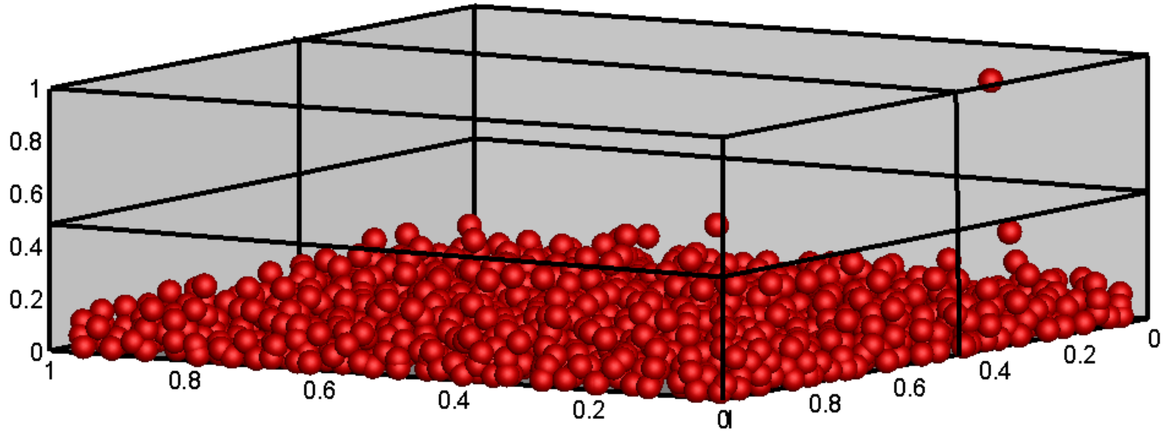
Parallelization Techniques

- **Fluid phase (CFD code) parallelism**
  - **Grid based field equations mapped to domain decomposition**
- **Particulate phase parallelism**
  - **N-body type of computations**

1. **Mirror domain technique**
   - **Each CPU has copy of all the particle data but only works on part of it**
     - No communication during computations
     - Large memory foot print
2. **Particle subset method**
   - **Particle workload is evenly divided amongst CPUs**
     - Ideal load balancing
     - Communication during computations
3. **Domain decomposition**
   - **Spatial decomposition irrespective of number of particles in each domain**
     - Easy to implement
     - Poor load balancing

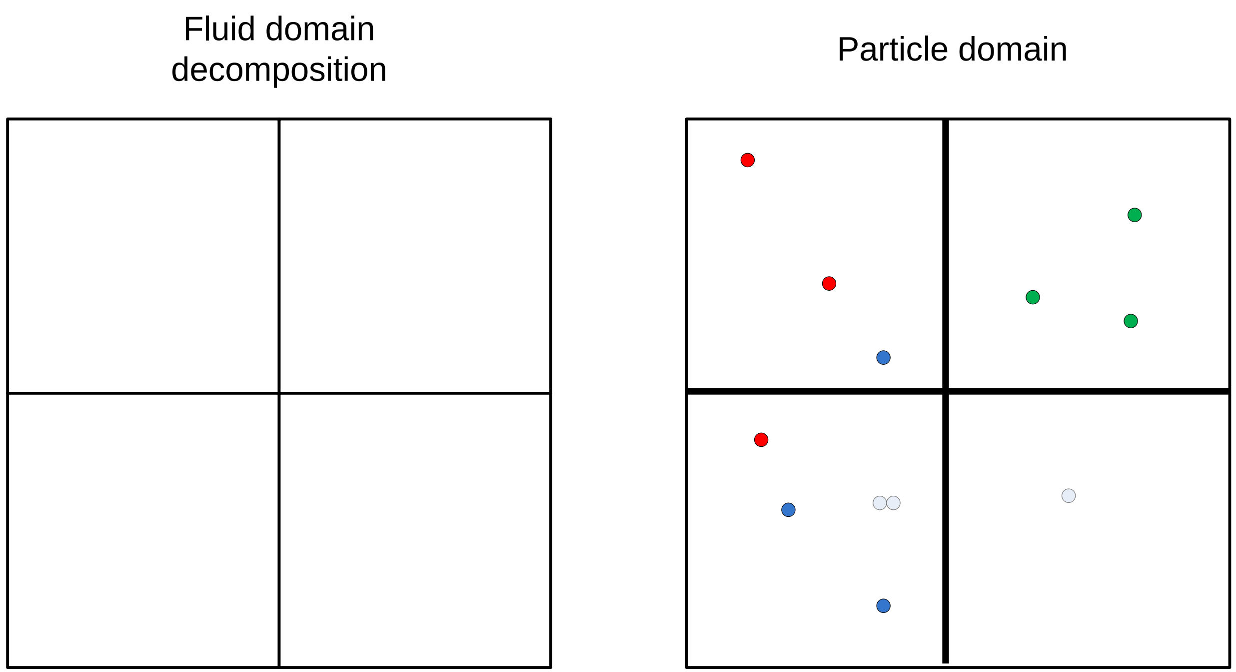
Parallelization of Eulerian-Lagrangian system

- **Parallelization of fluid-particulate systems**
  - **Domain decomposition**
    - **Poor load balancing for particulate system**



Parallelism for fluid-particulate system

- **Fluid – domain decomposition**
- **Particles – N-body decomposition**
  - **Particle subset method**



Implementation of multi-mode parallelism

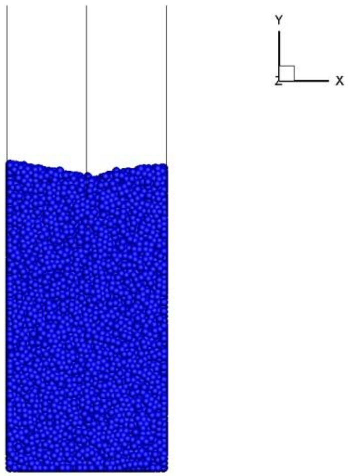
- Implementation in the MPI framework
  - All particle data needs to be gathered onto a single processor and evenly scattered across all the processors
  - After the particulate phase calculations, the drag forces need to be gathered and scattered to perform the fluid field calculation
  - Entire data structure is reshuffled twice every time step
- MPI implementation is inefficient for multi-mode parallelism
- MPI has high programming, development, and maintenance costs

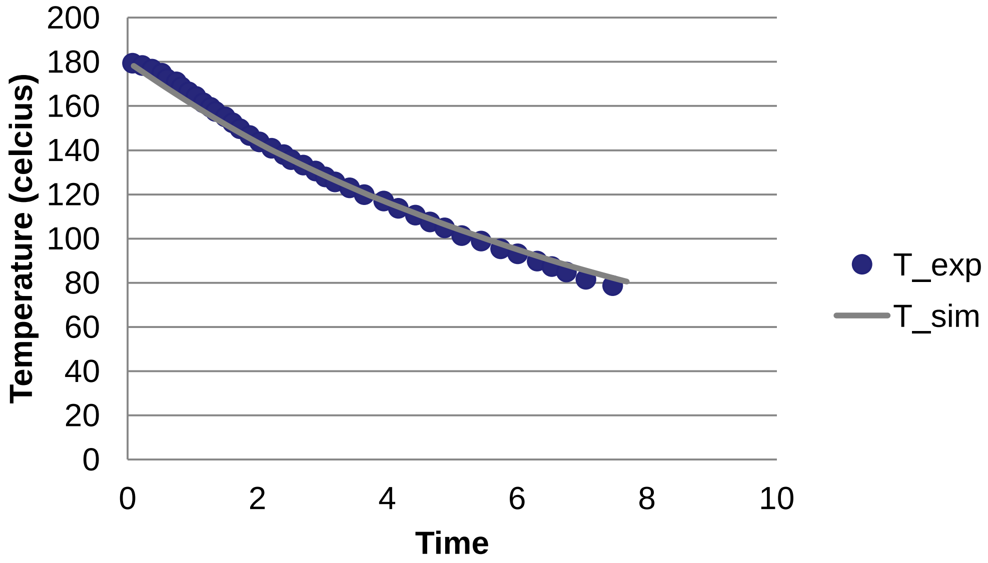
Implementation of CFD-DEM parallelism

- **Motivation for using OpenMP**
  - **OpenMP has flexible programming model**
    - **Domain decomposed (fluid phase)**
    - **Particle Subset (dispersed phase)**
  - **OpenMP is relatively easy to implement**
  - **OpenMP will have extensions (OpenACC) to support heterogeneous computing**
    - **GPGPU**
    - **Intel – Many Integrated Core (MIC)**
    - **AMD Fusion**

Micro scale heat transfer validation of OpenMP parallel implementation

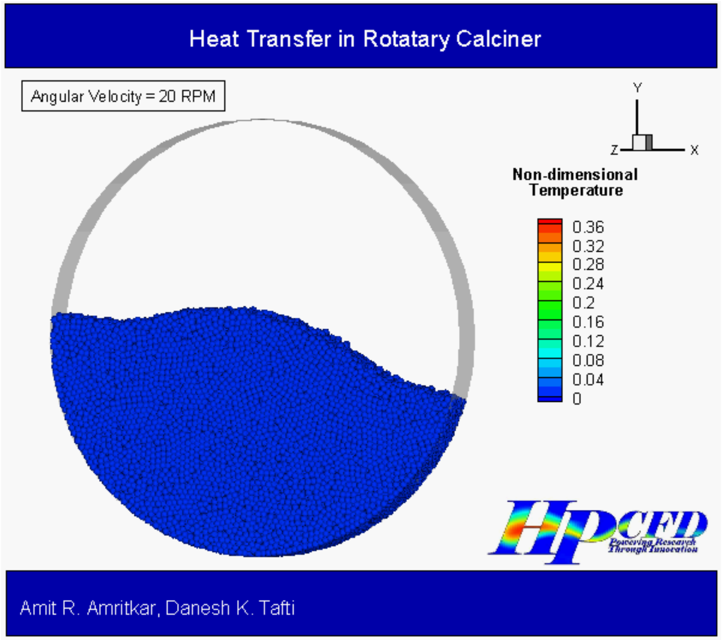
- **Validation study # 1**
  - **Hot particle is inserted in a dense bed (Collier et al., 2004)**
    - **3D periodic boundary condition**
    - **16,000 particles on 2 processors**
    - **0.58*u_mf**

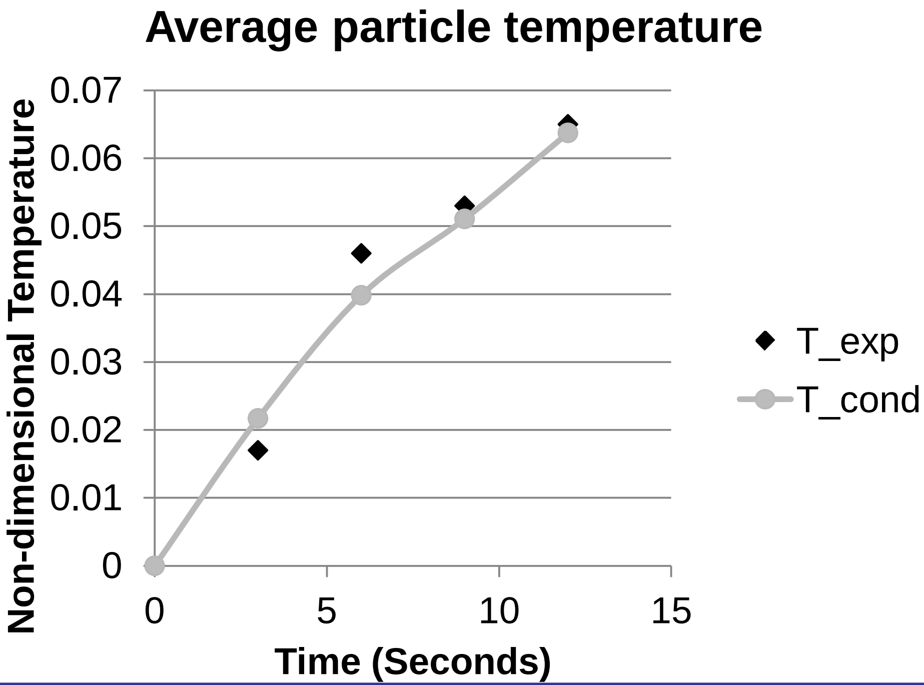




Micro scale heat transfer validation of OpenMP parallel implementation

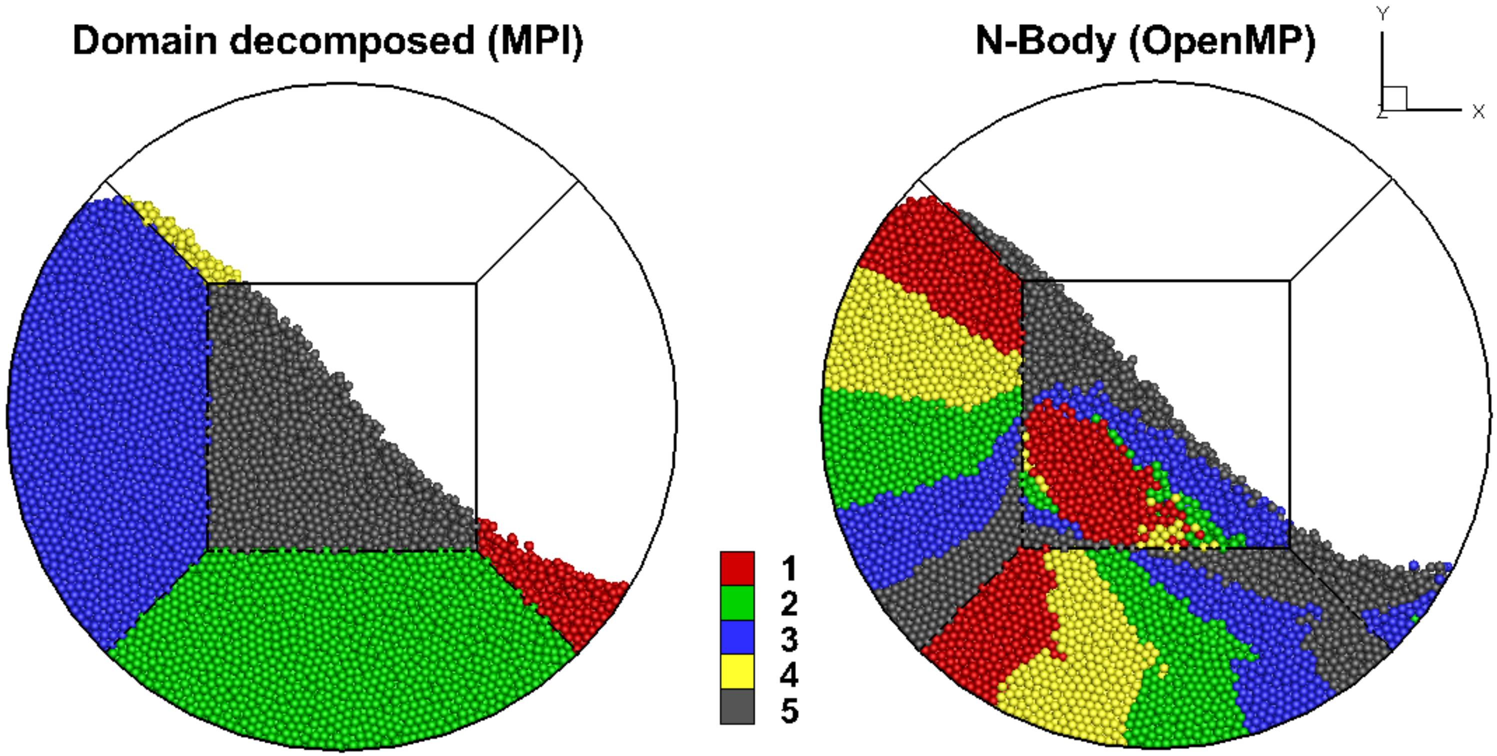
- **Validation study # 2**
  - **Heat transfer in rotary calciner (Chaudhuri et al., 2010)**
    - **Heated walls (curved surface) with particles rotating in a cylindrical drum**
    - **20 RPM with 20,000 particles**





Comparison of particle workload division in a rotary calciner

- **Load imbalance in domain decomposed mode**



Rotary calciner simulation

- **Calculations on HokieOne (SGI – UV100) shared memory machine**
- **10 milliseconds (1000 time steps) of simulation at 20 RPM**

	Number of Mesh Blocks	OpenMP	MPI
Time (seconds)	5 (20,000 particles)	159	329
Time (seconds)	25 (100,000 particles)	275.5	443

Performance of MFIX with OpenMP parallelism

- **Evaluation of OpenMP implementation performance for DEM**
  - **Top five most time-consuming subroutines**
calc_force_des
drag_fgs
cfnewvalues
leq_msolve
des_time_march

Threads	Total wall clock Time (s)	Time in calculating drag_fgs (s)
1	128	36
2	125	39
4	94	37
8	83	38

Future Work

- **Improve OpenMP performance**
  - **Immediate work will focus on parallelizing 'drag_fgs' of the OpenMP implementation to obtain higher performance**
  - **Investigate key subroutines and evaluate modifications for OpenMP parallelism to be effective**
- **Investigate OpenMP scalability for larger particle counts on tens of threads**

Summary

- **OpenMP is very effective in modeling fluid-particulate systems**
  - **Comparative ease of programming**
  - **Dynamic load balancing**
  - **Future support for accelerator (GPU/MIC)**

- ***MFIX-DES parallelization***
  - ***Drag force calculations***
  - ***Large scale computations***

Thank you

- **Questions?**