A Case Study of Combining Motif-Based Performance-Portable Libraries for Particle-Particle Particle-Mesh Method

Zoe Barbeau, Het Mankad, Samuel Reeve, Phillip Colella, Franz Franchetti, Jeffrey Vetter, and Keita Teranishi

__Motif:__ A compact set of algorithmic abstractions, that, when composed, can express a large class of performance-critical algorithms. Initial list includes dense and sparse linear algebra, structured and unstructured grid, particles, fast Fourier transform (FFT), and Monte Carlo simulations.

Motivation

- Rise in heterogeneity in current HPC systems.
- Huge demand of performance portable software to help domain scientists.
- ➢ ***How to obtain the best performance?*** Use motifs to rewrite the code to different composition of operations to improve performance
- ➢ ***Current Practice:*** Use multiple libraries focused on separate motifs to optimize code for a single scientific application.
- ➢ ***Challenge:*** To find the right combination of libraries to get performance-portability and programming productivity for a given application

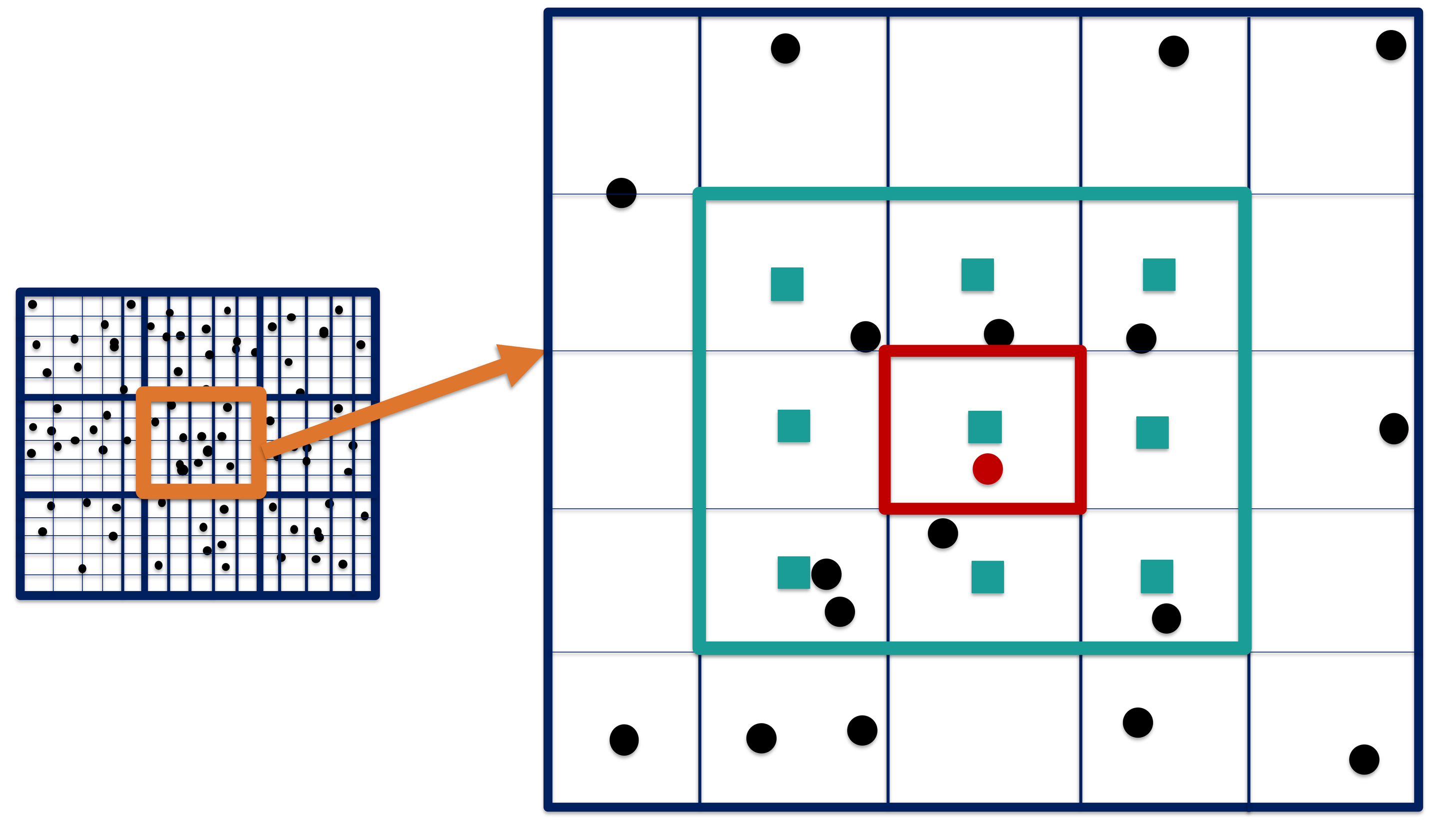
Method of Local Corrections (MLC)

- The Method of Local Corrections (MLC)[1] is a particle-particle particle-mesh method (PPPM)
- It is a vortex method defined for the 3D case as follows

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{\omega} = (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u},$$

$$\boldsymbol{u} = -\nabla \times \Delta^{-1}\boldsymbol{\omega}$$

- Here, $\boldsymbol{\omega}$ is the vorticity and $\boldsymbol{u}$ is the fluid velocity.
- The collection of particles is denoted as $\boldsymbol{\omega}(\boldsymbol{x}, t) \rightarrow \{\boldsymbol{\omega}_p(t), \boldsymbol{x}_p(t)\}_{p \in \mathcal{P}}$.
- The algorithm involves dividing the computation into two parts, namely long-range or far-field Poisson solve (Particle-Mesh) and the short-range or near-field particle interactions (Particle-Particle).
- Applications in molecular dynamics, cosmology, fluid dynamics, electrostatics, and plasma physics.
- ➢ ***Computational Cost:*** MLC reduces the usual cost of computing velocity from $O(N^2)$ to $O(NlogN)$ where N is the number of particles.



- ● Particle in the i^{th} bin where we want to compute the field
- ● Particles in correction radius of 2
- ■ Grid values forming interpolation stencil for the ● particle in the

MLC/PPPM Algorithm

➢ The RK4 scheme is used to solve the time integration and one step of RK4 involves solving the following algorithm 4 times

Deposition: Particle ⟶ Mesh

- Depositing the particle vorticity to near by grid points

$$u_j^{loc} \mathrel{+}= K_\delta(jh - \boldsymbol{x_p}) \cdot \boldsymbol{\omega_p}$$

- Apply Laplacian to the local velocity to obtain the local field F

Cabana[2]

- Using the Cabana linked list cell stencil to locally deposit velocity and calculate F

Particle Mesh (Long-Range)

- Compute the long-range velocity field by free space convolution using FFT

$$U_i = \sum_{j \in D_0} F_j G^h_{i-j}, i \in D$$

FFTX[3]

- Using the FFTX library that provides architecture optimized FFT code to compute convolution

Local Corrections and Interpolation

- Correct the velocity on the grid with local particle velocities for each grid point
- The corrected velocity on the grid is interpolated to particles

Cabana

- Cabana linked list cell stencil is used to find particles in the correction radius of grid point i and calculate their correction

Particle-Particle (Short-Range)

- Local particle interactions

$$\boldsymbol{u_p} \mathrel{+}= \sum_{q \in C^i} K_\delta(\boldsymbol{x_p} - \boldsymbol{x_q}) \cdot \boldsymbol{\omega_q}$$

Cabana

- Cabana `neighbor_parallel_for` is used to for particle-particle interactions

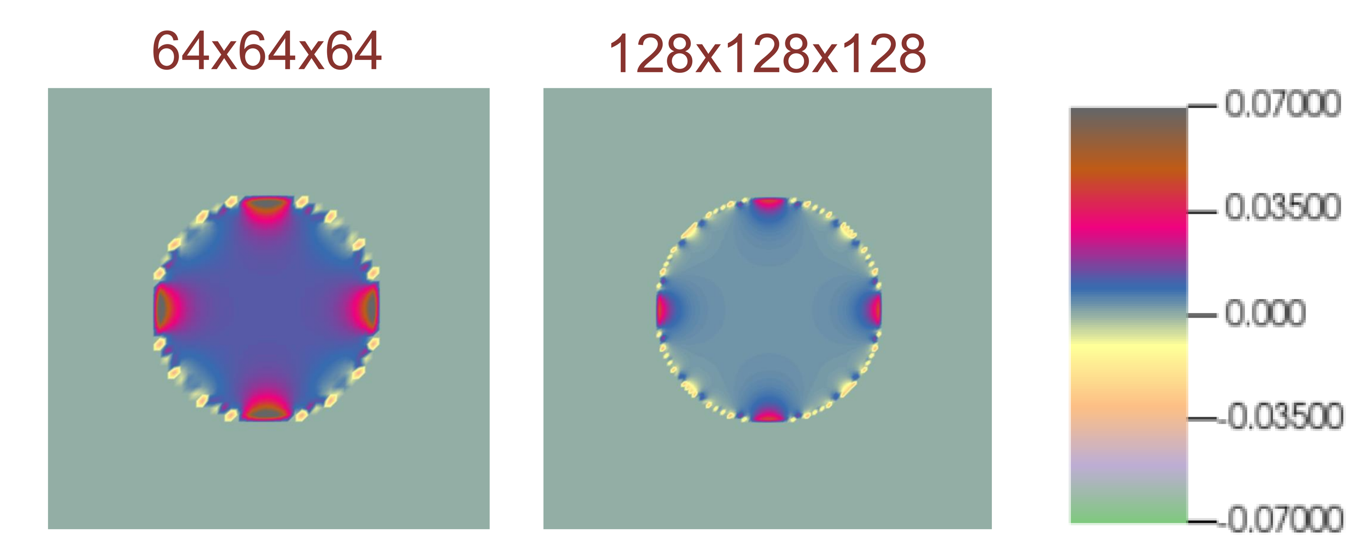
Hill's Vortex Problem for Code Verification

- Let U be the constant velocity and a be the radius of the Hill's vortex

$$\boldsymbol{\omega} = [\frac{15Uy}{2a^2}, \frac{-15Ux}{2a^2}, 0]^T$$

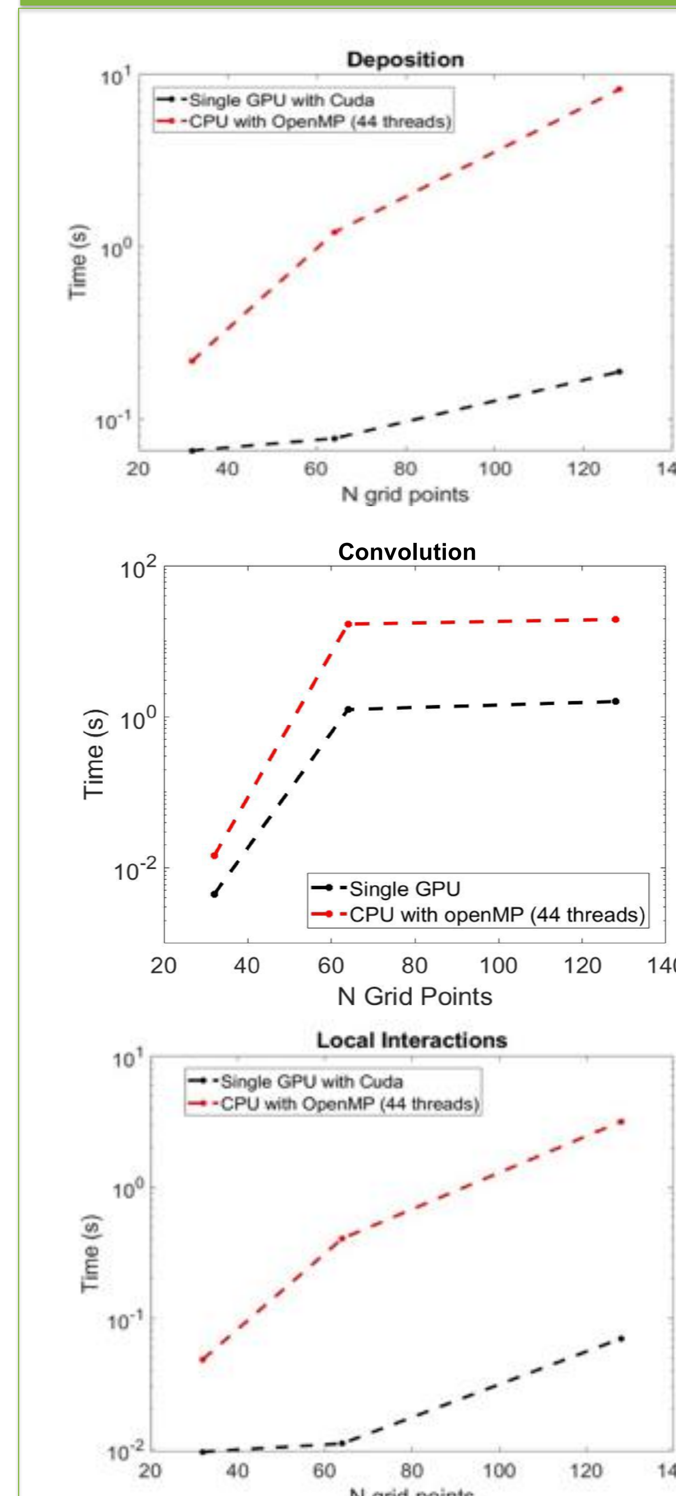
- GPU and CPU results match
- Particle velocity error L^2 - norm $O(h^{1.5})$ and max particle velocity error $O(h)$ is as expected

Z Component Particle Velocity Error for Grid Size 64 and 128

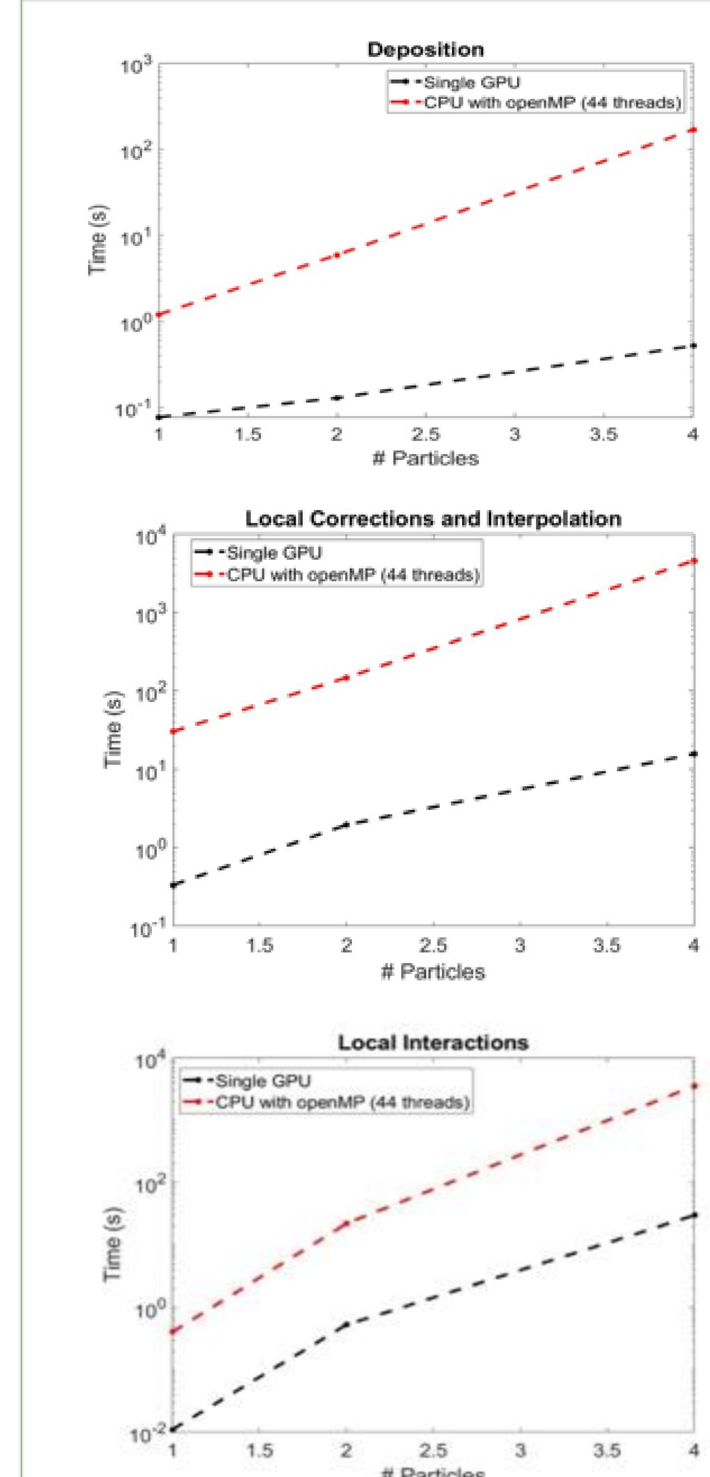


Runtime Results: CPU vs 1 NVIDIA A100 (Perlmutter)

of Particles = 1, Varying Grid Size



Varying # of Particles, Grid Size = 64x64x64



References

[1] Anderson C. R. *A Method of Local Corrections for Computing the Velocity Field due to a Distribution of Vortex Blobs,.*
[2] S. Slattery et. al. *Cabana: A Performance Portable Library for Particle-Based Simulations*
[3] Franchetti et. al. *FFTX and SpectralPack: A First Look*