



UNIVERSITY *of* DELAWARE

Parallel Implementation and Scalability Analysis of a Turbulent Particle Laden Flow Code

Particle Hydrodynamic Interactions
2D Domain Decomposition

Orlando Ayala

Acknowledgments:

H. Parishani, L. Liu, Dr. L.-P. Wang,

Dr. C. Torres, Dr. L. Rossi,

C. Andersen, H. Gao, Dr. B. Rosa

National Science Foundation

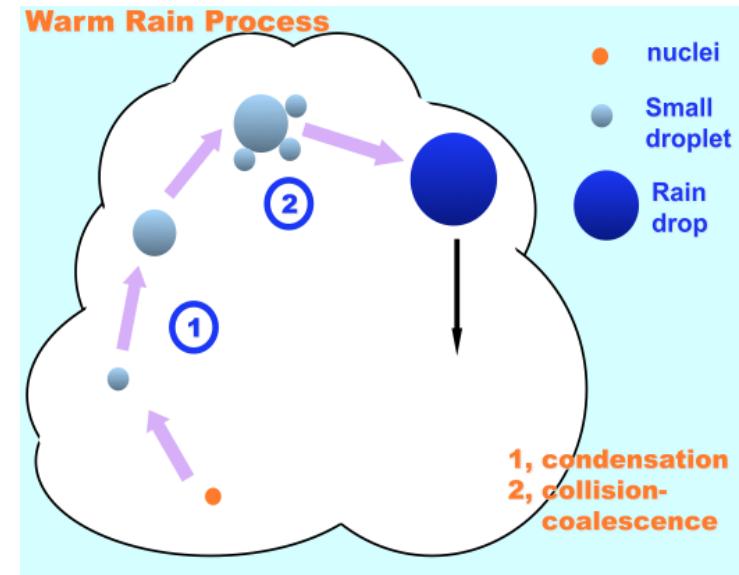
National Center for Atmospheric Research



Workshop on Multiphase
Turbulent Flows in the
Atmosphere and Ocean
August 13-17, 2012



How does air turbulence affect the collision rates of cloud droplets?





Resolve ~5 orders of magnitude in length scales

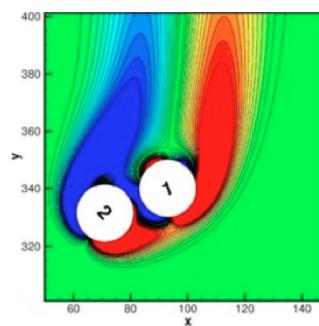
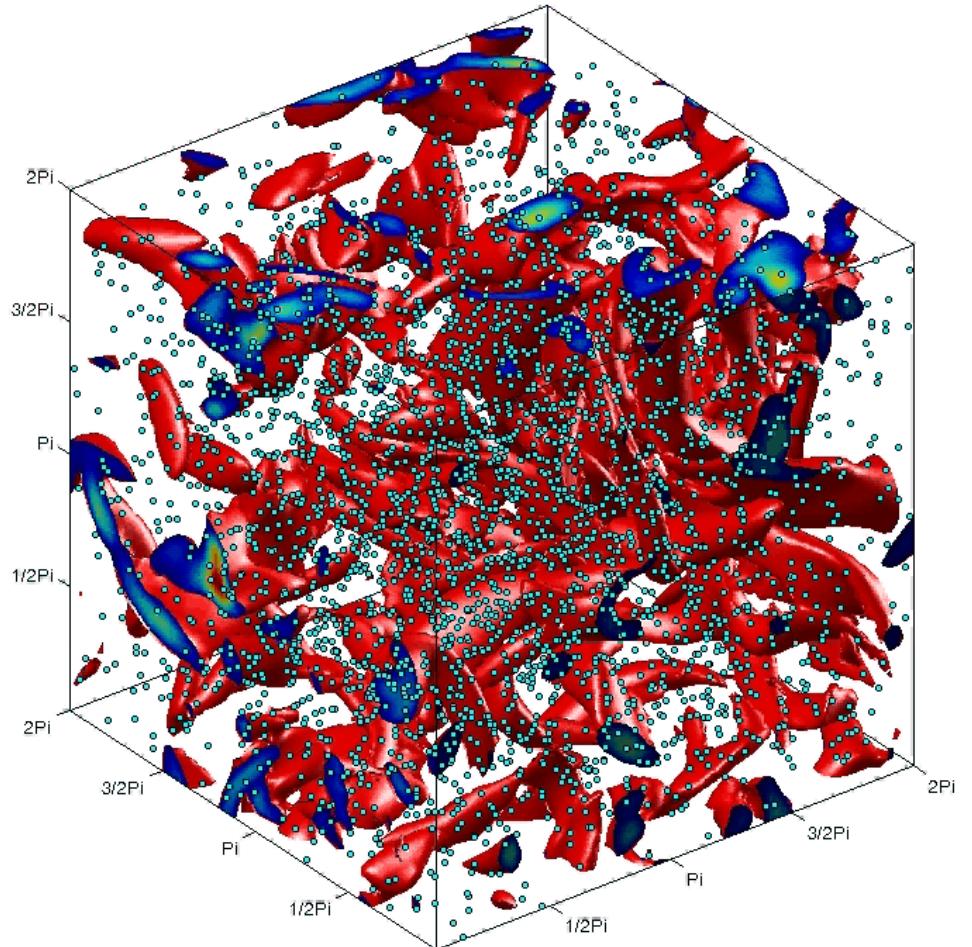
100 to 1000 mm

↑ Resolved numerically

$\Delta x \sim 1$ to 2 mm

↓ Treated analytically

$a \sim 0.02$ to 0.05 mm





Hybrid Direct Numerical Simulation

ASSUMPTIONS: 1) There is scale separation between Δx and localized disturbance flows; 2) The disturbance flow is Stokes flow \vec{u}_{STOKES} .

$$\frac{d\vec{V}^{(\alpha)}(t)}{dt} = \frac{\left[\vec{U}(\vec{Y}^{(\alpha)}(t), t) + \vec{u}(\vec{Y}^{(\alpha)}, t) \right] - \vec{V}^{(\alpha)}(t)}{\tau_p} + \vec{g}$$
$$\frac{d\vec{Y}^{(\alpha)}(t)}{dt} = \vec{V}^{(\alpha)}(t) \quad \text{here } \rho_p \gg \rho_f$$

$$\vec{u}(\vec{Y}^{(\alpha)}, t) = \sum_{\substack{m=1 \\ m \neq \alpha}}^{N_p} \vec{u}_{STOKES} \left(\frac{\vec{Y}^{(\alpha)} - \vec{Y}^{(m)}}{a_m}, \vec{V}^{(m)} - \vec{U}(\vec{Y}^{(m)}, t) - \vec{u}(\vec{Y}^{(m)}, t) \right)$$

A large linear system of $3N_p$ equations

$\vec{U}(\vec{x}, t)$ By interpolating in a box turbulence.
NS equations solved using a **Pseudo Spectral Method** (FFT) with large scale forcing.



Implications of increasing DNS grid resolutions

Assume a LWC of 1 g/m^3 .

Half $a_1 = 30 \mu\text{m}$ and half $a_2 = 20 \mu\text{m}$

	N	R_λ	Domain size (cm) ($400 \text{ cm}^2/\text{s}^3$)	Number of droplets
Published	32	23.5	4.2	1.0×10^3
	64	43.3	8.4	8.0×10^3
	128	74.6	16.9	6.6×10^4
On-going	256	123.0	34.0	5.4×10^5
	512	204.0	68.9	4.5×10^6
Target	1024	324.0	137.0	3.5×10^7

OpenMP

$$\text{Memory} \approx (9 \times N^3 + 4 \times 3 \times N_p) \times 8 \text{ bytes} \approx 74 \text{ Gbytes}$$

$$\text{Execution time} \approx 1379 \frac{s}{\text{timestep}} \times 14,000 \frac{\text{timestep}}{T_e} \times 20T_e \approx 4469 \text{ days} \quad \textcolor{red}{\sim 12 \text{ years!!}}$$

MPI PARALLELIZATION - Petascale Supercomputers

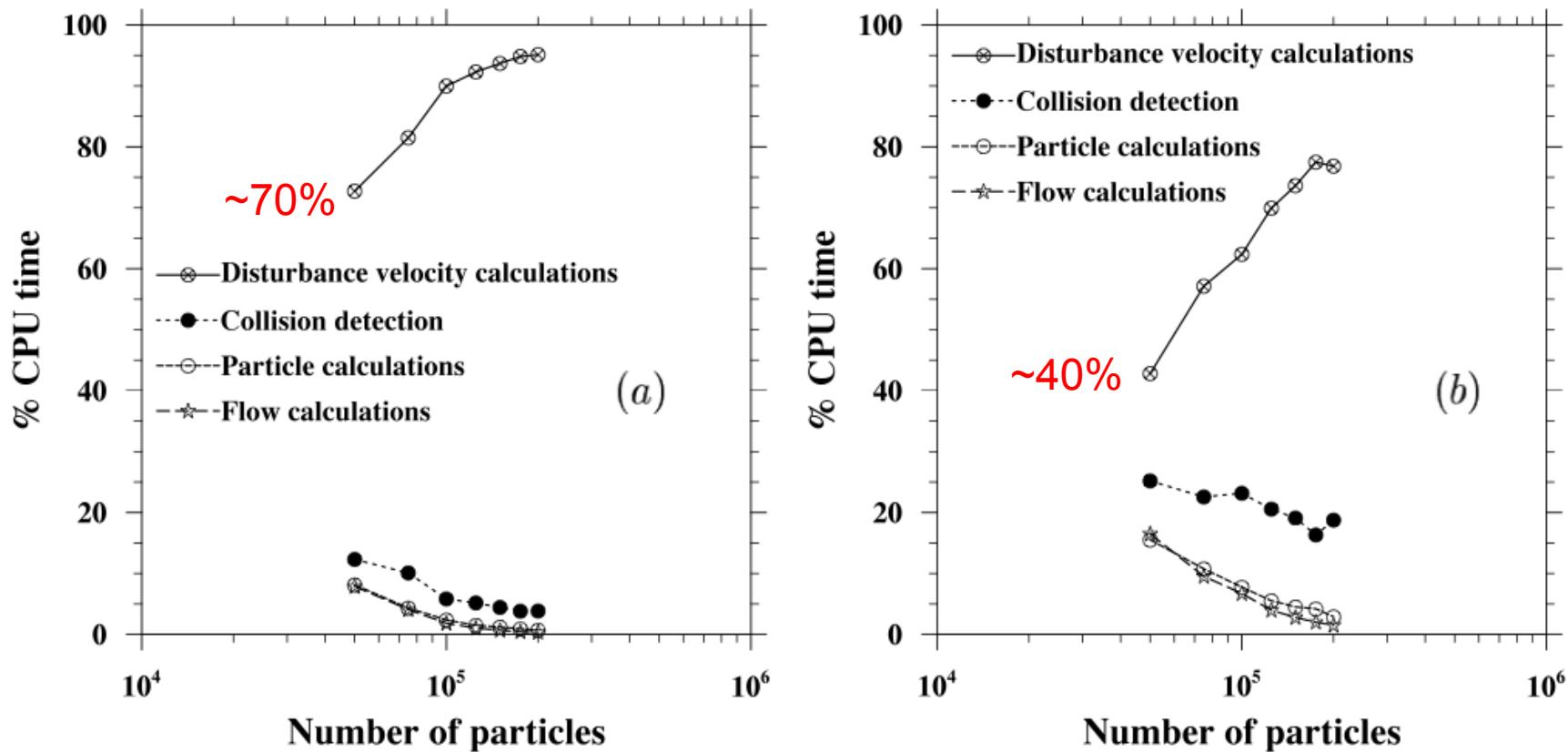


Figure 3.12: CPU time percentage of the total time used by each of the four computational tasks. (a) Monodisperse case using 32 processors, (b) Bidisperse case in turbulent flow using 32 processors. The bidisperse case with no turbulent flow is not shown as it presented same trend as figure (b).

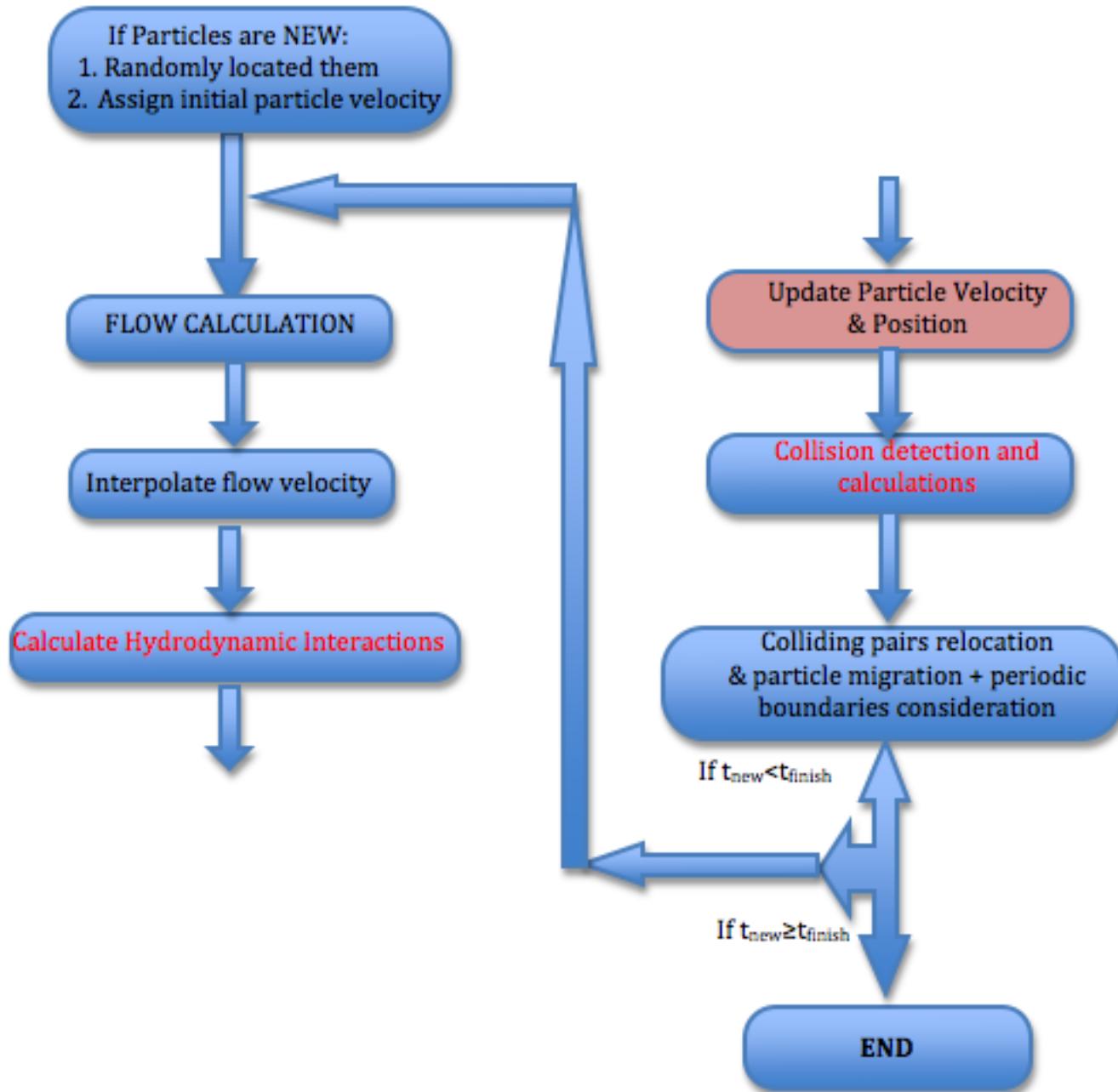


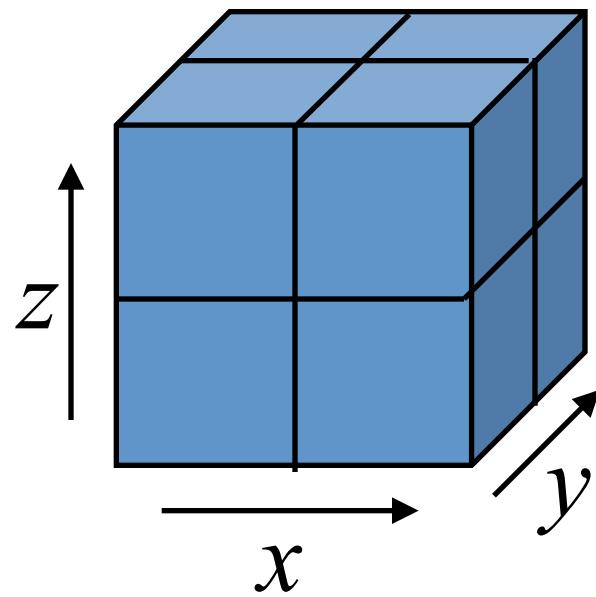
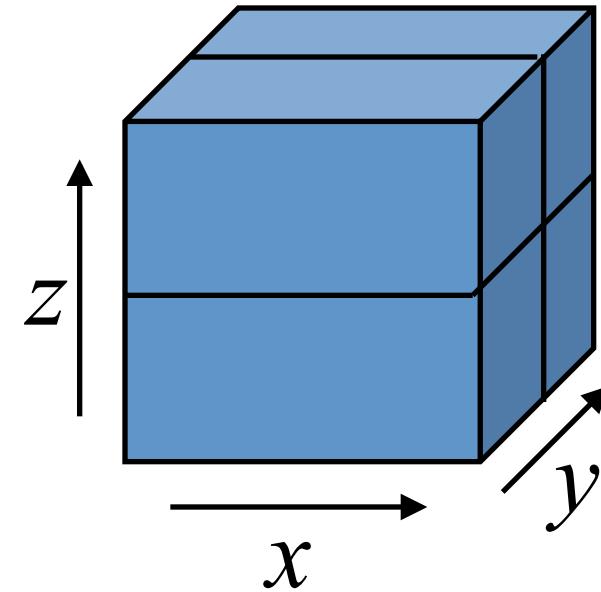
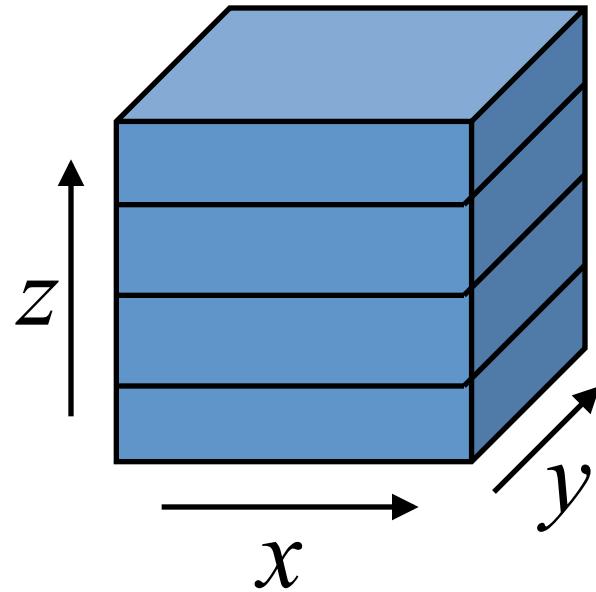
NSF OCI-0904534

Collaborative Research: PetaApps

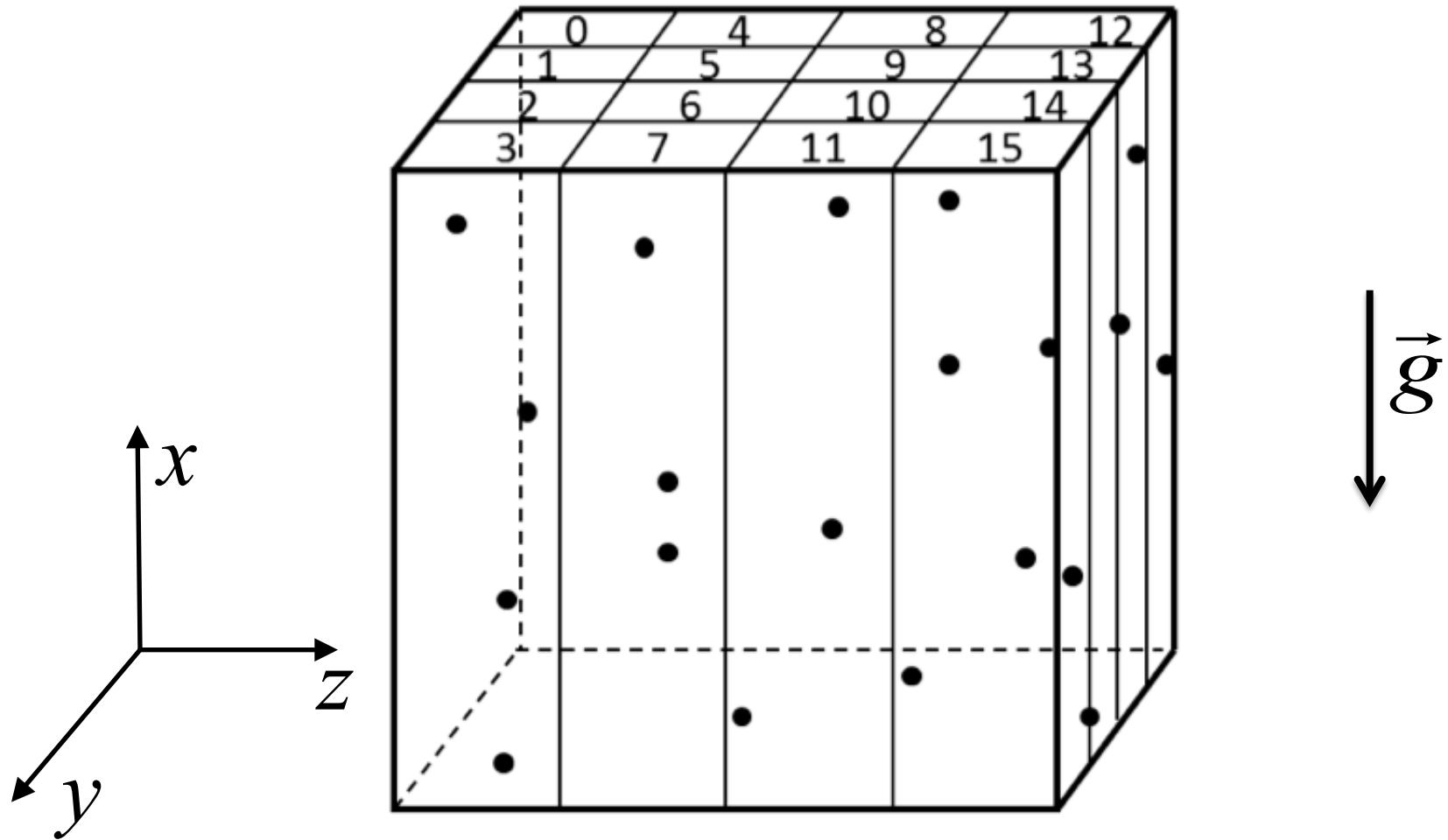
Enabling Multiscale Modeling of
Turbulent Clouds on Petascale
Computers

TO BRIDGE THE GAP BETWEEN LES AND DNS

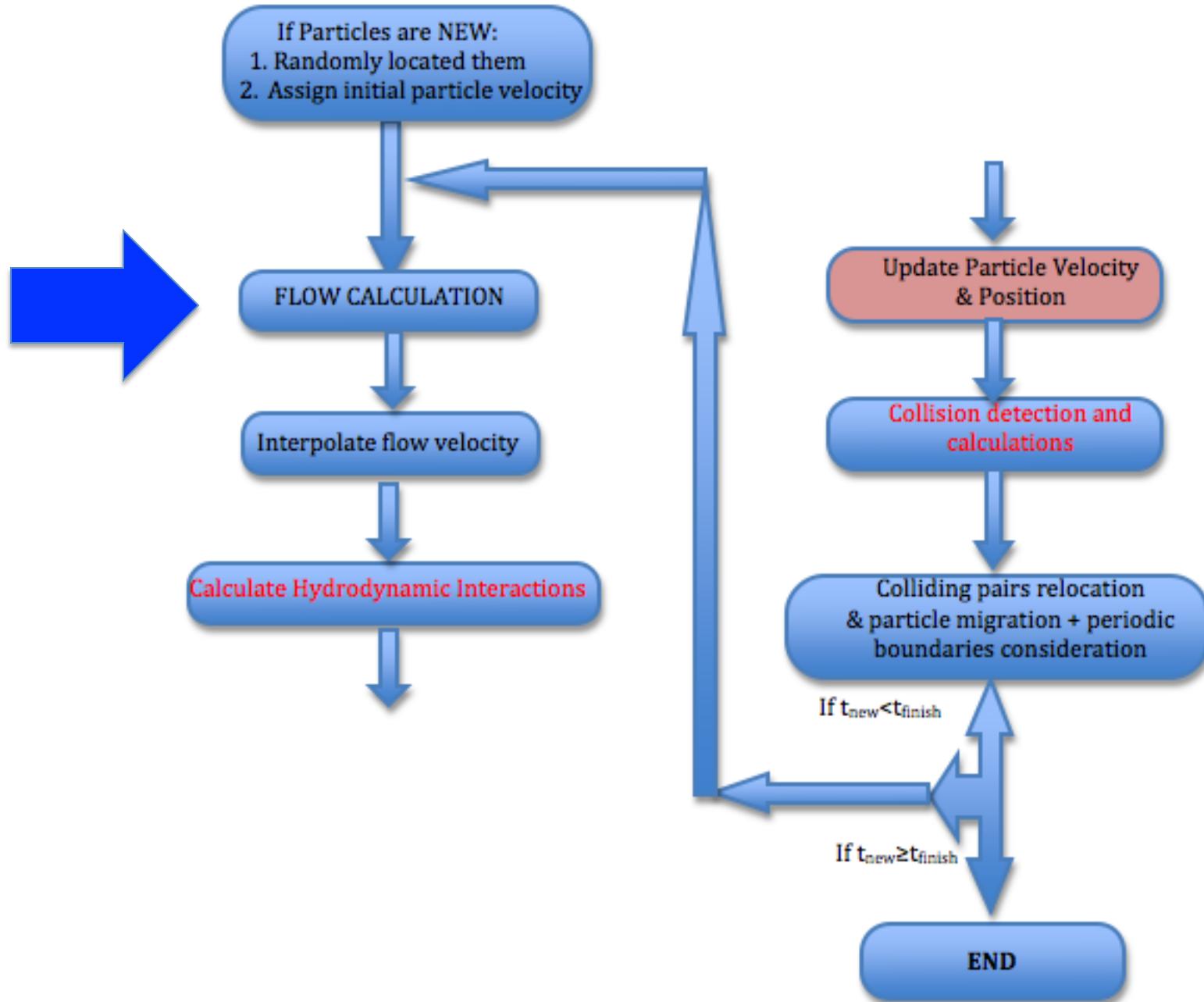


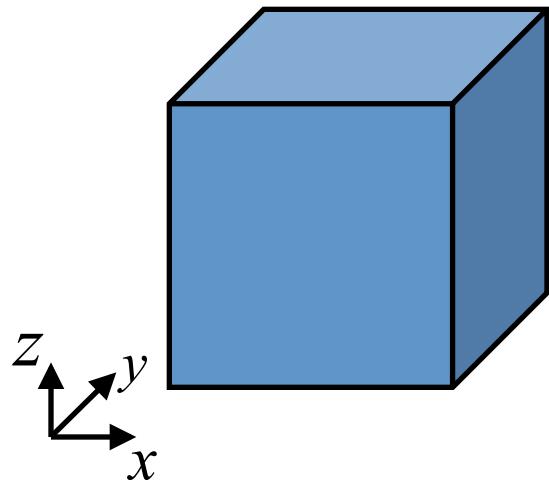


DIFFERENT DOMAIN
DECOMPOSITION
STRATEGIES



2D Domain Decomposition





$$\frac{\partial}{\partial t} \vec{u}(\vec{x},t) = \vec{u} \times \vec{\omega} - \nabla \left(\frac{p}{\rho} + \frac{u^2}{2} \right) + \nu \nabla^2 \vec{u} + \vec{f}(\vec{x},t)$$

$$\nabla \cdot \vec{u}(\vec{x},t) = 0$$

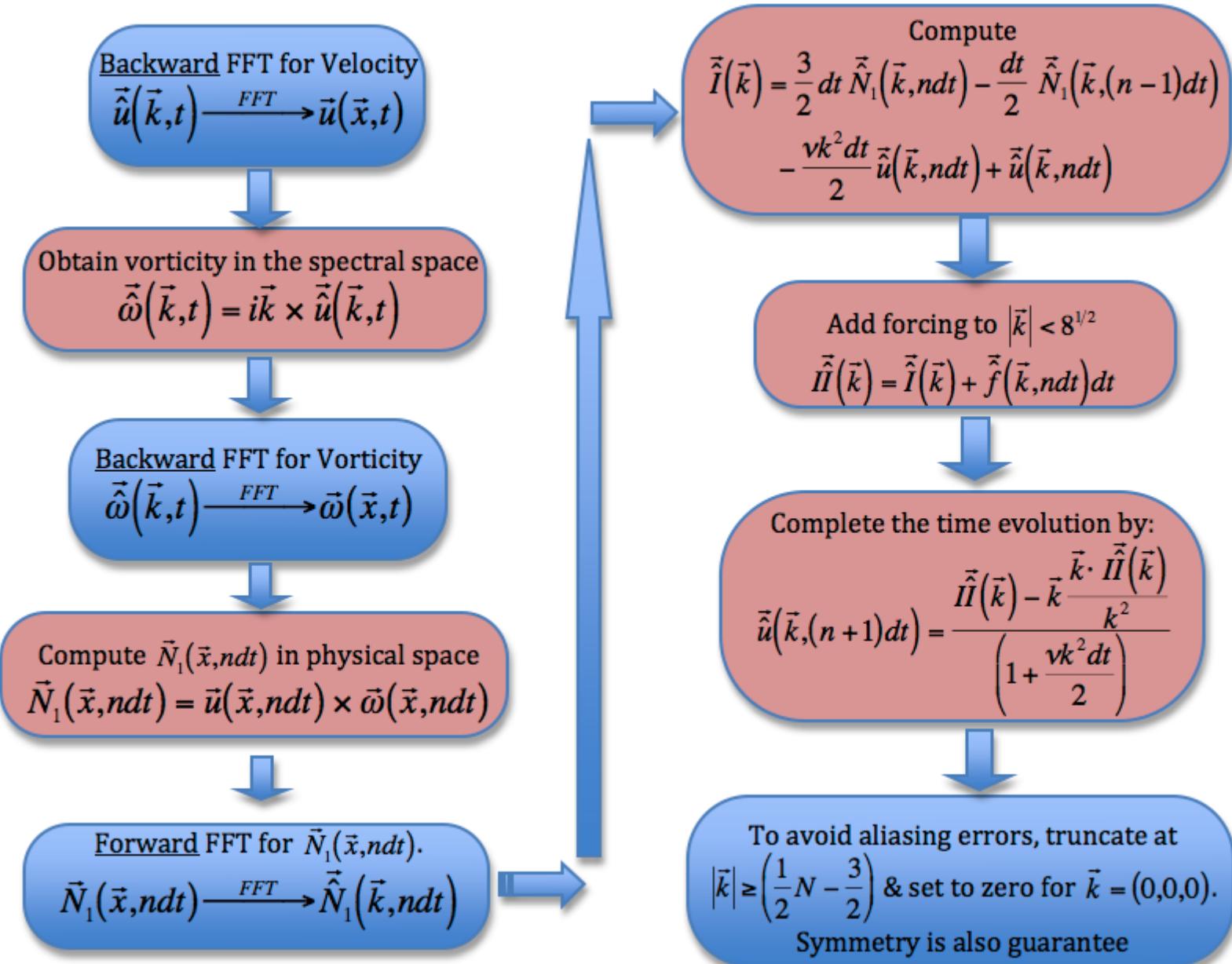
$$\frac{\partial}{\partial t} \vec{u}(\vec{x},t) = \underbrace{\vec{u} \times \vec{\omega}}_{\text{1st part of the nonlinear term}} - \underbrace{\nabla \left(\frac{p}{\rho} + \frac{u^2}{2} \right)}_{\text{2nd part of the nonlinear term}} + \nu \nabla^2 \vec{u} + \vec{f}(\vec{x},t)$$

viscous term
large scale force (low wave number)

$$\frac{\partial}{\partial t} \vec{u}(\vec{k},t) = \vec{N}_1(\vec{k},t) - ik \vec{N}_2(\vec{k},t) - \nu k^2 \vec{u}(\vec{k},t) + \vec{f}(\vec{k},t)$$

$$\nabla \cdot \vec{u}(\vec{x},t) = 0$$

$$\vec{N}_2(\vec{k},t) = -\frac{i\vec{k} \cdot \vec{N}_1(\vec{k},t)}{k^2}$$





Parallel implementation and scalability analysis of 3D fast Fourier transform using 2D domain decomposition

Orlando Ayala^{*a,b}, Lian-Ping Wang^a

^a*Department of Mechanical Engineering, 126 Spencer Laboratory, University of Delaware, Newark, Delaware 19716-3140, USA*

^b*Centro de Métodos Numéricos en Ingeniería, Escuela de Ingeniería y Ciencias Aplicadas, Universidad de Oriente, Puerto La Cruz, Venezuela*

Preprint submitted to Parallel Computing

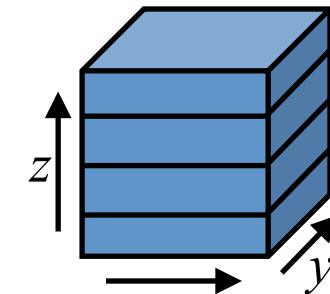
July 25, 2012



Approaches to parallelize FFT

- Dmitruk et al (2001)
1-D decomposition for 3-D FFT.

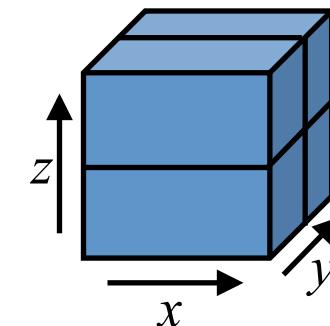
Number of processors (P) < Number of grid points (N)



- Takahashi (2009), Pekurovsky (2008),
Plimpton (1993)

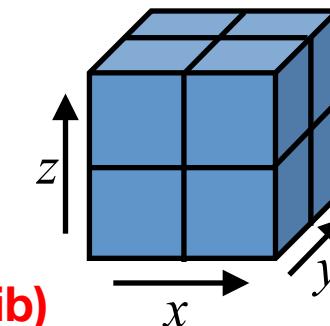
2-D decomposition for 3-D FFT.

P < N²



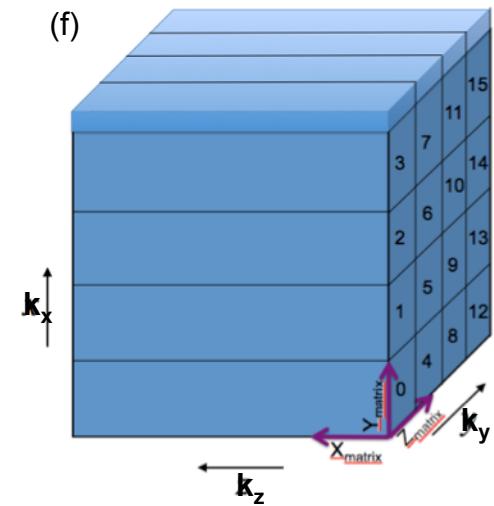
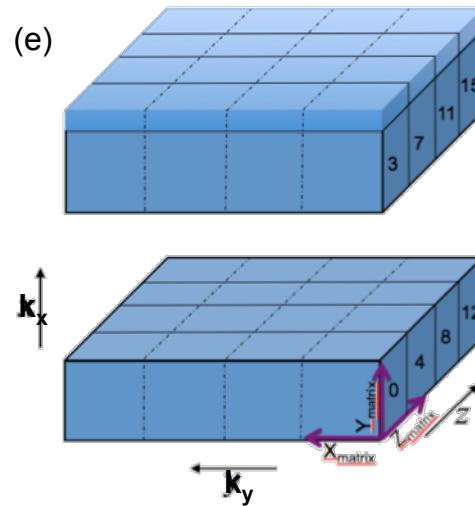
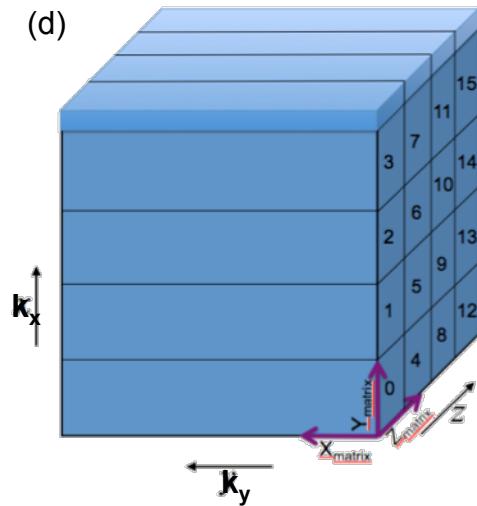
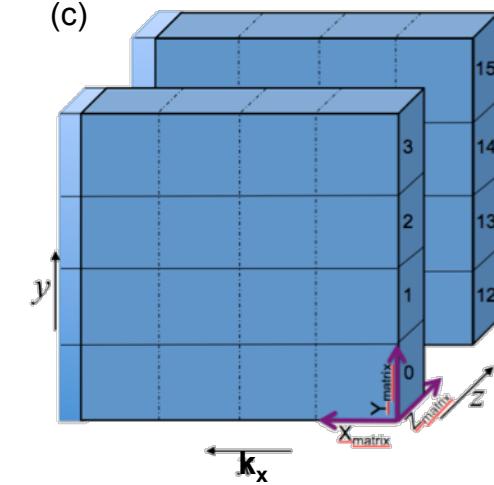
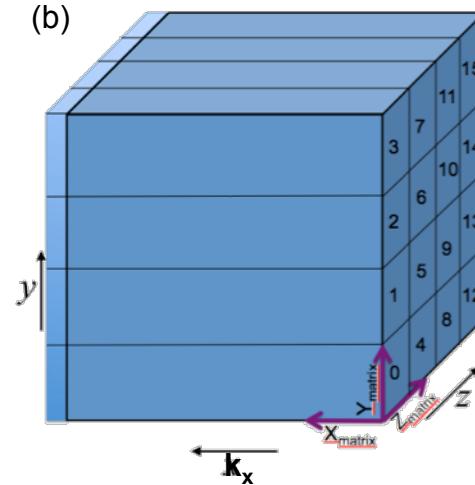
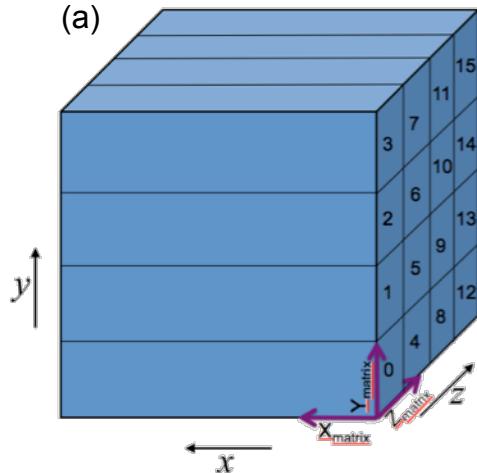
- Eleftheriou et al. (2003), (2005), Fang (2007)
3-D decomposition for 3-D FFT.

More communications (or develop new parallel 1DFFT lib)





The 2D Decomposition Strategy



Three Dimensional FFT Real to Complex using the 2D Decomposition Strategy: (a) Real Array, (b) 1D FFT along X, (c) Transpose between X and Y directions, (d) 1D FFT along Y, (e) Transpose between Y and Z directions, (f) 1D FFT along Z.



The 2D Decomposition Strategy

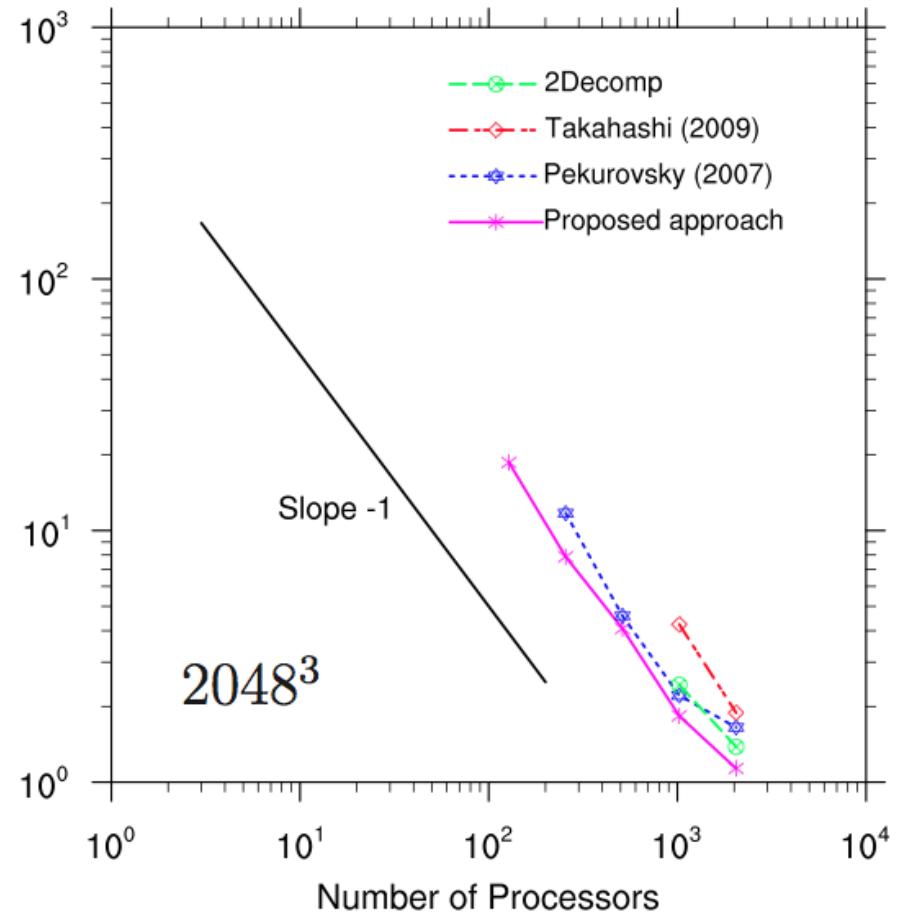
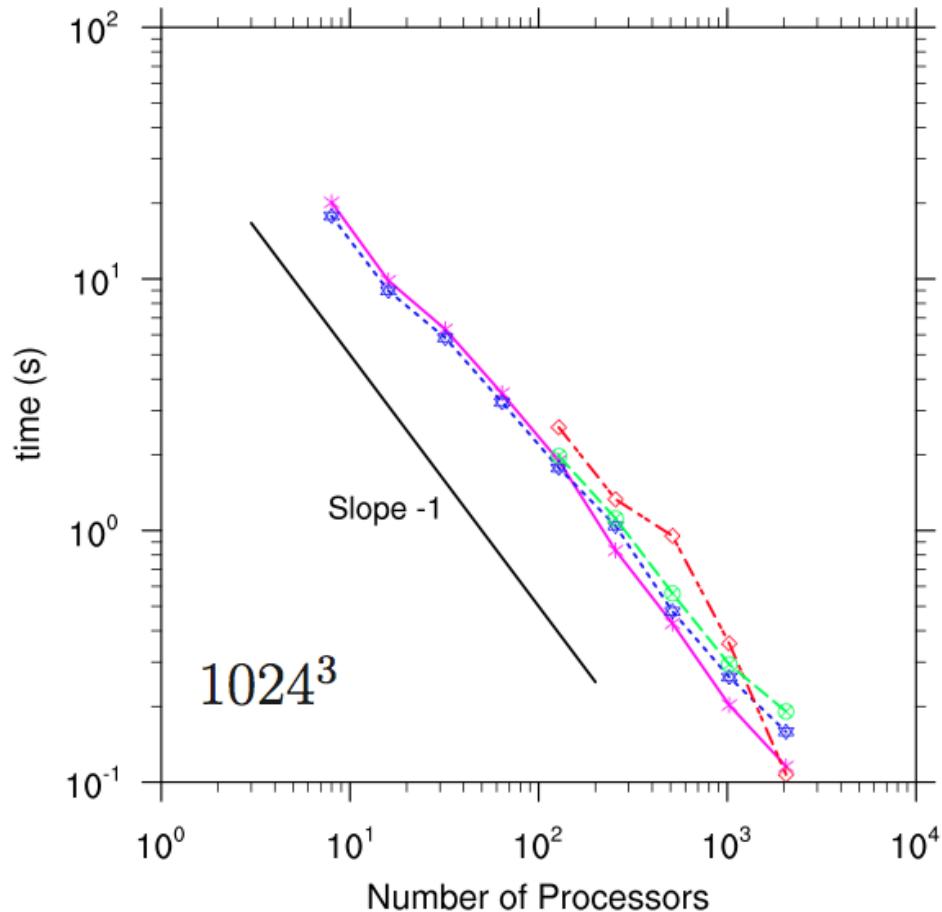
- Plimpton's strategy
 - Pack and send as much data as possible
 - MPI Send and MPI Irecv commands.
- Pekurovsky
 - Traditional MPI command MPIAlltoallv
- Takahashi
 - MPIAlltoAll to communicate data.

Our objective is to extend the approach used by Dmitruk *et al.* (2001) for 1D decomposition to 2D decomposition.

It is based on MPI_ISend and MPI_IRecv commands



Evaluation & Testing





Evaluation & Testing

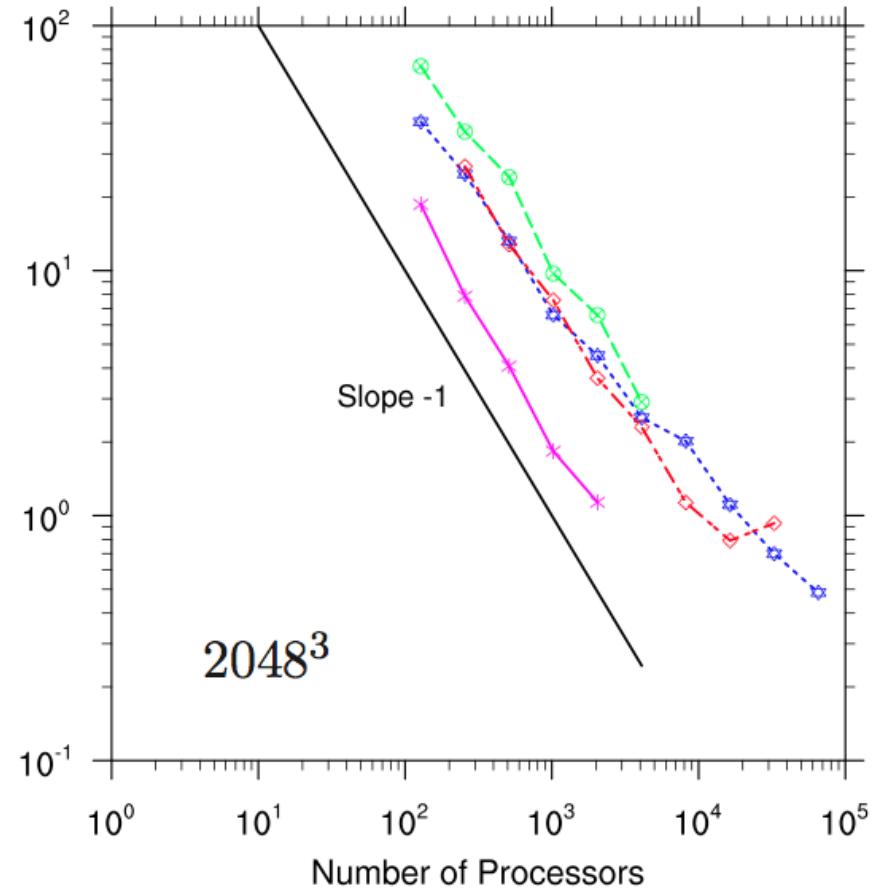
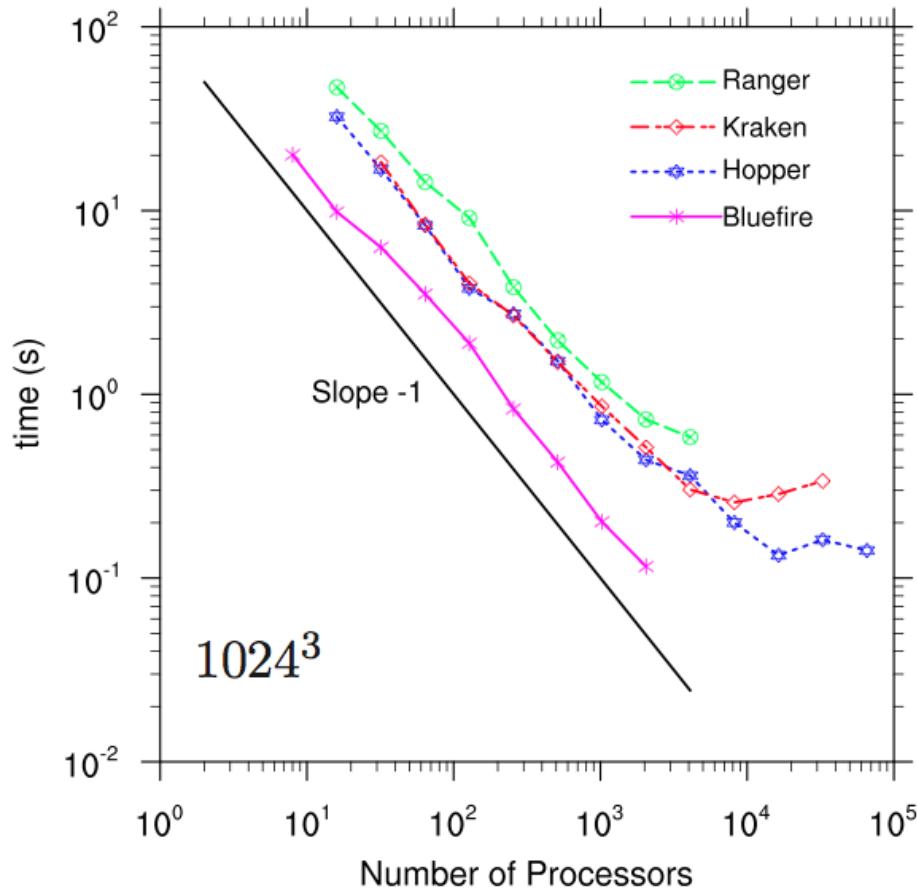
$$\frac{t_{MPI_ALLtoALL}}{t_{MPI_ISEND+IRECV}}$$

Message size	Internode	Intranode
2 bytes	0.190	0.296
16 bytes	0.243	0.294
128 bytes	0.385	0.496
1 Kbyte	0.769	1.274
8 Kbytes	1.148	1.573
64 Kbytes	1.222	1.435
512 Kbytes	4.478	3.598

Table 4: Speedup ratio between an MPI_ISend+MPI_IRecv communication strategy and an MPI_Alltoall communication strategy using 32 processors.



Evaluation & Testing



Dmitruk's et al (2001)

$$T = T_{COMP} + T_{COMM}$$

$$T_{COMP} = \frac{5}{2} \frac{N^3 \log_2(N^3)}{P} t_c + \left[2 \underbrace{\left(\frac{(N+2)N^2}{P} \right) \frac{1}{P}}_{\text{Size}} + (P-1) \underbrace{\left(\frac{(N+2)N^2}{P} \right) \frac{1}{P}}_{\text{Size}} \right] t_a$$

$$T_{COMM} = 2(P-1) \underbrace{\left(\frac{(N+2)N^2}{P} \right) \frac{1}{P}}_{\text{Size}} t_w + 2(P-1)t_s$$

t_c Computation time per floating point operation

t_a Memory-to-memory copy time per word

t_w Time for transmission of a single word between nodes per word

t_s Startup or latency time



Extension to 2D decomposition

for $P_y = P_z = \sqrt{P}$

Block size to communicate

$$\left[\left(\frac{(N+2)N^2}{P_y P_z} \right) \frac{1}{P_y} \right]$$

of Times to communicate

$$(P_y - 1)$$

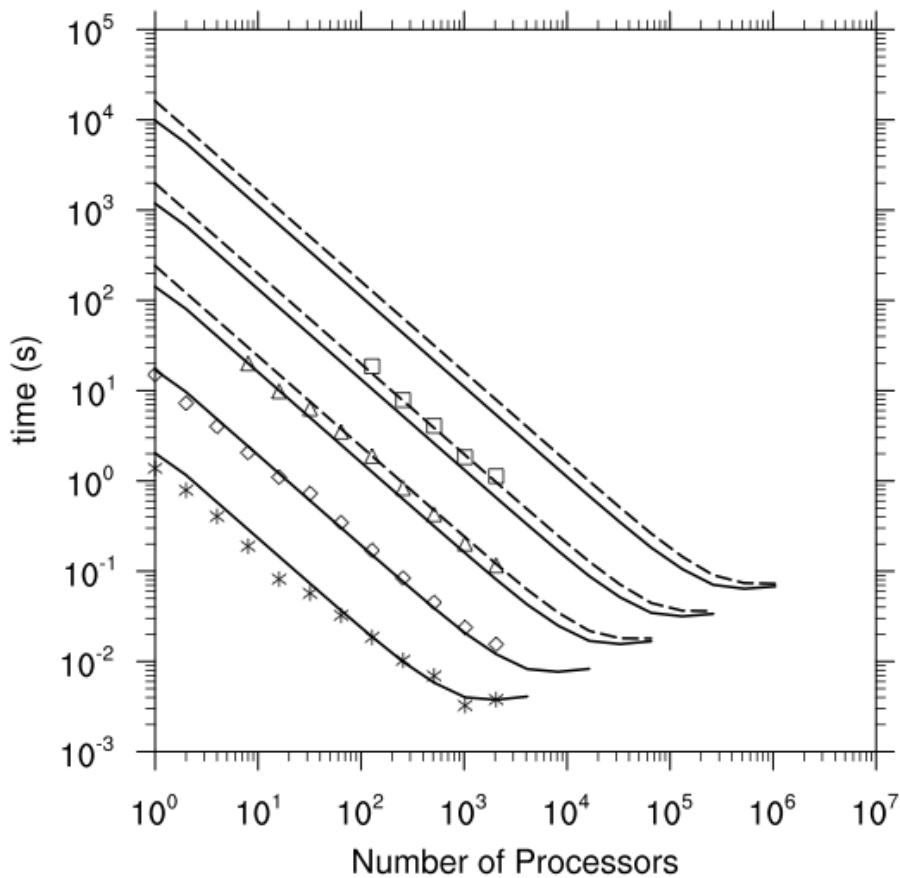
$$T_{COMP} = \frac{5}{2} \frac{N^3 \log_2(N^3)}{P_y P_z} t_c + 2 \left(2 \left(\frac{(N+2)N^2}{P_y P_z} \right) \frac{1}{P_y} + (P_y - 1) \left(\frac{(N+2)N^2}{P_y P_z} \right) \frac{1}{P_y} \right) t_a$$

$$T_{COMM} = 2 \left[2(P_y - 1) \left(\frac{(N+2)N^2}{P_y P_z} \right) \frac{1}{P_y} t_w + 2(P_y - 1) t_s \right]$$

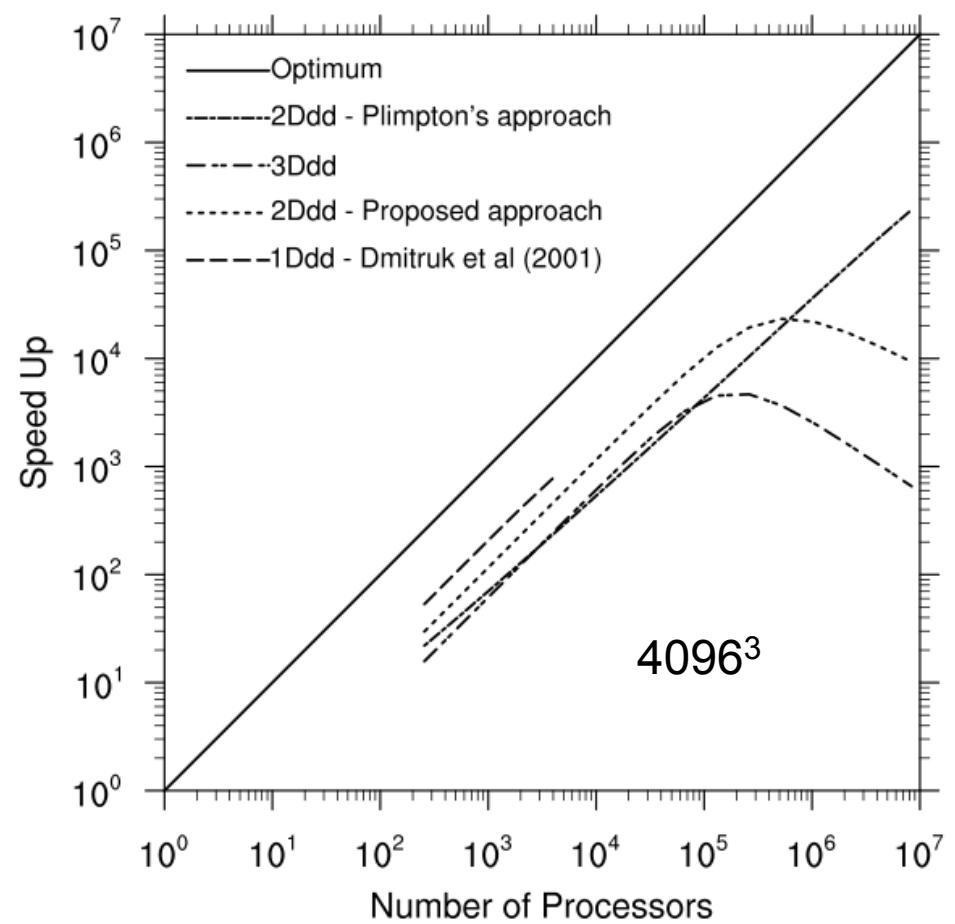
We also developed complexity analysis for
the case of 3D decomposition



Complexity Analysis

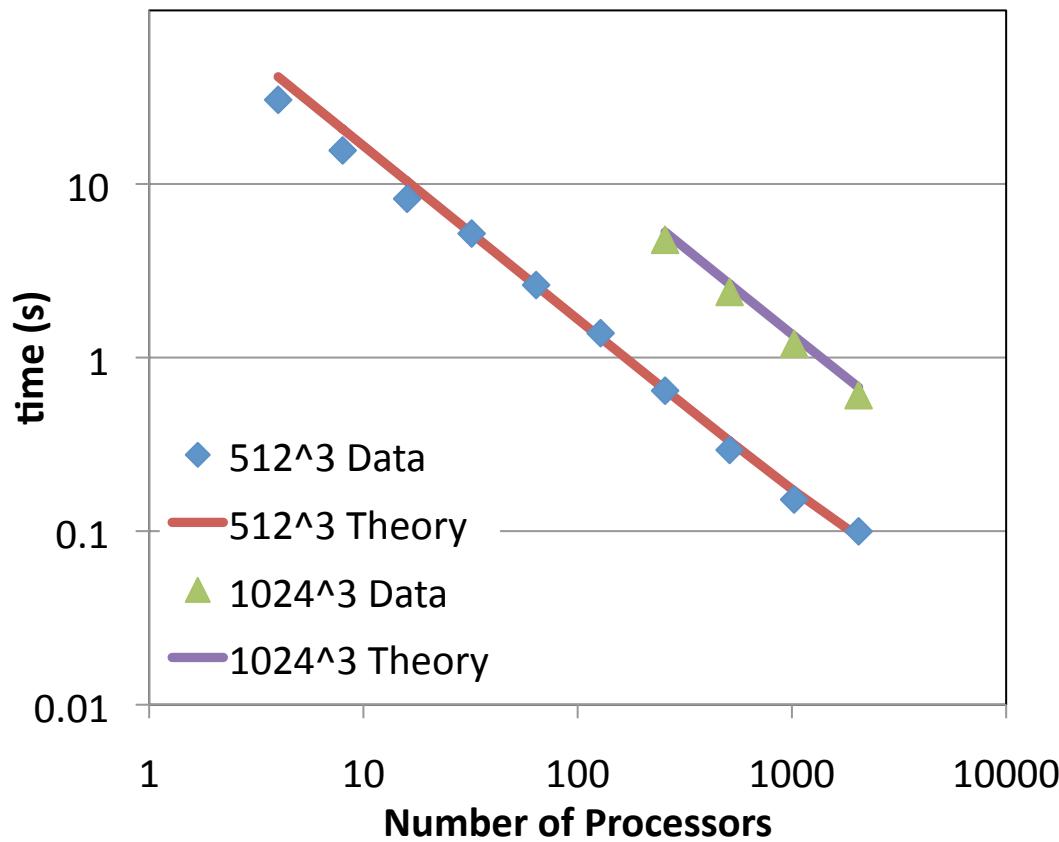


(*) for 256^3 , (◊) for 512^3 , (△) for 1024^3 ,
and (□) for 2048^3 .





Complexity Analysis whole code



$$\begin{aligned} t_{PSM_{2D}} = & 9 \cdot t_{FFT} \\ & + \left(93 \frac{N^3}{P} + 6 \frac{N^2}{P_z} + 7N \right) t_c \\ & + \left(9 \frac{N^3}{P} + 2 \frac{N^2}{P} + N^2 \left(\frac{1}{P_y} + \frac{2}{P_z} \right) \right) t_a \\ & + 4 \frac{N^2}{P_z} t_w + 4 t_s \end{aligned}$$

The theory predicts 90% of wall-clock time spent in FFT

Chen & Shan (1992) -> 94%



UNIVERSITY *of* DELAWARE

PSM

What about LBM or FDM?



Lattice Boltzmann Method

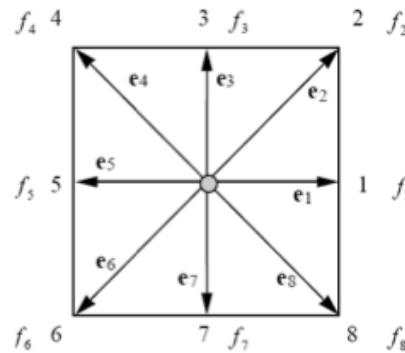
A mesoscopic approach solving the distribution functions “ f_i ” for a set of particles at each grid point:

$$f_i(x + \vec{e}_i \delta t, t + \delta t) - f_i(\vec{x}, t) = -\frac{1}{\tau} [f_i(\vec{x}, t) - f_i^{(eq)}(\vec{x}, t)]$$

$$f_i^{(eq)}(\vec{x}, t) = \rho w_i \left[1 + \frac{3}{c^2} \vec{e}_i \cdot \vec{u} + \frac{9}{2c^4} (\vec{e}_i \cdot \vec{u})^2 - \frac{3}{2c^2} \vec{u} \cdot \vec{u} \right]$$

Extension to multiple relaxation times.

Mesoscale particles move along the mesh links, with prescribed discrete velocities.



D2Q9 model

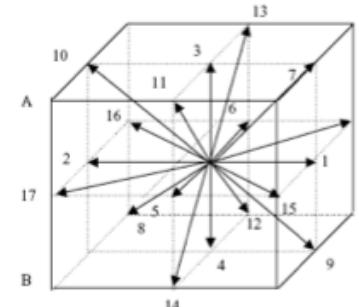


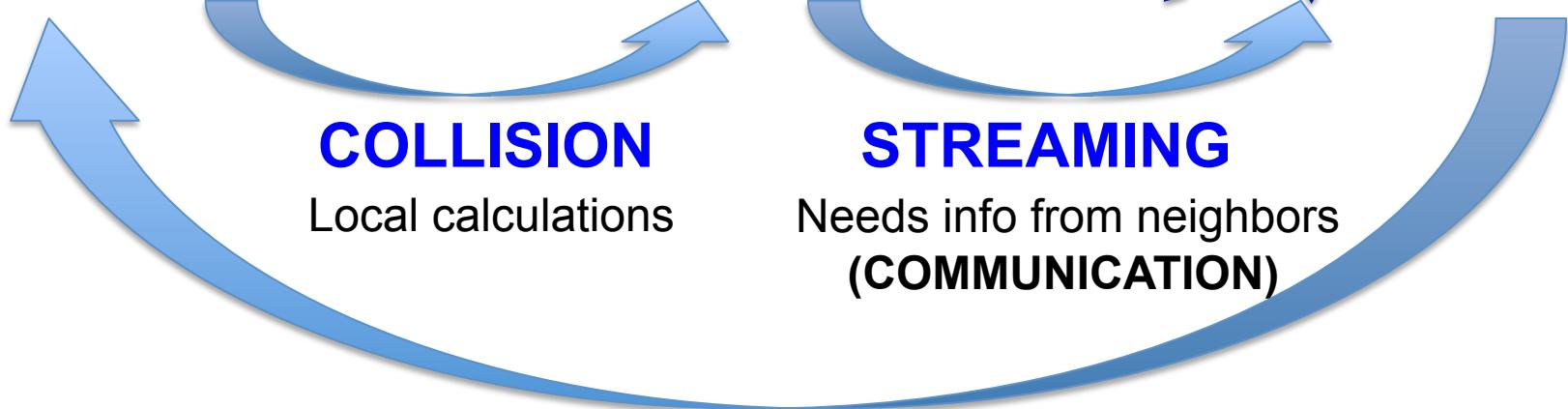
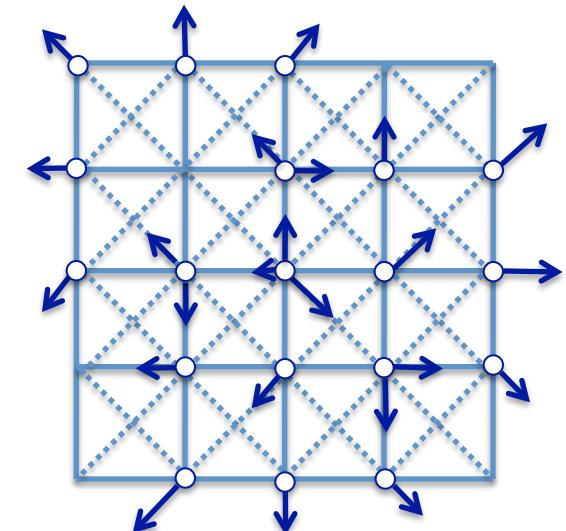
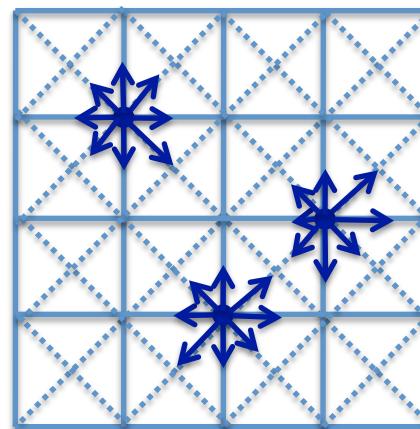
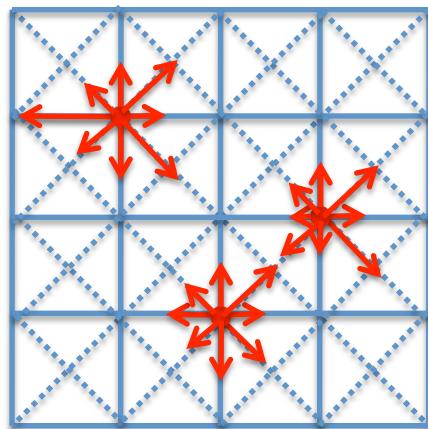
Fig. 2. The lattice velocities of D3Q19.

D3Q19 model



The connection with Navier-Stokes Equations

$$\rho = \sum_{\alpha} f_{\alpha}(\vec{x}, t), \quad \rho u = \sum_{\alpha} f_{\alpha}(\vec{x}, t) e_{\alpha}$$



Calculation of hydrodynamic variables (local)



Complexity Analysis

$$t_{LBM\,1D} = 474 \frac{N^3}{P} t_c + \left(79 \frac{N^3}{P} + 20N^2 \right) t_a + 20N^2 t_w + 20t_s$$

Comp Copy Transm Latency

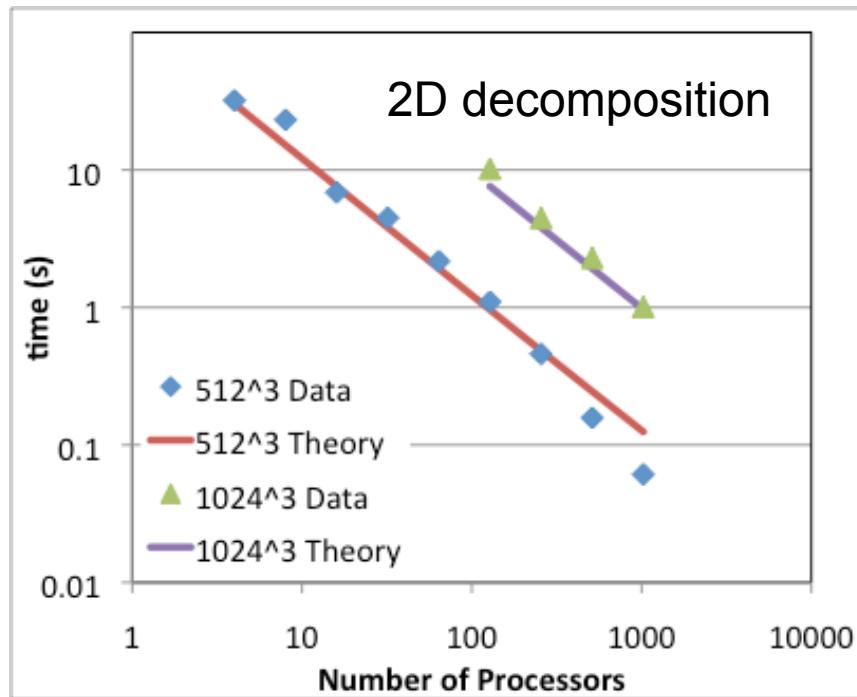
$$t_{LBM\,2D} = 474 \frac{N^3}{P} t_c + \left(79 \frac{N^3}{P} + 20N^2 \left(\frac{1}{P_y} + \frac{1}{P_z} \right) \right) t_a + 20N^2 \left(\frac{1}{P_y} + \frac{1}{P_z} \right) t_w + 40t_s$$

$$t_{LBM\,3D} = 474 \frac{N^3}{P} t_c + \left(79 \frac{N^3}{P} + 20N^2 \left(\frac{1}{P_y P_z} + \frac{1}{P_x P_z} + \frac{1}{P_x P_y} \right) \right) t_a + 20N^2 \left(\frac{1}{P_y P_z} + \frac{1}{P_x P_z} + \frac{1}{P_x P_y} \right) t_w + 60t_s$$

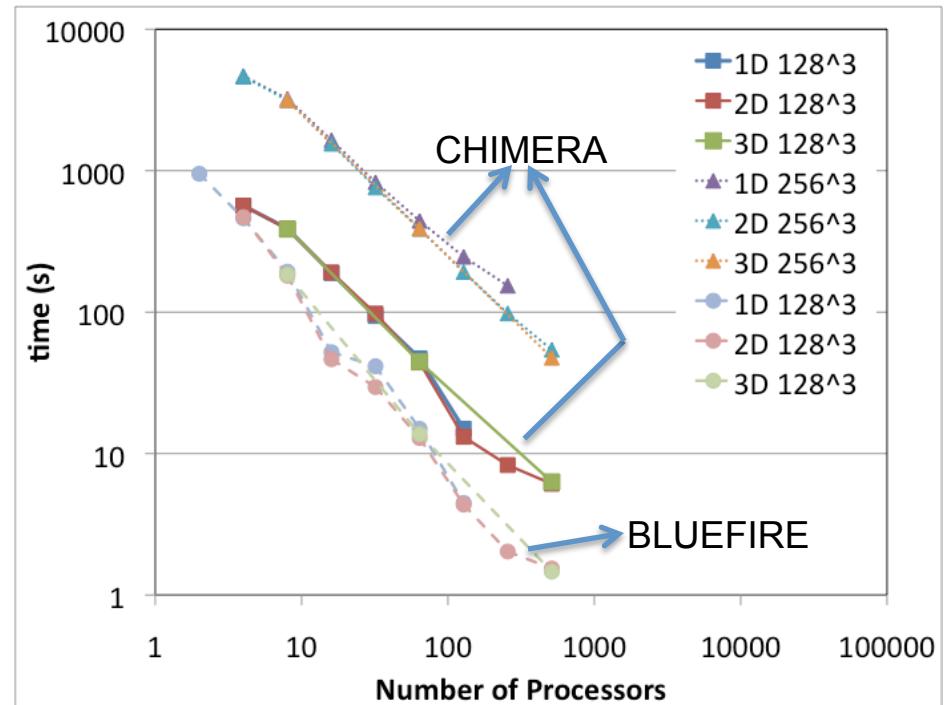
gets reduced



Complexity Analysis



Comparing complexity analysis to runtime data



Comparing three different domain decomposition schemes



Finite Difference Method

One of the most common approach used is the Fractional Step Method by Kim and Moin (1985) which is a 2nd order

$$\frac{\hat{u}_i - u_i^n}{\Delta t} = \frac{1}{2} (3H_i^n - H_i^{n-1}) + \frac{1}{2} \frac{1}{\text{Re}} \left(\frac{\delta^2}{\delta x_1^2} + \frac{\delta^2}{\delta x_2^2} + \frac{\delta^2}{\delta x_3^2} \right) (\hat{u}_i + u_i^n), \quad (1)$$

$$\frac{u_i^{n+1} - \hat{u}_i}{\Delta t} = -G(\phi^{n+1}), \quad (2)$$

with

$$D(u_i^{n+1}) = 0, \quad (3)$$

Approximate factorization:

$$(1 - A_1)(1 - A_2)(1 - A_3)(\hat{u}_i - u_i^n) = \frac{\Delta t}{2} (3H_i^n - H_i^{n-1}) + 2(A_1 + A_2 + A_3) u_i^n. \quad (4)$$

$$\left(\frac{\delta^2}{\delta x_1^2} + \frac{\delta^2}{\delta x_2^2} + \frac{\delta^2}{\delta x_3^2} \right) \phi^{n+1}(i, j, k) = \frac{1}{\Delta t} D\hat{u} \quad (5)$$

For the Poisson equation -> ADI with Thomas algorithm or
solve in the spectral space (FFT)

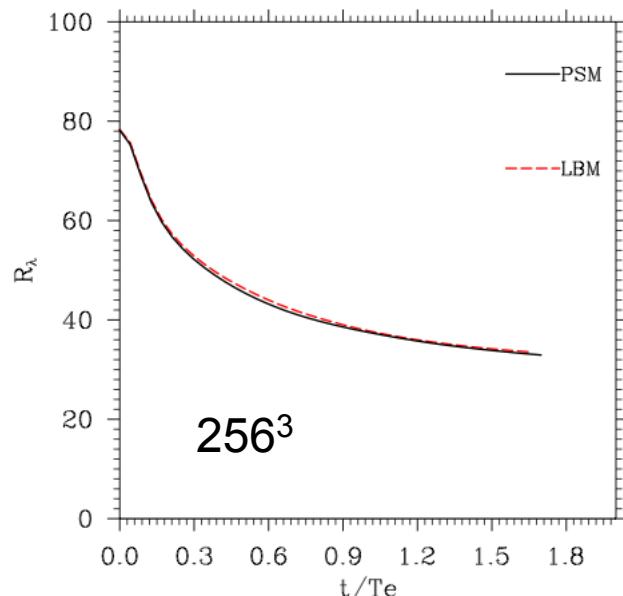


Complexity Analysis

Following a parallel FDM algorithm:

$$\begin{aligned} t_{FDM_{2D}} = & \left(210 + 5 \log_2(N^3) \right) \frac{N^3}{P} t_c + \left(39 \frac{N^3}{P} + 16N^2 \left(\frac{1}{P_y} + \frac{1}{P_z} \right) \right) t_a \\ & + \left(39 \frac{N^3}{P} + 12N^2 \left(\frac{1}{P_y} + \frac{1}{P_z} \right) \right) t_w + \left(32 + 6(P_y + P_z) \right) t_s \end{aligned}$$

- Poisson solver takes 13% of the total computation (using FFT). People has found it is around 30% (using iterative methods).



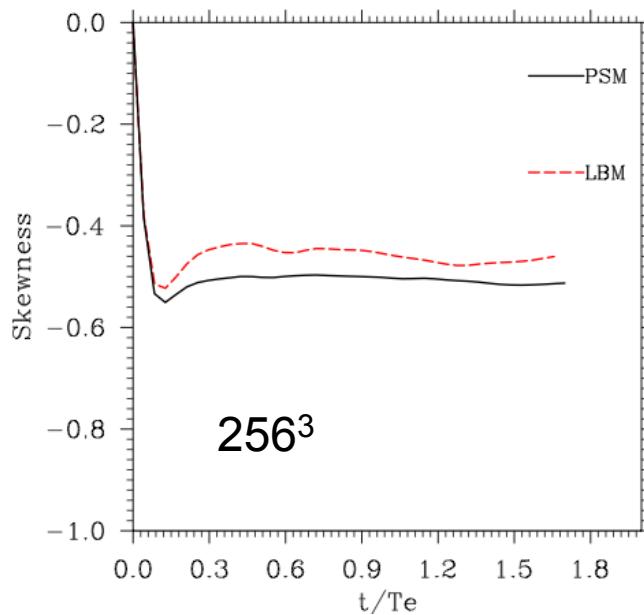
- **LBM vs PSM**

- Satofuka & Nishioka (1999) and Gao H. et al (2011) found good agreement. No detail on small scales.
- Peng et al (2010) paid attention to small scales and concluded:

COMPUTATIONAL EFFORT - LBM
SIMILAR PSM
ACCURACY

$$\approx (2*2*2) * 2 * \frac{\text{Space Resolution}}{\text{Time Resolution}}$$

COMPUTATIONAL EFFORT - LBM



- **PSM vs FDM**

- Orlandi P. (2000): found perfect agreement.
- Moin & Mahesh (1998): $\Delta x_{FDM} \approx 0.5 \Delta x_{PSM}$
- Wait & Smolarkiewicz (2008) ->

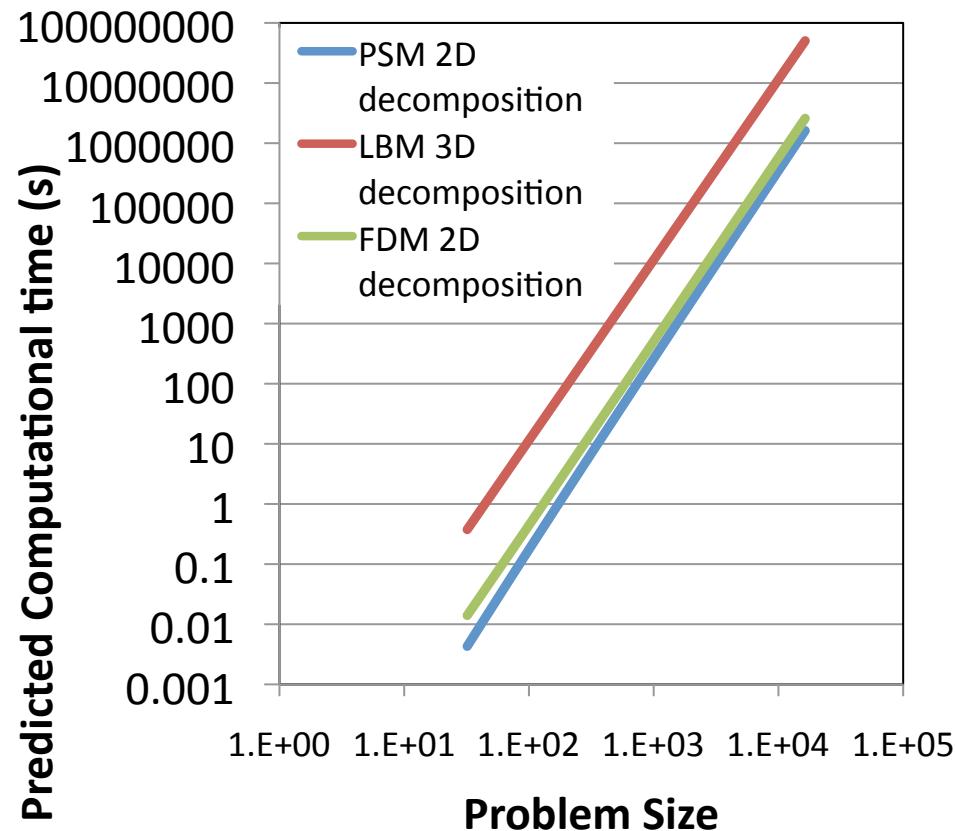
$$\Delta x_{FDM} = 1.5 \Delta x_{PSM} \quad \Delta t_{FDM} \approx 2 \Delta t_{PSM}$$

COMPUTATIONAL EFFORT - FDM
SIMILAR PSM
ACCURACY

$$\approx (2*2*2) * 1/2 * \frac{\text{COMPUTATIONAL EFFORT - FDM}}{\text{COMPUTATIONAL EFFORT - LBM}}$$



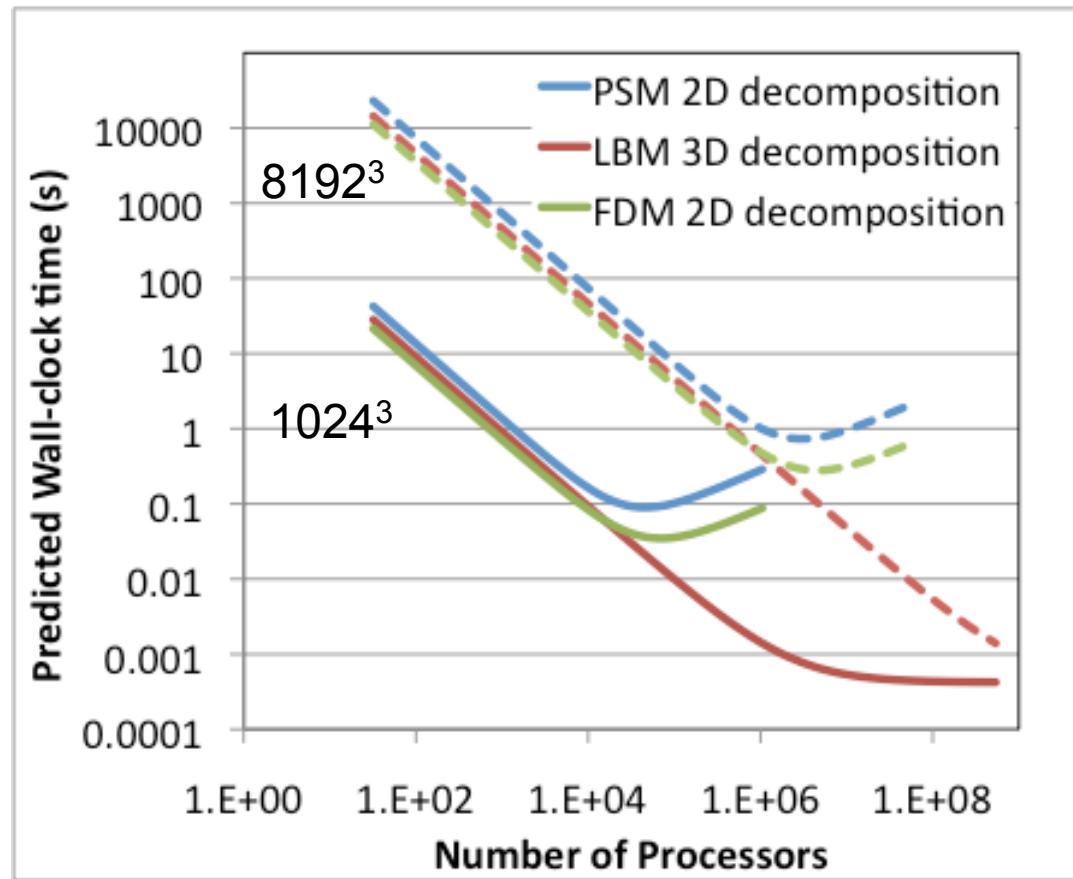
Using the theoretical expressions for wall-clock time



Comparing the algorithms as
serial codes



Don't care much
about accuracy





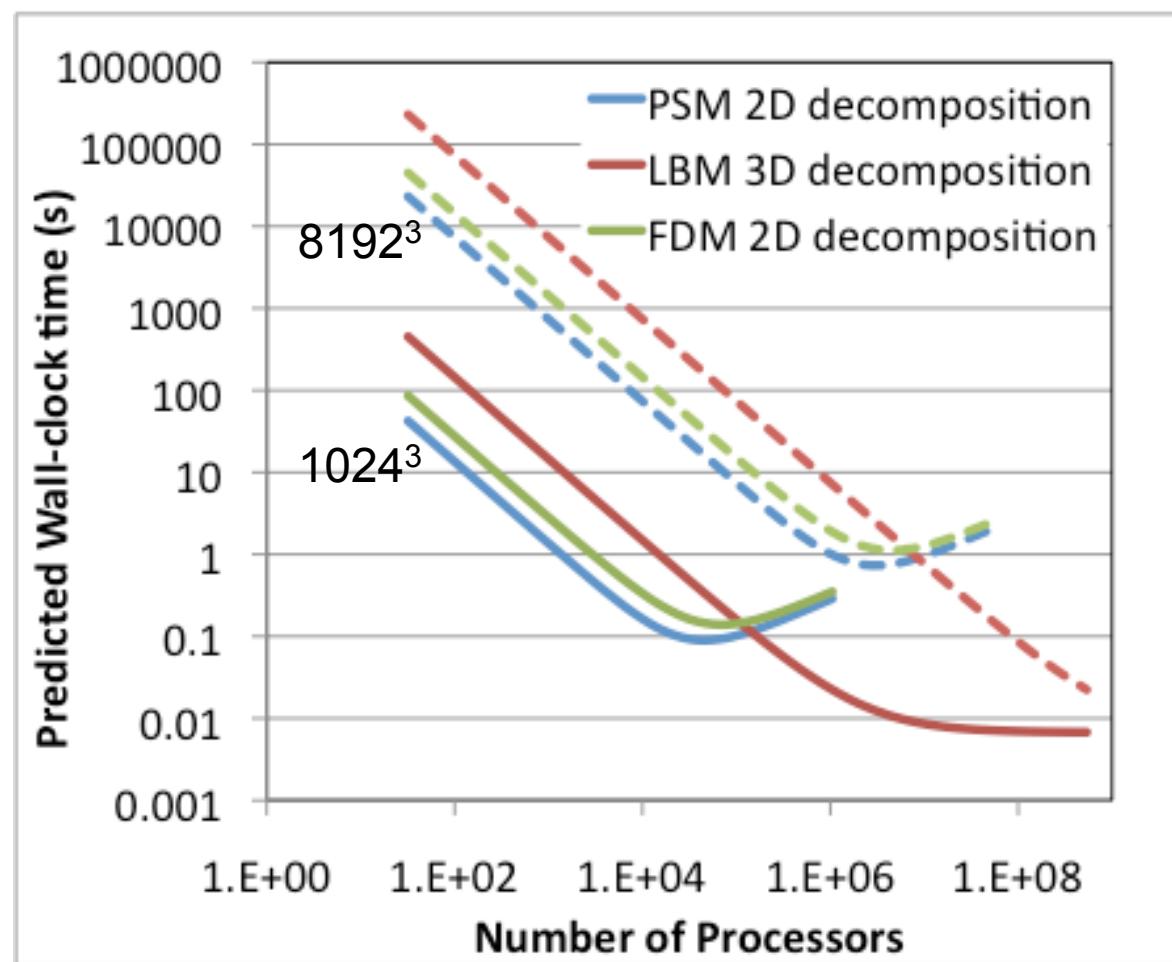
For similar PS accuracy

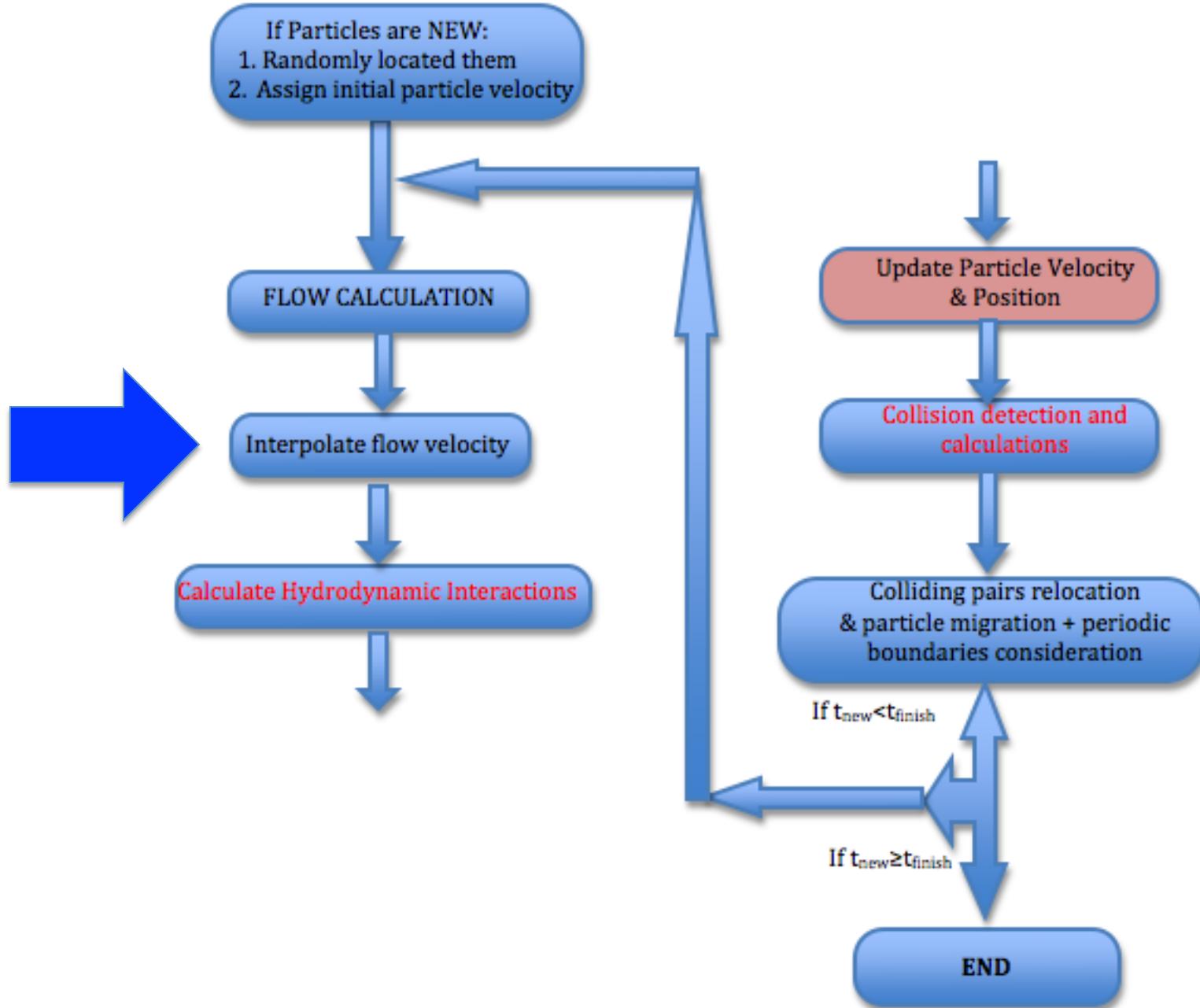
COMPUTATIONAL
EFFORT - LBM
SIMILAR PSM
ACCURACY

$\approx 16^*$ COMPUTATIONAL
EFFORT - LBM

COMPUTATIONAL
EFFORT - FDM
SIMILAR PSM
ACCURACY

$\approx 4^*$ COMPUTATIONAL
EFFORT - PSM





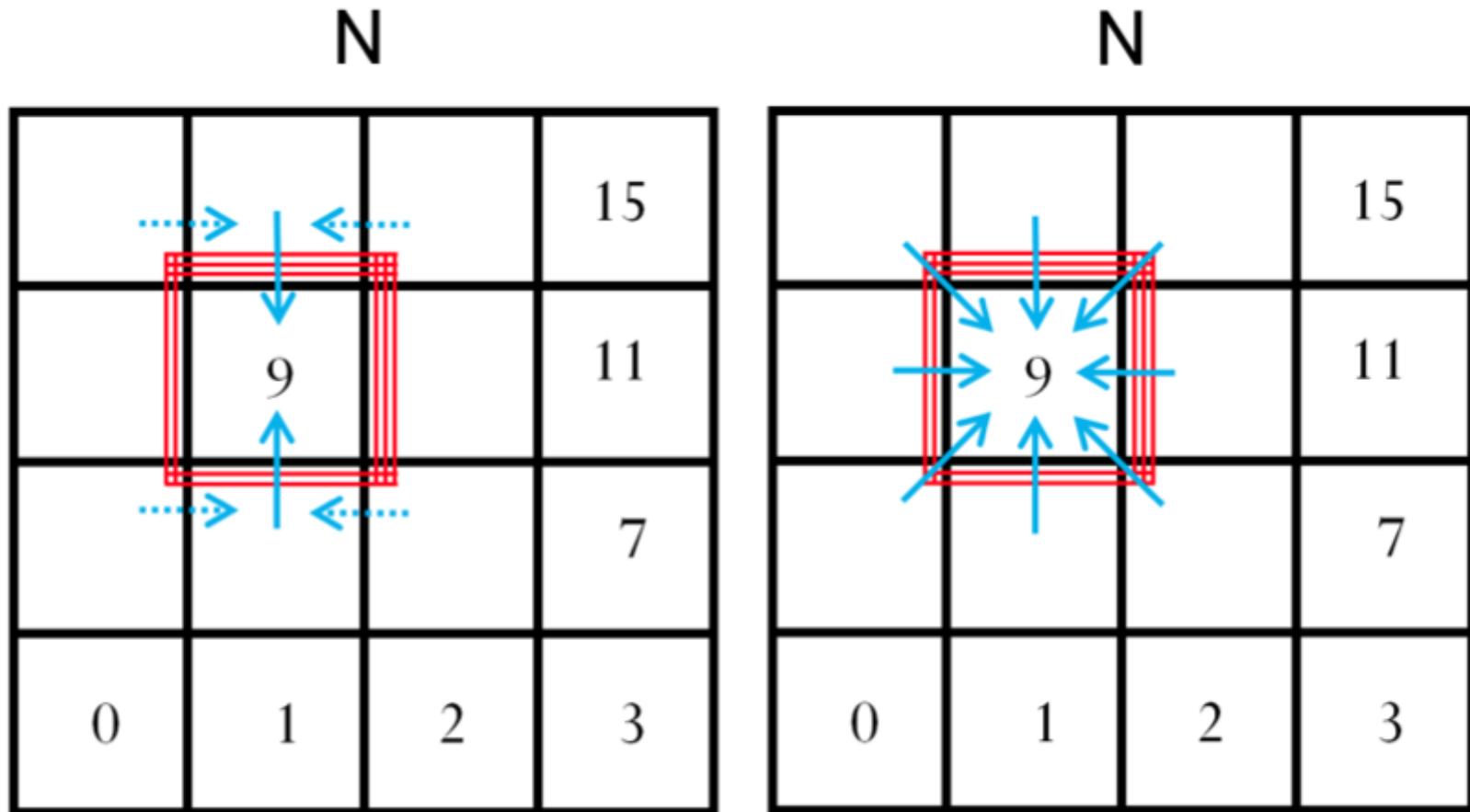
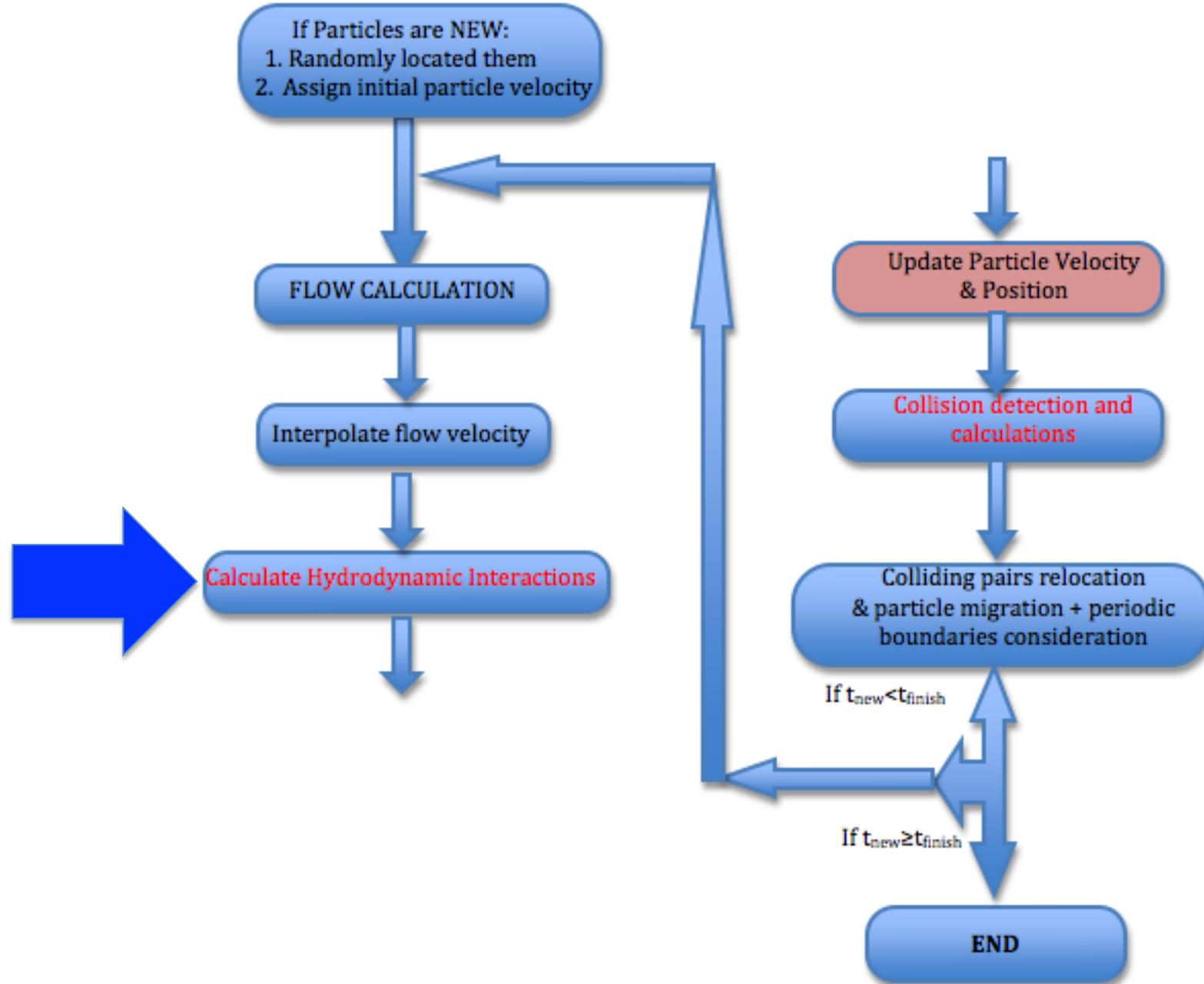


Figure 2. Two different communication strategy for fluid velocity interpolation. Red lines show the layers of grid velocity data to be communicated.

$$P_{y_{MAX} \atop Interp} = P_{z_{MAX} \atop Interp} = \frac{N}{3}$$

We chose the 2nd method as it is less machine dependent ³⁶





Analysis and scaled implementation of the “improved superposition method” of hydrodynamic interaction of cloud droplets

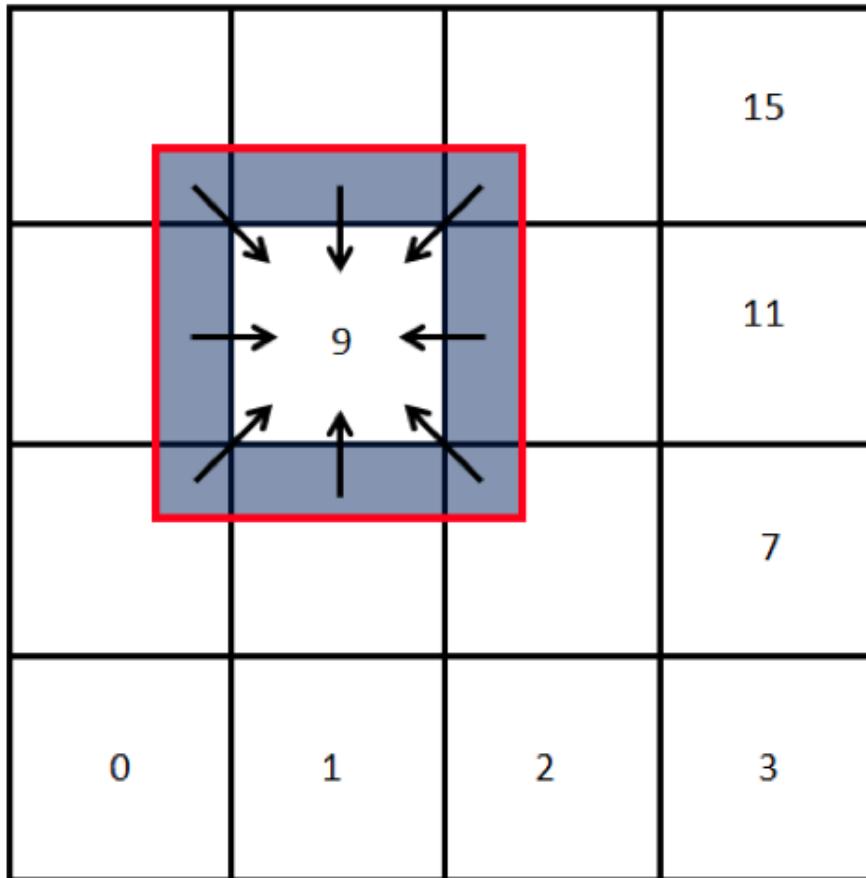
C. E. Torres^a, H. Parishani^b, O. Ayala^b, L. Rossi^a, L.-P. Wang^b

^a*Department of Mathematics, University of Delaware, Newark, DE 19716-3140, USA*

^b*Department of Mechanical Engineering, University of Delaware, Newark, DE
19716-3140, USA*

Preprint submitted to J. Computational Physics

April 19, 2012

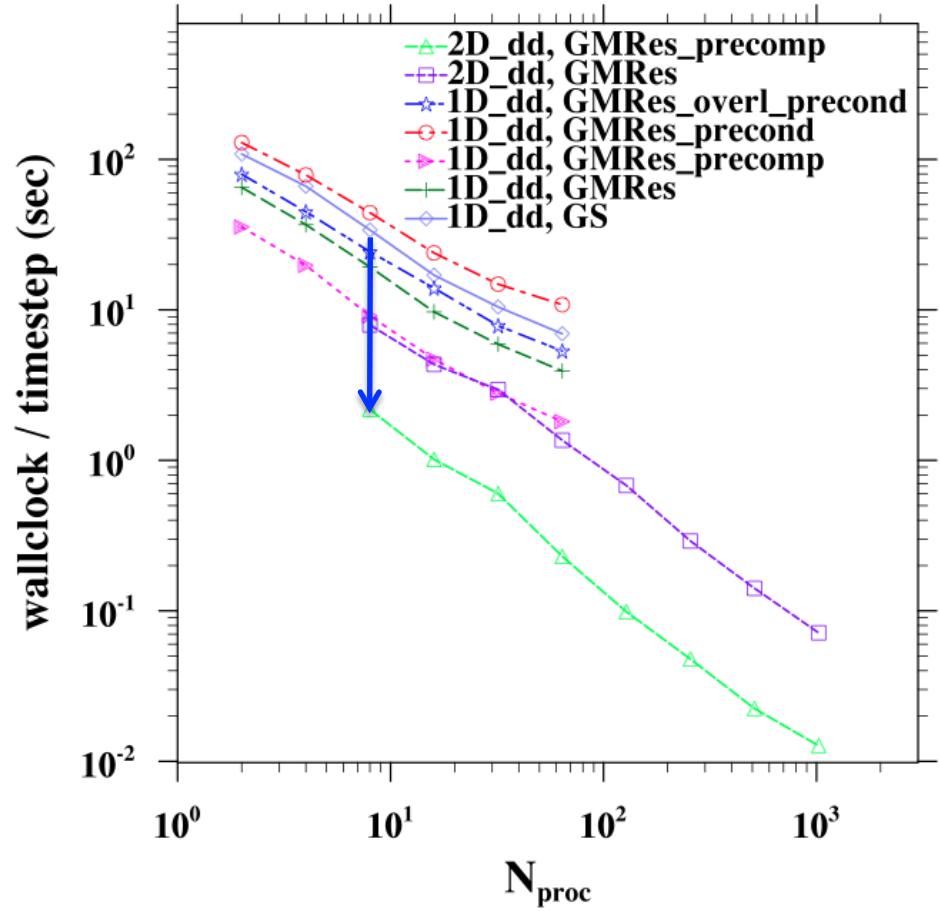
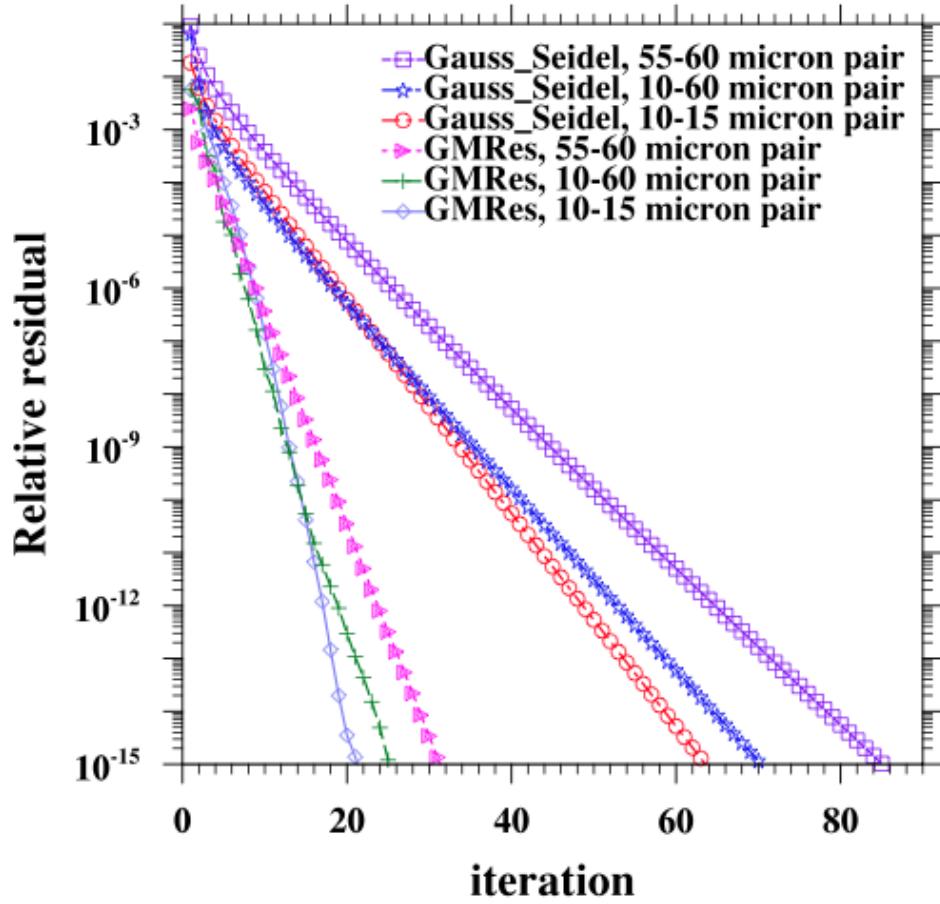


$$L_{\text{HALO}} = 50r_{\max}$$

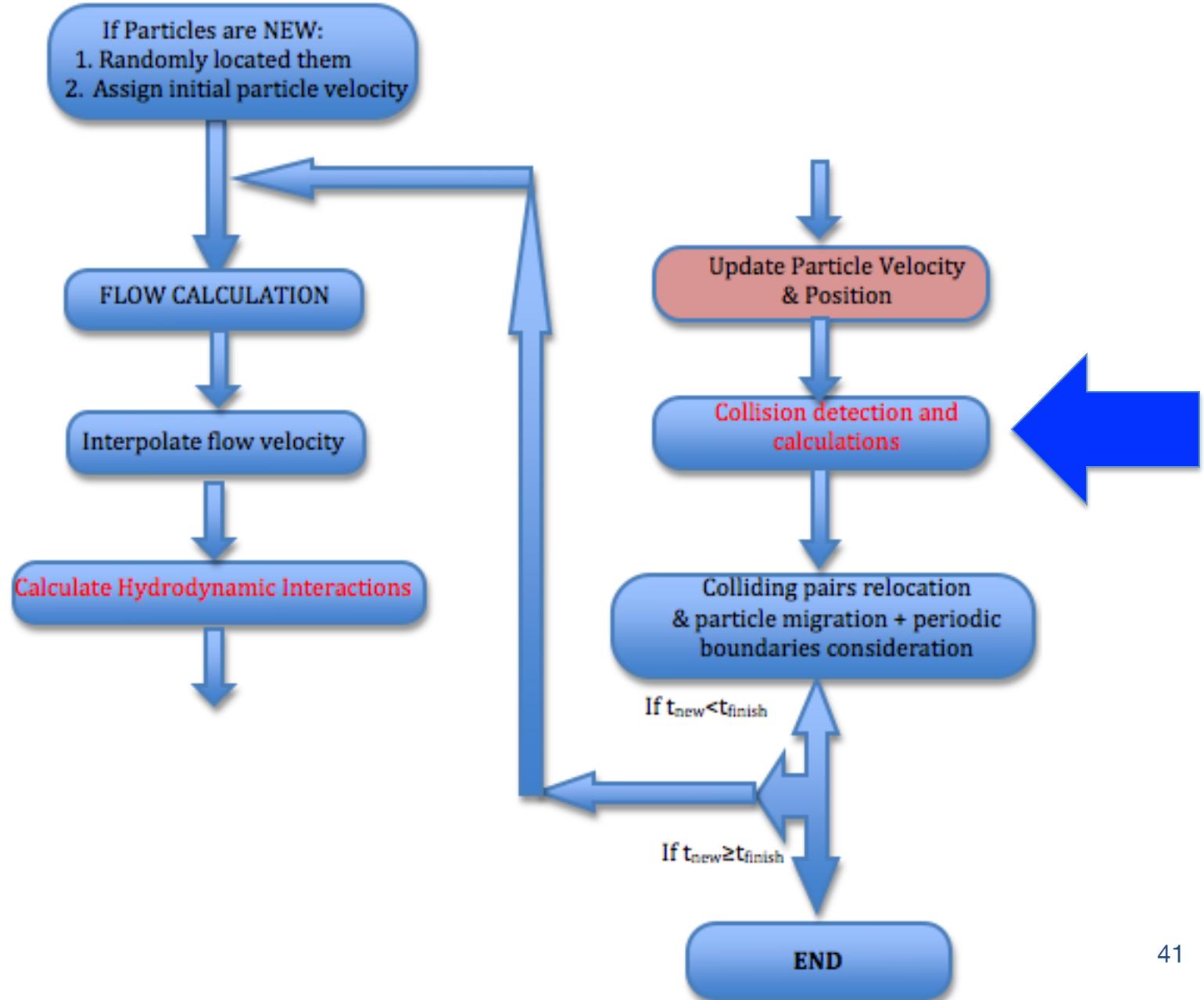
We used the cell index method and the concept of linked list

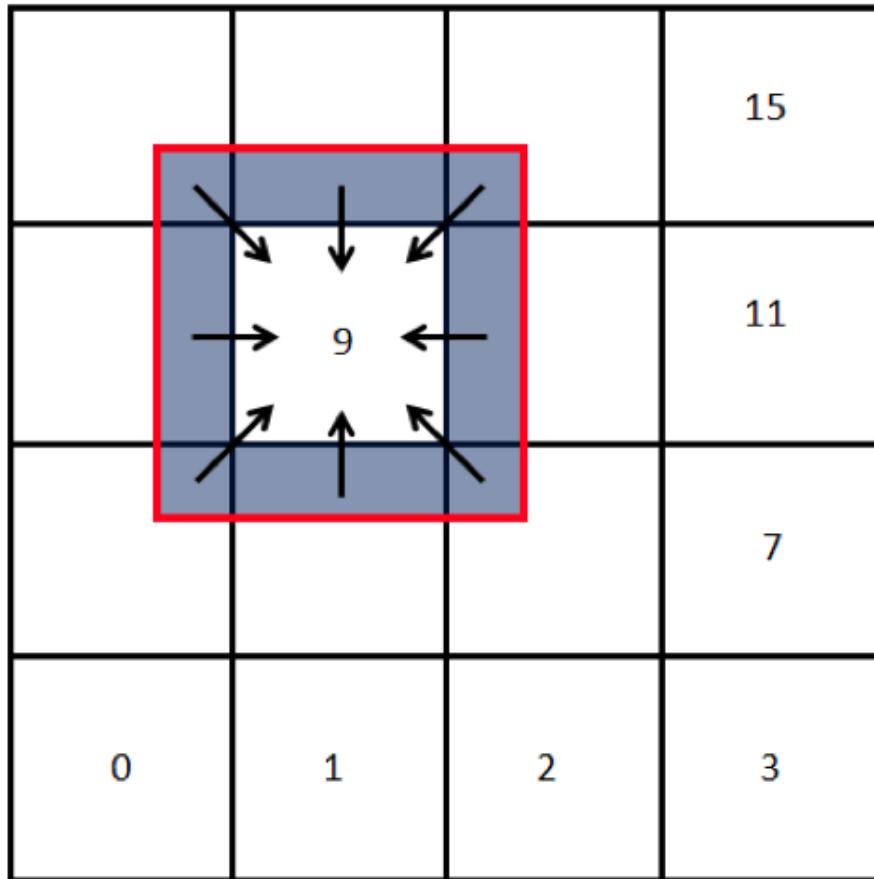
$$P_{y_{\text{MAX}}^{\text{Interp}}} = P_{z_{\text{MAX}}^{\text{Interp}}} = \frac{L}{50r_{\max}}$$

Figure 2: Dark region around subdomain 9 shows the “halo region” in 2D domain decomposition. Particles’ data in this region is made available for subdomain 9 via communications from 8 neighboring subdomains. Arrows indicate the direction of data communication.



About ONE order of
magnitude (or more) faster



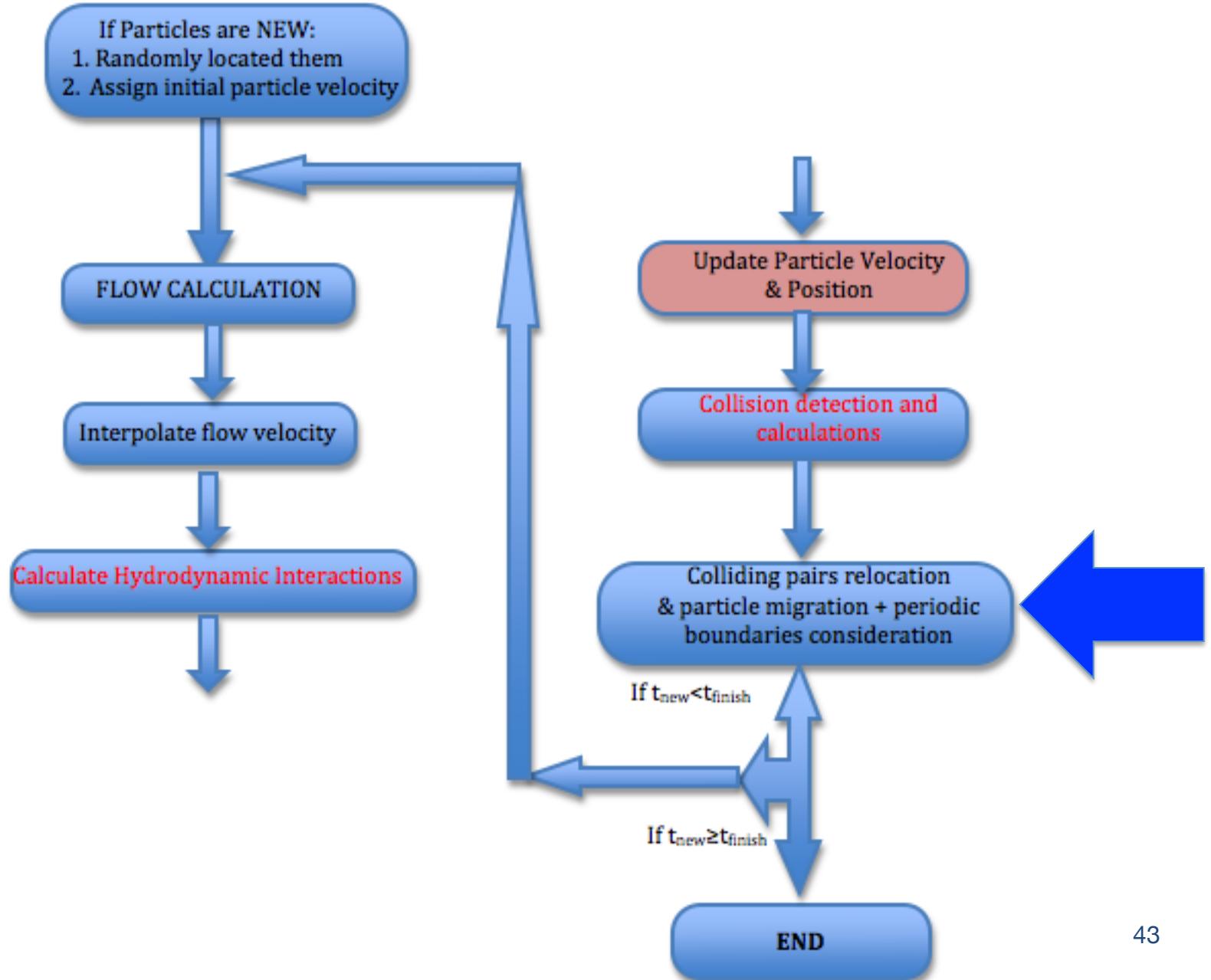


$$L_{\text{HALO}} = 20r_{\max}$$

We used the cell index method and the concept of linked list

$$P_{y_{\text{MAX}}^{\text{CollDetec}}} = P_{z_{\text{MAX}}^{\text{CollDetec}}} = \frac{L}{20r_{\max}}$$

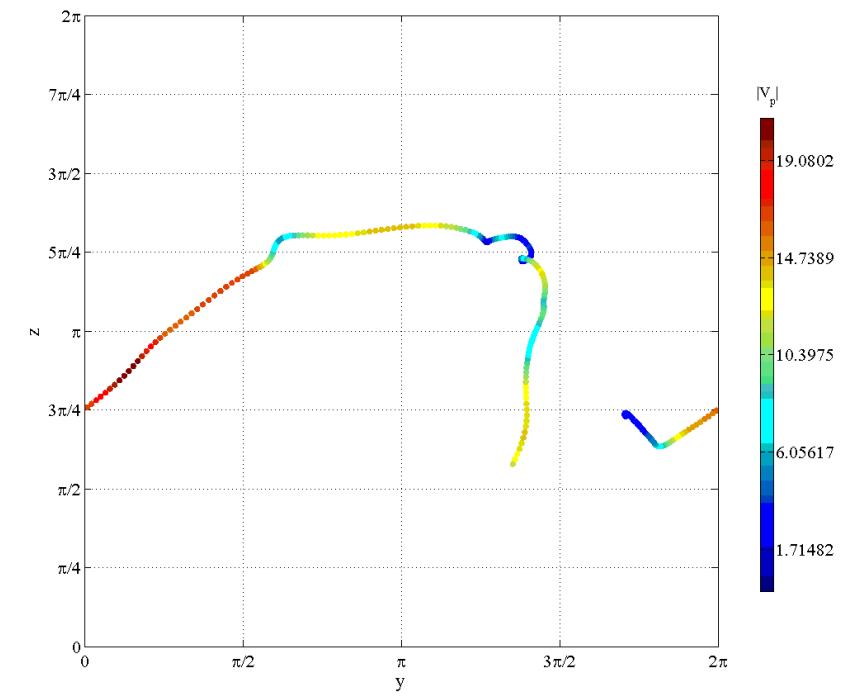
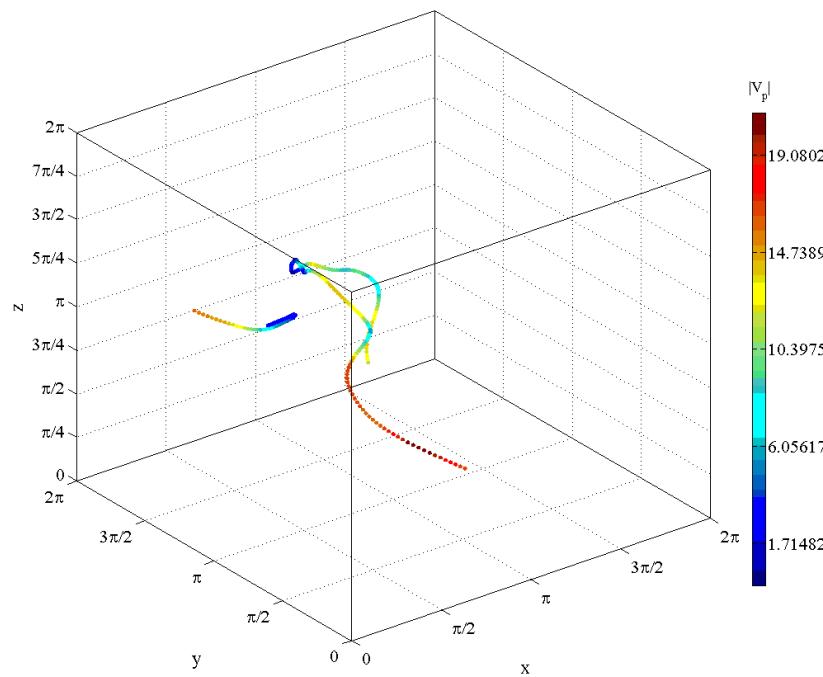
If HDI, only one communication





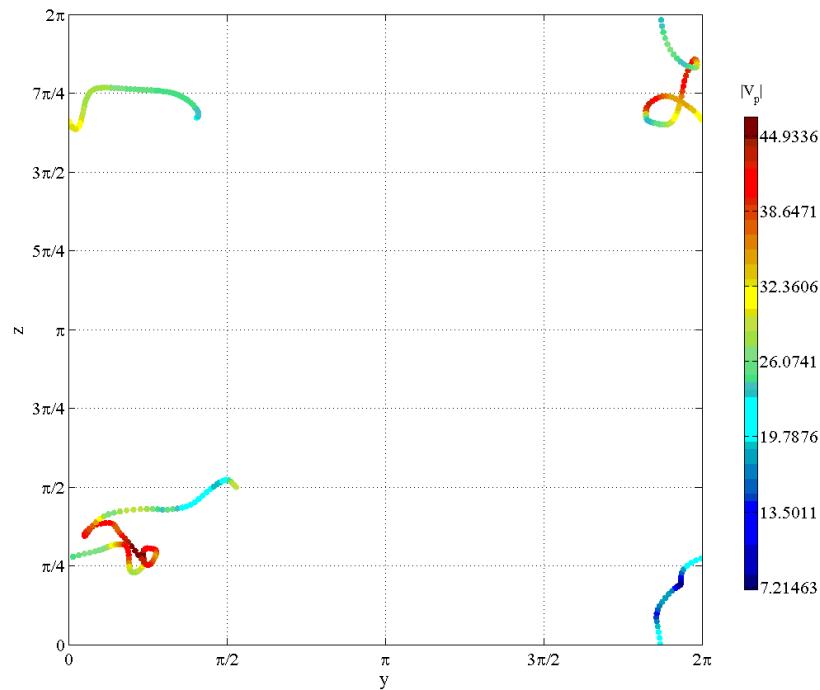
- 128^3 simulation
- 20 micron ($Sv=1.78$)
- 3Te
- 0.5M droplets

When migrating particles, we update the cell-index sorting

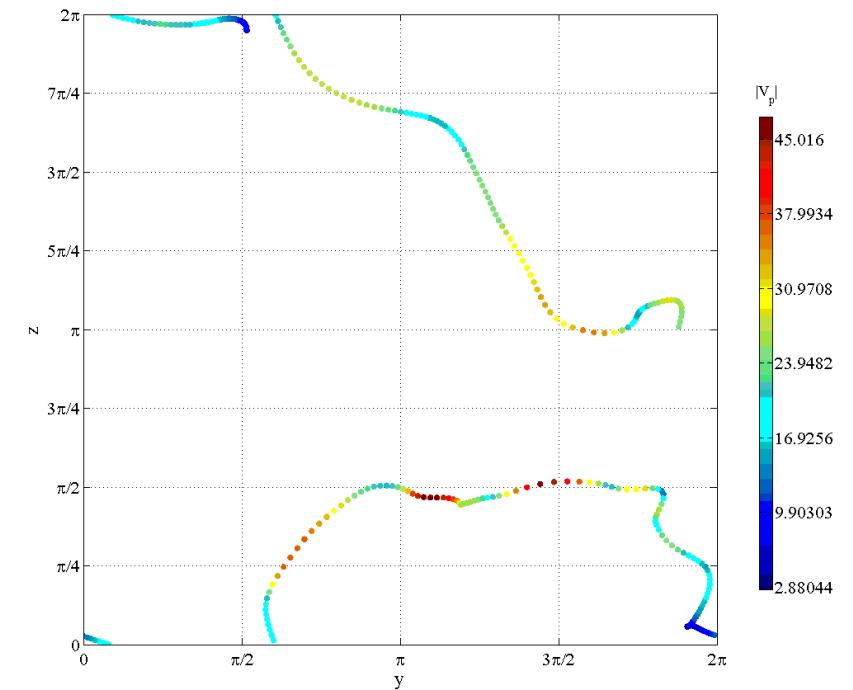


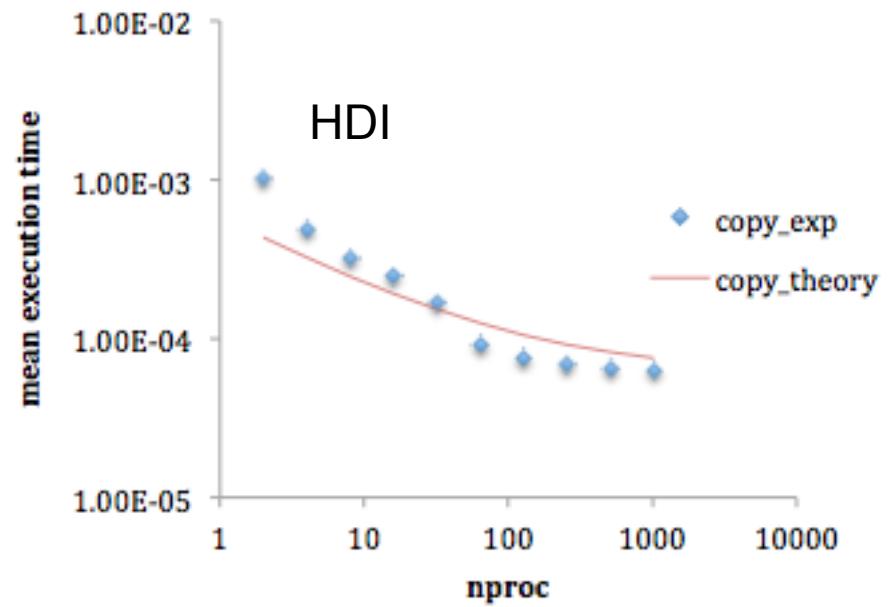
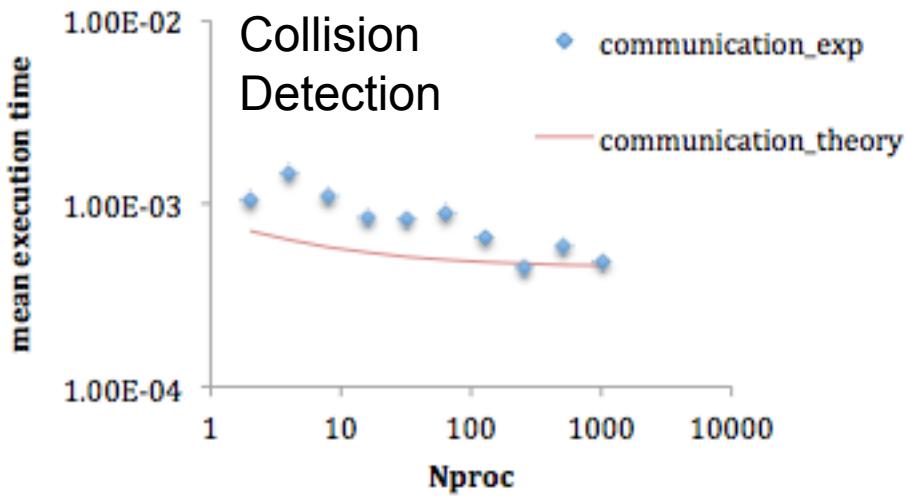
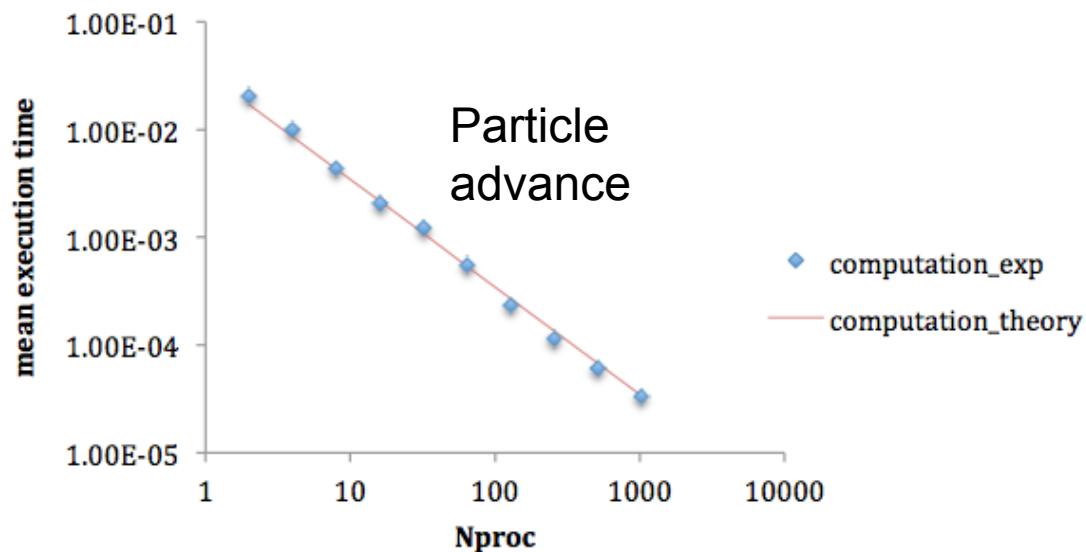


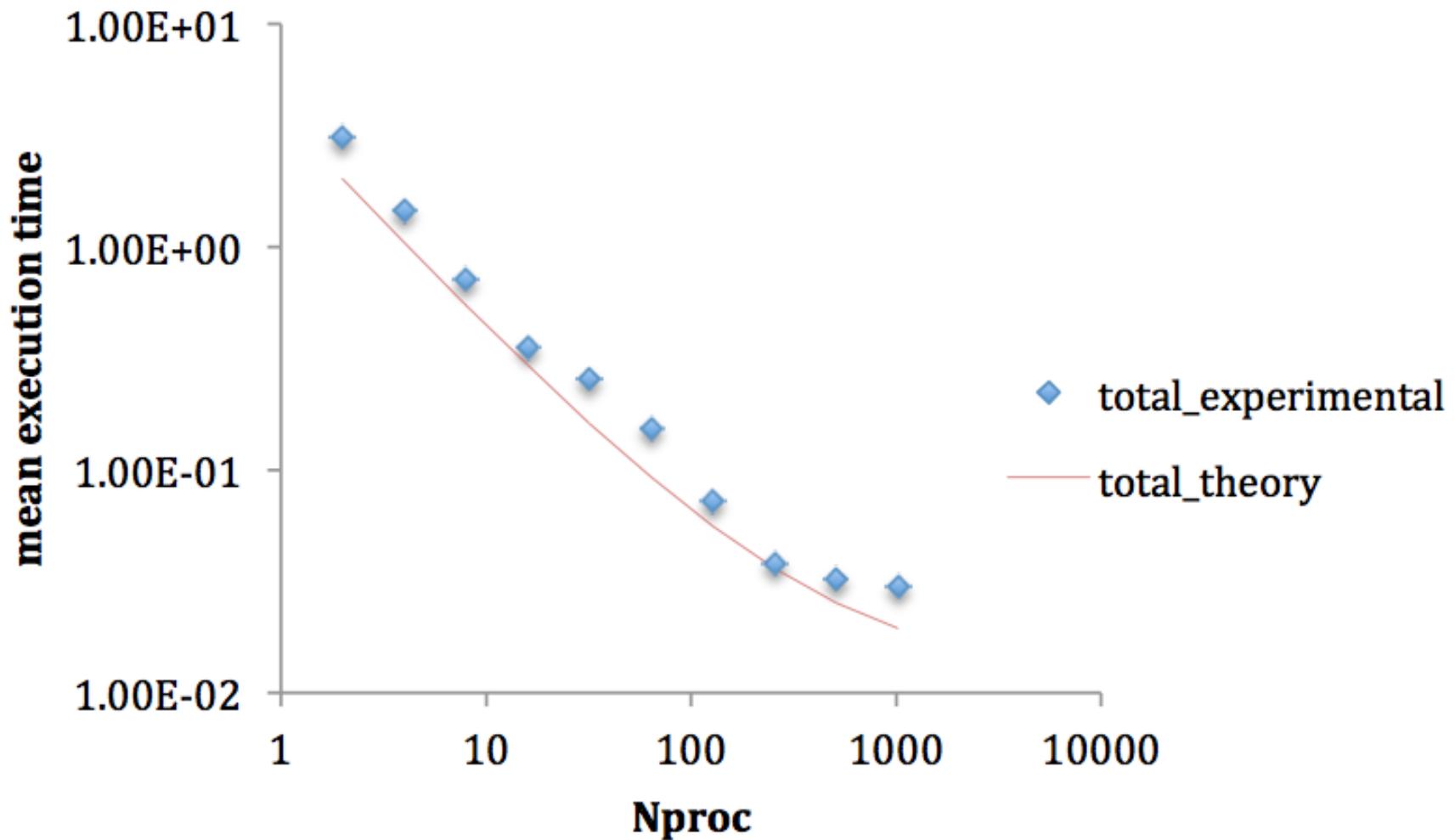
- 128^3 simulation
- 40 micron ($Sv=7.13$)
- 3Te

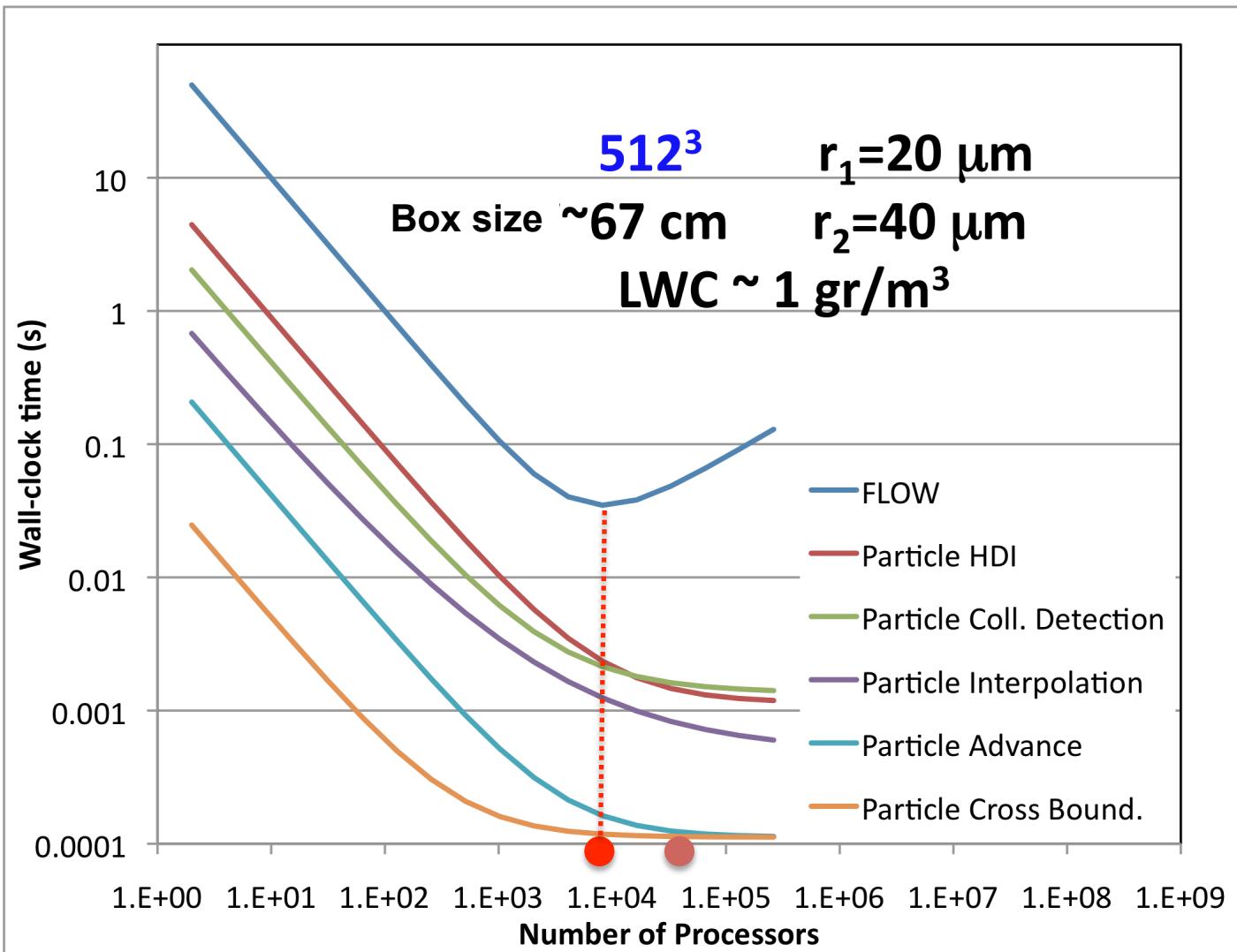


- 128^3 simulation
- 20 micron ($Sv=1.78$)
- 3Te







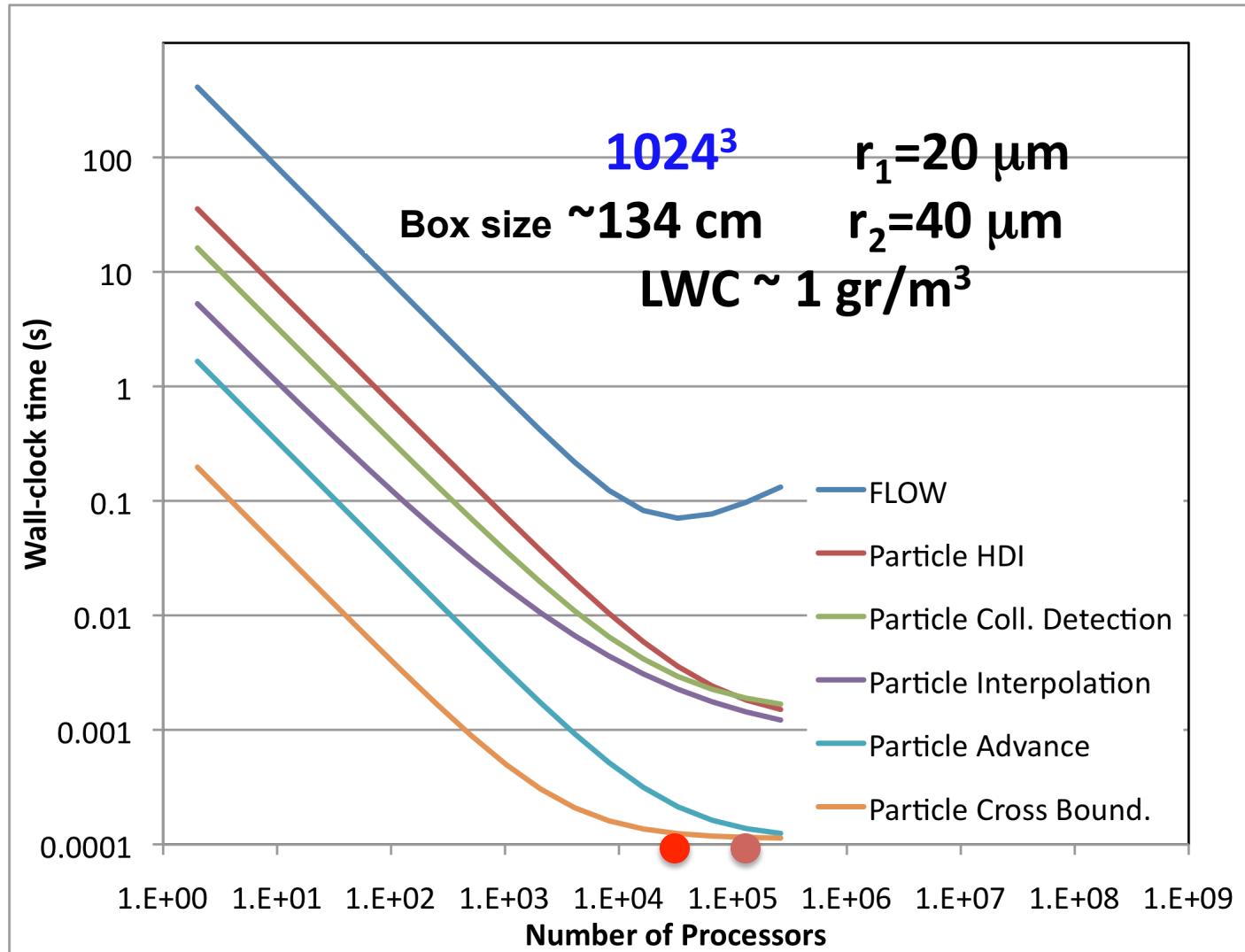


Particles:

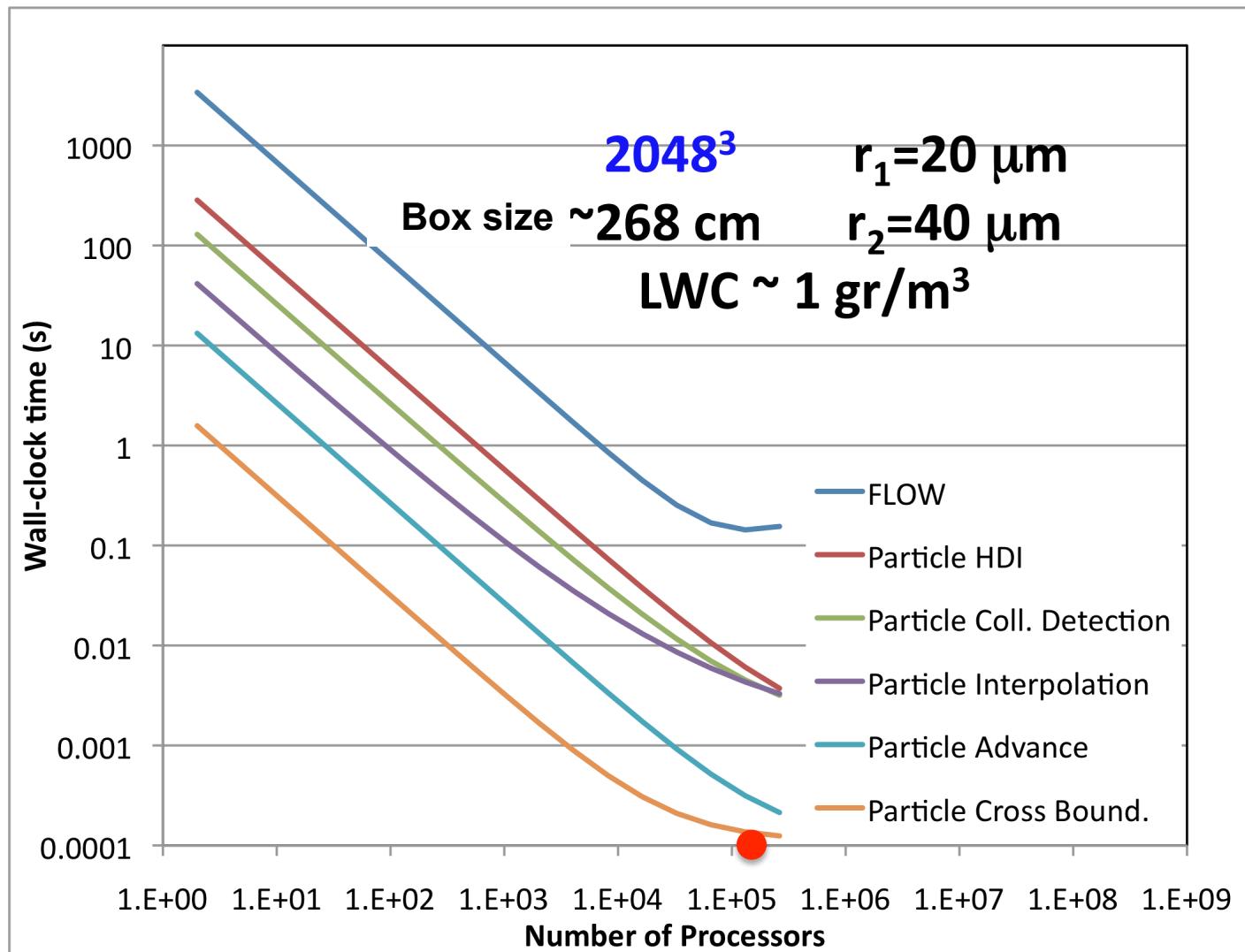
$$P_{y_{\max}} = P_{z_{\max}} = N/3$$

$$P_{MAX} = N^2/9$$

FFT:
$$P_{Max} = \left[\left(\frac{5}{4} \log_2(N^3) t_c + 3t_a + 2t_w \right) \frac{N^3}{t_s} \right]^{2/3}$$



$$\text{Execution time} \approx 0.1 \frac{s}{\text{timestep}} \times 14,000 \frac{\text{timestep}}{T_e} \times 20T_e \approx 8 \text{hours}$$



$$\text{Execution time} \approx 0.2 \frac{s}{\text{timestep}} \times 27,000 \frac{\text{timestep}}{T_e} \times 30T_e \approx 2 \text{days}$$



- ✧ We have developed a highly scalable parallel code to model homogeneous isotropic turbulence with inertial particles using 2D domain decomposition.
- ✧ We were able to optimize HDI calculations. Now the flow (or FFT) calculations are the bottleneck (P_{\max}).
- ✧ In comparison to LBM and FDM, PSM is still a good choice to model HI turbulence in Supercomputers. It is more accurate and, for similar accuracy, it is comparable in terms of parallel performance.
- ✧ IMPROVEMENTS
 - OpenMP+MPI (but if large P , might not help).
 - GPU+CPU.
 - Use Plimpton's scheme for 3DFFT for large P .