Direct Numerical Simulation and Turbulence Modeling for Fluid-Structure Interaction in Aerodynamics

Thibaut Deloze\textsuperscript{a}, Yannick Hoarau\textsuperscript{b}, Marianna Braza\textsuperscript{a}\textsuperscript{*}

\textsuperscript{a}IMFT, 2 allée du Professeur Camille Soula, Toulouse 31400, France
\textsuperscript{b}IMFS, 2 rue Boussingault, Strasbourg 67000, France

Abstract

The present work focuses on code development of an efficient and robust CFD solver in aeronautics: the NSMB code, Navier-Stokes MultiBlock. A specific advanced version of the code containing Turbulence Modeling Approached developed by IMFT will be the object of the present optimization. The present project aims at improving the performances of the MPI version of the code in order to use efficiently the fat node part of Curie (TGCC, France). Different load balancing strategies have been studied in order to achieve an optimal distribution of work using up to 4096 processes.

"Project ID: PA398"

1. Introduction

The NSMB solver, used during the decade of 1990’s and 2000’s by principal European research institutes and European industry in aeronautics, has been significantly upgraded in the last ten years in terms of implementing new FLOW PHYSICS MODELLING, able to capture complex phenomena arising in aeronautics and aeroelasticity, by the collaborative efforts of CFS-Engineering, IMFT, IMF-Strasbourg, EPFL. A specific advanced version of the code containing Turbulence Modeling Approached\textsuperscript{1} developed by IMFT will be the object of the present optimization. The present project aims at improving the performances of this version of the code in order to use efficiently the largest number of processors currently available. This will allow investigating NEW PHYSICS in the turbulence processes arising in aeronautics, thanks to an increase by orders of magnitude the number of degrees of freedom and by refining the time-steps. Fully implicit numerical schemes will be privileged, dealing properly with the non-linearity of Turbulence in the flow equations. Furthermore, new methods of turbulence modeling and of coupling with deformable solid structures will be implemented in an optimal basis of MPI architecture, to increase the predictive capability of the solver comparing to other currently existing methods, concerning the unsteady

\textsuperscript{*} Corresponding author. \textit{E-mail address:} Marianna.braza@imft.fr.
aerodynamic loads, the nuisance instabilities as buffet and flutter, as well as strong flow detachments. These constitute currently the stiffest problems in aeronautics, fluid-structure interaction and flow control.

The aim of the preparatory access was first to validate the different strategies on this test case for a significant number of blocks. This represents a crucial step to enable simulations on several thousand of cores. We plan to consider problems of higher size to get a satisfactory speed-up on thousand of cores. It is expected that the performances will be higher, for the real size configurations of order 100 million of finite volumes. The influence of the number of blocks, the number of processors but also the distribution of blocks on the nodes available has been studied. Indeed, the sizes of blocks and the connections should be considered for an optimal distribution of works.

2. Methods

2.1. NSMB solver

The tested application is NSMB for “Navier-Stokes Multi-Block”. It solves the Navier-Stokes equations for flow simulation in a wide range of states (laminar, turbulent, compressible, incompressible ...) on structured meshes. Solved linear system is created on each block and the communication of connection boundary conditions creates the link of multi-blocks. The resolution is done independently on each block with a LU-SGS method implemented in Fortran 90. The work focused on establishing a state of efficiency of the parallelization of NSMB for a medium size problem. The goal is to determine the performance, but also to locate the steps blocking or penalizing the parallelization using SCALASCA.

2.2. Domain decomposition

Given a problem of fixed size, the computational domain is decomposed into a significant number of sub-domains (called blocks) and computations on these blocks (up to 4096 blocks ) are performed using different loadbalancing strategies that assign at each MPI process a set of blocks. Three techniques are used to define the distribution of sets of blocks to processes:

- a distribution based on the block size (for uniform distribution of the total size of finite volume per node) called greedystnd,
- a distribution based on the SCOTCH algorithm taking into account the communication between blocks and the size (it minimizes the connections while keeping a uniform distribution of finite volumes per node),
- the METIS tree on the same principle.

2.3. Description of the test cases

In order to study the performance of the code, two configurations have been chosen:

- a wing (OAT15A) at incidence, from subsonic to transonic speeds based on a 3D mesh of 8 million of finite volumes (cf. Fig. 1),
- two cylinders in tandem (static and vibrating) for the landing gear generic configuration (aeroacoustics prediction at high Reynolds number, Re=166000), based on two 3D meshes, 12 millions (3D spanwise length, with D the cylinder diameter) and 60 million (18D spanwise length) (cf. Fig. 2).

These problems are now studied with increased mesh sizes of order 60 M to illustrate the performances in a real size environment.
2.4. Description of the machines

Three machines (x86 architectures) have been used to perform these tests:

- Jade 1 (CINES, France) made of Intel Harpertown processor with 8 cores per node (2 sockets of 4 cores),
- Jade 2 (CINES, France) made of Intel Nehalem processor with 8 cores per node (2 sockets of 4 cores) and QPI technology which provides a better bandwidth than Jade 1,
- Fat nodes of Curie (TGCC, France) made of Intel Nehalem processor with 32 cores per node (4 sockets of 8 cores) and QPI technology.
3. Main results

3.1. Scalability

Scalability tests are done in two stages. Initially, a short test (10 outer iterations with up to 100 inner iterations) was conducted and then a test in real conditions of study (200 outer iterations after transient period). Most of the scaling tests have been performed on a single test case corresponding to an order 12 Million mesh volumes. The speedup and efficiency of our application on the cylinders test case has been studied up to 4096 cores. The performances are satisfactory (efficiency of 88% for 512 cores and 76% for 1024 cores, see Fig. 3).

![Speedup and efficiency graphs](image)

Fig. 3: Speedup (left) and efficiency (right) versus the number of CPU for the tandem cylinders test-case (12M finite volumes) and for the short test.

3.2. Loadbalancing strategies

A first test of the scalability performances has been achieved using medium size grids. The PRACE preparatory access allowed testing different loadbalancing strategies and comparing their respective efficiency up to 4096 processes. The same tests have been performed on three different machines. The hotspots analysis using SCALASCA revealed that the behaviour of the application is significantly different with the three loadbalancing strategies. We noticed it would be wise to first study these behaviours and the impact on the overall performance of the application on the different test cases. Thus, we focused on the current pure MPI version of NSMB and assessed the performance of the loadbalancing strategies.

In term of simulation wall clock time, the METIS tree algorithm has proven to be a very efficient strategy. In term of scalability, the METIS and SCOTCH algorithms give the best results, see Fig. 4, but their performance is non-linear. Arrangements are sometimes preferable for a lower number of processors and we found a quasi-linear speedup (‘superlinear speedup) on Jade 1.
The splitting into a high number of blocks is important for a proper use of the METIS and SCOTCH algorithms and provides a wider flexibility in the distribution (see Fig. 5).

Fig. 4: Speedup versus the number of CPU for the tandem cylinders test-case (12M finite volumes) and for the ‘real condition’ test computed on curie (left) and on Jade 1 (right) using the three load balancing strategies.

Fig. 5: Wall time clock in second (left) and speedup (right) versus the number of CPU for the OAT15A test-case (8,3M finite volumes) using scotch and for different blocks cutting (64, 128, 256 and 512 blocks).
The tests performed have also shown an importance of the architecture, because the METIS and SCOTCH algorithms have different performances depending on the machine used (Fig. 6).

Moreover, the mesh size plays a significant role in the CPU time. Indeed the larger mesh of 60 M shows a more linear behavior of the speedup and an improved scalability (Fig. 6, right).

We noticed that the call to SCOTCH library has to be modified to better share the work between processes and take advantage of this strategy. The process graph (modeling our parallel program) is not build in the appropriate way (default strategy). The building of this process graph represents a crucial step. Indeed, it has to be mapped to the architecture graph (modeling the architecture of the machine).

3.3. OpenMP/MPI hybridization

In order to take advantage of the fat node part of Curie: considerable number of cores per node (32 cores per node) and available memory (128 GB per node), it has been consider to work on a hybrid version of NSMB (OpenMP/MPI). This hybridization task asks for a huge cost in term of human resources and could not be fully accomplished within the 6 month period of access to Curie. The preparatory access enabled us to better examine the chosen resolution algorithm and the parallelization strategy of NSMB. This is essential to help us to determine judicious locations into the resolution algorithm where OpenMP directives have to be added. The preparatory access enabled us to realize some important tests regarding this specific fat node architecture. We actually did not have access to architecture with so many cores per node.

4. Conclusions

The test of the parallelisation of NSMB on a real medium problem has allowed determining its scalability (relative to a moderate mesh size, given the number of CPU hours available: 250,000. The present work shows the importance of the distribution of tasks per node. The choices based on METIS or SCOTCH show better performance when opportunities permit (number of blocks exceeds the number of cores). These tests have shown also the importance of multi-blocks cutting. It may be considered the involvement of the METIS/SCOTCH algorithms in cutting himself. The second track to increase the parallelization of the code is the use of direct solver library (like PSBLAS, MUMPS or SuperLU) on the whole problem. The loss of time in solving the total problem is compensated by an increased convergence and lower internal iterations accelerating the resolution.
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-211528 and FP7-261557. The work is achieved using the PRACE Research Infrastructure resources from Curie (TGCC, France) with support from CINES staff.

References