Parallel Computing Characteristics of CUPID code under MPI and Hybrid environment

Jae Ryong Lee[a*], Byoung Jin Jeon[b], Hyoung Gwon Choi[b], Han Young Yoon[a]
[a]*Korea Atomic Energy Research Institute,1045 Daeduk-daero, Daejeon, Korea, 305-353*
[b]*Seoul National University of Science and Technology, 232 Gongneung-ro, Nowon-gu, Seoul, Korea,139-743*
[*]*Corresponding author: jrlee@kaeri.re.kr*

1. Introduction

KAERI has developed a three-dimensional thermal hydraulics code, CUPID for the thermal-hydraulic analysis of nuclear reactor components, such as reactor vessel, steam generator, containment, etc [1-2]. It adopts a three-dimensional, transient, three-field model for two-phase flow and includes various physical models and correlations of the interfacial mass, momentum and energy transfer.

In this paper, a characteristic of parallel algorithm is presented for solving an elliptic type equation of CUPID via domain decomposition method using the MPI and the parallel performance is estimated in terms of a scalability which shows the speedup ratio. In addition, the time-consuming pattern of major subroutines is studied. Two different grid systems are taken into account: 40,000 meshes for coarse system and 320,000 meshes for fine system. Since the matrix of the CUPID code differs according to whether the flow is single-phase or two-phase, the effect of matrix shape is evaluated. Finally, the effect of the preconditioner for matrix solver is also investigated. Finally, the hybrid (OpenMP+MPI) parallel algorithm is introduced and discussed in detail for solving pressure solver.

2. Numerical Methodology

2.1 Governing equation

The CUPID code [1] adopts the two-fluid model for two-phase flows. In the two-fluid model, the mass, energy, and momentum equations for liquid and vapor phases are established separately, and then, they are linked by the interfacial mass, momentum, and energy transfer models. For a mathematical closure, the constitutive relations for the interfacial momentum transfer, the interfacial heat transfer and the wall heat partitioning are necessary.

2.2 Parallelization of the CUPID code

Domain decomposition method

The CUPID code is parallelized on the basis that the users can easily access it on any computing hardware. The SPMD (Single-Program-Multiple-Data) model in conjunction with domain decomposition scheme is employed. Domain partition should be first taken into consideration for parallel computing with MPI. The CUPID code adopts the METIS API [16] for domain decomposition of unstructured mesh system. The number of computational cells in each subdomain is kept almost the same to satisfy the load balance for all processors. Each decomposed subdomain consists of internal cells, internal interface and external interface cells as illustrated in Fig. 1 [3].

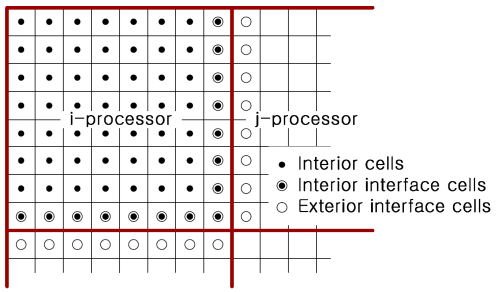

Fig 1. Classification of cells in a subdomain

Parallel preconditioning technique

ILU-type preconditioners [4] show good convergence rate in serial computations. However, there is a communication overhead when the ILU preconditioner is adopted in parallel computation based on domain decomposition method. Therefore, one needs to modify the algorithms of ILU preconditioners so that they reduce the communication overhead. In this respect, diagonal preconditioning and block ILU preconditioning without overlap are employed in the CUPID code.

Compressed sparse row format

The discretized governing equation yields a sparse linear system. In many real CFD-related engineering applications, the matrix consists of a diagonal term and a few off-diagonal terms. That is, most of elements in the rest of the system matrix are zero. Since it is not effective to handle a whole sparse matrix, a memory-saving method to effectively manage the large sparse matrix is used. Of various methods to make the matrix from the discretized governing equation which yields an irregular non-zero pattern, CSR scheme is adopted, which is one of the commonly used for massive computing with unstructured grids. CSR save only non-zero value and its position (row and column). Basically, it requires three one-dimensional arrays, au(:), ja(:) and ia(:) of length NN, NN, and n+1, respectively, where NN is the total number of nonzero elements in the matrix and n is the number of rows.

OpenMP & MPI hybrid parallelization

Recently, a hybrid parallel computing has being raised to maximize the capability of the multi-core supercomputer. Within single computing nodes, the OpenMP is easier to implement and know to be efficient. As a feasibility study, the diagonal preconditioning pressure solver is tested for hybrid parallel computing. To avoid other parameter which would cause confusion to an understanding of the hybrid results, a simple symmetric matrix is chosen. The pressure matrix is subdivided as the number of computing nodes (5 nodes), and the threads for activating OpenMP calculation varies up to 12 cores.

2.3 Scalability analysis

The scalability of the parallel simulation is defined as the speedup ratio of computation time against the number of processors used as follows;

$$\Gamma = \frac{T_1}{T_n} \qquad (1)$$

where T: computation time, n: number of processors

Two types of grid system are taken into account in this study: 40,000 meshes for coarse system and 320,000 meshes for fine system. In the given geometry, both single- and two-phase calculations are conducted. In addition, since most of computation time is due to the pressure solver, the computation time and the scalability for some major subroutines is compared.

3. Results and Discussion

3.1 Scalability analysis for MPI parallel computing

Figure 2 shows the scalability for the coarse mesh system. Both single- and two-phase results with different preconditioners are taken into account. For coarse mesh, the parallel performance with less than 4 processors shows linear scalability but does not more than that. Maximum speedup is expected 20 times faster than serial execution with 40 processors. It is because a communication time among processors is more dominant than an execution time within subdomain. Thus, with the provided mesh size, the domain is excessively partitioned. For the coarse mesh system, the scalability of single- and two-phase calculations for up to 8 processors is almost the same as 7.17 ~ 7.7 and the effect of preconditioner seems negligible. The speedup ratio becomes saturated for more than 8 processors and calculations are not recommended over 20 processors. Although the ILU preconditioner shows a better performance, the difference is not significant for the coarse mesh system.

A fine mesh system improves the speedup ratio as shown in Fig. 3 since the communication time is small compared to computation time. The single-phase calculation executes over 10 times faster than two-phase one. The reason for this difference is mainly due to the pressure solver and will be discussed in detail at following section. For single-phase calculation, the speedup ratio exceeds the ideal speedup. This super-scalable speedup is possible since more cache memories are involved in the parallel calculation [5]. The two-phase calculation shows less scalable than single-phase one. It is one of the important characteristics of the multi-phase CFD simulation and mainly due to the occurrence of asymmetric pressure matrix. Iteration number of the asymmetric matrix solver is large compared to that of single phase which results in more communication time. For a given flow condition, the ILU preconditioner enhances the execution. The preconditioner plays an important role to reduce the computation time. In general, the ILU preconditioner is known to decrease the number of iteration of a matrix solver, and more effective for an asymmetric matrix. Even though the diagonal preconditioner show better scalability, the ILU preconditioner is effective to shorten the real runtime. Thus, the ILU preconditioner is recommended for solving two-phase flow calculation with fine mesh system.

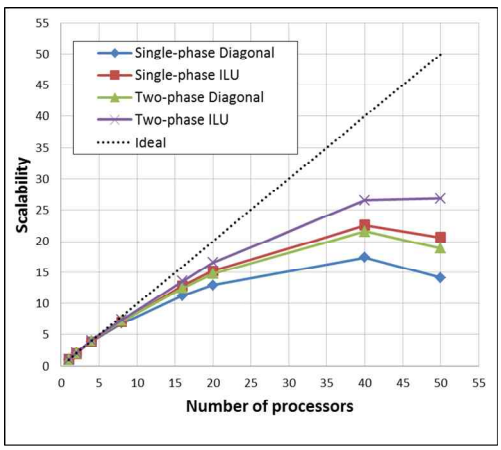

Fig 2. Scalability for coarse mesh system

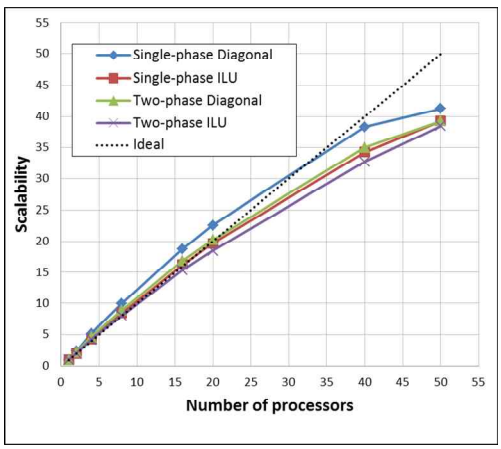

Fig 3. Scalability for fine mesh system

3.2 Scalability analysis for hybrid parallel computing

Fig 4 and 5 shows the comparison between MPI and hybrid parallel computing. The hybrid computing shows better parallel performance than pure MPI. For coarse mesh, the scalability of hybrid computing is similar with one of pure MPI computing. However, as increasing of matrix size, the hybrid computing shows remarkable efficiency. For large matrix size (about 1.5 million grids), the maximum speedup for hybrid computing is about 42, whereas one for pure MPI computing is 28 based on usage of 60 processors.

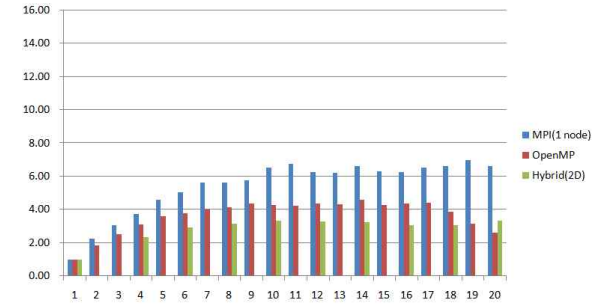

Fig 4. Scalability for hybrid computing for coarse mesh

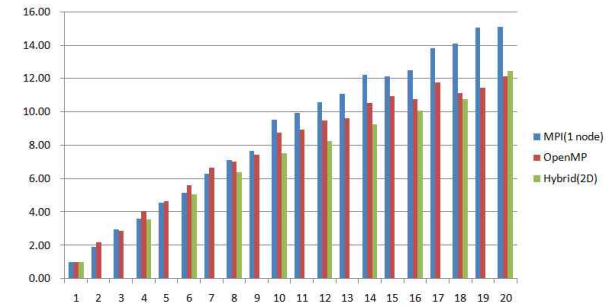

Fig 5. Scalability for hybrid computing for fine mesh

4. Conclusions

Component-scale thermal-hydraulics code, CUPID has been developed for two-phase flow analysis, which adopts a three-dimensional, transient, three-field model, and parallelized to fulfill a recent demand for long-transient and highly resolved multi-phase flow bahavior. In this study, the parallel performance of the CUPID code was investigated in terms of scalability. The CUPID code was parallelized with domain decomposition method. The MPI library was adopted to communicate the information at the neighboring domain. For managing the sparse matrix effectively, the CSR storage format is used. To take into account the characteristics of the pressure matrix which turns to be asymmetric for two-phase flow, both single-phase and two-phase calculations were run. In addition, the effect of the matrix size and preconditioning was also investigated.

The fine mesh calculation shows better scalability than the coarse mesh because the number of coarse mesh does not need to decompose the computational domain excessively. The fine mesh can be present good scalability when dividing geometry with considering the ratio between computation and communication time. For a given mesh, single-phase flow simulation with diagonal preconditioning shows the better speedup. However, for the two-phase flow simulation, the ILU preconditioning is recommended since it reduces the overall simulation time.

In addition, the feasibility study of OpenMP and MPI hybrid parallel computing was discussed. The MPI library was used for node-to-node communication among partitioned subdomains, and the OpenMP threads were activated in every single node using multi-core computing environment. The results of hybrid computing show good performance comparing the pure MPI parallel computing.

Acknowledgement

This work was supported by Nuclear Research & Development Program of the NRF (National Research Foundation of Korea) grant funded by the MEST (Ministry of Education, Science and Technology) of the Korean government(Grant code: 2012M2A8A4025647).

REFERENCES

[1] J.J. Jeong, H.Y. Yoon, I.K. Park and H.K. Cho, “The CUPID code development and assessment strategy”, Nuclear Engineering and Technology, 42, pp.636-655, 2010.

[2] H.Y. Yoon, H.K.Cho, J.R. Lee, I.K. Park and J.J. Jeong, “Multi-scale thermal-hydraulic analysis of PWRs using the CUPID code”, Nuclear Engineering and Technology, 44, pp.831-846, 2012

[3] G.F. Carey, Y. Shen and R.T. Mclay, “Parallel Conjugate Gradient Performance for Least-Square Finite Elements and Transport Problems”, International Journal for Numerical Methods in Fluids, 28, pp.1421-1440, 1998

[4] D.S. Kershaw, “The incomplete Cholesky-conjugate gradient method for the iterative solution of systems of linear equations”, Journal of Computational Physics, 26, pp.43-65, 1978.

[5] S.G. Akl, “Superlinear performance in real-time parallel computation”, Journal of Supercomputing, 29, pp.89-111, 2004