Transactions of the Korean Nuclear Society Spring Meeting
Jeju, Korea, May 29-30, 2014

Parallelization of Subchannel Analysis Code MATRA

Seong-Jin Kim*, Dae-Hyun Hwang, Hyouk Kwon
Korea Atomic Energy Research Institute, Daejeon, Korea
*Corresponding author: sjkim2@kaeri.re.kr

1. Introduction

A subchannel analysis code MATRA [1] is under development at KAERI for the evaluation of thermal margin in various types of reactor cores as well as for the multi-physics calculations coupled with neutronics and/or fuel performance codes. A stand-alone calculation of MATRA code used up pertinent computing time for the thermal margin calculations while a relatively considerable time is needed to solve the whole core pin-by-pin problems. In addition, it is strongly required to improve the computation speed of the MATRA code to satisfy the overall performance of the multi-physics coupling calculations [2]. Therefore, a parallel approach to improve and optimize the computability of the MATRA code is proposed and verified in this study. The parallel algorithm is embodied in the MATRA code using the MPI communication method [3] and the modification of the previous code structure was minimized. An improvement is confirmed by comparing the results between the single and multiple processor algorithms. The speedup and efficiency are also evaluated when increasing the number of processors.

2. Description of Parallel Algorithm

In this section, the algorithm used to embody the parallel computing function into the MATRA code is proposed and described. The parallel algorithm was developed by minimizing the modification of the previous single computing algorithm.

2.1 Single Process Algorithm

The MATRA code applies an implicit method called by SCHEME algorithm. The first order backward difference scheme for both the time and space is applied. For the convection term, an upwind scheme is specially applied. In the present single computing SCHEME algorithm, a solving progress direction is identical to the main flow direction, that is, from upstream to downstream. Therefore, the solution for thermal-hydraulic field is sequentially found as time-marching.

The calculation flow of the SCHEME can be divided by two distinct do-loops, an inner do-loop and an outer do-loop. In the inner loop, the cross flows between subchannels, axial flow, properties, temperature, and so on, are solved at the axial location JX. At this time, the solved values at the axial location JX-1 are applied to the convection terms. In the same way, all of the calculating domain is solved from the bottom to the top, sequentially, plane by plane. The entire axial sweep means the single outer iteration, and the calculation is repeated until the convergence criteria are satisfied. This is the key of the parallel algorithm as described in the next section.

2.2 Parallel Process Algorithm

In the SCHEME algorithm of the MATRA code, all unknown values including the cross flow and axial flow are solved at the present axial node, and the domain to solve then moves to the next axial node. At this time, the previous single processor algorithm can calculate only one axial plane at once. The concept of the parallel algorithm of the MATRA code is that the planes of the same time step are simultaneously solved in the multiple processors. The concept of this parallel algorithm was shown in Fig. 1. In the figure, JX means the axial location. It was assumed that the number of axial node is 50 and the parallelized rank is 5, which means that the 5 processors are used to calculate. As shown in Fig. 2, when the axial plane #1 is calculated on processor #1, the axial planes #2 ~ #4 are calculated on processors #2 ~ #4, simultaneously. In this example, the 4 axial planes are solved during the time during which one axial plane will be calculated in the previous SCHEME algorithm. After each plane on each processor is solved, the next planes are calculated in the same way. Even though each outer iteration consists of 51 calculations in the single computing algorithm, 13 calculations are needed to execute one outer iteration in this parallel example. In addition, the master processor collects the solved results at each plane and broadcasts the merged results before the calculation of the next time step is started.

The calculation flow of the parallel computing compared to a single computing algorithm in section 2.1 is shown Fig. 2. At the first outer iteration, all of the processors execute the same calculation with the previous single computing algorithm to generate an initial value at each plane. This initialization might increase the calculation time. However, it can help to decrease the total number of iterations in order to converge. The axial planes are divided and calculated on all processors from the second outer iteration.

3. Verification of Parallel Algorithm

Verification of the developed parallel algorithm of the MATRA code was executed for three subchannel models, which are a 5 by 5 bundle, 1/8 core, and whole core of SMART reactor. The parameters used to investigate the dimensions of the subchannel model are summarized in Table I. The comparisons between the single and the parallel computing result of the MATRA code for the subchannel model in Table I were conducted in view of the channel exit, bundle average, and MDNBR channel. The results are summarized as the relative and absolute difference term between the single, and parallel computing and are shown in Table II. It is shown that the results of two calculations are identical. At this time, the convergence criteria applied the same values. The axial distributions of CHFR are depicted, as shown in Fig. 3. The MDNBR differences between the single and the parallel computing are evaluated as it was within +/- 0.005.

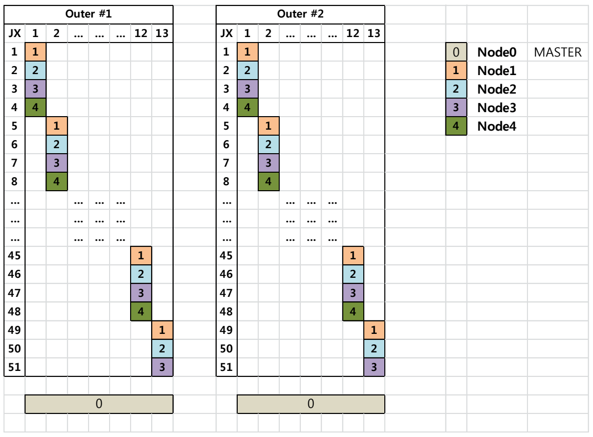

Fig. 1. The examples of the parallel algorithm (when number of axial plane and the SIZE are 50 and 5, respectively)

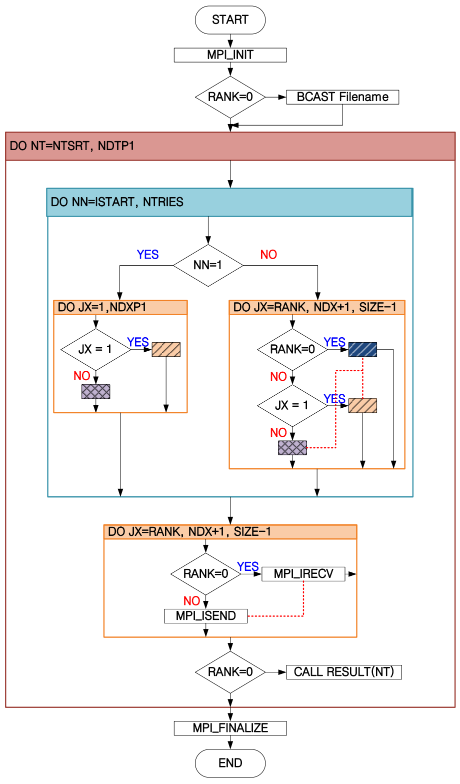

Fig. 2. Calculating flow of the parallel algorithm

Table I: Description of subchannel model for SMART

	5 X 5	1/8 Core	Whole Core
Channel #	36	2,331	16,780
ROD #	25	2,119	15,048
Axial Node #	50	50	50
GAP #	60	4,536	33,252

Table II: Verification results of parallel computing

Channel Exit Value										
		Enthalpy	Temp.	Density	Equil.	OSV	TRUE	Void	Flow	Mass Flux
		%	oC	kg/m3	-	-	-	-	%	%
5x5	MEAN	-0.044	0.000	0.307	0.000	0.000	0.000	0.000	0.009	0.009
	STD	0.002	0.000	0.027	0.000	0.000	0.000	0.000	0.050	0.050
	MAX	-0.040	0.000	0.342	0.000	0.000	0.000	0.000	0.083	0.083
	MIN	-0.047	0.000	0.255	-0.001	0.000	-0.001	-0.001	-0.082	-0.082
1over8	MEAN	0.007	0.016	-0.038	0.000	0.000	0.000	0.000	-0.002	-0.002
	STD	0.037	0.082	0.266	0.001	0.000	0.000	0.000	0.109	0.109
	MAX	0.138	0.300	0.526	0.002	0.000	0.000	0.000	0.335	0.335
	MIN	-0.068	-0.140	-0.994	-0.001	0.000	0.000	0.000	-0.344	-0.339
whole	MEAN	-0.002	-0.005	0.006	0.000	0.000	0.000	0.000	0.006	0.006
	STD	0.033	0.076	0.217	0.000	0.000	0.000	0.000	0.135	0.135
	MAX	0.114	0.250	0.373	0.002	0.000	0.000	0.000	0.566	0.565
	MIN	-0.063	-0.150	-0.774	-0.001	0.000	0.000	0.000	-0.265	-0.265

Bundle Averaged Value												
		Delta-P	Enthalpy	Temp.	Density	Equil.	OSV	TRUE	Void	Flow	MassFlux	Velocity
		%	%	oC	kg/m3	-	-	-	-	%	%	%
5x5	MEAN	-0.285	-0.006	-0.001	0.048	0.000	0.000	0.000	0.000	0.000	0.000	-0.019
	STD	0.103	0.015	0.002	0.139	0.000	0.000	0.000	0.000	0.000	0.000	0.054
	MAX	0.000	0.021	0.000	0.333	0.000	0.000	0.000	0.000	0.000	0.000	0.090
	MIN	-0.542	-0.045	-0.010	-0.300	-0.001	0.000	0.000	0.000	0.000	0.000	-0.150
1over8	MEAN	0.030	0.001	0.001	0.007	0.000	0.000	0.000	0.000	0.000	0.000	-0.001
	STD	0.038	0.001	0.004	0.006	0.000	0.000	0.000	0.000	0.000	0.000	0.002
	MAX	0.287	0.002	0.010	0.017	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	MIN	0.000	0.000	0.000	-0.006	0.000	0.000	0.000	0.000	0.000	0.000	-0.005
whole	MEAN	0.031	-0.001	-0.002	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	STD	0.013	0.001	0.004	0.008	0.000	0.000	0.000	0.000	0.000	0.000	0.002
	MAX	0.094	0.000	0.000	0.021	0.000	0.000	0.000	0.000	0.000	0.000	0.005
	MIN	0.000	-0.002	-0.010	-0.008	0.000	0.000	0.000	0.000	0.000	0.000	-0.005

MDNBR Channel Value												
		Delta-P	Enthalpy	Temp.	Density	Equil.	OSV	TRUE	Void	Flow	MassFlux	Velocity
		%	%	oC	kg/m3	-	-	-	-	%	%	%
5x5	MEAN	-0.377	-0.011	-0.004	0.123	0.000	0.000	0.000	0.000	0.019	0.019	-0.019
	STD	0.194	0.017	0.007	0.150	0.000	0.000	0.000	0.000	0.047	0.047	0.083
	MAX	0.000	0.008	0.000	0.425	0.000	0.000	0.000	0.000	0.153	0.153	0.105
	MIN	-0.936	-0.049	-0.020	-0.084	-0.001	0.000	-0.001	-0.001	-0.082	-0.082	-0.220
1over8	MEAN	0.038	-0.026	-0.059	0.182	0.000	0.000	0.000	0.000	0.309	0.309	0.282
	STD	0.072	0.020	0.043	0.151	0.000	0.000	0.000	0.000	0.203	0.203	0.209
	MAX	0.092	0.000	0.000	0.464	0.000	0.001	0.000	0.000	0.708	0.708	0.682
	MIN	-0.195	-0.060	-0.120	0.000	-0.001	0.000	0.000	0.000	-0.056	-0.056	-0.133
whole	MEAN	-0.047	0.016	0.037	-0.108	0.000	0.000	0.000	0.000	-0.060	-0.061	-0.044
	STD	0.060	0.014	0.032	0.096	0.000	0.000	0.000	0.000	0.054	0.054	0.056
	MAX	0.000	0.036	0.080	0.000	0.001	0.000	0.000	0.000	0.044	0.044	0.078
	MIN	-0.285	0.000	0.000	-0.253	0.000	0.000	0.000	0.000	-0.168	-0.167	-0.159

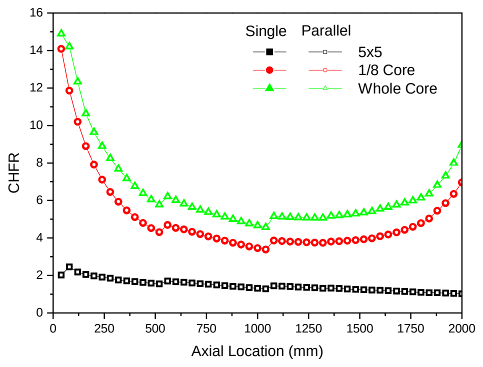

Fig. 3. The SCHEME algorithm of MATRA code solves the unknowns according to the axial direction, step by step

4. Speedup of MATRA Parallel Version

A speedup of the MATRA parallel version was evaluated. The speedup is one of the parameters in multi-processing systems, which is defined as how much a parallel computing is faster than single computing. The required time to obtain the converged solution is compared by increasing the number of processors and is summarized in Table III. The NP means the number of processors used to calculate and MASTER, MEAN, MAX mean the spent time of a master processor, the averaged-spent time of all processors, and the maximum spent time out of all processors, respectively. It was noted that the effects of parallel computing are not shown when the problem dimension is small such as in the 5 by 5 bundle case. The speedup and efficiency of two cases, the 1/8 core and whole core case are shown in Figs. 4 and 5, respectively. As shown in the figures, the speedup and efficiency are decreased as the number of processor is increased according to Amdahl's Law. Moreover, it was shown that those were decreased as the problem size is increased because the cost of communication among clusters is increased. However, the performance of the MATRA code is greatly improved in view of the computation time.

Table III: Speedup and efficiency of MATRA parallel version (unit: sec)

5x5						
NP	1	6	11	16	21	26
MASTER	0.914	0.588	0.792	1.521	1.545	1.501
MEAN	0.914	0.645	0.671	1.443	1.876	1.606
MAX	0.914	0.802	0.855	1.785	2.317	2.076
Speedup	1.000	1.140	1.069	0.512	0.394	0.440
Efficiency	1.000	0.190	0.097	0.032	0.019	0.017
1/8 Core						
NP	1	6	11	16	21	26
MASTER	394.873	58.739	40.489	32.176	28.213	26.746
MEAN	394.873	57.912	39.610	31.215	27.566	25.284
MAX	394.873	58.739	40.489	32.176	28.213	26.746
Speedup	1.000	6.723	9.753	12.272	13.996	14.764
Efficiency	1.000	1.120	0.887	0.767	0.666	0.568
Whole Core						
NP	1	6	11	16	21	26
MASTER	1934.319	499.486	359.990	268.785	257.287	232.862
MEAN	1934.319	493.613	362.154	280.076	267.774	245.164
MAX	1934.319	499.486	362.628	282.304	271.874	250.208
Speedup	1.000	3.873	5.334	6.852	7.115	7.731
Efficiency	1.000	0.645	0.485	0.428	0.339	0.297

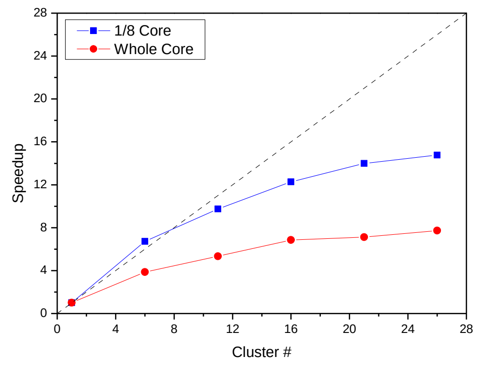

Fig. 4. Speedup of MATRA parallel version

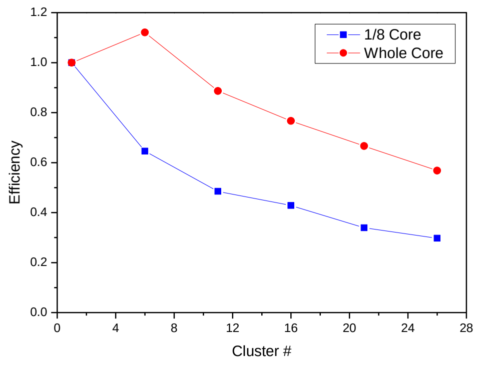

Fig. 5. Efficiency of MATRA parallel version

5. Conclusion

The parallel algorithm was implemented to the subchannel code MATRA using the MPI. The performance of the parallel algorithm was verified by comparing the results with those from the MATRA with the single processor. It is also noticed that the performance of the MATRA code was greatly improved by implementing the parallel algorithm for the 1/8 core and whole core problems.

ACKNOWLEGEMENTS

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. 2012M2A8A4026261)

REFERENCES

[1] S.J. Kim, at al., Development and Assessment of Core T/H Code's Real Time Model for SMART Simulator, KAERI/TR-4904/2013, Korea, Jan., 2013.
[2] R. K. Salko, et al., Suggestions for COBRA-TF Parallelization and Optimization, CASL Technical Report: CASL-U-2013-0003-000, December, 2012.
[3] P. S. Pacheco, Parallel Programming with MPI, Morgan Kaufmann, 1996.