Feb. 27, 2018

Simulation Development of Planetary Magnetosphere toward Exascale Computing Era

Keiichiro FUKAZAWA[1], Yuto KATO[2], Yohei MIYAKE[3], and Takeshi NANRI[4]

1. Academic Center for Computing and Media Studies, Kyoto University
2. Graduate School of Science Geophysics, Tohoku University
3. Graduate School of System Informatics, Kobe University
4. Research Institute for Information Technology, Kyushu University

Simulation of Jovian magnetosphere 1

Early global simulation of magnetosphere around 2000

Starting just around the Galileo spacecraft observation

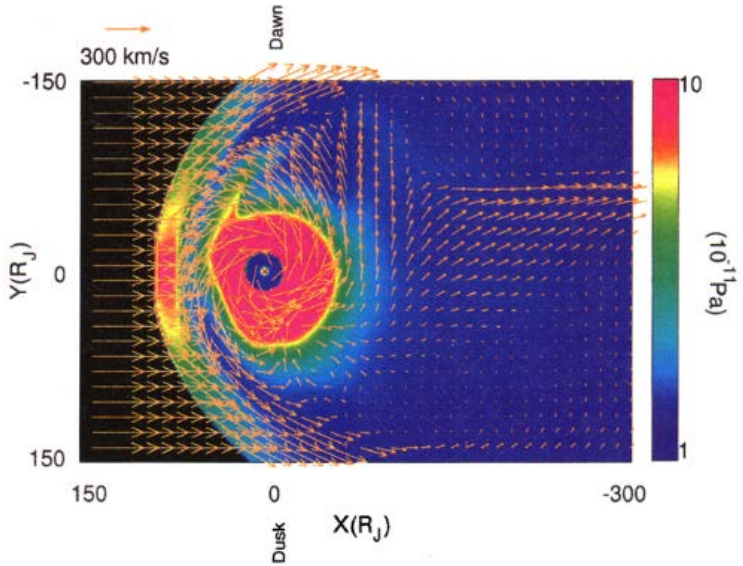


Fig. 1. Plasma pressure and flow vector in the equatorial plane [*Ogino et al.*, 1998]

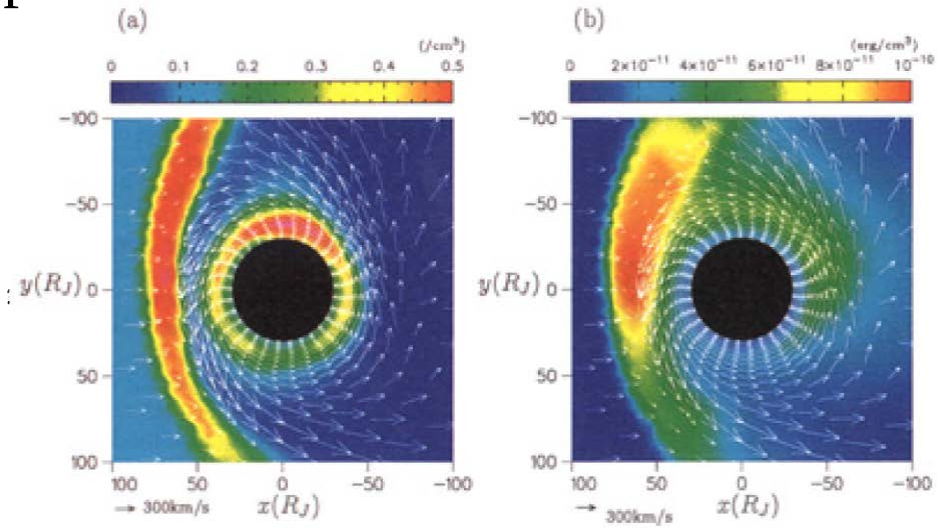


Fig. 2. Plasma density (a) and pressure with flow vector in the equatorial plane [*Miyoshi and Kusano*, 2001]

Simulation size: 300 × 200 × 100=366MB

Performed with the shared memory vector parallel supercomputer

Now we can calculate it with our smartphone!!

Simulation of Jovian magnetosphere 2

Simulation from 2005 to 2010

Periodic plasmoid ejection at 2005

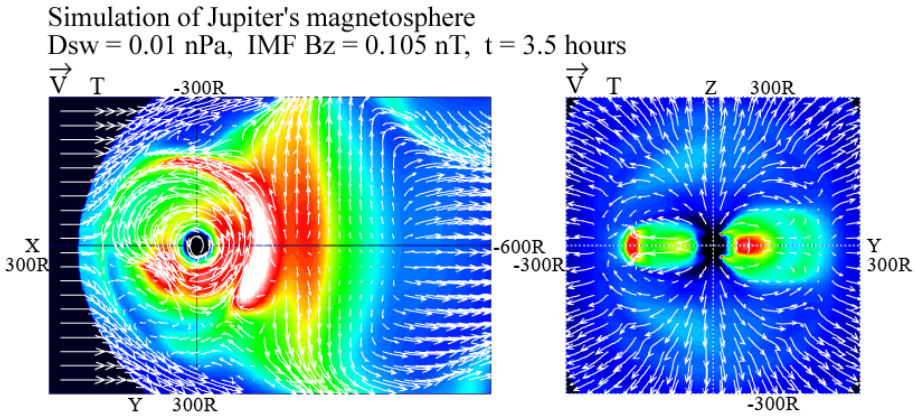


Fig. 3 Periodic plasmoid ejection [*Fukazawa et al., GRL*, 2005]

- Simulation size: 600 × 400 × 200=3GB
- Calculated with the distributed memory scalar supercomputer
- Now we can do using our laptop PC

Periodic plasmoid ejection at 2010

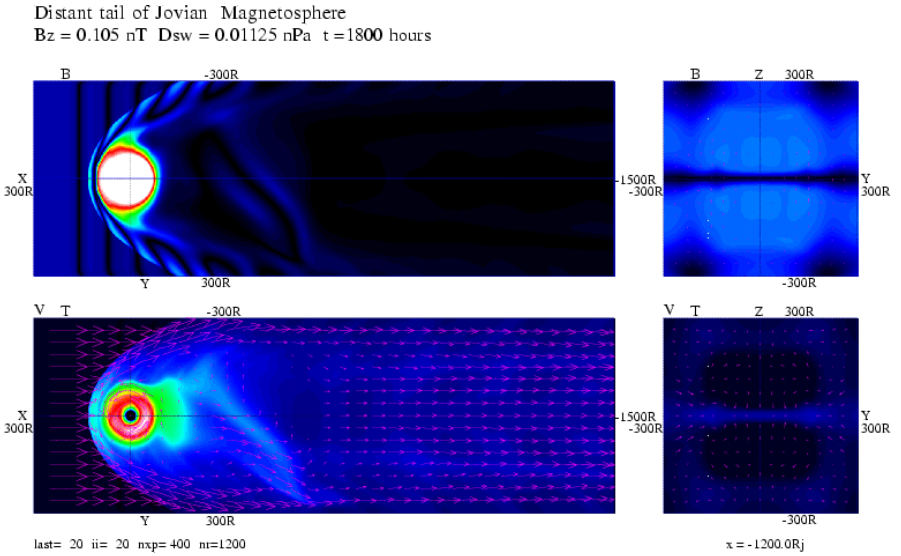


Fig. 4. Periodic plasmoid ejection [*Fukazawa et al., JGR*, 2010]

- Simulation size: 1200 × 400 × 400=12GB
- Using PC cluster type supercomputer
- Now we can perform by a computer server.

Latest simulation of Jovian magnetosphere

Now we can perform the 10,000 times larger simulation of magnetosphere compared to 10 years ago

Recent simulation setting

- Simulation size: 6000 × 4000 × 2000=3TB
- Resolution: 0.15R_J at maximum (we used 1.5R_J in 2000 or so)
- Inner boundary: nearby Io torus 5~7R_J (we have set it 15~20R_J in 2000)
- Using massively parallel supercomputer

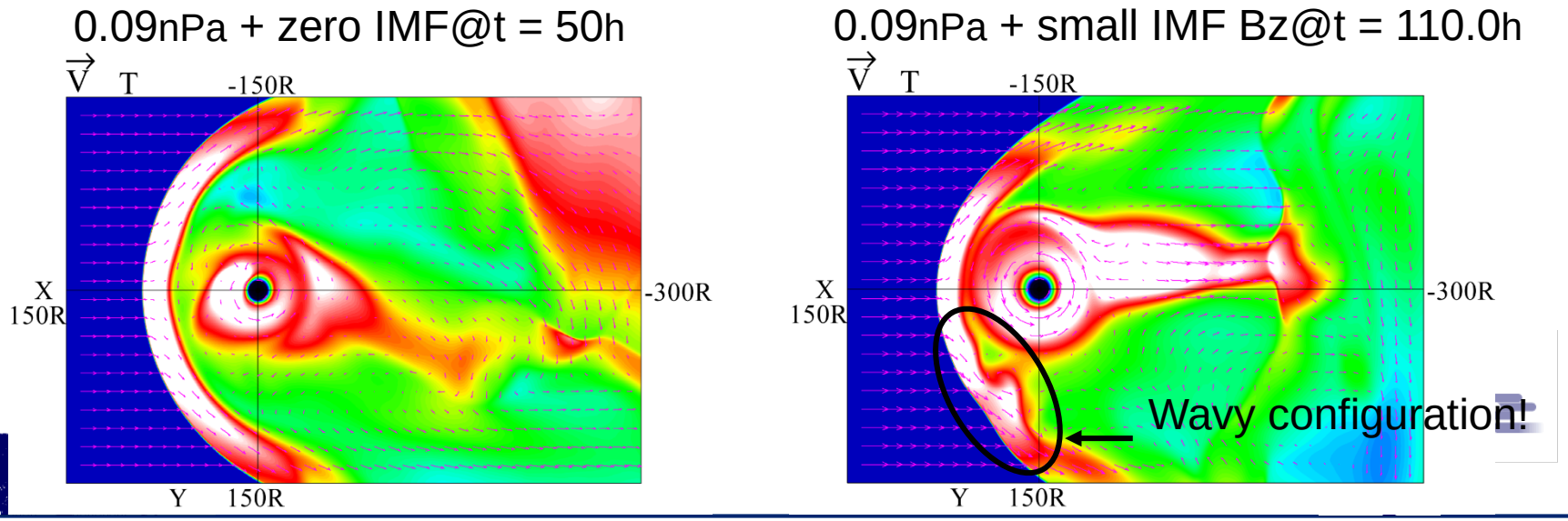


Fig. 5. High resolution simulation of Jovian magnetosphere

京都大学
KYOTO UNIVERSITY

Issues of recent Jovian simulation

There are some technical problems

Calculation time: it takes 5days to proceed the simulation for 2.5hr with 30TFlop.

- 30TFlops=1000 latest Xeon CPUs

Post processing: hard to treat the 3TB simulation data on the server

- In general a PC server only has around 128GB memory (1/24 size)

Data transfer: hard to move the simulation data through the Internet

- It takes 400m/data by 1Gbps throughput network

Storage: hard to acquire the space to store the simulation data

- 300TB (100 data) disk space is required

Issues of recent Jovian simulation

Scalability in exa-scale computing

The exascale computing systems will be developed using over 3 million computer nodes around 2020.

The scalability of our MHD code using 30 thousand nodes of K-Computer decreases by 10% (weak scaling).

If we will not do anything to our code in exa-era, the scalability will decrease by 20%.

→This may be come from the synchronization between massively nodes.

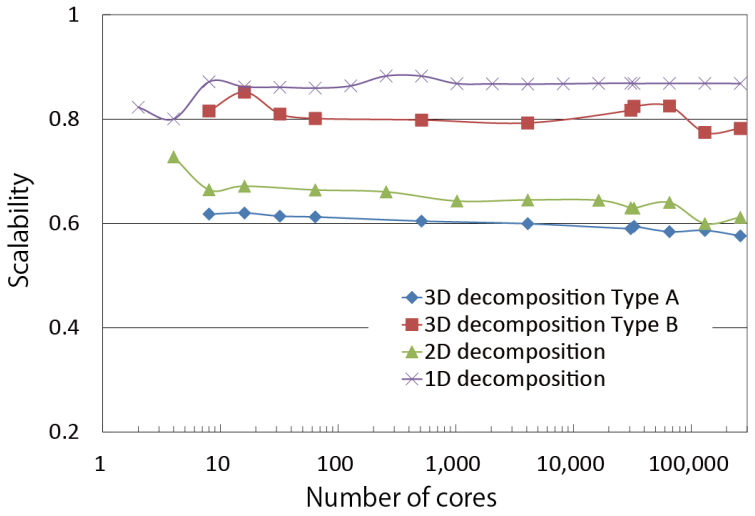


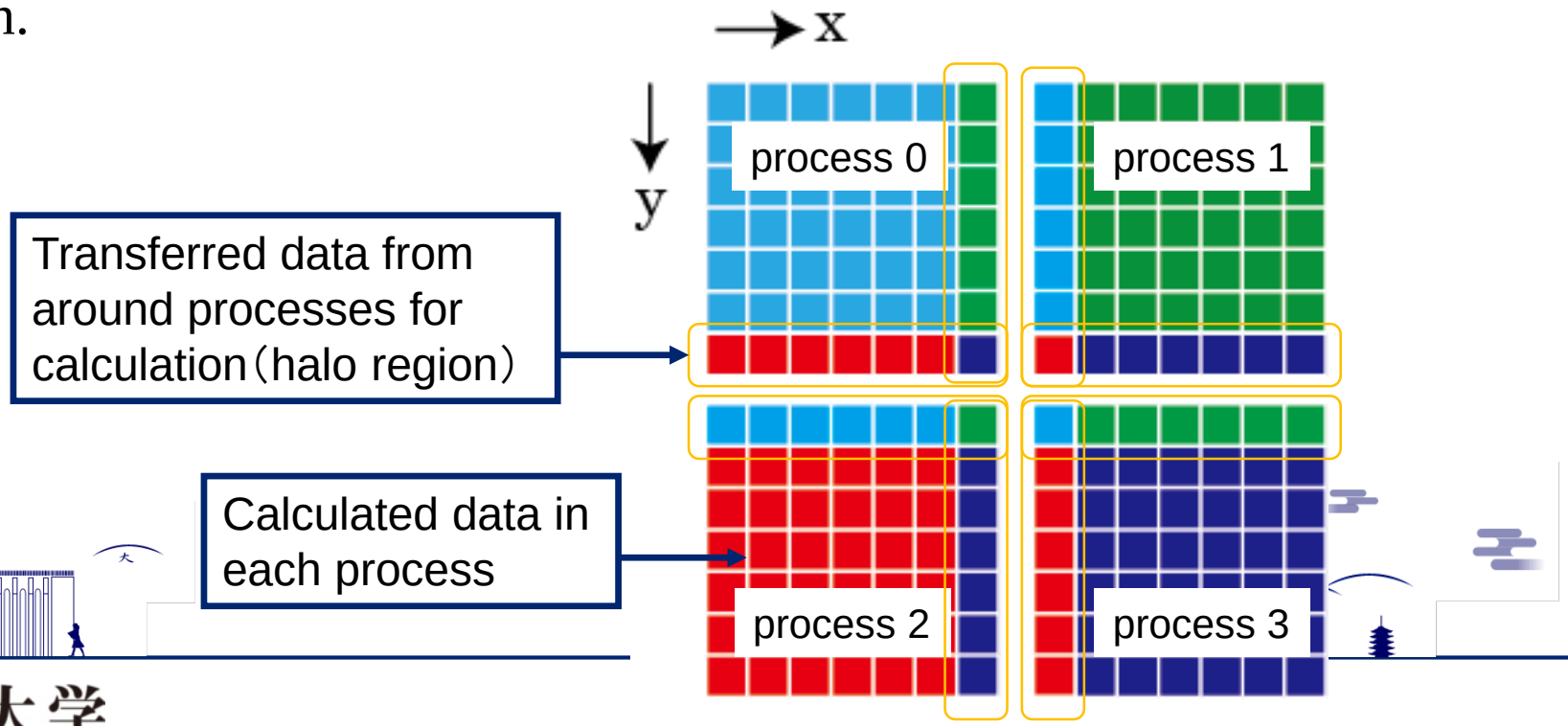
Fig. 6. Scalability of MHD code with K-computer [*Fukazawa et al.* 2013]

Cause of scalability fall

Issue of Halo Communication

In the boundary communication for the domain decomposition, pack/unpack of communication data and the order of communication decrease the parallel scalability.

Using the Halo thread, the time of pack/unpack and communication will be hidden.



Proposed Model 1

Parallel Implementation

Domain decomposition for parallel calculation

- Internode communication is only the halo communication in this case
- mpi_isend/irecv and mpi_wait are used

Halo thread for halo region

- Only the Halo thread treats the calculation and communication in the halo region where has the dependence of communication and calculation
- Then it is not required the synchronization in calculation threads (except for the Halo thread) due to the communication.
- It is thought that the computer system which has the communication core will increase, then the Halo thread will be work more effectively than the only communication thread.

Proposed Model 2

Flowchart of MHD code w/o Halo thread

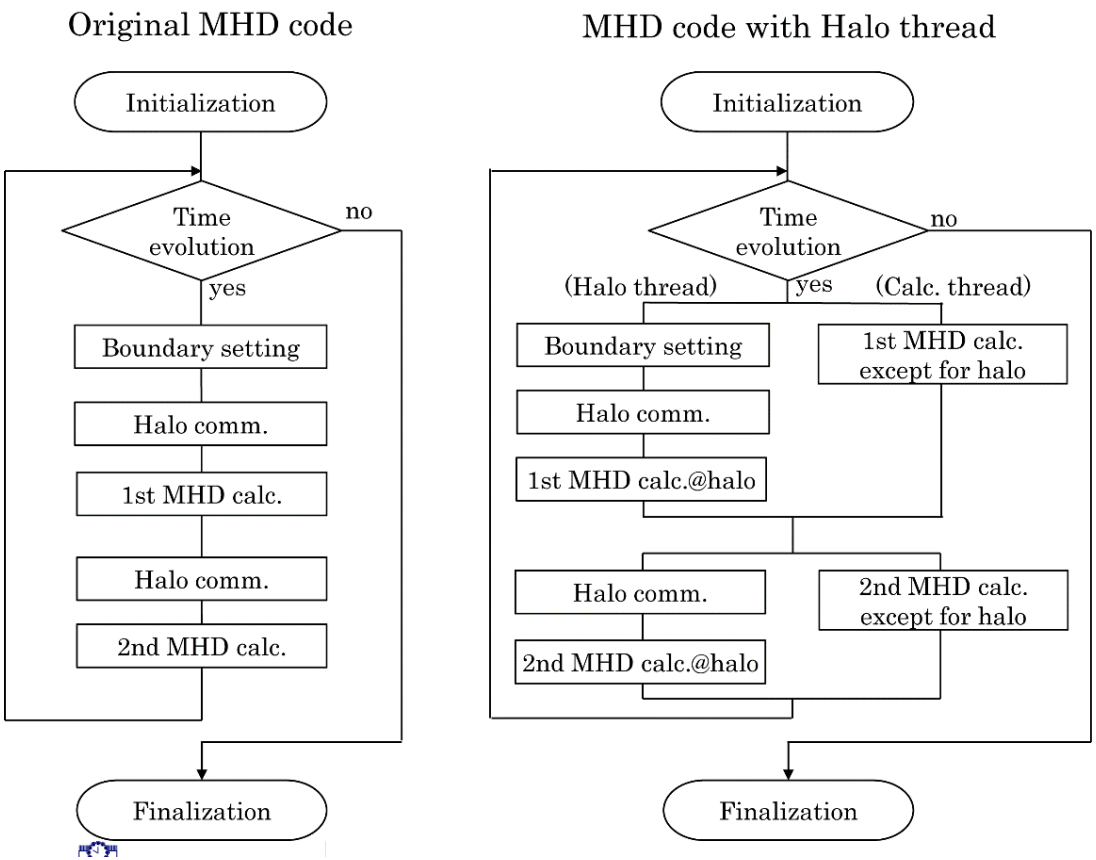


Fig. 7 MHD flowchart (left: original code, right: with the Halo thread)

Introducing the Halo thread, there is no synchronization in calculation threads.

Proposed Model 3

Implementation of Halo Thread

```
  call init_mhd(f) ! Initilization
!
!----Time evoluation---!
  do time = 1, 1000
!
!----Thread setting----!
!$OMP PARALLEL PRIVATE(myid,mylid,ks,ke,ii)
    myid = omp_get_thread_num()
    nthreads = omp_get_num_threads() - 1
    mylid = myid - 1
    kmod = mod(nzz-2, nthreads)
    kdiv = floor(real((nzz-2)/nthreads))
!
    if (kmod > mylid) then
      ks = mylid * (kdiv + 1) + 1
      ke = ks + kdiv
    else if (kmod == mylid) then
      ks = mylid * (kdiv + 1) + 1
      ke = ks + kdiv - 1
    else
      ks = mylid * kdiv + kmod + 1
      ke = ks + kdiv - 1
    end if
!
!----Halo thread----!
    if(myid == 0) then
      call boundary(f)    ! boundary setting
      call halo3d(f)      ! Halo communication
!
      do k = zs, ze
        call mhd_calc(f) ! MHD calc. at Halo
      end do
!
!----Calc thread----!
    else
      do k = ks+1, ke-1
! MHD calc. except for Halo
        call mhd_calc(f)
      end do
    end if
!
!$OMP END PARALLEL
!
.
.
.
  end do
```

Performance Measurements

Strong and weak scaling performance

Strong scaling:800 × 800 × 1600, 200 × 200 × 400

Weak scaling :100 × 100 × 100/process

FX10(SPARC64 Ixfx + Tofu)

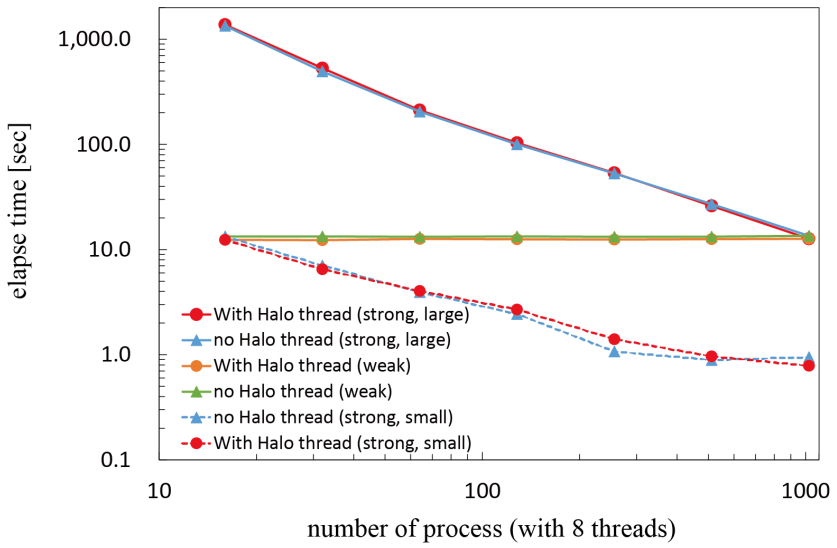


Fig. 8. Performance of strong and weak scaling with/without on FX10 (8 threads).

HA8000(Ivy Xeon + InfiniBand FDR)

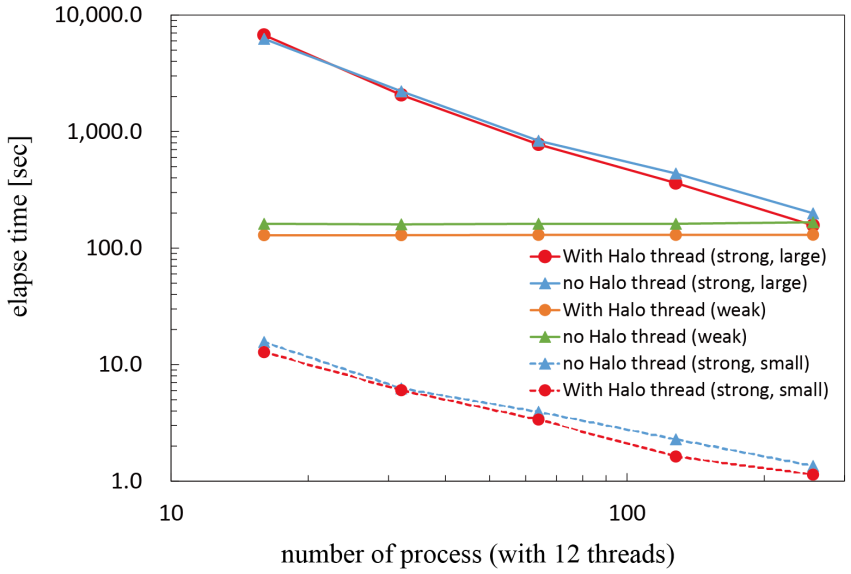
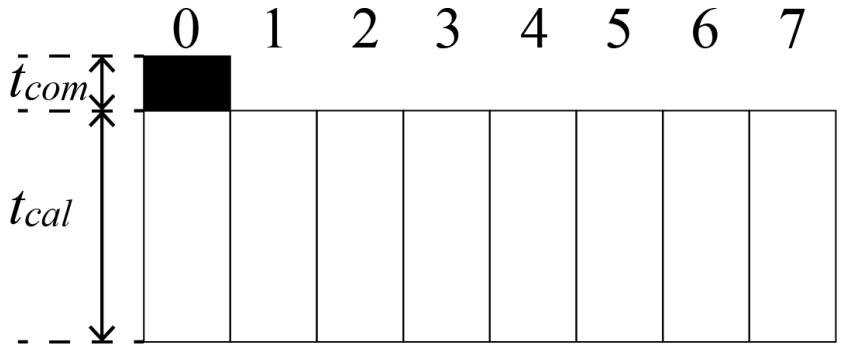


Fig. 9. Performance of strong and weak scaling with/without on the HA8000 (12 threads).

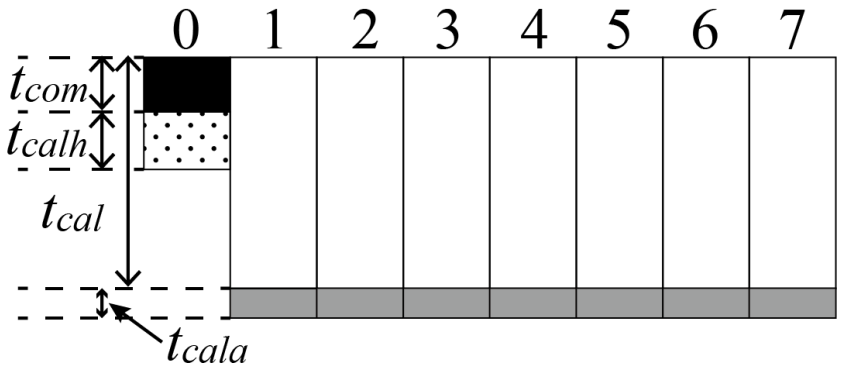
Effective Condition

Condition of good performance using the Halo thread

i) no Halo thread



ii) Halo thread



t_{com}: Halo communication time

t_{cal}: MHD calculation time without Halo thread

t_{calh}: MHD calculation time of halo region

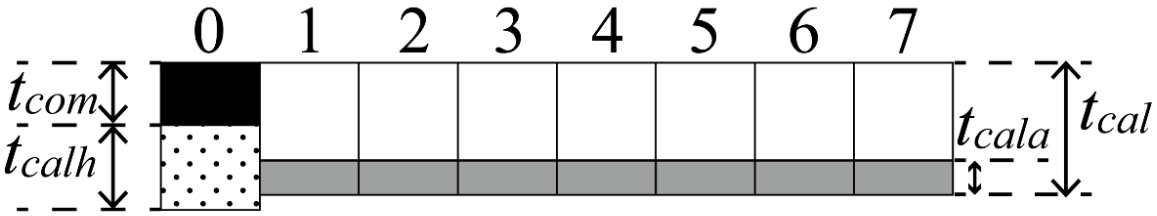
t_{cala}: Additional MHD calculation time due to decrease of calculation thread

When $t_{com} > t_{cala}$, there is an advantage of Halo thread

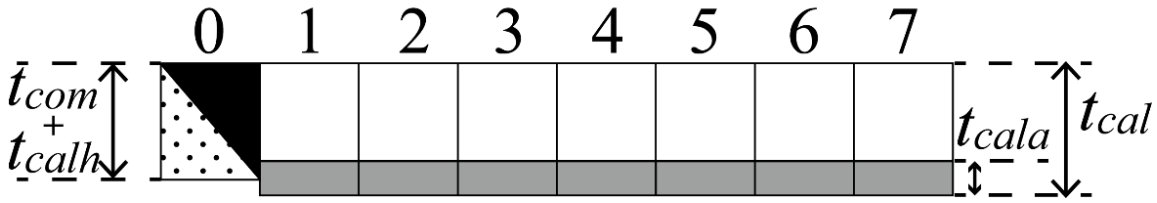
Limitation of Halo Thread Effect

Necessity of overlapping of calculation and communication

i) no overlapping



ii) overlapping



As the strong scaling condition, if $t_{com} + t_{calh} > t_{cala}$ then, the advantage of Halo thread will be “zero”.

Optimization of Halo Thread

Introduction of the Halo functions

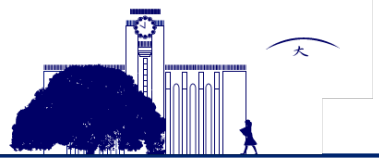
To overlap the halo communication and calculation, Halo functions are developed. In general the destination and data of halo communication are fixed or can be defined easily. Considering these information, Halo functions optimize the pack/unpack operation and communication. Halo functions are independent from Halo thread so you can use them without Halo thread.

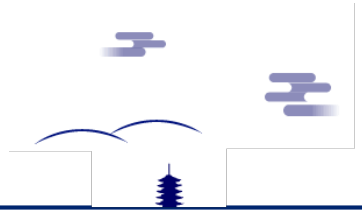
Explanation of Halo function

- **`Halo_init`**: initializes a set of information for halo communication according to the specified parameters such as the address and the dimensions of the target array, and the shape of the process grid with which the array is distributed among processes. The initialized information consists of the logical coordinate of the process in the process grid and the addresses used for the halo communication.

Explanation of Halo function

- **`Halo_isend`**: a non-blocking function that starts sending the halo region to the target process according to the specified direction.
- **`Halo_irecv`**: a non-blocking function that starts receiving the halo region from the source process according to the specified direction.
- **`Halo_test`**: checks if the specified “Halo_isend / irecv” has been completed.
- **`Halo_wait`**: waits for the completion of the specified “Halo_isend / irecv”.
- **`Halo_finalize`**: releases the memory initialized in “Halo_init”.





Implementation of Halo Functions

No overlapping

```
 call halo_init(f) ! Halo Initilization
!
!----Time evoluation---!
  do time = 1, 1000
!
!----Thread setting----!
!$OMP PARALLEL PRIVATE(myid,mylid,ks,ke,ii)
.
.
.
!----Halo thread----!
    if(myid == 0) then
      call boundary(f)   ! boundary setting
      do l = 1, 26
        call halo_irecv(f) ! Halo recieve
        call halo_isend(f) ! Halo send
        call halo_wait ! for receive
        call halo_wait ! for send
!
        do k = zs(l), ze(l)
          call mhd_calc(f) ! MHD calc. at Halo
        end do
!
      end do
!----Calc thread----!
    else
      do k = ks+1, ke-1
        call mhd_calc(f) ! MHD calc.
      end do
    end if
!
!$OMP END PARALLEL
.
.
.
  end do
```

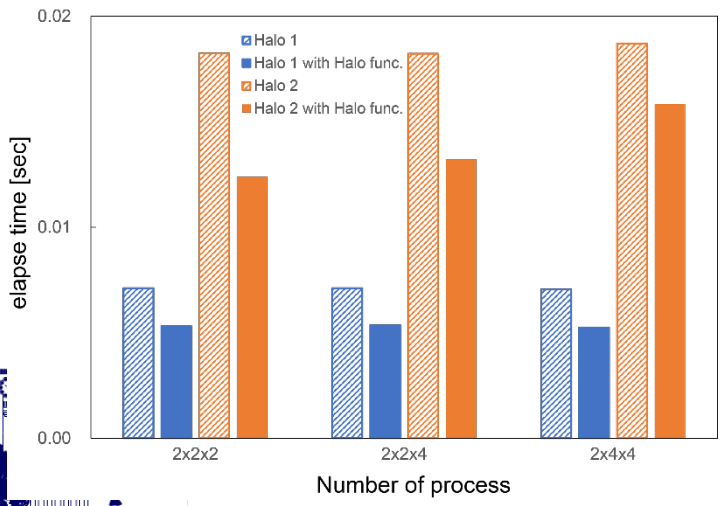
overlapping

```
 call halo_init(f) ! Halo Initilization
!
!----Time evoluation---!
  do time = 1, 1000
!
!----Thread setting----!
!$OMP PARALLEL PRIVATE(myid,mylid,ks,ke,ii)
.
.
.
!----Halo thread----!
    if(myid == 0) then
      call boundary(f)   ! boundary setting
      do l = 1, 26
        call halo_irecv(f) ! Halo recieve
        call halo_isend(f) ! Halo send
      end do
!
      do l = 1, 26
        call halo_wait ! for receive
        do k = zs(l), ze(l)
          call mhd_calc(f) ! MHD calc. at Halo
        end do
      end do
!
      do l = 1, 26
        call halo_wait ! for send
      end do
!----Calc thread----!
    else
      do k = ks+1, ke-1
        call mhd_calc(f) ! MHD calc.
      end do
    end if
!
!$OMP END PARALLEL
.
.
.
  end do
```

Performance using Halo functions

- To tune the halo exchange and overlap the communication with calculation in the parallel stencil computation, we have developed “Halo functions” and introduced them to MHD simulation code.
- As the results we obtained almost double performance enhancement at maximum.
- In addition, we can perform the overlapping of communication with calculation easily.

FX100 (SPARC64 XIfx + Tofu2)



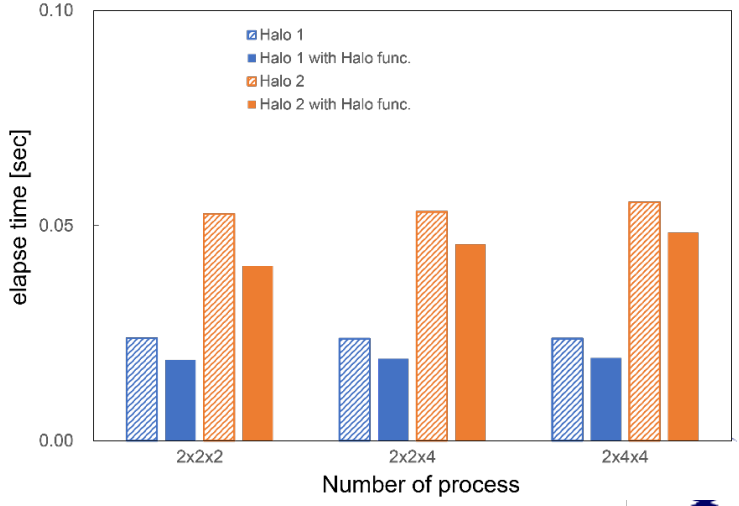


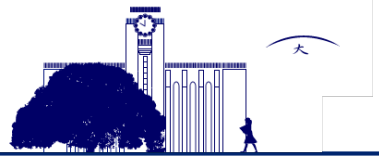
Fig. 10. Halo communication time w/o Halo functions on Fujitsu FX100

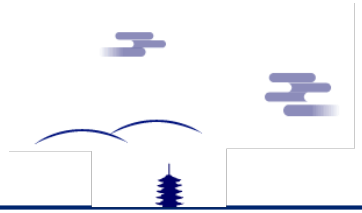
Overlapping Effect

- ✓ We achieved 41 % (no overlapping) and 109 % (overlapping) @FX100 performance increase in total simulation time compared to the regular halo communication.
- ✓ Using Halo functions, we have obtained the 14 ~ 41% performance gain in the total elapse time. Additionally, introducing the overlapping, we can achieve the 66 ~ 109 % performance gain.

Table 1. Performance enhancement ratio of MHD code with Halo functions to the regular halo exchange

@FX100	2×2×2 100^3	2×2×4 100^3	2×4×4 100^3	2×2×2 200^3	2×2×4 200^3	2×4×4 200^3
No overlapping	1.1494	1.1533	1.1546	1.4179	1.4115	1.4105
overlapping	1.7181	1.6700	1.6690	2.0959	2.0829	2.0743



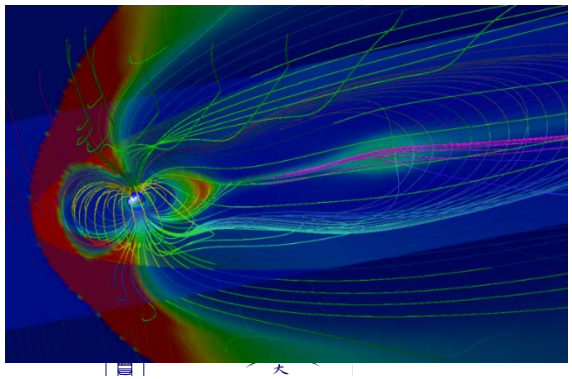


Other Simulation for Exa computing

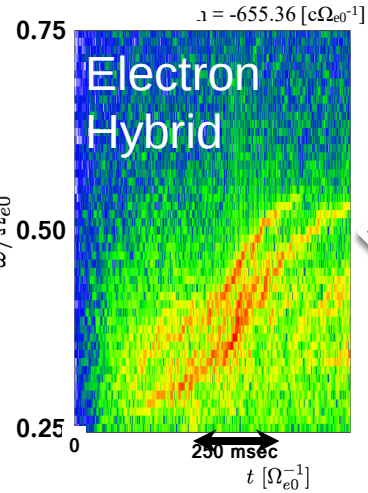
Macro-Meso-MIcro coupling simulation

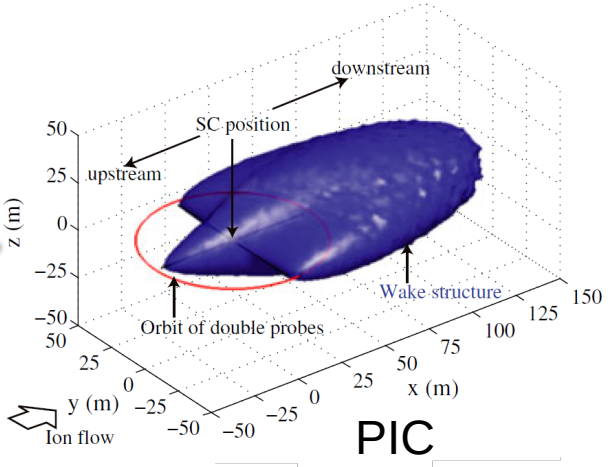
1. Global simulation of Terrestrial magnetosphere
2. Electron Hybrid simulation of high energy particle in Terrestrial magnetosphere
3. PIC simulation of electro-magnetic environment around a satellite

We plan to couple these (1, 2, 3) simulation in the exaflops computing.

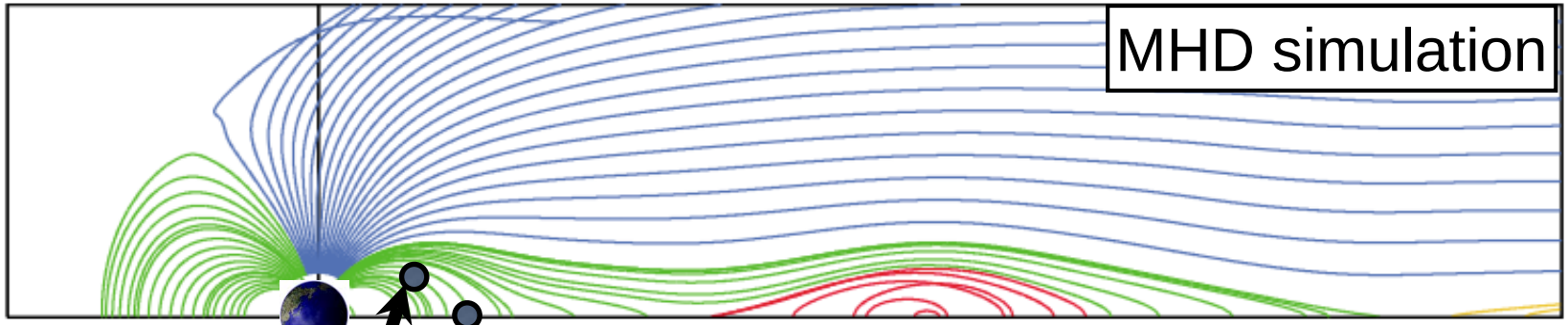


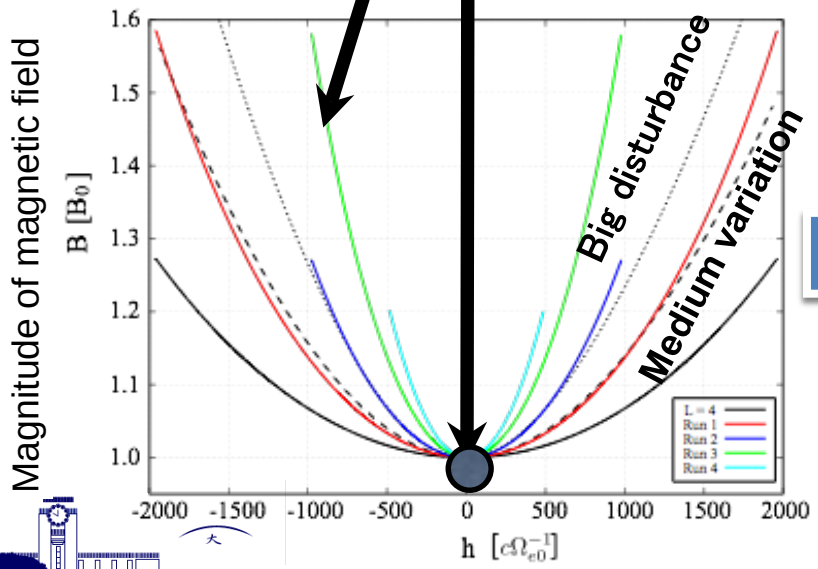
Global MHD

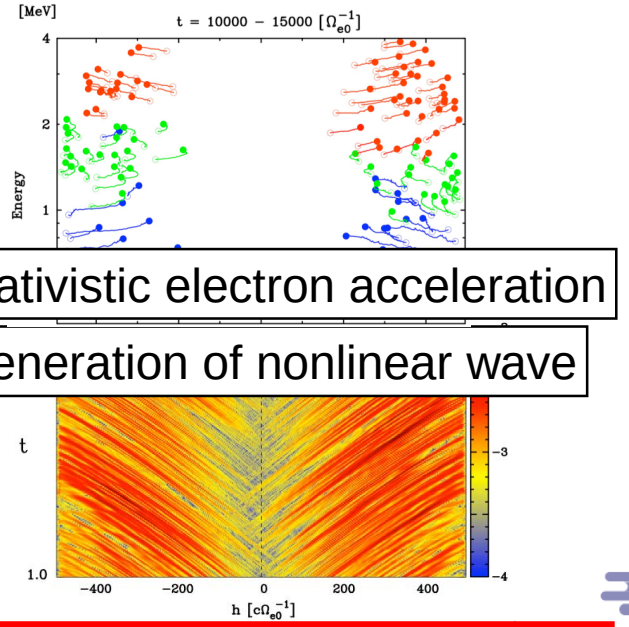




Micro-Macro Coupling Simulation of Terrestrial Magnetosphere







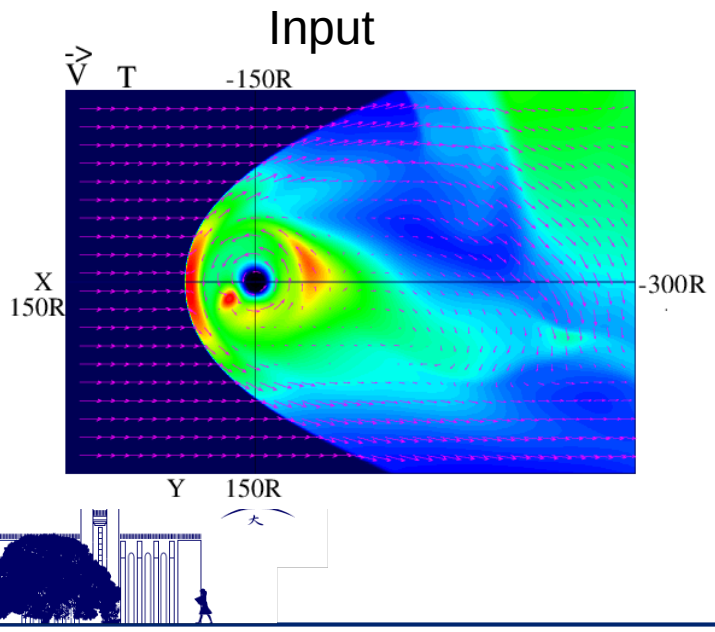
京都大学
KYOTO UNIVERSITY

Status of coupling simulation

Coupling the electron hybrid simulation

Input the magnetic field data of MHD simulation to the electron hybrid simulation by Dr. Kato[*Kato and Omura*, 2013]

- Provide the background magnetic field which is the condition of chorus emission
- Recently we can set the inner boundary nearby Io so that we can couple our simulation.



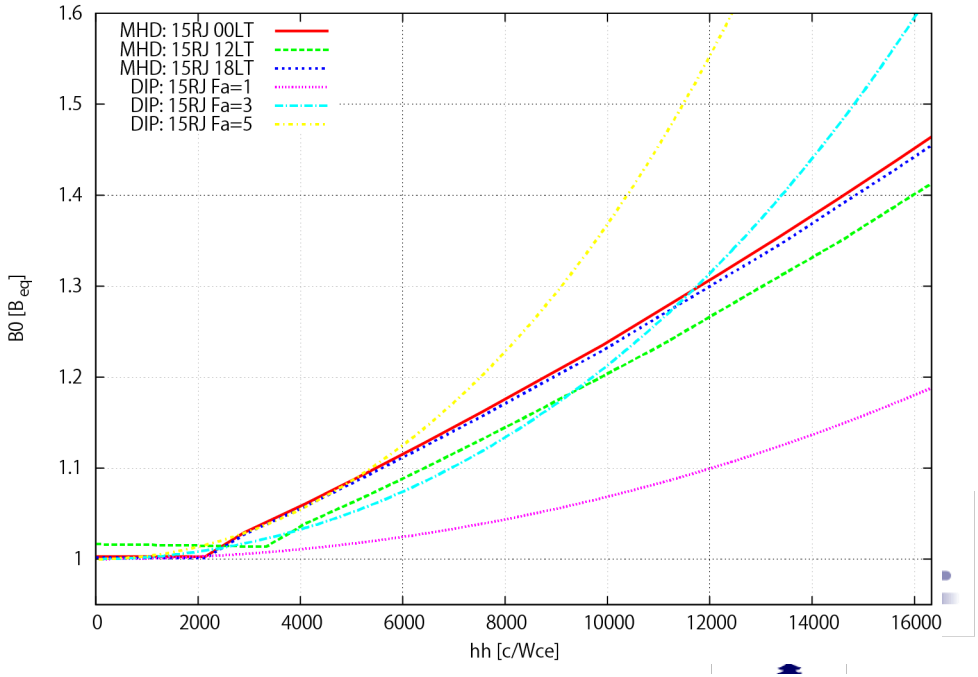


Fig. 11. Comparison of MHD simulation data and model

Summary

- ✓ In the exa computing era, there is an issue of parallel computing scalability.
- ✓ To overcome the issue, we have introduced the Halo thread to our MHD simulation code and examined the performances.
- ✓ Using the Halo thread we obtained the good performance in both weak and strong scaling.
- ✓ To avoid the limitation of the Halo thread effect, we have developed the Halo functions which can communicate between process effectively and be easy to overlap the calculation and communication.
- ✓ The Halo functions achieves good performance compared to the usual MPI communication.
- ✓ In addition, we will be able to perform the micro-macro coupling simulation in the exa era, so we have started to develop the coupling model.
- ✓ Now we have connected the MHD simulation and electron hybrid simulation weakly.