Cross checking OpenFOAM and Fluent results of CFD simulations in ENEA-GRID environment

The OpenFOAM (Open Field Operation and Manipulation) CFD Toolbox can simulate anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics, electromagnetics and the pricing of financial options. OpenFOAM is produced by OpenCFD Ltd, is freely available and open source, licensed under the GNU General Public Licence.

The core technology of OpenFOAM is a flexible set of efficient C++ modules. These are used to build a wealth of: solvers, to simulate specific problems in engineering mechanics; utilities, to perform pre- and post-processing tasks ranging from simple data manipulations to visualisation and mesh processing; libraries, to create toolboxes that are accessible to the solvers/utilities, such as libraries of physical models.

OpenFOAM is supplied with numerous pre-configured solvers, utilities and libraries and so can be used like any typical simulation package. However, it is open, not only in terms of source code, but also in its structure and hierarchical design, so that its solvers, utilities and libraries are fully extensible.

OpenFOAM uses finite volume numerics to solve systems of partial differential equations ascribed on any 3D unstructured mesh of polyhedral cells. The fluid flow solvers are developed within a robust, implicit, pressure-velocity, iterative solution framework, although alternative techniques are applied to other continuum mechanics solvers. Domain decomposition parallelism is fundamental to the design of OpenFOAM and integrated at a low level so that solvers can generally be developed without the need for any 'parallel-specific' coding. (www.opencfd.co.uk/openfoam).

Here is an example of cross checking results of a CFD simulation between OpenFOAM and Fluent.

The test case is a modification of the "lid driven cavity" case of the official tutorial distribution of OpenFOAM.

As Figure 1 shows the fluid motion is made by the moving wall at the top of the cavity;

Initially the case run with a Reynolds number of 10, where the Reynolds number is defined above and where d and U are the characteristic length and velocity respectively and u is the cinematic viscosity supposed to be 0.01m2s-1

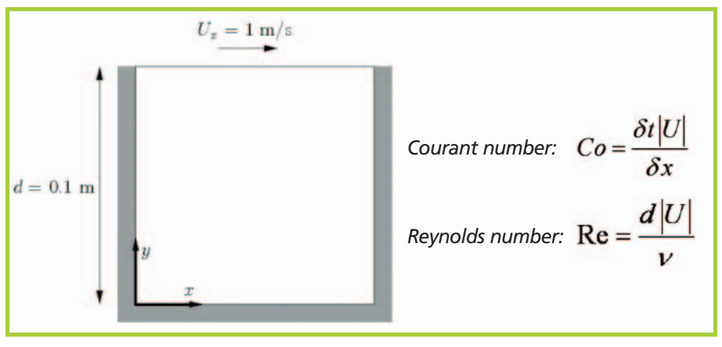

Figure 1 Problem specification of the cavity test

Several meshes refinement have been considered (also for the studies of performances in terms of scalability); in this particular example the mesh is a simple quadrilateral mesh of 41 x 41 cells.

The unsteady solution is obtained with a timestep choice that makes it sure the Courant number is less than 1 everywhere in the flow domain. The Courant number is defined for one cell as is reported above where dt is the time step, U is the velocity through the cell and dx is the cell size in the direction of the velocity.

We chose dt based on the worst case that correspond to dt = 0.005s.

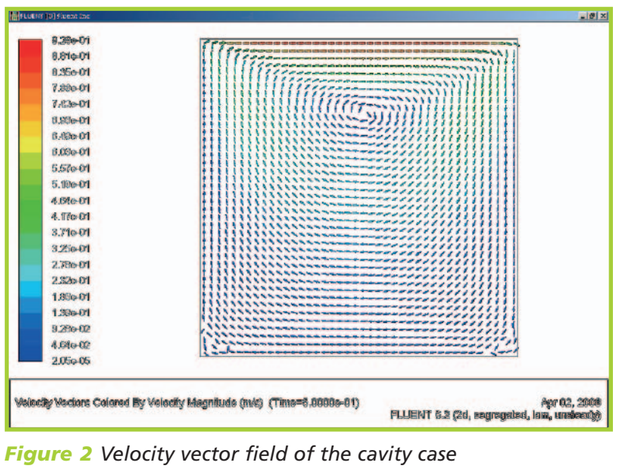

Figure 2 Velocity vector field of the cavity case

Figure 2 shows the velocity vector map coloured by velocity magnitude calculated by Fluent (vector length is constant and not related to velocity magnitude).

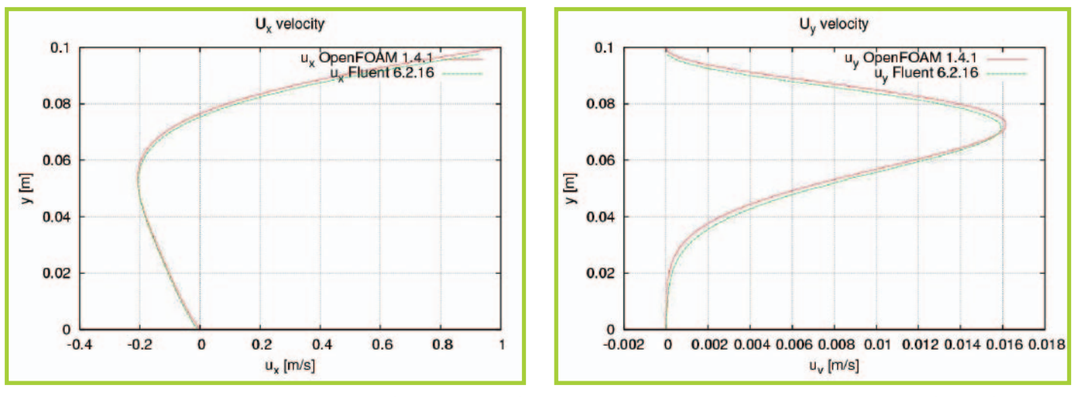

Figure 3 and Figure 4: *Comparison of velocity profiles along the imaginary vertical middle surface of the cavity (at x=0.05m) be-tween Fluent and OpenFOAM*

Figures 3 and 4 show a comparison of velocity profiles along the imaginary vertical middle surface of the cavity (at x=0.05m) obtained from simulations based on OpenFOAM and Fluent implementations of the cavity case. The latter was created manually with Fluent and the former is provided by the OpenFOAM tutorial.

The Fluent implementation of the cavity case was converted in its OpenFOAM equivalent one. To make an automatic conversion of the mesh we used the utility fluentMeshtoFoam provided by OpenFOAM while manual changes were necessary to convert settings of boundary conditions, initial conditions, physical characteristics of the fluid (viscosity etc.) and the characteristic parameters of the numerical simulation.

The case obtained by this conversion was found to be fully equivalent to the original case provided by the OpenFOAM tutorial.

We are now studying the capabilities and limitations for converting to OpenFOAM CFD cases made by Fluent. The goal is to make this procedure as much as possible robust, efficient and automatic on the ENEA-GRID.

overview

On numerical fluid dynamics, computational cost plays a crucial role. Many phenomena can't be treated with approximate numerical methods but by others more complex, such as RANS (Reynolds Averaged Navier Stokes) for example, requiring the adoption of resolution meshes need a large amount of computational calculation.

It is therefore important to have a calculating tool that can handle very hard CFD simulations in reasonably execution time. In the last years computing power of processors has reached a technological limit and then the technological solution adopted to solve these CFD problems is to divide the work among parallel processors (parallel working).

For a CFD solver therefore it is important to be scalable or, in other words, to have a significant improvement in performance as the number of processors used to solve the problem.

On numerical fluid dynamics, computational cost plays a crucial role. Many phenomena can't be treated with approximate numerical methods but by others more complex, such as RANS (Reynolds Averaged Navier Stokes) for example, requiring the adoption of resolution meshes need a large amount of computational calculation.

It is therefore important to have a calculating tool that can handle very hard CFD simulations in reasonably execution time. In the last years computing power of processors has reached a technological limit and then the technological solution adopted to solve these CFD problems is to divide the work among parallel processors (parallel working).

For a CFD solver therefore it is important to be scalable or, in other words, to have a significant improvement in performance as the number of processors used to solve the problem.

Figure 5 shows the performances of parallel simulations on a modification of the case previously discussed (cavity: 2D, 1000x1000 cells, laminar, incompressible, unsteady) obtained by varying the number of cores.

As ENEA users benchmarking group, we have used the EFDA-itm.eu cluster (http://www.efda-itm.eu) composed of 3 front-end and 16 worker multi-core nodes. Each node has 2 processors Dual-Core AMD Opteron(tm) Processor 2218 (4 cores) 2.6GHz, 16GB memory, Gigabit and Infiniband interconnections. This architecture is very similar to the architecture of CRESCO.

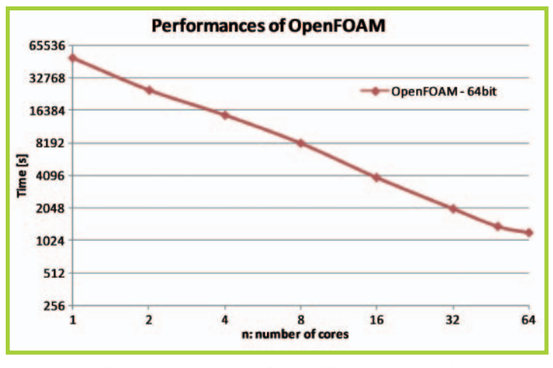

Figure 5 Performances of OpenFOAM: scalabilty.

Many simulations have been done by enabling or not the Infiniband interconnection (IB) to show the improvements due to this powerful technology.

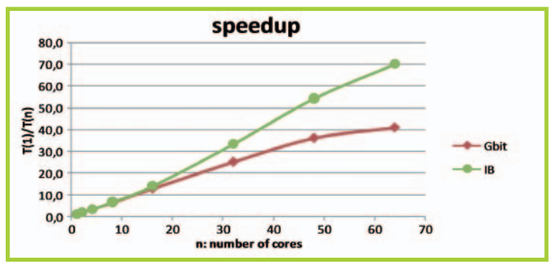

Figure 6 Performances of OpenFOAM: speedup variation due to interconnection.

Figure 6 shows the speedup (obtained respect to the serial simulation) in both cases obtained by using Gigabit and IB interconnection.

Simulations using IB have been done with an implementation of OpenMPI (www.openmpi.com) that use the OpenFabrics (www.openfabrics.org) specification software of Infiniband.

Figure 7 shows the gain on the simulation time obtained by choosing IB rather than Gigabit interconnection (we want to remember the reader that each node has 4 cores therefore the number of nodes involved in the simulation is obtained by dividing the number of core by 4).

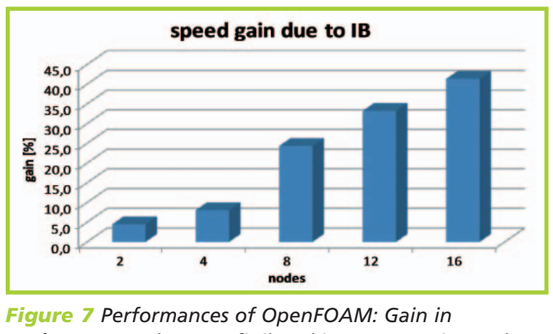

Figure 7 Performances of OpenFOAM: Gain in performan-ces due to Infiniband interconnection rather than Gigabit.

References:
www.fluent.com; OpenFOAM User Guide; www.opencfd.co.uk/openfoam; www.openmpi.com

Autori: F. Ambrosino, S. Raia, A. Funel, S. Podda, S. Migliori
ENEA, Dipartimento Tecnologie Fisiche e nuovi Materiali
e-mail contact: fiorenzo.ambrosino@portici.enea.it ➤ www.cresco.enea.it