Transactions of the Korean Nuclear Society Spring Meeting
Jeju, Korea, May 10-11, 2007

Performance Evaluation of MARS Solution Schemes on Parallel Computers

K. D. Kim, and B. D. Chung
Thermal-Hydraulic Safety Research Division, Korea Atomic Energy Research Institute, 150 Dukjin-dong, Yuseong, Daejeon, 305-353, kdkim@kaeri.re.kr

1. Introduction

MARS 3.1 [1] has the fully multi-dimensional thermal-hydraulic modeling capability. This makes it possible to examine the detailed behaviors where multi-dimensional effects may occur. This improvement, however, makes the problem size too large to efficiently handle it by using the default direct sparse matrix solver which is the same solver used in RELAP5 [2]. To effectively solve large linear equations for a multi-dimensional component, Pre-conditioned bi-conjugate gradient (PBCG) matrix solver, which solves large sparse coefficient matrices iteratively [3], has been incorporated in MARS. The significant enhancements are achieved for MARS running with the PBCG method over a previously used direct sparse matrix solver. For most one-dimensional problems, there is no speedup or as fast; however, for the problems with wide bandwidths, especially those with three-dimensional regions, significant speedups has been achieved. [4] An accompanying advantage of using PBCG solver is that this solver is easy to use to take advantage of a parallel machine since it does not have recursive calculation.

Recently, the parallel computers are widely used in engineering calculations because these are providing computational capabilities on desktop class machines previously found only on large, more expensive mainframes. The most common parallel computers are distributed memory parallel computers which are usually massive CPU clusters and shared memory parallel computers which have relatively small number of CPUs. The matrix solver portion of MARS has been implemented on both computer architectures to evaluate the parallel performance of MARS because this part is the most computationally intensive portion. This paper will briefly describe an implementation of the MARS code into parallel computers and the evaluation results.

2. Application of Parallel Architectures to MARS

PBCG is inherently parallelizable since its operations are independent of each other. No additional optimization efforts have been initiated for the other part of the MARS code since the objective of this study is initial investigation to evaluate the parallel performance.

2.1 Application of Distributed Memory Computer

MPI (Message Passing Interface) is a library specification for message-passing between processors, proposed as a standard, for high performance on massively parallel distributed memory machines. [5] MPI is used for the implementation of PC cluster to the MARS code. Normal way to implement this type of machines is spread major parts of problem over the independent processors. Therefore, the efficiency is degraded if amount of data that must be transferred between processors becomes large and load for each processor is not balanced. To evaluate the parallel performance of the PBCG matrix solver on a distributed memory machine, the matrix solver portion of MARS was parallelized over more than one processor using MPI libraries. The schematic of this application is shown in Fig. 1.

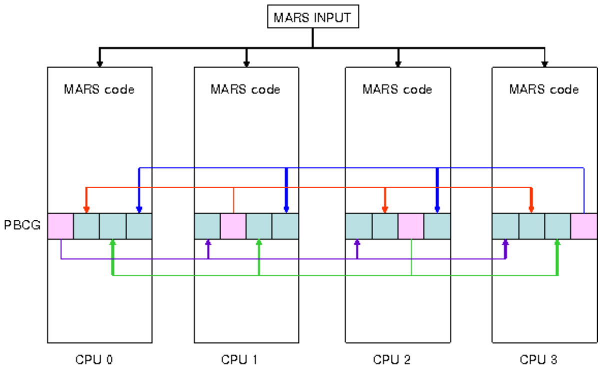

Figure 1. Sample application of a distributed memory parallel machine to MARS

A Linux based cluster PC was used to implement MARS to distributed memory parallel computer and investigate the parallel performance. The initial investigation was not so successful mainly because the amount of data to be transferred between processors is too large and distributed computational load for each processor is too small. Table 1 shows the evaluation results.

Table 1. CPU times (second) on cluster computer

No. of CPUs	1	2	3	4
Total	699.3	824.0	993.9	1275.9
Matrix solution	168.0	211.1	264.6	421.5
Speedup	1	0.85	0.7	0.55

It may be improved if we divide a problem into a few domains and spread the domains over the process. However, this approach is difficult to implement and requires large efforts. The attempt has been made for

RELAP5-3D to make more than one sub-domains of a problem solved on different processors by using PVM (Parallel Virtual Machine) [6]. No significant improvement has been investigated and it has been used mainly for coupled calculation with other application program.

2.2 Application of Shared Memory Computer

Effective way to use shared memory parallel machines is spread independent vector operations at do loop level over processors. Shared memory with multi-processors minimizes the time for data transfer. The efficiency for this type of machines also depends upon amount of data that must be transferred between processors, but the effect is much less significant. The PBCG matrix solver is inherently suitable for a shared memory machine since its operations in do loops are independent of each other. The schematic of this application is shown in Fig. 2.

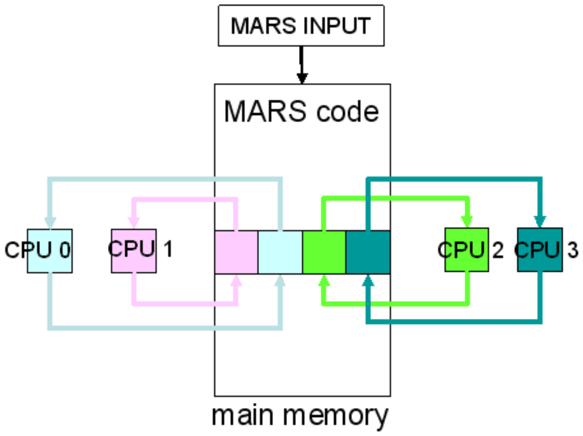

Figure 2. Sample application of a shared memory parallel machine to MARS

A Window-XP based desktop computer is used to implement and investigate the enhancement of the newly adapted iterative solution scheme over a previously used direct matrix solver. The target machine has shared memory architecture with two 64-bit dual-core Intel Xeon 5110 processors. Each processor contains two independent CPU cores with cache memories. An enhanced SSE3 (Streaming SIMD Extension) vector engine allows two floating-point operations to be simultaneously executed in the SIMD (Single Instruction, Multiple Date) format. Similar computer configuration could be achieved with an affordable price range.

To demonstrate the application of a parallel computer approach to the simulation of the MARS code, PANDA problem was used. This problem, which uses a multi-d component, consists of 3400 computational cell to simulate a 3D tank. The tank was initially filled with air and steam was injected at time zero into the tank. For this investigation, Intel Fortran compiler 9.1 was used. Intel compiler automatically generates parallelization at do loop level for shared memory parallel machine. Table 2 shows the evaluation results. The significant enhancement could be possible since the portion of iterative matrix solver uses about 70% of CPU time for large problems. As expected, it was not so successful when direct matrix solver was used.

Table 2. CPU times (second) on shared memory computer

Solver type	Serial	Parallel	Speedup
Direct solver	5039	4488	1.13
Iterative solver	5087	2412	2.11

3. Conclusion

The application of distributed memory parallel computer to the MARS code was not so successful. But the initial investigation proves that PBCG can take full advantage of using a shared memory parallel computer without much effort. Recently, leading computer processor manufacturers such as Intel and AMD have produced the Central Processing Units (CPU), which have parallel and/or vector capabilities, at an affordable price. Therefore, the successful application results for advanced parallel computer architecture to MARS are encouraging us to take advantage of using parallel technology.

ACKNOWLEDGEMENTS

This work has been performed as a part of the Nuclear R&D Program supported by the Ministry of Science and Technology (MOST) of the Republic of Korea.

REFERENCES

[1] MARS Code Manual: Volume I: Code Structure, System Models, and Solution Methods, KAERI/TR-2812/2004, Korea Atomic Energy Research Institute, 2004.

[2] RELAP5/ MOD3.3 Manual: Volume 1: Code Structure, System Models and Solution Methods, US NRC, NUREG/ CR-5535, 2001.

[3] W. H. PRESS, S. A. TEUKOLSKY, W. T. VETTERLING, B. P. FLANNERY, Numerical Recipes in C – The Art of Scientific Computing, 2nd Ed., Chap. 2, Cambridge University Press, New York, 1992.

[4] K. D. Kim, H. S. Lim, B. D. Chung, "Application of an Iterative Solution Scheme to MARS," Tran. Of the Korean Nuclear Society Autumn Meeting, Gyeongju, Korea, Nov. 2-3, 2006.

[5] Argon National Laboratory (ANL), The Message Passing Interface (MPI) Standard, http://www-unix.mcs.anl.gov/mpi, 2006.

[6] Walter L. Weaver III, Programmers Manual for the PVM Coupling Interface in RELAP5-3D Code, Idaho National Laboratory, INL/EXT-05-00203, March 2005.

[7] Lawrence Livermore National Laboratory (LLNL), OpenMP, http://www.llnl.gov/computing/tutorials/openMP/, 2006.