

OpenFOAM on ARCHER

Mark Filipiak, EPCC

23 Sep 2015

The EPSRC logo consists of the letters 'EPSRC' in a bold, purple, sans-serif font. It is framed by two horizontal teal lines, one above and one below the text.

EPSRC

The NERC logo features the letters 'NERC' in white on a dark green rectangular background. To the right, the words 'SCIENCE OF THE ENVIRONMENT' are written in white on a yellow-green rectangular background.

NERC SCIENCE OF THE ENVIRONMENT

The archer logo features a red and white bullseye target icon on the left, followed by the word 'archer' in a white, lowercase, sans-serif font on a black rectangular background.

archer

The Cray logo features the word 'CRAY' in a bold, blue, sans-serif font. Below it, the words 'THE SUPERCOMPUTER COMPANY' are written in a smaller, blue, sans-serif font.

CRAY
THE SUPERCOMPUTER COMPANY

The epcc logo features the lowercase letters 'epcc' in a dark blue, sans-serif font, flanked by two vertical red lines.

epcc



OpenFOAM versions installed

OpenFOAM version	Compute nodes	Serial nodes
2.2.2	✓	✓
2.3.0	✓	✗
2.3.1	✓	✗
2.4.0	✓	not yet

Compute versus serial

Compute nodes have MPI and increased vectorisation (AVX), use them for

- parallel solvers (icoFoam, ...)
- parallel utilities (snappyHexMesh, ...)
- serial utilities as part of a job (blockMesh, ...)

Serial nodes have lots of memory, use them for

- pre- and post-processing that requires large amounts of memory (reconstructPar, ...)



Using OpenFOAM

For 2.2.2 (2.3.0, 2.3.1 are similar)

```
module swap PrgEnv-cray PrgEnv-gnu
unset FOAM_INST_DIR WM_PROJECT_SITE
source /work/y07/y07/cse/OpenFOAM/OpenFOAM-
2.2.2/etc/bashrc
```

For 2.4.0

```
module load openfoam/2.4.0
unset FOAM_INST_DIR WM_PROJECT_SITE
source $OPENFOAM_DIR/OpenFOAM-2.4.0/etc/bashrc
```



Compiling OpenFOAM

For 2.2.2 (2.3.0, 2.3.1 are similar)

See the ARCHER OpenFOAM webpage

For 2.4.0

```
module load openfoam-build/2.4.0
```

and follow the instructions in

```
module help openfoam-build/2.4.0
```



Don't compile OpenFOAM

Compiling OpenFOAM takes 9 hours

If you are modifying an application or library

- `source/OpenFOAM-x.y.z/etc/bashrc`
- copy the application or library source directory to your user OpenFOAM directory
- and compile it

See Section 3.2 of the OpenFOAM User Guide for details



ARCHER specifics: `/work`

All files need to be on `/work` to be accessible from the compute nodes:

- case directories and files (`FOAM_RUN`)
- user applications (`FOAM_USER_APPBIN`)
- user libraries (`FOAM_USER_LIBBIN`)

```
source ...../OpenFOAM-x.y.z/etc/bashrc
```

sets the user directory on `/work`, for example

```
/work/z01/z01/mjf/OpenFOAM/mjf-2.4.0/run
```



/work is not backed up

Use a version control system (subversion, git) and repository for your source files (applications and libraries)

When working on a case, mirror your dictionaries and constant data to the RDF using `rsync`

- You may want to mirror the case results as well but this will be slow for large numbers of files
- You can use `rsync` for data transfer also

When you have finished working on a case and want to archive it, create a tar file and copy it to the RDF.



ARCHER specifics: `aprun`

The ARCHER command to start an MPI program on the compute nodes is `aprun`

```
aprun -n 2400 icoFOAM -parallel &> if.log
```

Use `aprun` for serial utilities that are run as part of a job on the compute nodes

```
aprun -n 1 decomposePar &> dec.log
```



ARCHER specifics: PrgEnv

OpenFOAM on Archer has been built using the Gnu programming environment

Any applications or libraries that you build must also be built with the Gnu programming environment, use

```
module swap PrgEnv-cray PrgEnv-gnu
```

OpenFOAM can be built with the Intel compiler – currently cannot use `codeStream` with the Intel compiler



ARCHER specifics: ParaView

OpenFOAM has not been built with ParaView

Use the centrally installed ParaView

```
module load paraview
```

This does mean that the file readers and user interface panel provided by OpenFOAM are not available - use ParaView's built-in versions



/work is a Lustre file system

OpenFOAM

Each process writes a file for each output field at each output time

Number of files =
number of output fields ×
number of output times ×
number of processes

e.g. $4 \times 50 \times 240 = 48,000$

Lustre

Not optimised for large numbers of small files

- Opening and closing files can be slow (MDS)
- Large numbers of processes (\gg number of OSTs) can contend for access

Matching OpenFOAM to Lustre (1)

Setting the stripe count to one may improve file read and write performance

```
lfs setstripe -c 1  
/work/z01/z01/mjf/OpenFOAM/mjf-2.4.0
```

- The stripe count is inherited by all files and directories that you create in or copy to the directory
- Files that are moved using `mv` don't change their stripe count



Matching OpenFOAM to Lustre (2)

If you find that I/O takes a significant fraction of your job time, try changing output settings in `controlDict`

- Increase `writeInterval`
- Use binary format for the fields `writeFormat binary`
- For steady-state solutions `purgeWrite 1`
- (perhaps) Don't read dictionaries at every time-step
`runTimeModifiable no`

There is an HDF5 library for OpenFOAM which may be help, but there are some limitations



Matching OpenFOAM to Lustre (3)

OpenFOAM is dynamically linked and has dynamically loaded libraries (`libs` and `functionObjectLibs`) and run-time code compilation (`codeStream`)

- each process opens these shared objects (`.so`) and reads (via `mmap`) parts of an object as they are needed, for example when a function is called
- Some of these shared objects are on `/work` so there can be many accesses to many small files, which may be slow

If start-up takes a large fraction of the run time, the DLFM package may help, please contact the ARCHER helpdesk for assistance



Visualisation with ParaView - choices

choice	Rendering	User interface
transfer the results to your desktop and run ParaView there	fast <small>(if you have a good graphics card)</small>	fast
Run pvserver in parallel on the compute nodes with a ParaView client on your desktop	fast <small>(uses kAUs)</small>	fast
Run pvserver on a post-processing node with a ParaView client on your desktop	slow	fast
Run ParaView on a post-processing node	slow	slow



Links and References

- ARCHER OpenFOAM webpage:
<http://www.archer.ac.uk/documentation/software/openfoam>
- OpenFOAM User Guide:
<http://cfd.direct/openfoam/user-guide>
- Data Management Guide:
<http://www.archer.ac.uk/documentation/data-management>
- OpenFOAM HDF5 output
<http://openfoamwiki.net/index.php/Contrib/IOH5Write>
- ARCHER DLFM guide
https://www.archer.ac.uk/documentation/white-papers/dynamic-import/ARCHER_wp_dynamic-import.pdf



Reusing this material



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

http://creativecommons.org/licenses/by-nc-sa/4.0/deed.en_US

This means you are free to copy and redistribute the material and adapt and build on the material under the following terms: You must give appropriate credit, provide a link to the license and indicate if changes were made. If you adapt or build on the material you must distribute your work under the same license as the original.

Note that this presentation contains images owned by others. Please seek their permission before reusing these images.

