Massive Parallel Computation for an Efficient Whole Core Transport Calculation

Jin Young Cho* and Seungsu Yuk
Korea Atomic Energy Research Institute, 111, Daedeok-daero 989beon-gil, Yuseong-gu, Daejeon, 34057, Korea
**Corresponding author: yhchoi@kaeri.re.kr*

1. Introduction

The radial MOC and the axial Pn coupled calculation [1,2,3] is widely used to obtain the sub-pin level whole core transport solution which requires too much consuming time and tremendous memory. To resolve these problems, the domain decomposition scheme is commonly applied to the whole core transport code. DeCART applies the MPI parallel computation to the axial domain to reduce the memory requirement to the number of axial planes, and the OpenMP to the radial domain to increase the computational speed. MPACT applies the MPI parallel computation to the axial and the radial domain, and shows good computational speed.

In this paper, the massive parallel computation by introducing the axial and the radial domain decomposition scheme is applied to the whole core transport calculation. By introducing the massive parallel computation, the calculation can be parallelized up to the one node calculation per CPU. This paper analyze the computational performance of the massive parallel computation.

2. Massive Parallelization

2.1 Domain Decomposition and Processor Grouping

To achieve the massive parallel computation, the plane-wise and the assembly-wise domain decomposition scheme is introduced. By this scheme, the least domain assigned to one CPU can be reduced to one assembly node.

Due to the introduction of the assembly-wise domain decomposition, the main change is that the ray tracing scheme based on the core ray is modified to the assembly ray basis. In the core ray base ray tracing, the incoming angular flux at the core boundary go through the core domain by the linked modular ray and finishes at the other core boundary. However, in the assembly base ray tracing, the incoming angular flux at the assembly boundary should finish at the other assembly boundary and transfer the outgoing angular flux to the adjacent assembly. Therefore, the assembly base ray tracing requires the incoming angular flux information by the communication with the neighboring assemblies, and transfers the outgoing angular flux data to the neighboring assemblies.

For an efficient control of processors involving in whole core transport calculation, the processor grouping is performed in two ways. The first grouping is for communication between the radial neighboring nodes which is frequently required in the radial MOC and the whole CMFD calculation. The second grouping is for communication between the axial neighboring nodes which is required in the axial Pn and the whole CMFD calculation. The new processor groups are named as MPI_PL_WORLD and MPI_ASY_WORLD. In the new processor groups, the data gathering and the reduction is performed first through MPI_ASY_WORLD by assembly Master, and next through MPI_PL_WORLD consisting of assembly Masters by whole Master.

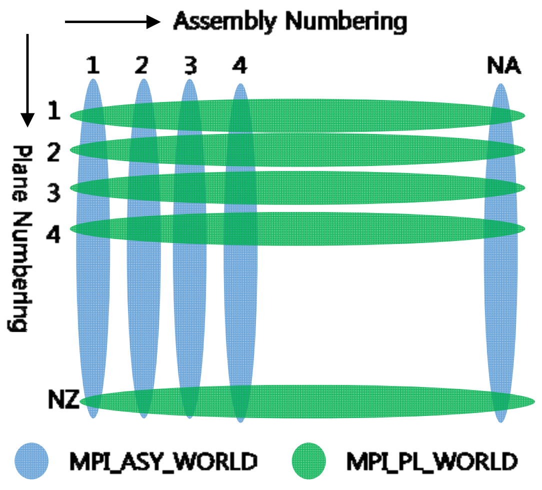

Fig. 1. Communication Domain Splitting and Processor Grouping

2.2 Communication in the MPI_PL_WORLD

The communication between assemblies occurs when performing the ray tracing for MOC equation and performing the node sweeping for CMFD equation. The MOC equation requires the incoming angular flux solution of the neighboring node. Therefore, the outgoing angular fluxes at assembly boundaries are stored every iteration, and then transferred to the neighboring assembly. The CMFD equation requires the scalar flux of the neighboring nodes. The data communication is performed at each inner iteration.

The data communication with the neighboring nodes is performed through the MPI_PL_WORLD. Figure 2 shows the communication order in the MPI_PL_WORLD. For Cartesian geometry, maximum 4 communications with the 4 neighboring nodes are required to obtain the neighboring node data. If the sum of the row and column assembly index is even number, the processor determines the communication order clockwise from the bottom surface, and receives the data first and sends next. Otherwise, the processor determines the communication order clockwise from the top surface, and sends the data first and receives next.

The data communication occurs simultaneously according to the communication order.

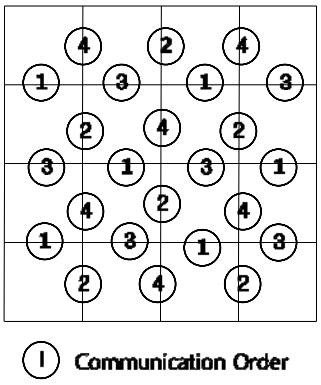

Fig. 2. Communication Order in MPI_PL_WORLD

2.3 Communication in the MPI_ASY_WORLD

The communication between planes occurs when performing the node sweeping for the axial Pn and for the CMFD equations. The Pn equation requires the flux moment solutions of the neighboring planes, and the flux moment solutions of a plane are transferred to the neighboring planes every inner iteration. The CMFD equation requires the scalar flux of the neighboring planes. The data communication is performed after one inner iteration.

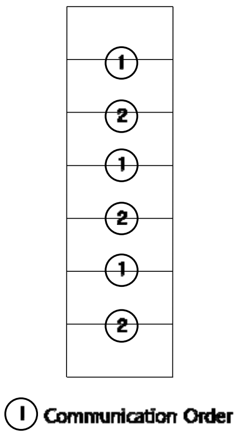

Fig. 3. Communication Order in MPI_ASY_WORLD

The data communication with the neighboring planes is performed through the MPI_ASY_WORLD. Figure 3 shows the communication order in the MPI_ASY_WORLD. In the axial direction, maximum 2 communications with the 2 neighboring planes are required to obtain the neighboring plane data. If the plane index is odd number, the processor communicates with the lower plane first and next with the upper plane, and receives the data first and sends next. Otherwise, the processor communicates with the upper plane first and next with the lower plane, and receives the data first and sends next.

The data communication occurs simultaneously according to the communication order

3. Performance Examination

The performance of the massive parallel computation is examined first for the C5G7 2-D and 3-D problem, and next for SMART quarter core test problem. In the evaluation, the massive parallel computation is also compared with the OpenMP based radial parallel computation. In the examination, 0.02 cm for ray spacing, 32 azimuthal angles, and 2 polar angles are used.

3.1 C5G7 2-D Problems

One assembly node computation per CPU is a target goal of the massive parallel computation scheme. Therefore, the target computing time is also the computing time for one assembly. The computational efficiency of the massive parallel computation is discussed in this aspect.

Table Ⅰ: Computational Efficiency for C5G7-2D Benchmark Problem

	Size	Total	CMFD	MOC	Comm*
Time, sec	FA	16.53	0.11	16.16	
	OC	17.97	0.21	17.37	0.21
	QC.	17.91	0.23	17.35	0.21
	FC	18.14	0.37	17.25	0.38
η**	OC	0.92	0.52	0.93	
	QC	0.92	0.48	0.93	
	FC	0.91	0.30	0.94	

* Communication
** Efficiency = T_{SA}/T, T_{SA}=Time for FA
OC: Octant Core, QC: Quarter Core
FC: Full Core

Table I shows the computation efficiency of the massive parallel computation. The C5G7 problem consists of total 36 assemblies with 16 fuel and 20 reflector assemblies, respectively. Most of the computing time of more than 95 % is charged for MOC calculation. The parallel performance of MOC calculation shows good performance of about 0.93. However, the CMFD calculation shows less performance ranging from 0.52 to 0.30. The parallel performance is associated with the computing load and the number of communication. The CMFD calculation has lower computing load than the MOC calculation, but requires more frequent communication with the neighboring node. Also the CMFD module requires more number of iterations according to the problem size. Those are the reasons for the degradation in the parallel efficiency of CMFD module to the MOC module. The total parallel efficiencies range from 0.92 to 0.91.

3.2 C5G7 3-D Problems

In the 3-D problem, it's not easy to obtain the reference computing time for the single node where the axial Pn calculation is omitted. Therefore, the reference computing time is obtained instead for the single assembly which consists of 4 axial nodes by performing the parallel computation with 4 CPUs, the computational efficiency of the massive parallel computation is discussed by comparing with the single assembly calculation.

Table II shows the computation efficiency of the massive parallel computation. In the table, the CMFD time includes the axial Pn computing time. As for the 2-D problem, most of the computing time of more than 90 % is charged for MOC calculation, which is less than 2-D case due to the burden for the axial Pn calculation. The parallel performance of MOC calculation shows good performance ranging from 1.00 to 0.97. However, the CMFD calculation shows less performance ranging from 0.62 to 0.45. The axial Pn calculation shows the parallel performance ranging from 0.65 to 0.50 which are the better performance than the CMFD calculation.The total parallel efficiencies range from 0.98 to 0.86.

Table Ⅱ: Computational Efficiency for C5G7-3D Benchmark Problem

	Size	Total	CMFD	Pn	MOC	Comm*
Time, sec	FA	19.37	1.63	1.37	17.46	0.08
	OC	19.72	2.61	2.12	16.70	1.42
	QC.	20.50	2.63	2.09	17.50	1.55
	FC	22.42	3.61	2.72	17.96	2.78
η**	OC	0.98	0.62	0.65	1.05	
	QC	0.94	0.62	0.66	1.00	
	FC	0.86	0.45	0.50	0.97	

* Communication
** Efficiency = T_{SA}/T, T_{SA}=Time for FA
OC: Octant Core, QC: Quarter Core
FC: Full Core

Table III shows the computing time breakup when using OpenMP according to the number of threads with 4 axial decomposition using MPI. For octant core problem, the total number of CPUs when using the 6 threads is same CPUs with the massive parallelization scheme in Table II, and shows similar computational speed. For quarter core problem, the total number of CPUs when using the 9 threads is same CPUs with the massive parallelization scheme in Table II, and shows worse computational speed. The 12 threads is the limit number of threads in this computing environment. Therefore, for full core problem, the 12 threads is used and shows not good performance. By comparing Table III with Table II, it can be concluded that the MPI based massive parallelization shows better performance than using OpenMP in the computing time and in the utilization of computing environment.

Table Ⅲ: OpenMP Performance with 4 Domain Decomposition for C5G7-3D Benchmark Problem, sec

Size	Thread	Total	CMFD	Pn	MOC	Comm*
OC	12	13.94	2.62	1.67	9.51	0.18
	6	21.18	3.75	2.67	15.58	0.20
QC	12	30.23	6.08	3.90	18.74	0.49
	9	35.60	6.49	4.30	23.79	0.45
FC	12	133.29	26.08	16.62	61.08	1.53

* Communication
OC: Octant Core, QC: Quarter Core
FC: Full Core

3.3 Realistic Quarter Core 3-D Test Problems

To examine the applicability of the massive parallelization, the realistic SMART quarter 3-D core problem is solved. The original problem which consists of 24 axial planes with 20 fuel planes and 4 reflector planes is modified to 14 plane problems with 10 fuel planes. This problem requires total 238 CPUs which is acceptable in this LINUX environment for the massive parallel computation.

Table IV shows the computation efficiency of the massive parallel computation. As for the C5G7 problem, most of the computing time of more than 95 % is charged for MOC calculation. The parallel performance of MOC calculation shows good performance of about 0.90. However, the CMFD calculation shows poor performance of about 0.32. The total parallel efficiency is about 0.80.

Table Ⅳ: Computational Efficiency for Realistic 3-D Core Problem

	Size	Total	CMFD	Pn	MOC +SG	Comm*
Time, sec	FA	337.8	12.1	8.8	321.0	1.0
	OC	420.3	37.7	25.6	356.7	13.4
η**	OC	0.80	0.32	0.34	0.90	

* Communication
** Efficiency = T_{SA}/T, T_{SA}=Time for FA
SG: Subgroup Calculation, OC: Octant Core

4. Conclusion

In this paper, the massive parallel computation is introduced to whole core transport calculation, the performance is examined for C5G7 2-D and 3-D problems and for the realistic 3-D core problem. The goal of the massive parallelization in this paper is assign one 3-D node per CPU. The examination shows that the parallel efficiencies are 0.89 and 0.77 for 2-D and 3-D C5G7 problems, and 0.80 for the realistic 3-D core problem. These efficiencies are very good considering the number of CPUs attending the computations. Therefore, it can be concluded that the massive parallel computation works good for the whole core transport calculation. In the parallel efficiency, the CMFD and axial Pn calculation showed poor efficiency, and deteriorates the total efficiency. Therefore, more efficient parallel scheme needs to be developed in the future.

ACKNOWLEDGMENT

This study was supported by the National Research foundation of Korea (NRF) grant funded by the Korea Government (MSIT). (No. 2017M2A8A1092448).

REFERENCES

[1] J. Y. Cho et al., “Axial SPN and Radial MOC Coupled Whole Core Transport Calculation,” Journal of NUCLEAR SCIENCE and TECHNOLOGY, 44, 9, 1156 (2017).

[2] Y. S. Jung et al., “ Practical Numerical Reactor Employing Direct Whole Core Neutron Transport and Subchannel Thermal/Hydraulic Solvers,” Ann. Nucl. Energy, 62, 357 (2013).

[3] MPACT TEAM, MPACT Theory Manual, Tech. rep., Oak Ridge National Laboratory and the University of Michigan (2015).