GENESIS
Generalized-ensemble simulation system

RIKEN AICS
理化学研究所 計算科学研究機構
粒子系生物物理研究チーム
Computational Biophysics Research Team
RIKEN Advanced Institute for Computational Science

# GENESIS tutorial 2:

## Advanced MD (how to speed up GENESIS)

Workshop “Frontiers in Computational Biophysics and Biochemistry”

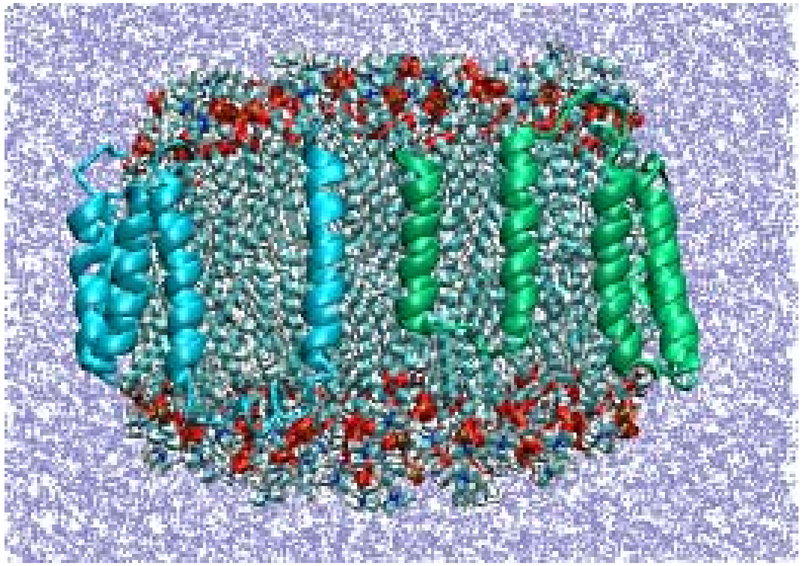
2017/02/28

J. Jung

# Contents

- 15:30 – 17:00 GENESIS tutorial 2
  - ➢ General comments in GENESIS
  - ➢ How to accelerate MD speed of GENESIS : GPU
  - ➢ How to accelerate MD speed of GENESIS : Multiple time step integration
  - ➢ How to accelerate MD speed of GENESIS : Further suggestions

# Test system



- 92,224 atoms
- 92,118 bonds
- 74,136 angles
- 74,130 dihedral angles

# Molecular Dynamics (MD)

1. Energy/forces are described by classical molecular mechanics force field.
2. Update state according to equations of motion.

$$\frac{d\mathbf{r}_i}{dt} = \frac{\mathbf{p}_i}{m}$$

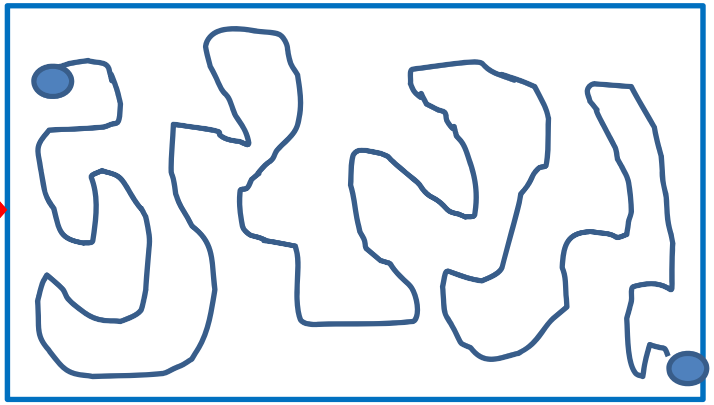
$$\frac{d\mathbf{p}_i}{dt} = \mathbf{F}_i$$

Equation of motion

$$\mathbf{r}_i(t+\Delta t) = \mathbf{r}_i(t) + \frac{\mathbf{p}_i}{m}\Delta t$$

$$\mathbf{p}_i(t+\Delta t) = \mathbf{p}_i(t) + \mathbf{F}_i\Delta t$$

Integration

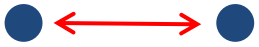


Long time MD trajectory
=> Ensemble generation

Long time MD trajectories are important to obtain thermodynamic quantities of target systems.

# Potential energy in MD using PME

$$E_{\text{total}} = \sum_{\text{bonds}} k_b (b - b_0)^2$$



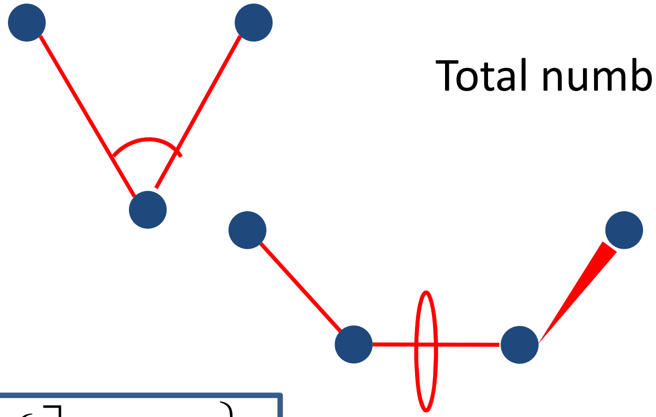
O($\underline{N}$)

Total number of particles

$$+ \sum_{\text{angles}} k_a (\theta - \theta_0)^2$$

O($N$)

$$+ \sum_{\text{dihedrals}} V_n [1 + \cos(n\omega - \gamma)]$$



O($N$)

$$+ \boxed{\sum_{j=1}^{N-1} \sum_{i=j+1}^{N} \left\{ \varepsilon_{ij} \left[ \left( \frac{r_{0ij}}{r_{ij}} \right)^{12} - 2 \left( \frac{r_{0ij}}{r_{ij}} \right)^{6} \right] + \frac{q_i q_j}{r_{ij}} \right\}}$$

O($N^2$)

**Main bottleneck in MD**

$$\boxed{\sum_{|i-j|<R} \left\{ \varepsilon_{ij} \left[ \left( \frac{r_{0ij}}{r_{ij}} \right)^{12} - 2 \left( \frac{r_{0ij}}{r_{ij}} \right)^{6} \right] + \frac{q_i q_j \operatorname{erfc}(\alpha r_{ij})}{r_{ij}} \right\}} + \boxed{\sum_{\mathbf{k} \neq 0} \frac{\exp(-\mathbf{k}^2 / 4\alpha^2)}{\mathbf{k}^2} \mathrm{FFT}(Q(\mathbf{k}))}$$

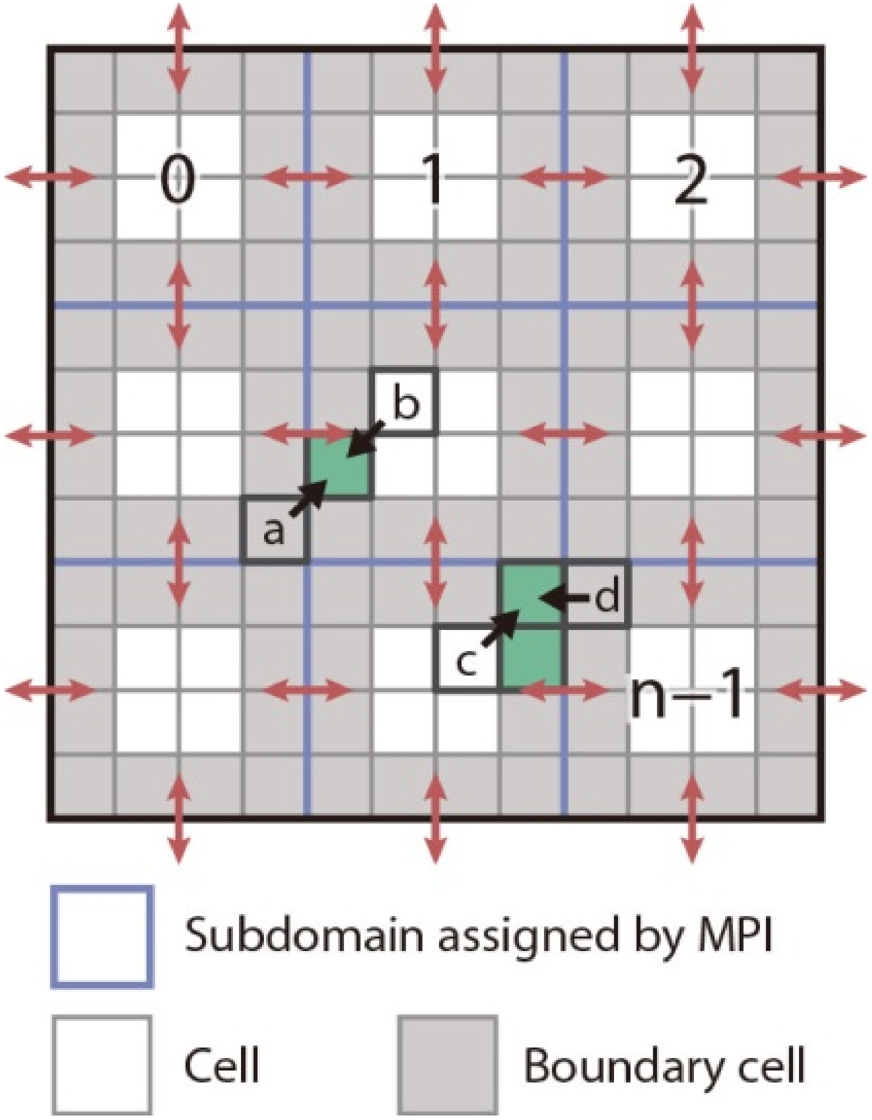
Real space, O($N$)

Reciprocal space, O($N\log N$)

# Domain decomposition in GENESIS

- In GENESIS, the total system is subdivided by subdomain according to MPI processors.

- Subdomain is again subdivided by cells (basic unit) which has dimensions larger than half of cutoff distance.



# Comments on system size (1)

- The subdomain size should be greater than cutoff distance

- Cell size should be greater than half of cutoff distance

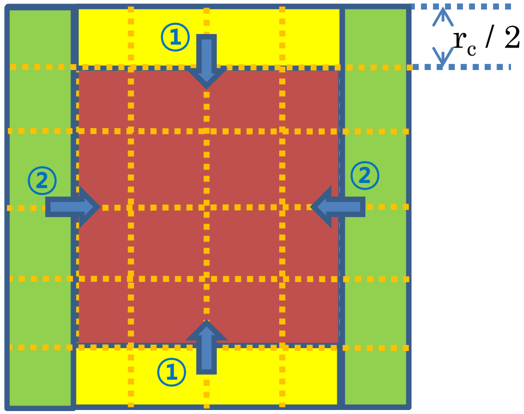
- In each subdomain, there should be at least two cell in each direction (the number of cells in each dimension should be more than twice of the number of domains).

- You can check this from output file

```
Setup_Boundary_Cell> Set Variables for Boundary Condition
  domains (x,y,z) =          2         2         2
  ncells (x,y,z)  =          8         8         8
```

# Comments on system size (1)

- The subdomain size should be greater than cutoff distance
- Cell size should be greater than half of cutoff distance
- In each subdomain, there should be at least two cell in each direction (the number of cells in each dimension should be more than twice of the number of domains for fast communication).



```
Setup_Boundary_Cell> Set Variables for Boundary Condition
  domains (x,y,z) =          4         2         2
  ncells (x,y,z)  =         16        16        10
```

# Comments on system size (2)

- There should be at least 5 cells in each dimension. If not, program finishes with error message.

**Input**

```
[ENERGY]
switchdist    = 9.0
cutoffdist    = 10.0
pairlistdist  = 12.0

[BOUNDARY]
type  = PBC
box_size_x    = 37.7
box_size_y    = 37.7
box_size_z    = 37.7
```

**Outut**

```
Setup_Boundary_Cell> too small boxsize/pairlistdist. shorter
pairlistdist or larger boxsize or less MPI processors should
be used.  rank_no =     0
```

# You can estimate cell sizes and numbers

$$n_x = \frac{B_x}{r_c/2} - \mathrm{mod}\left(\frac{B_x}{r_c/2}, P_x\right)$$

$n_x$ : number of cells in $x$ dimension

$B_x$ : System size in $x$ dimension

$r_c$ : pairlist cutoff distance

$P_x$ : Subdomain number in $x$ dimension

```
[ENERGY]
switchdist     = 10.0
cutoffdist     = 12.0
pairlistdist   = 13.5

[BOUNDARY]
type  = PBC
box_size_x     = 108.8612
box_size_y     = 108.8612
box_size_z     = 77.758
```

```
mpirun –np 1 {PATH}/spdyn test1.inp
```

```
domains (x,y,z) =   4     2     2
ncells (x,y,z)  =  16    16    11
```

```
mpirun –np 16 {PATH}/spdyn test1.inp
```

```
domains (x,y,z) =   4     2     2
ncells (x,y,z)  =  16    16    10
```

# If there are constraints for bonds involving hydrogens, we should consider larger cell sizes

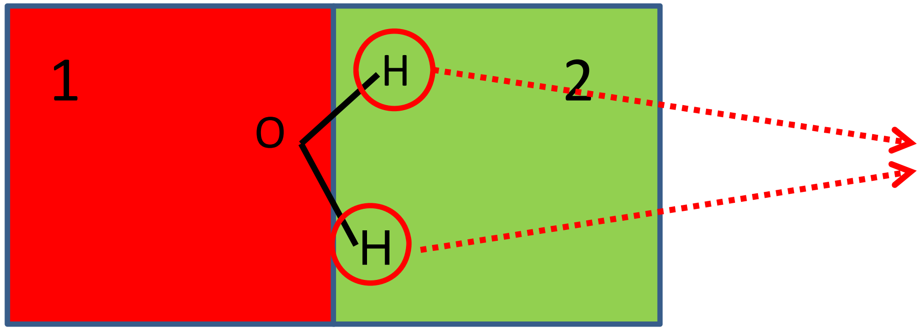
$$n_x = \frac{B_x}{r_c/2} - \mathrm{mod}\left(\frac{B_x}{r_c/2}, P_x\right)$$

No constraints

$$n_x = \frac{B_x}{(r_c+2)/2} - \mathrm{mod}\left(\frac{B_x}{(r_c+2)/2}, P_x\right)$$

Constraints

- If there are constraints, only heavy atoms are considered for assigning subdomain and cells
- Hydrogen atoms have the same subdomain and cell assignment as the heavy atoms bonded.
- In this case, we should consider pairwise distance up to parilsit cutoff + 2r$_{OH}$



assigned to cell index 1

# Comments on MPI processors

- MPI processors is automatically defined if domain information is not written in the control input.
- If domain information is written in the control input, the number of domains should be identical to the number of MPI processors.

Input : test3.inp

```
[BOUNDARY]
type  = PBC
domain_x       = 2
domain_y       = 2
domain_z       = 2
```

execution

```
mpirun –np 16 {PATH}/spdyn test3.inp
```

output

```
Setup_Processor_Number> # of process is not domain_x *
domain_y * domain_z  rank_no =     0
```

# Comments on PME grid numbers (1)

- GENESIS is currently using FFTE for 1-dim’l FFT kernel
- In FFTE, only the multiple of 2, 3, and 5 are available as PME grid numbers.
- If you write grid numbers that are not multiple of 2, 3, or 5, the program automatically increases the grid numbers.

Input : test4.inp

```
[ENERGY]
pme_ngrid_x   =  99
pme_ngrid_y   =  99
pme_ngrid_z   =  79
```

execution

```
mpirun –np 1 {PATH}/spdyn
test4.inp
```

output

```
WARNING: PME grid number is different from the input
pme_ngrid(x,y,z)=          100          100          80
```

# Comments on deciding PME grid numbers (2)

Furthermore, grid numbers should be divisible by some combination of domain numbers

- pme_grid_x : multiple of (2 × domain_x)
- pme_grid_y : multiple of (domain_y × domain_z)
- pme_grid_z : multiple of (domain_x × domain_z) and multiple of (domain_y × domain_z)

Input : test4.inp

```
[ENERGY]
pme_ngrid_x    = 99
pme_ngrid_y    = 99
pme_ngrid_z    = 79
```

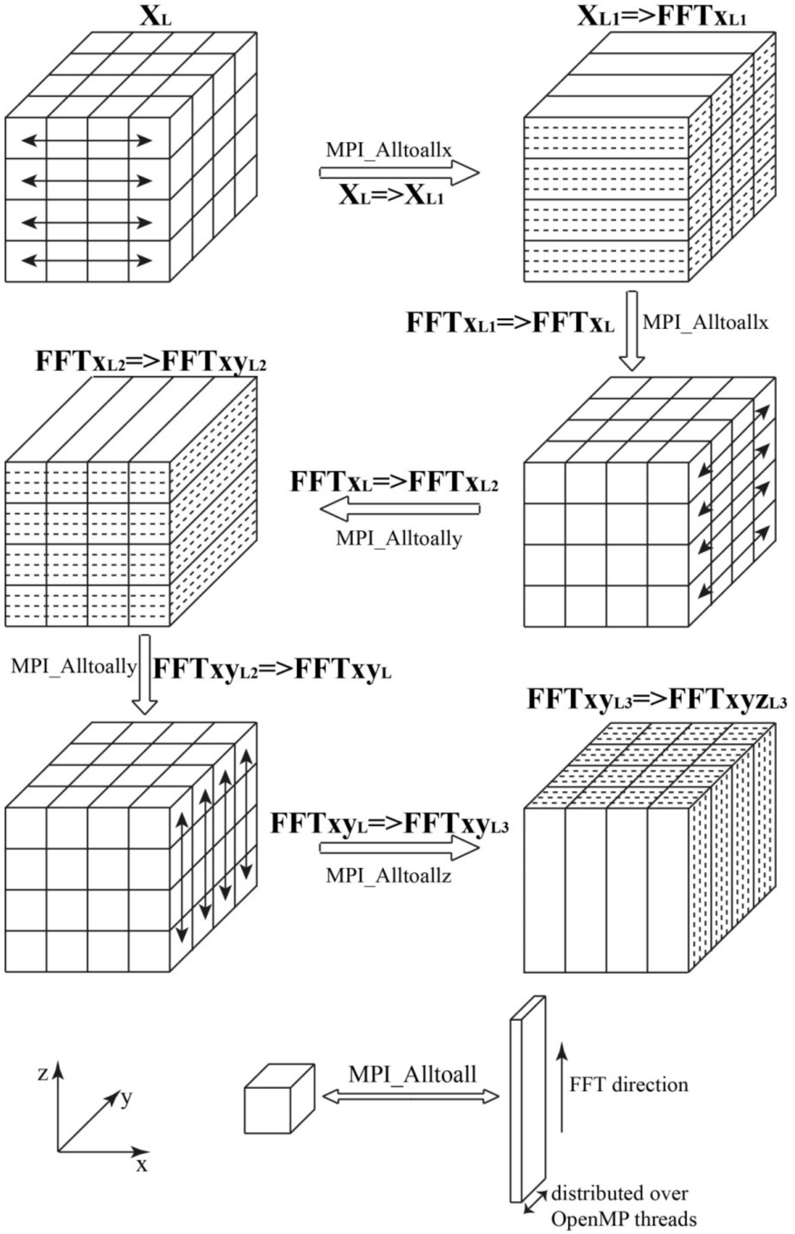
execution

```
mpirun –np 16 {PATY}/spdyn
test4.inp
```

output

```
 WARNING: PME grid number is different from the input
  pme_ngrid(x,y,z)=         100         120          80
```

# Why there are restrictions of PME grid numbers



- GENESIS make use of volumetric decomposition for FFT which requires one dimensional MPI_Alltoall for three directions
- To perform MPI_Alltoall, some restrictions exist (divisibility)
- Without restrictions, we can perform MPI_Alltoallv, but it decreases the performance

# My suggestions

- Write domain information explicitly in the control input and run with corresponding MPI processors
- Given domain information, write PME grid numbers that is not changed in execution (grid spacing should be less than 1.5).

Input : test5.inp

```
[ENERGY]
pme_ngrid_x    = 128
pme_ngrid_y    = 128
pme_ngrid_z    =  96

[BOUNDARY]
type           = PBC
domain_x       = 4
domain_y       = 2
domain_z       = 2
```

execution

```
mpirun –np 16 {PATH}/spdyn
test4.inp
```

# Exercise

1. Run test1.inp and test2.inp with 16 and 1 MPI processors. Check the domain size and cell sizes from the outputs.

2. Run test3.inp with 16 and 8 MPI processors.

3. Run test4.inp with various MPI processors and check domain numbers and PME grid numbers.

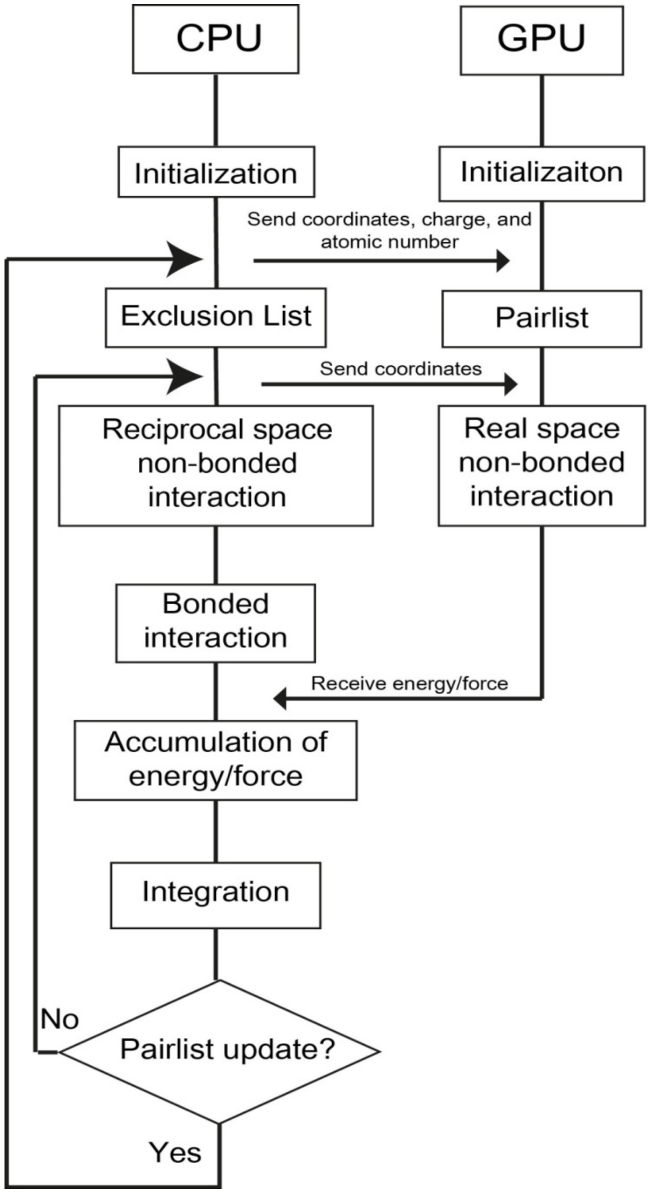
4. Run test5.inp with 16 MPI processors. Write inputs for 32, 64, and 128 MPI processors in which PME grid numbers do not change when executed.

Please run mpirun with orte_base_help_aggregate 0
(mpirun –mca orte_base_help_aggregate 0 ./spdyn )

# Contents

- 15:30 – 17:00 GENESIS tutorial 2
  - ➢ General comments in GENESIS
  - ➢ How to accelerate MD speed of GENESIS : GPU
  - ➢ How to accelerate MD speed of GENESIS : Multiple time step integration
  - ➢ How to accelerate MD speed of GENESIS : Further suggestions

# Overview of CPU+GPU calculations


CPU
GPU
Initialization
Initializaiton
Send coordinates, charge, and atomic number
Exclusion List
Pairlist
Send coordinates
Reciprocal space non-bonded interaction
Real space non-bonded interaction
Bonded interaction
Receive energy/force
Accumulation of energy/force
Integration
No
Pairlist update?
Yes

1. Computation intensive work : GPU
   - Pairlist
   - Real space non-bonded interaction
2. Communication intensive work or no computation intensive work : CPU
   - Reciprocal space non-bonded interaction with FFT
   - Bonded interaction
   - Exclusion list
3. Integration is performed on CPU due to file I/O.

# Compile of GENESIS for GPU usage

- CPU+GPU with single precision :

  -
    ```
    ./configure –enable-single –enable-gpu –with-cuda={path}
    ```
- CPU+GPU with double precision :

  -
    ```
    ./Configure –enable-gpu –with-cuda={path}
    ```
- Only CPU with single precision :

  -
    ```
    ./configure –enable-single
    ```
- Only CPU with double precision :

  -
    ```
    ./configure
    ```
- Here, we prepared four binaries

**CPU single**： `/home2/data/genesis/bin.CPU.sp/spdyn`

**CPU double**： `/home2/data/genesis/bin.CPU.dp/spdyn`

**CPU+GPU single**： `/home2/data/genesis/bin.GPU.sp/spdyn`

**CPU+GPU double**： `/home2/data/genesis/bin.GPU.dp/spdyn`

# Comparison of speed
# (Time for 2000 MD steps)

- CPU : Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz, 24 cores
- GPU : NVIDIA GeForce GTX 1080
- 8 MPIs with 3 OpenMPs
- Test input file : test6.inp

| Condition | Time (sec) |
|---|---|
| CPU+GPU (double) | 40.685 |
| CPU+GPU (single) | 26.307 |
| CPU only (double) | 109.52 |
| CPU only (single) | 101.284 |

# Contents

- 15:30 – 17:00 GENESIS tutorial 2
    - General comments in GENESIS
    - How to accelerate MD speed of GENESIS : GPU
    - How to accelerate MD speed of GENESIS : Multiple time step integration
    - How to accelerate MD speed of GENESIS : Further suggestions

# Multiple time step integration (1)

- **Real space bonded interaction** : small amount of computation

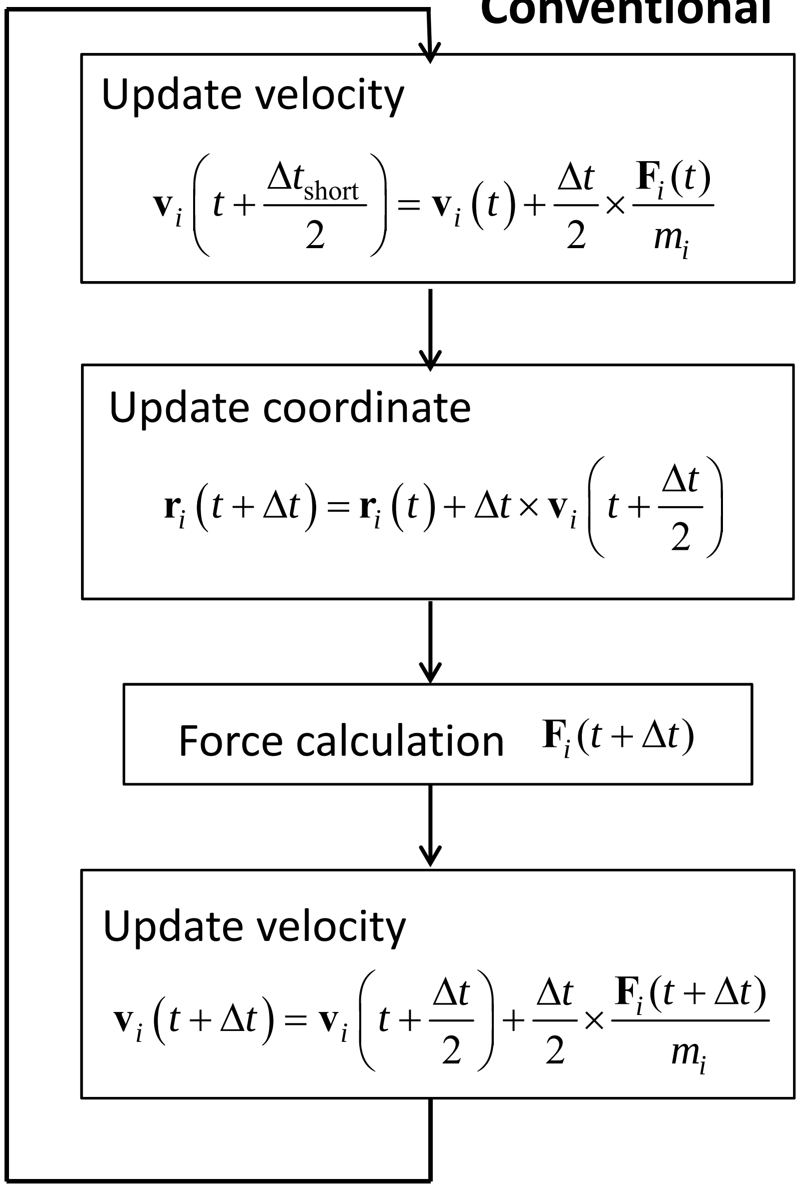
- **Real space nonbonded interaction** : large amount of computation, well scalable by increasing processors (main bottleneck when using small number of processors)

- **Reciprocal space nonbonded interaction** : medium amount of computation, not well scalable by increasing processors (main bottleneck when using large number of processors)

- **Purpose of multiple time step integration** : Not to calculate the communication-intensive reciprocal space nonbonded interactions every step to increase the performance

# Multiple time step integration (2)

## Conventional



Update velocity

$$\mathbf{v}_i\left(t+\frac{\Delta t_{\text{short}}}{2}\right)=\mathbf{v}_i\left(t\right)+\frac{\Delta t}{2}\times\frac{\mathbf{F}_i(t)}{m_i}$$

Update coordinate

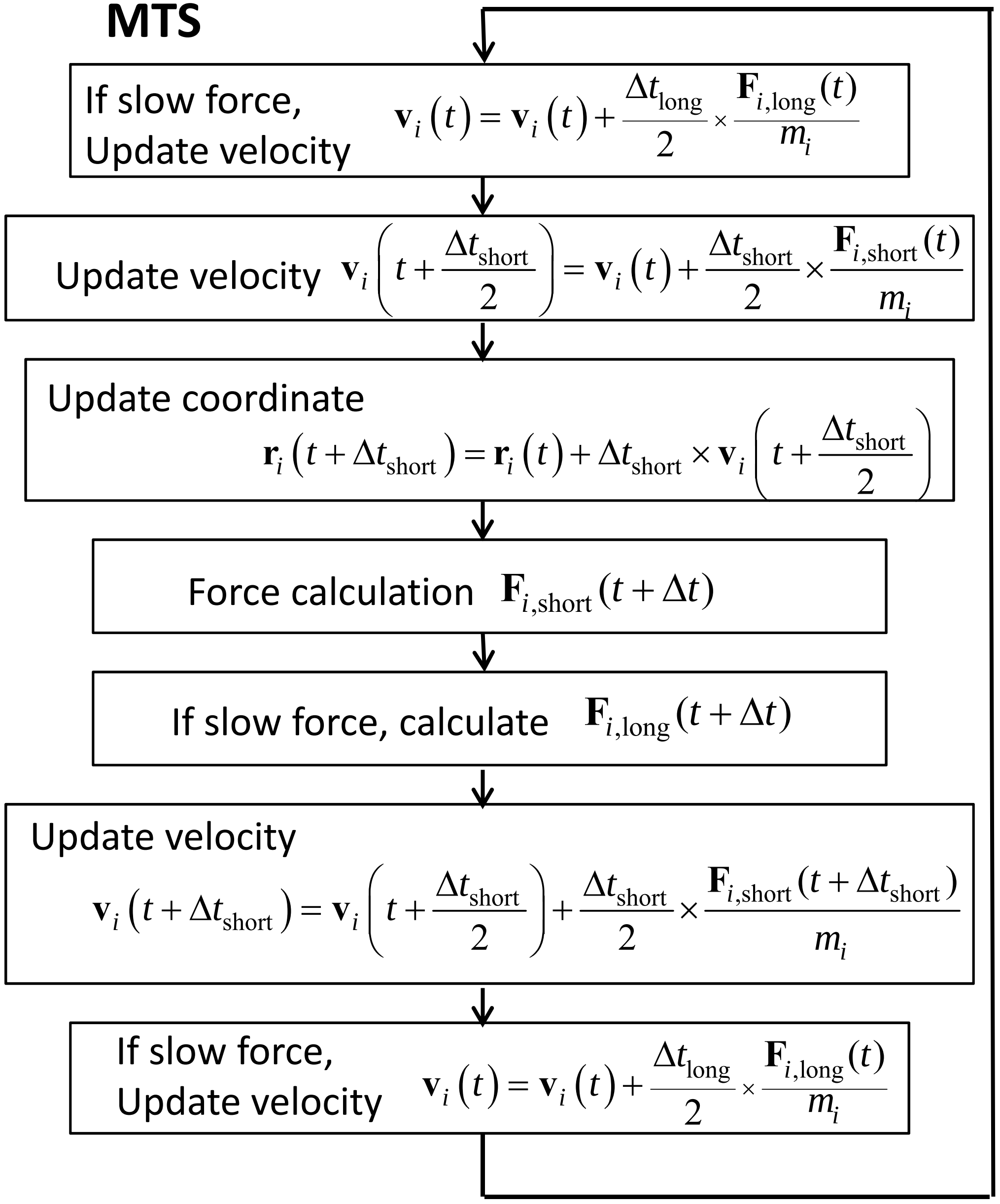
$$\mathbf{r}_i\left(t+\Delta t\right)=\mathbf{r}_i\left(t\right)+\Delta t\times\mathbf{v}_i\left(t+\frac{\Delta t}{2}\right)$$

Force calculation $\mathbf{F}_i(t+\Delta t)$

Update velocity

$$\mathbf{v}_i\left(t+\Delta t\right)=\mathbf{v}_i\left(t+\frac{\Delta t}{2}\right)+\frac{\Delta t}{2}\times\frac{\mathbf{F}_i(t+\Delta t)}{m_i}$$

## MTS



If slow force, Update velocity $\mathbf{v}_i\left(t\right)=\mathbf{v}_i\left(t\right)+\frac{\Delta t_{\text{long}}}{2}\times\frac{\mathbf{F}_{i,\text{long}}(t)}{m_i}$

Update velocity $\mathbf{v}_i\left(t+\frac{\Delta t_{\text{short}}}{2}\right)=\mathbf{v}_i\left(t\right)+\frac{\Delta t_{\text{short}}}{2}\times\frac{\mathbf{F}_{i,\text{short}}(t)}{m_i}$

Update coordinate

$$\mathbf{r}_i\left(t+\Delta t_{\text{short}}\right)=\mathbf{r}_i\left(t\right)+\Delta t_{\text{short}}\times\mathbf{v}_i\left(t+\frac{\Delta t_{\text{short}}}{2}\right)$$

Force calculation $\mathbf{F}_{i,\text{short}}(t+\Delta t)$

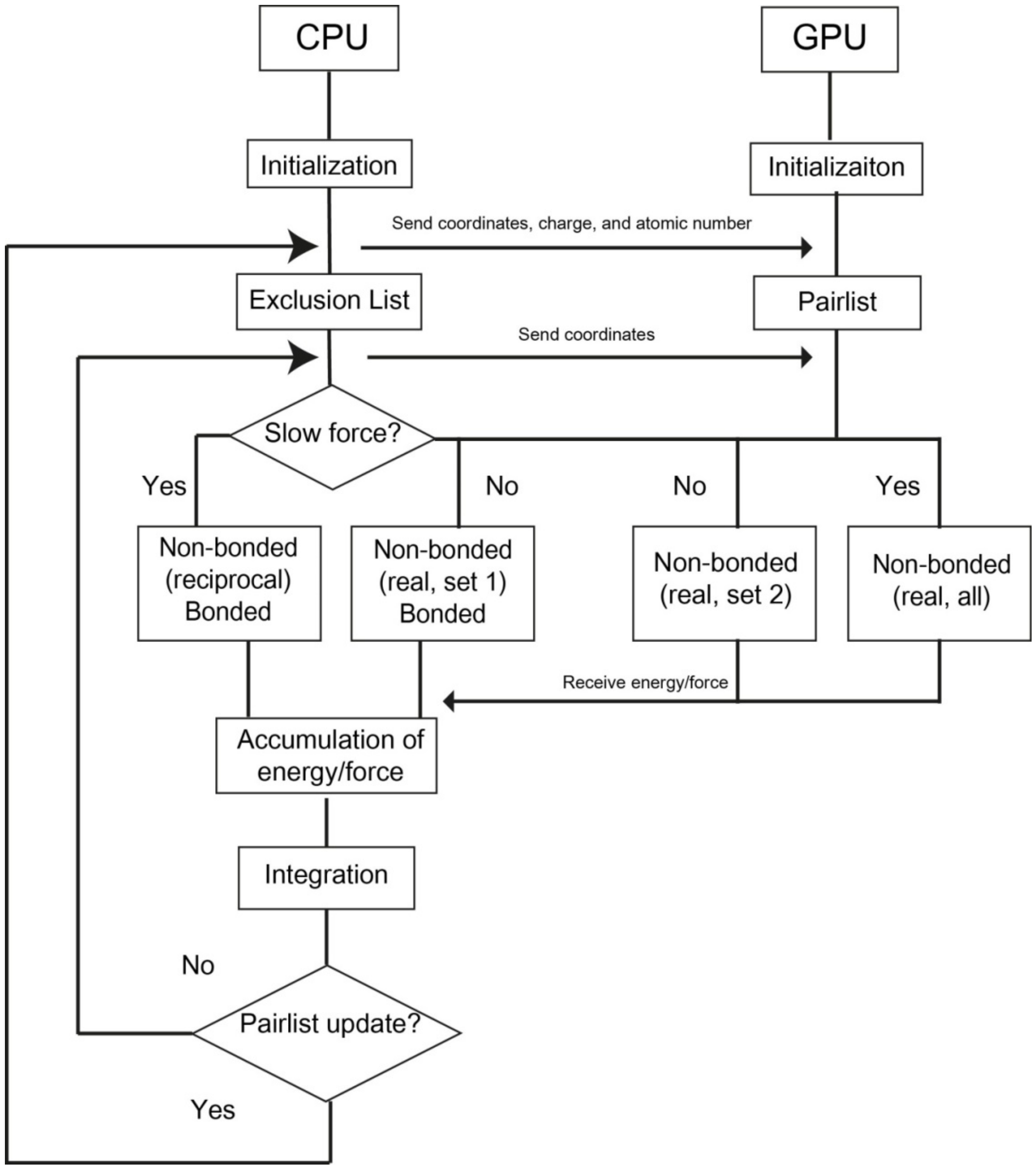
If slow force, calculate $\mathbf{F}_{i,\text{long}}(t+\Delta t)$

Update velocity

$$\mathbf{v}_i\left(t+\Delta t_{\text{short}}\right)=\mathbf{v}_i\left(t+\frac{\Delta t_{\text{short}}}{2}\right)+\frac{\Delta t_{\text{short}}}{2}\times\frac{\mathbf{F}_{i,\text{short}}(t+\Delta t_{\text{short}})}{m_i}$$

If slow force, Update velocity $\mathbf{v}_i\left(t\right)=\mathbf{v}_i\left(t\right)+\frac{\Delta t_{\text{long}}}{2}\times\frac{\mathbf{F}_{i,\text{long}}(t)}{m_i}$

# Multiple time step integration with GPU



1. In the multiple time step integrator, we do not perform reciprocal space interaction every step.

2. If reciprocal space interaction is not necessary, we assign subset of real space interaction on CPU to maximize the performance.

3. Integration is performed on CPU only.

# Performance result with Multiple time step integration

Input : test6.inp

```
[DYNAMICS]
integrator = VVER
```

Input : test7.inp

```
[DYNAMICS]
integrator        = VRES
nsteps            = 2000
elec_long_period  = 2
thermostat_period = 2
barostat_period   = 2
```

**Timing for 2000 MD steps**

| Condition | Test6.inp | Test7.inp |
|---|---|---|
| CPU+GPU (double) | 40.685 | 39.153 |
| CPU+GPU (single) | 26.307 | 22.749 |
| CPU only (double) | 109.52 | 99.520 |
| CPU only (single) | 101.28 | 94.691 |

Speed up due to multiple time step becomes more critical as we increase the number of processors.

# Contents

- 15:30 – 17:00 GENESIS tutorial 2
  - ➢ General comments in GENESIS
  - ➢ How to accelerate MD speed of GENESIS : GPU
  - ➢ How to accelerate MD speed of GENESIS : Multiple time step integration
  - ➢ How to accelerate MD speed of GENESIS : Further suggestions

# Energy output

- If energy output is required, the program calculates both energy and force
- If energy output is not required, the program calculates only force.
- Please do not print energy output frequently.

Input : test6.inp

```
[DYNAMICS]
eneout_period = 100
```

Input : test8.inp

```
[DYNAMICS]
eneout_period = 1
```

**Timing for 2000 MD steps**

| Condition | Test6.inp | Test8.inp |
|---|---|---|
| CPU+GPU (double) | 40.685 | 48.032 |
| CPU+GPU (single) | 26.307 | 29.897 |
| CPU only (double) | 109.52 | 137.89 |
| CPU only (single) | 101.28 | 120.73 |

# Decision of MPI and OpenMP processors (1)

- MPI
  - Distributed memory parallelization
  - Communication is necessary

- OpenMP
  - Shared memory parallelization
  - Communication is not necessary

- MPI/OpenMP hybrid parallelization in GENESIS
  - Domain decomposition by MPI
  - In each domain, force decomposition by OpenMP

# Decision of MPI and OpenMP processors (2)

- Our suggestions
  - For small number of processors, flat MPI is preferred
  - For multiple nodes, OpenMP threads could be helpful
  - More than 16 OpenMP threads is not recommended.

**Timing for 2000 MD steps**

| Condition | MPI=16, OMP=1 | MPI=4, OMP=4 |
|---|---|---|
| CPU+GPU (double) | 43.765 | 50.496 |
| CPU+GPU (single) | 30.447 | 36.406 |
| CPU only (double) | 151.34 | 158.89 |
| CPU only (single) | 145.84 | 146.85 |

# Exercise

1. Compare the performance by running test6.inp (conventional integration) and test7.inp (multiple time step integration)

2. Compare the performance by running test6.inp (energy output every 100 steps) and test8.inp (energy output every step)

3. Compare the performance by running test9.inp with various combinations of MPIs and OpenMPs