Available on-line at www.prace-ri.eu

Partnership for Advanced Computing in Europe

Scalability Analysis and OpenMP Hybridization of a Code for Direct Numerical Simulation of a Real Wing

Maciej Cytowski[a,*], Matteo Bernardini[b]

[a]Interdisciplinary Centre for Mathematical and Computational Modelling, University of Warsaw
[b]Universitá di Roma La Sapienza, Dipartimento di Ingegneria Meccanica e Aerospaziale

Abstract

The project aimed at extending the capabilities of an existing flow solver for Direct Numerical Simulation of turbulent flows. Starting from the scalability analysis of the MPI baseline code, the main goal of the project was to devise a MPI/OpenMP hybridization capable of exploiting the full potential of the current architectures provided in the PRACE framework. The project was very successful, the new hybrid version of the code outperformed the pure MPI version on IBM Blue Gene/Q architecture (FERMI).

Project ID: PA1454

1. Introduction

Direct Numerical Simulation (DNS) is a natural candidate for physical investigation of turbulent flows because in a direct computation all scales of motion are resolved, thus avoiding to appeal to any turbulence model (as happens for Large Eddy Simulations or Reynolds-Averaged Navier Stokes computations). The main drawback of DNS is given by the extensive computational resources required, since a very small grid spacing is necessary to obtain an accurate resolution of the dissipative scales. Moreover the computational resources dramatically increase with Reynolds number following an approximately cubic power law ($\mathrm{Re}^{11/4}$ in wall-bounded flows).

Current High Performance Computing (HPC) systems now offer the possibility of performing Petascale computations but suitable code design has to be planned to get the highest performance, trying to match the underlying hardware features. For instance, the IBM Blue Gene/Q architecture (available in PRACE) features a complex multinode arrangement with 16 computing cores per node, and 4 hardware threads each. In this view, a possible strategy to improve the performance is to implement an MPI/OpenMP hybridization to reduce the number of MPI processes and minimizing the communication time. Starting from the baseline code, which already has a full MPI parallelization based on Cartesian topologies, an hybrid MPI/OpenMP porting has been attempted and the resulting performance has been analyzed and compared to the pure MPI version using common tools.

2. Scalability analysis, MPI/OpenMP hybridization and results

It should be noted that the baseline code presents very high scalability (both weak and strong) on the IBM Blue Gene/Q system and was frequently used on the FERMI system in production runs. The MPI scalability of the solver (using 16 MPI processMPI processesr node) has been tested up to 8192 cores. However, when trying to exploit the total amount of hardware threads offered by IBM Blue Gene/Q (up to 64 cores per node), a degradation of (intranode) performance is observed, with a gain of a factor 1.3 and 1.4 using 32 and 64 cores per node, respectively. Our main objective was to improve the intranode scaling characteristics through the implementation of an hybrid MPI/OpenMP parallelization. Also, such approach would reduce the total number of MPI tasks used which may be crucial when reaching Petascale performance.

The full MPI/OpenMP hybridization of the code was achieved during the project. At the beginning, performance profile of the code was analyzed with Scalasca tool [1]. In a second step, the most time consuming stages of the code were investigated and successfully hybridized with OpenMP. This was achieved mainly by loop based OpenMP parallelization. Among others this process involved parallelization of Fortran routines dedicated to:

[*] Corresponding author.
e-mail. m.cytowski@icm.edu.pl

- computations of temperature and viscosity fields,
- computations of convective fluxes,
- adding of dissipative fluxes in the flow regions with shock waves,
- evaluation of viscous fluxes.

Original code was compiled with the use of following IBM XL Fortran compiler invocation:

```
mpixlf90 -O3 -qarch=qp -qtune=qp -qautodbl=dbl
```

After applying the OpenMP parallelization the code was compiled with:

```
mpixlf90_r -O3 -qarch=qp -qtune=qp -qautodbl=dbl -qsmp=omp
```

For the sake of this project we have designed three scalability test cases in order to evaluate the MPI/OpenMP hybridization performance on the FERMI system. All test cases involve simulations of turbulent flows on a regular grid:

- **Test A** - small test with grid size $1024 \times 256 \times 192$ which fits into RAM memory of a single FERMI node,
- **Test B** - medium test with grid size $1024 \times 256 \times 4096$ designed to be executed on up to 256 nodes of FERMI system,
- **Test C** - large test with grid size $4096 \times 384 \times 2048$ designed to be executed on larger partitions of FERMI system.

Results of the execution of Test A are presented in Table 1. The main objective of this test was to find the most appropriate MPI/OpenMP mode to be used within a single node of FERMI. The size of the test case was chosen to utilize the memory of a single computing node almost entirely. The fastest execution mode was the one with 8 MPI processes each additionally parallelized with 8 OpenMP threads. However, we have decided to use the **4 MPI x 16 OpenMP mode** for further testing. Firstly, the performance of this mode was very similar to the **8 MPI x 8 OpenMP mode**. Secondly, we wanted to further reduce the number of MPI processes on a single node, since we expect that it might be crucial for achieving Peta-scalability of the application.

Table 1. Test A (small): scalability and performance comparison of pure MPI and MPI/OpenMP hybrid versions of the code on a single node of FERMI system.

Single node mode	**Walltime**	**Speed-up** **vs. 1 pure MPI**
1 MPI (pure MPI)	657.80s	1.00x
1 MPI x 64 OpenMP	30.77s	21.37x
2 MPI x 32 OpenMP	17.43s	37.73x
4 MPI x 16 OpenMP	**12.35s**	**53.26x**
8 MPI x 8 OpenMP	11.13s	59.10x
16 MPI x 4 OpenMP	11.29s	58.26x
32 MPI x 2 OpenMP	11.46s	57.39x
64 MPI (pure MPI)	12.03s	54.67x

The most attractive execution mode within a node was used in further extensive multiple node scalability testing (both Test B and Test C). Results of the execution of Test B are presented in Table 2. The main objective of running the medium test case on the FERMI system was to investigate whether our predictions about the performance of application in hybrid MPI/OpenMP mode are true. At this stage we have found that few additional loops are limiting the multi-node scalability of the code and need to be OpenMP parallelized. In particular this parallelization was applied to the loop performing the solution update of the Runge-Kutta scheme. It was impossible to investigate this during Test A benchmarking due to the small size of the job.

Having successfully tested the MPI/OpenMP hybridization of the code for both small and medium test cases we were ready to perform final extensive scalability testing on FERMI. Final results are presented in Table 3. As can be seen our MPI/OpenMP hybridization outperformed the pure MPI version of the code. Therefore the main objective of the project was achieved.

3. Summary

We have successfully implemented a hybrid MPI/OpenMP version of an existing flow solver for Direct Numerical Simulation of turbulent flows. The final version of the code was extensively tested and is now ready for production runs on the FERMI system. The hybrid MPI/OpenMP version of the code outperformed pure MPI implementation.

The final code resulting from the present preparatory project, will be used for a campaign of direct numerical simulations, mainly on standard airfoil shapes (i.e. the NACA 0012) for which abundant experimental data in the transitional regime are available ([2],[3]).

Table 2. Test B (medium): scalability and performance comparison of pure MPI and MPI/OpenMP hybrid versions of the code on multiple nodes of FERMI system.

Number of nodes	MPI 64 MPI / node		MPI/OpenMP (4 MPI x 16 OpenMP) / node	
	Walltime	Speed-up vs. 64 nodes, MPI	Walltime	Speed-up vs. 64 nodes, MPI
64	16.19s	1.0x	12.23s	1.32x
128	9.44s	1.71x	6.52s	2.48x
256	6.00s	2.69x	4.09s	3.95x

Table 3. Test C (large): scalability and performance comparison of pure MPI and MPI/OpenMP hybrid versions of the code on multiple nodes of FERMI system.

Number of nodes	MPI 64 MPI / node		MPI/OpenMP (4 MPI x 16 OpenMP) / node	
	Walltime	Speed-up vs. 256 nodes, MPI	Walltime	Speed-up vs. 256 nodes, MPI
256	21.56s	1.00x	10.49s	2.05x
512	10.97s	1.96x	5.42s	3.97x
1024	6.27s	3.43x	3.38s	6.37x

Acknowledgements

This work was financially supported by the PRACE project funded in part by the EUs 7th Framework Programme (FP7/2007-2013) under grant agreement no. RI-312763. The work was achieved using the PRACE Research Infrastructure resources at the FERMI (CINECA) supercomputer.

References

1. Scalasca Website: http://www.scalasca.org
2. S. Pirozzoli, M. Bernardini, Phys. Fluids, 25,021704 (2013).
3. T. Sayadi and P. Moin, Phys. Fluids 2012, pp. 114103114103 (2012).