Optimizing the particle mesh method for extreme scale simulations

Yuki Kaneko, Chiba University

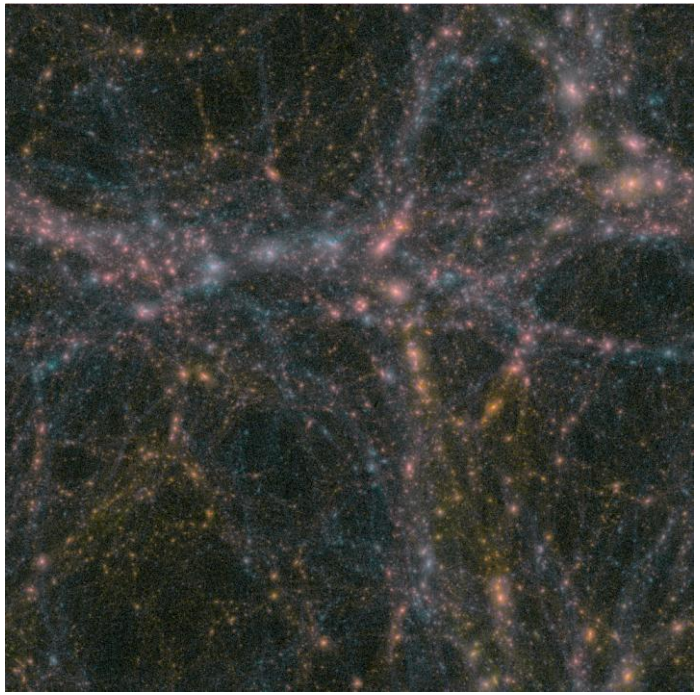
The 10th East Asia Numerical Astrophysics Meeting(EANAM10)

Sep. 15 2025, Jeju Island, South Korea

Outline

1. Introduction
2. Method
3. Result
   - Uniform distribution
   - Non-uniform distribution
   - Overall comparison
4. Summary & Future work

Introduction: N body simulation

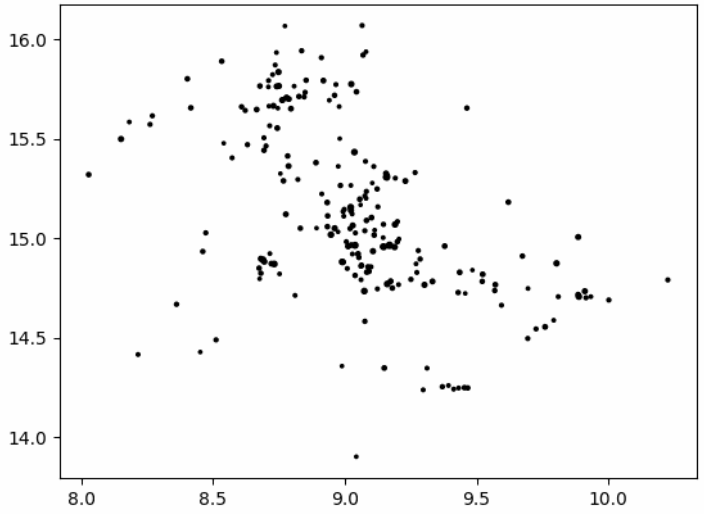


- gravitational N-body system

→many particles interacting through gravity
(e.g., solar system, galaxies)

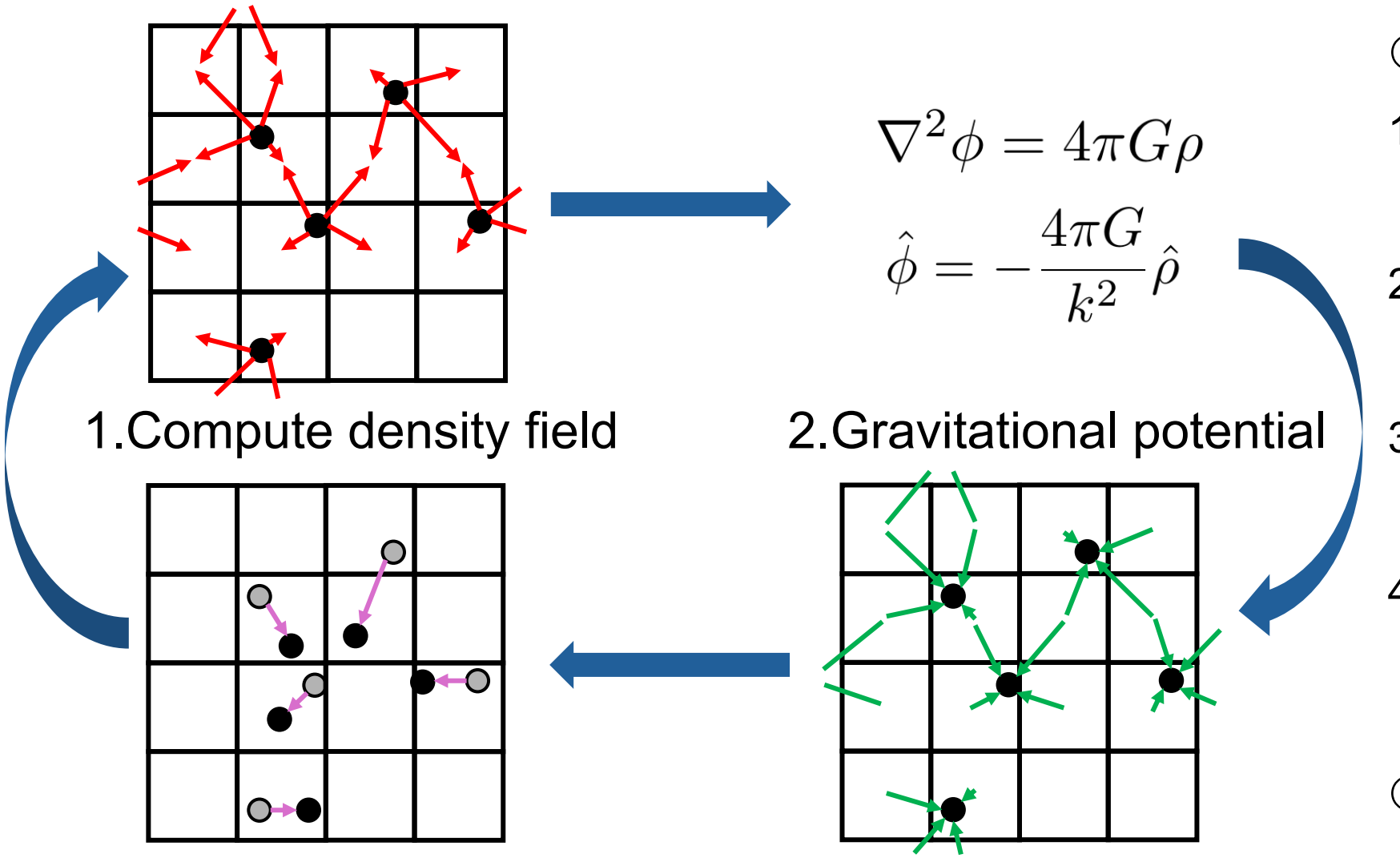
- Acceleration given by Newton's law of gravitation

$$\frac{d^2\boldsymbol{r}_i}{dt^2} = \sum_{j\neq i}^{N} \mathrm{G}m_j \frac{\boldsymbol{r}_j - \boldsymbol{r}_i}{\left(\left|\boldsymbol{r}_j - \boldsymbol{r}_i\right|^2 + \epsilon^2\right)^{\frac{3}{2}}}$$



- Direct method
  This is very large computation
  because of $O(N^2)$ computations
- Tree method
- Particle Mesh method
- TreePM

Introduction: Particle Mesh Method



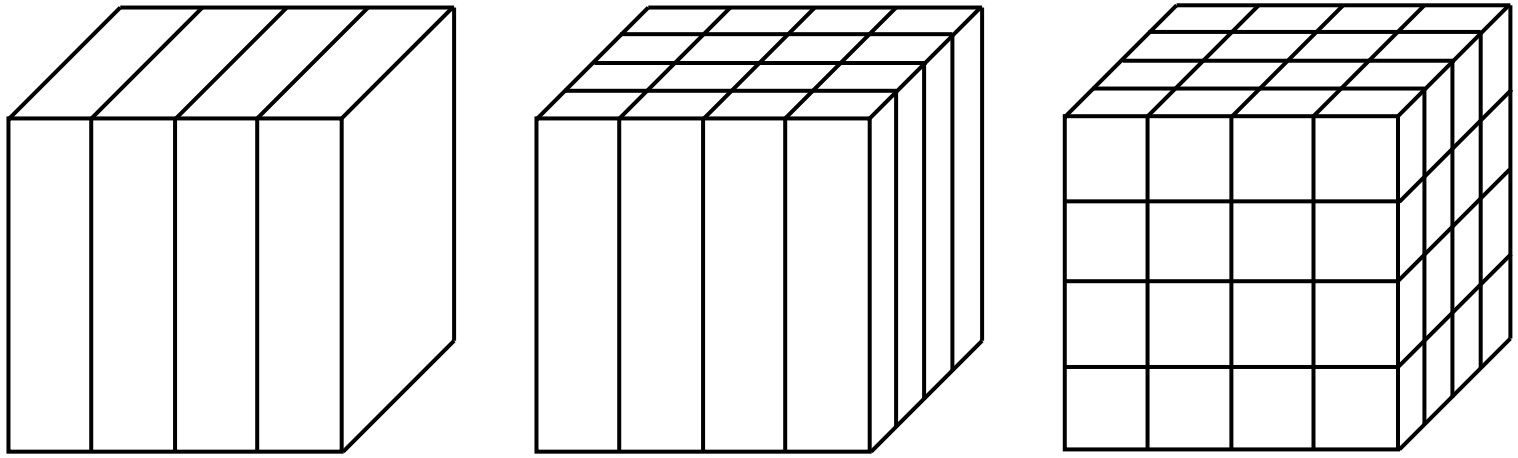
○Basic Steps

1. Assign particle masses to mesh
2. Solve Poisson equation for gravitational potential
3. Interpolate forces back to particles
4. Update particle positions and velocities

○FFT is used to solve Poisson efficiently

Introduction: Process decomposition

FFT Libraries and their decomposition schemes



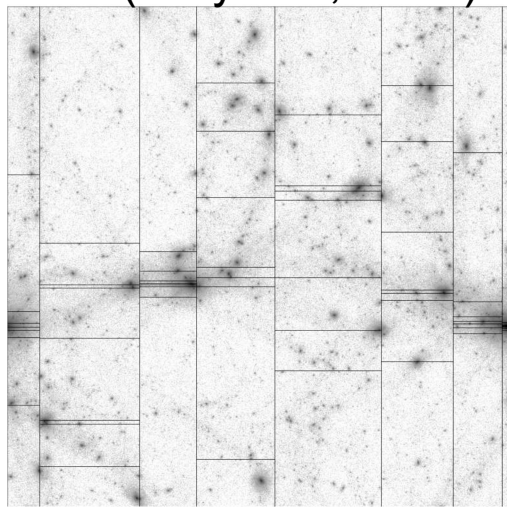
Slab
(1D decomposition)
e.g. FFTW, FFTE

Pencil
(2D decomposition)
e.g. FFTE

Cube
(3D decomposition)
e.g. heFFTe

Reshape

(Ishiyama, 2009)



FFTW : Frigo & Johnson, (2005)
FFTE : Takahashi, (2020)
heFFTe : Ayala et al., (2020)

The domain decomposition used for particle calculations differs from that used in FFT. Therefore, an **additional communication** for exchanging data among all processes is required to reconcile these decompositions during parallel execution.

Introduction: motivation

・Particle computations use a domain decomposition optimized for particle interactions, while FFT relies on library-specific decompositions.

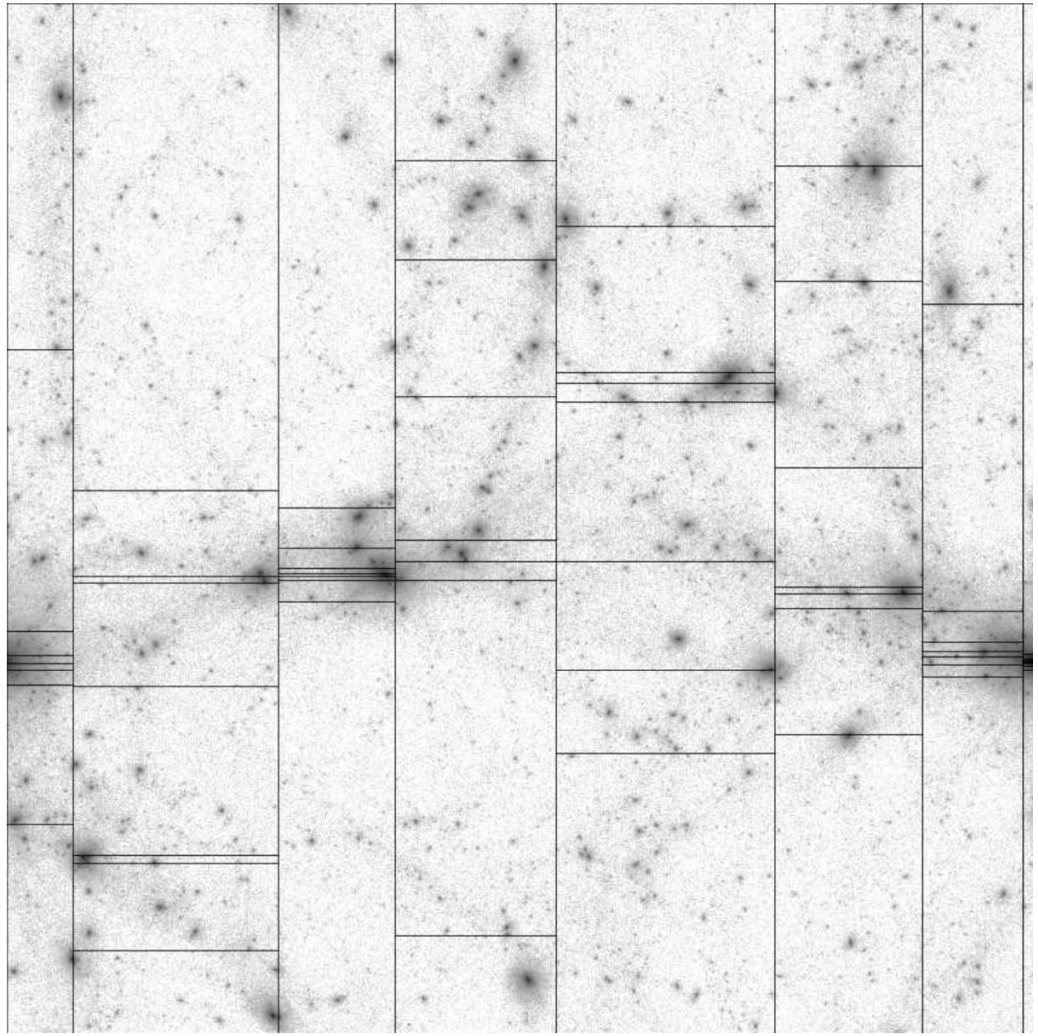
・These decompositions do not match, so data must be communicated and transposed between processes.

・At large scales, this communication cost becomes the main bottleneck.

・The communication overhead depends on the FFT library and its decomposition scheme.

・Therefore, it is important to compare different FFT libraries to evaluate their efficiency in large-scale simulations.

Method



Example of non-uniform domain decomposition

Two types of particle distributions are used in the experiments

1.Uniform distribution:

Particles are evenly distributed, corresponding to early-universe conditions (high-z).
Scale free

2.Non-uniform distribution:

Particles are clustered, representing late-time universe conditions (low-z).
box size : 1.0Gpc/h redshift : 0
This non-uniform distribution is obtained from actual cosmological simulations.

Method

- **FFT Libraries**

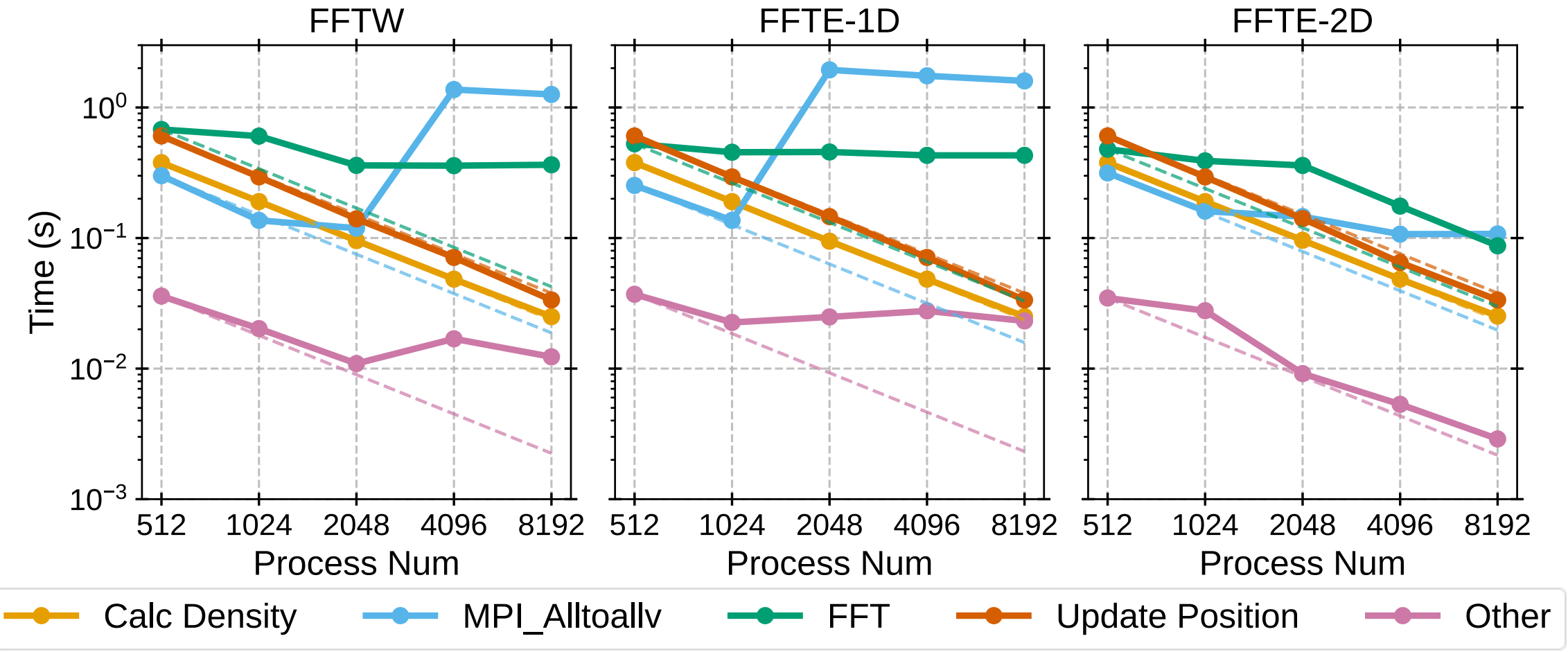
  1. Slab : FFTW, FFTE-1D 2. pencil : FFTE-2D

- **Particle number** : 4096^3
- **Mesh size** :

  1. 2048^3 2. 4096^3

- **System** : Supercomputer Fugaku
  - 48 cores per node and 2.0 GHz clock frequency
  - total node : 158,976 nodes
  - Nodes used in this study : 128-2048
  - 12threads and 4processes per node

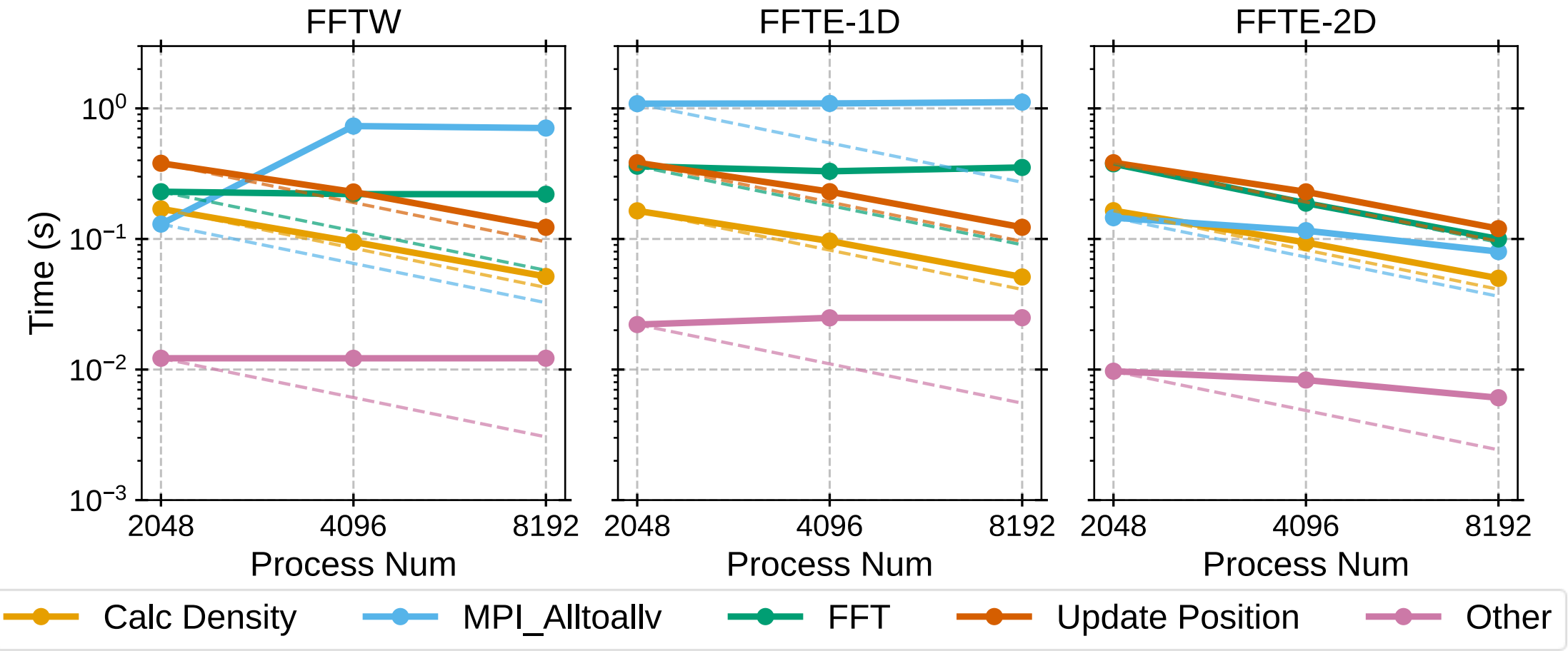
Result Uniform distribution(Ng = 2048)



・FFTW/FFTE-1D (slab): saturate when the number of processes reaches the slab limit.

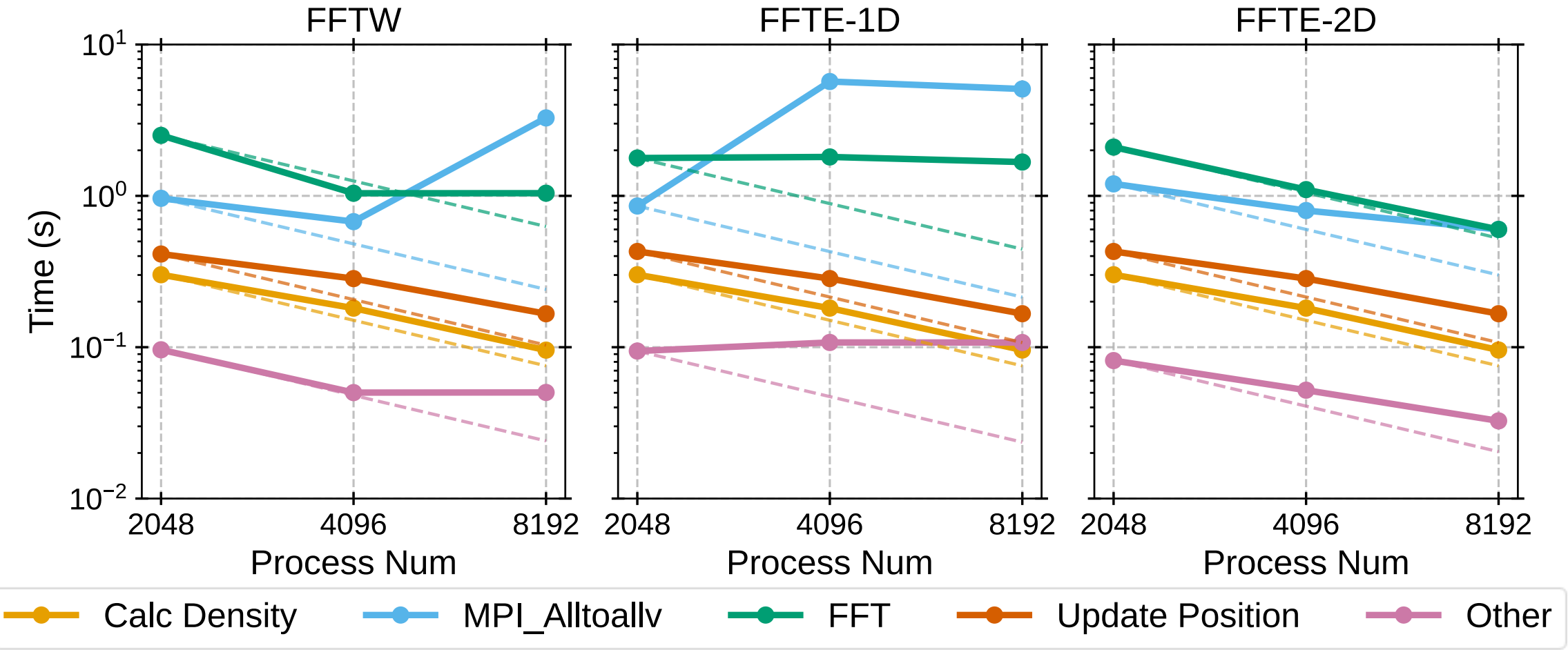
・FFTE-2D (pencil): Scales to larger process numbers

Result Non-uniform distribution(Ng = 2048)



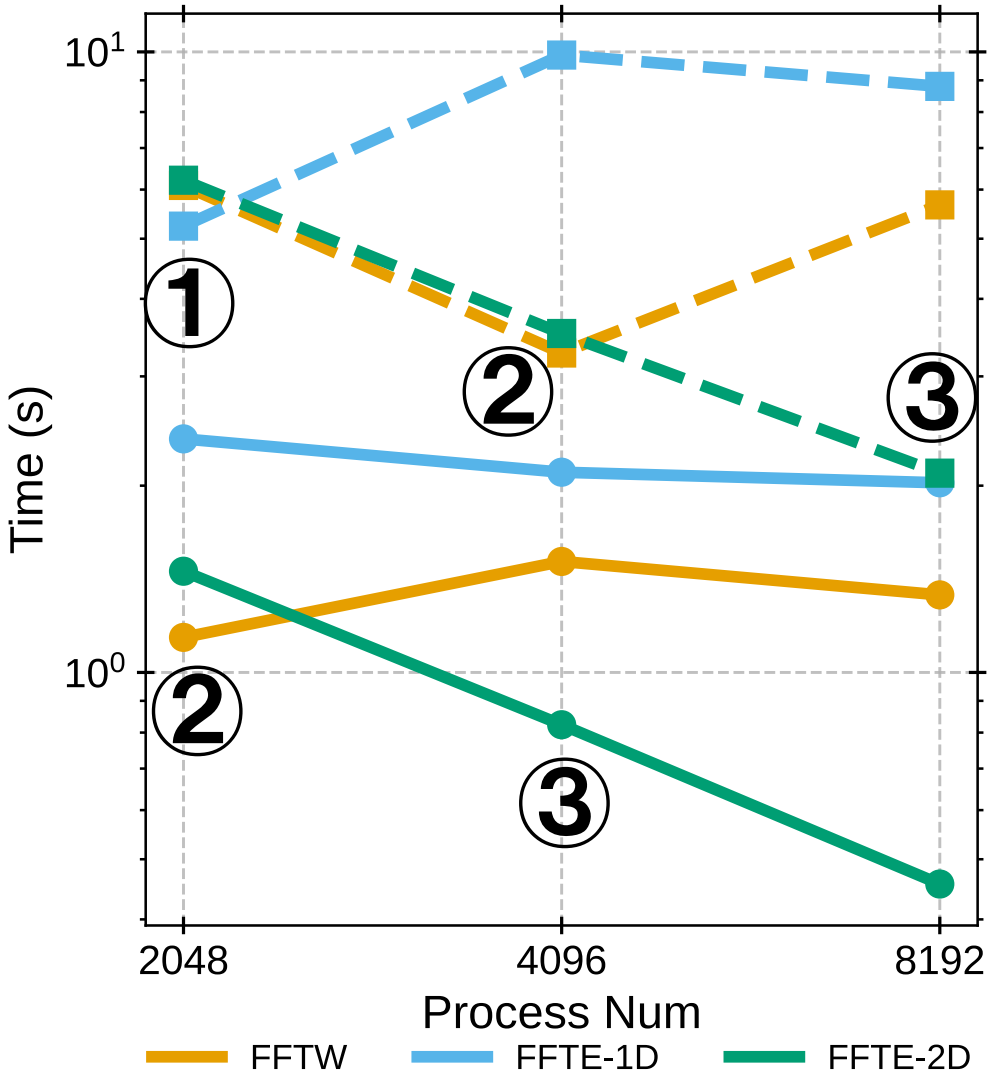
- Process range is limited due to particle imbalance
- Runtime increases slightly, but parallel efficiency remains similar to the uniform case

Result Non-uniform distribution (Ng = 4096)



- The scaling behavior remains similar to previous results.
- But with larger grids, FFTW scales up to 4096 processes.

Result Overall comparison



Solid line: Ng = 2048, Dashed line: Ng = 4096
(both non-uniform)

①FFTE-1D is fastest when Process Num < Ng.

②FFTW is fastest when Process Num = Ng.

③FFTE-2D is fastest when Process Num > Ng.

Summary and future work

・ Slab decomposition saturates when the number of processes reaches the grid size, while pencil decomposition continues to scale beyond it.

・The performance strongly depends on the decomposition:
FFTE-1D is fastest when Process Num < Ng
FFTW is fastest when Process Num = Ng
FFTE-2D is fastest when Process Num > Ng

・Future work :
・Evaluate cube-based FFT decomposition (heFFTe)

・Investigate performance differences across node topologies on Fugaku

・Further optimize communication overhead