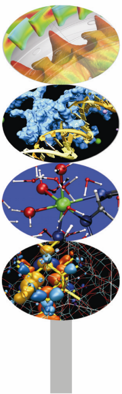
HPC enabling of OpenFOAM® for CFD applications



Cluster configuration and installation of OpenFOAM

26-28 March 2014, Casalecchio di Reno, BOLOGNA.

SuperComputing Applications and Innovation Department, CINECA

Table of Contents

Objectives and Topic

- Objective
  Show the installation of OpenFOAM into HPC cluster, and show the changes between the different type of OpenFOAM installations.
- Topics
  - OpenFOAM installation
  - Network installation: configuration and policy
  - OpenFOAM directory organization
  - Local installation
  - Check your installation
  - Add-ons: `pyfoam`, `swak4foam`

Table of Contents

OpenFOAM Installation

OpenFOAM can be installed for many users (network installation) or for a single user (local installation):

- **Network installation**: This installation is suitable when a group of people is supposed to use OpenFOAM, and when not everyone want to learn how to install and compile it. All users will use exactly the same (base) installation.
  Pro: A single installation for each version of OpenFOAM, maintained by the CINECA UserSupport.
  Cons: limited to major release and most common used tools (`swak4foam`, `pyfoam`).
- **Local installation**: This is the most common way of installing OpenFOAM. The installation will be located in `HOME/OpenFOAM/OpenFOAM-2.x.y`.
  Pro: Each user will 'owns' his proper installation and may update it any time.
  Cons: Requires extra disk space if there are several users with their own installations (minor issue), and all users must know how to install OpenFOAM and the Third-Party products (major issue)

CINECA policies:

- Network installation only major 2.3.0, 2.4.0, ...2.n.0 by default.
- Minor installation 2.3.1, 2.3.2, ...2.3.4 upon request.
- `profile base` ⇒ last two majors + 1 minor. `profile advanced` ⇒ other versions
- Git and .x only local installation

Table of Contents

Configure and install on PLX cluster

- The configuration on PLX, that will be used for the tutorial, is shown at a glance.
- You DO NOT need to do it. Already installed and tested.
- You can use it, as template, if you want to install your local/git/modified version.
- The configuration is set in the `bashrc` file, which is located in
  `/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0/etc/bashrc`

```
# File
#     etc/bashrc
# Description
#     Startup file for OpenFOAM
#     Sourced from ~/.profile or ~/.bashrc

export WM_PROJECT=OpenFOAM
export WM_PROJECT_VERSION=2.3.0

#- Compiler location:
foamCompiler=system

#- Compiler:
export WM_COMPILER=Gcc

#- Architecture:
export WM_ARCH_OPTION=64

#- Precision:
export WM_PRECISION_OPTION=DP

#- Optimised, debug, profiling:
export WM_COMPILE_OPTION=Opt

#- MPI implementation:
export WM_MPLIB=SYSTEMOPENMPI
```

The MPI installed on PLX is used, by loading the specific module

```
module load profile/advanced
module load gnu/4.7.2
module load openmpi/1.6.3--gnu--4.7.2
env | grep OPENMPI_HOME
OPENMPI_HOME=/cineca/prod/compilers/openmpi/1.6.3/gnu--4.7.2
```

Installation is performed by sourcing the relative bashrc

```
source OpenFOAM-2.3.0/etc/bashrc
```

and compiling the source code

```
export WM_NCOMPPROCS=8
cd OpenFOAM-2.3.0
./Allwmake
```

Compilation with 8 procs needs $\sim$ 1 hour

Loading the OF module on PLX

After the module have been installed, you can check the available modules

```
module avail openfoam
openfoam/2.2.0-gnu-4.7.2 openfoam/2.2.1-gnu-4.7.2 openfoam/2.3.0-gnu-4.7.2
```

and load it on PLX, by typing in your shell

```
module load autoload
module load openfoam/2.3.0-gnu-4.7.2
```

The corresponding `bashrc` is sourced and the environment variables, libraries and applications are loaded.

The environmental variables and the aliases are set in the module. Typing

```
module show  openfoam/2.3.0-gnu-4.7.2
```

or

```
env | grep WM
env | grep FOAM
```

You will see them

```
$FOAM_INSTALL_DIR= /cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2 ---> Installation dir
$FOAM_MPI=openmpi-system ---> type of MPI used
$FOAM_LIBBIN=
/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0/platforms/linux64GccDPOpt/lib
FOAM_SOLVERS=/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0/applications/solvers
WM_PROJECT_VERSION=2.3.0 ---> Current version used
WM_PROJECT_DIR=/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0
$WM_PROJECT_USER_DIR=/plx/userinternal/ispisso0/OpenFOAM/ispisso0-2.3.0 --->It is the user directory. User dependent variable!
$FOAM_RUN=/plx/userinternal/ispisso0/OpenFOAM/ispisso0-2.3.0/run ---> User dependent variable!
..
```

Other versions: to see the other available modules in *profile/adavanced*, that are the minor releases

```
module load profile/advanced
module av  openfoam
```

$\boxed{\texttt{openfoam/2.1.1-gnu-4.7.2}}$ `openfoam/2.2.1-gnu-4.7.2`
`openfoam/2.2.0-gnu-4.7.2 openfoam/2.3.0-gnu-4.7.2`

Table of Contents

OpenFOAM directory organization

Examine the directory organization using the tree command

```
tree -L 1 -d $WM_PROJECT_DIR
/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0
|-- applications   ............ source executable
|-- bin            ............ shell script for file manipulation such as foamEtcFile, foamLog, foamInstallationTest
|-- doc            ............ the doc directory contains the documentation of OpenFOAM
|-- etc            ............ the etc directory contains env set-up, global OpenFOAM instructions and thermoData
|-- platforms      ............ the platforms directory contains the compiled binary of executable and libraries
|-- src            ............ the directory source contains the source code for all the libraries
|-- tutorials      ............ the directory tutorial contains example cases for each solver
`-- wmake          ............ script and rules for compiling
```

The applications directory

Examine the directory organization using the tree command

```
 tree -L 1 -d $WM_PROJECT_DIR/applications
/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0/applications
|-- solvers   ............ contains source code for the distributed solvers
|-- test      ............ contains source code that test and show example of usage os some of the OpenFOAM classes
`-- utilities ............ contains source code for the distributed utilities
```

```
    tree -L 1 -d solvers
|-- DNS
|-- basic
|-- combustion
|-- compressible
|-- discreteMethods
|-- electromagnetics
|-- financial
|-- heatTransfer
|-- incompressible
|-- lagrangian
|-- multiphase
`-- stressAnalysis

12 directories
```

```
    tree -L 1 -d test
|-- BinSum
|-- Circulator
|-- CompactIOList
|-- CompactListList
|-- DLList
|-- Dictionary
|-- Distribution
|-- DynamicField
|-- DynamicList
|-- Field
|-- FixedList
|-- GAMGAgglomeration
|-- HashPtrTable
|-- HashSet
|-- HashTable
|-- HashTable2
|-- HashTable3
|-- Hashing
|-- HashingSpeed
........
.....
...

126 directories
```

```
    tree -L 1 -d utilities
|-- mesh
|-- miscellaneous
|-- parallelProcessing
|-- postProcessing
|-- preProcessing
|-- surface
`-- thermophysical

7 directories
```

The `src` directory

Examine the directory organization using the tree command

```
 tree -L 1 -d $WM_PROJECT_DIR/src   ........... (35 dir, only a bunch of them)
/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0/src
|-- ODE
|-- OSspecific
|-- OpenFOAM  This library includes the base classes: containers, data types, operators, Input/Output. The bone of OpenFOAM
|-- Pstream  This library includes the parallel communication classes based on message passing library (MPI)
|-- combustionModels
|-- conversion
|-- dynamicFvMesh
|-- dynamicMesh This library contains moving meshes algorithmics
|-- engine
|-- fileFormats
|-- finiteVolume  This library provides all the classes for FV discretization, such as fvMesh, divergence, laplacian,
gradient, discretization operators, matrix solvers and boundary conditions
|-- fvAgglomerationMethods
|-- fvMotionSolver
|-- fvOptions

|-- lagrangian
|-- mesh
|-- meshTools
|-- parallel  This library contains the source code for decomposition, reconstruction and distribution of the domain
|-- postProcessing
...
|-- sampling
....
|-- topoChangerFvMesh
....
`-- turbulenceModels   ...........contains several turbulence models (RANS, DES, LES, Smagorinsky, SpalartAllmaras..)

35 directories
```

The platforms directory

The platforms directory contains the compiled binaries of executables (solvers, utilities) and libraries.

$WM_PROJECT_DIR/platforms=linux64GccDPOpt

```
tree -L 2 -d $WM_PROJECT_DIR/platforms
/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0/platforms
`-- linux64GccDPOpt
    |-- bin ......................It includes icoFoam, blockMesh, etc. >> total of 225 binaries
    `-- lib ......................It includes libOpenFOAM.so, libPstream.so, etc. >> total more than 100 libraries
```

A subfolder is created for each compiler configuration.
The configuration is set in the etc/bashrc file. The subfolder contains separate folders for executables and libraries.
The name of the directory is assembled according to:

<OS name><N bits><compiler name><precision><option>

When compiling on PLX, e.g. a <linux> machine at <64> bit, with <Gcc> compiler in <DP> Double Precision, with <Opt> Optimized compilation option, the folder name will be

linux64GccDPOpt

This allows for having different compiled versions of OpenFOAM on the same directory path, without duplicating the installations.

Table of Contents

Local installation

- If you want to install a minor release (*.x) or update a git release (updated by the the developer in the source forge), you have to install a local version.
- As example, it is described how to install a 2.3.x version from git in your $*HOME* space, based on the OpenFOAM UserGuide
- Please use the script available in one of the Network Installation. For the last version 2.3:

```
$FOAM_INST_DIR/OF_2.3.0_LOCAL_INSTALL_CINECA_PLX.tar
```

Configuration and installation of the local version

```
cd $HOME
cp /cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OF_2.3.0_LOCAL_INSTALL_CINECA_PLX.tar .
tar -xf OF_2.3.0_LOCAL_INSTALL_CINECA_PLX.tar
rm OF_2.3.0_LOCAL_INSTALL_CINECA_PLX.tar
cd OF_2.3.0_LOCAL_INSTALL_CINECA_PLX
ls
OF_2.3.0_LOCAL_INSTALL_CINECA_PLX.sh PATCHES
```

Modification and customization of the script

```
OF_2.3.0_LOCAL_INSTALL_CINECA_PLX.sh
# I section: customize here you OpenFOAM installation
# select the downloading type you want
export TYPE=2  #  1=current version /  2=GIT .x create (git clone)  /  3=GIT .x update (git pull)
# II section: set variables depending on the OpenFOAM version
# III section: download/patch/install
```

Local installation

Execute the script to download, configure and install your local version of OpenFOAM:

```
./OF_2.3.0_LOCAL_INSTALL_CINECA_PLX.sh
```

The installation dir, in this case, is located in your `$HOME` space

```
FOAM_INST_DIR=$HOME/OpenFOAM
```

The OpenFOAM environment of the local installation is set in

```
$HOME/OpenFOAM/OpenFOAM-2.3.x/etc/bashrc
```

To load the environment of your local installation

```
module load gnu/4.7.2
module load openmpi/1.6.3--gnu--4.7.2
source $HOME/OpenFOAM/OpenFOAM-2.3.x/etc/bashrc
```

and you will have

```
$WM_PROJECT_VERSION=2.3.x
$FOAM_INST_DIR=/plx/userinternal/ispisso0/OpenFOAM
$WM_PROJECT_USER_DIR=/plx/userinternal/ispisso0/OpenFOAM/ispisso0-2.3.x
```

Customize your profile: Tip & Tricks

You can automatically load your preferred OpenFOAM version in your `bashrc`, by adding to it the following lines

```
  # .bashrc
# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi

# User specific aliases and functions
of230()
{
        module purge
        module load autoload/0.1/verbose
        module load openfoam/2.3.0-gnu-4.7.2
}

of23x_local()
{
        source $HOME/OpenFOAM/OpenFOAM-2.3.x/etc/bascrc
}
```

When you will open a new shell, you can use one of the defined function to load the specific OpenFOAM environment. The corresponding `$WM_PROJECT_DIR/etc/bashrc` will be sourced and the OpenFOAM environment will be set up.

```
[ispisso0@node342 ~]$ of230
### auto-loading modules gnu/4.7.2 openmpi/1.6.3--gnu--4.7.2
### auto-loading modules gnu/4.7.2 openmpi/1.6.3--gnu--4.7.2
echo $WM_PROJECT_VERSION
2.3.0
echo $WM_PROJECT_DIR
/cineca/prod/applications/openfoam/2.3.0-gnu-4.7.2/openmpi--1.6.3--gnu--4.7.2/OpenFOAM-2.3.0
```

Table of Contents

Check your installation

After having loaded the module, type the following commands to test the installation

```
foamInstallationTest
```

If everything is in order, you will see at the end of the log the message

```
 Summary
-------------------------------------------------------------------------------
Base configuration ok.
Critical systems ok.
Done
```

and run by command line the `icoFoam` tutorial

```
cp -r $FOAM_TUTORIAL/incompressible/icoFoam/cavity $FOAM_RUN
run
cd cavity
blockMesh
Build  : 2.3.0-f5222ca19ce6
Exec   : blockMesh
Date   : Mar 20 2014
Time   : 12:41:36
Host   : "node342"
PID    : 8583
Case   : /plx/userinternal/ispisso0/OpenFOAM/ispisso0-2.3.0/run/cavity
nProcs : 1
....
icoFoam
Create time
Create mesh for time = 0
Reading transportProperties
Reading field p
Reading field U
Reading/calculating face flux field phi
Starting time loop
Time = 0.005
.....
Time=0.5
End
```

Table of Contents

Add-ons: `pyfoam`, `swak4foam`

Useful and common-used add-ons to standard OpenFOAM distribution are:

- `pyFoam`: pyFoam is a Python front-end to the OpenFOAM. It introduces interactivity into OpenFOAM, simplifies connection with third-party functionality and streamlines design of custom user solvers. It is used to control the OpenFOAM runs and manipulate OpenFOAM data. Website
- `swak4foam`: swak4foam is a library that combines the functionality of groovyBC and funkySetFields: it offers the user the possibility to specify expressions involving the fields and evaluates them. This library offers a number of utilities (for instance funkySetFields to set fields using expression), boundary conditions (groovyBC to specify arbitrary boundary conditions based on expressions) and function objects that allow doing many things that would otherwise require programming. Website