Science and Technology Facilities Council

Hartree Centre

CCP-WSI

a Collaborative Computational Project in Wave Structure Interaction

Serial to Parallel OpenFOAM Cases

Dr. Raynold Tan, Dr. Xiaohu Guo

Table of Content

Science and Technology Facilities Council
Hartree Centre

Science and Technology Facilities Council

Hartree Centre

Fundamentals

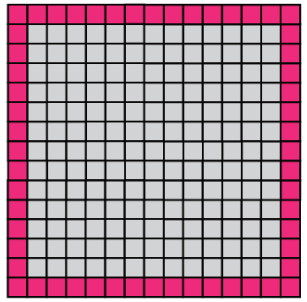
Fundamentals

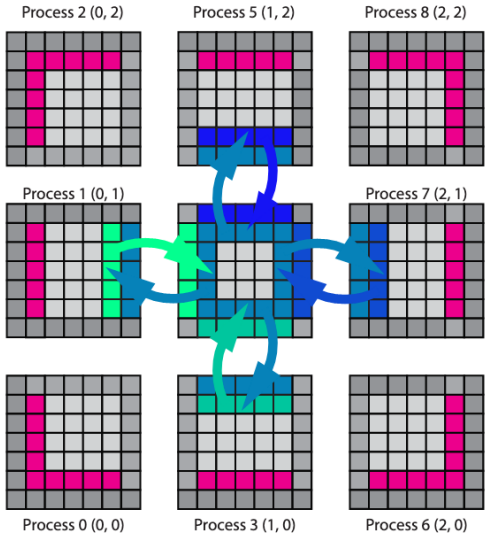
When running a parallel simulation, the mesh is decomposed into individual parts for each MPI Process. Each separate mesh are connected together with special processor boundary patches. Each processor-specific constant/polyMesh/boundary files will contain this type of entry in addition to the physical processor boundary patch:

```
procBoundary1to0
{
    type            processor;
    inGroups        1(processor);
    nFaces          26;
    startFace       2244;
    matchTolerance  0.0001;
    transform       unknown;
    myProcNo        1;
    neighbProcNo    0;
}
procBoundary1to3
{
    type            processor;
    inGroups        1(processor);
    nFaces          23;
    startFace       2270;
    matchTolerance  0.0001;
    transform       unknown;
    myProcNo        1;
    neighbProcNo    3;
}
```

Fundamentals

Halo Layer Approach

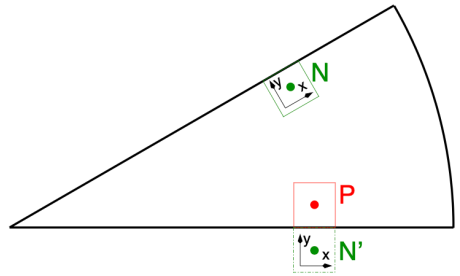




Halo layer covers all processor boundary and is updated through the parallel communication call
Each processor mesh has to account for additional halo cells
All face and cell for loops need to account for the presence of the halo cells
No. of layers of halo depend on the stencil width of discretization scheme

Zero Halo Layer Approach

Instead of storing halo cell info, processor boundary update combines the process of taking in neighbouring cell values from communication call (which is stored in cache memory) and perform evaluation



$$\phi_f = f_x\phi_P + (1 - \mathrm{f_x})\phi_N$$

In parallel, ϕ_P and ϕ_N live on different processors. Assuming ϕ_P is local and ϕ_N is fetched through communication, this is a once per solution cost and obtained by pairwise communication.
For a processor boundary between domain A and B, evaluation of face values is carried out in 3 steps:

1. Collect subset of internal cell values from local domain and send values to neighbouring processor
2. Receive neighbour values from neighbouring processor
3. Evaulate local processor face value using interpolation

Fundamentals

The decomposeParDict is required by decompose utilities and for any solver or utilities running in parallel. It is located in: name_case/system/

Note: you have to source the OpenFOAM-v2212/etc/bashrc in order to use the commands below.

Commands to decompose or reconstruct a case folder in serial:

- decomposePar
- reconstructPar

Note: you can see the full range of options available using decomposePar/reconstructPar -help

Commands to decompose or reconstruct a case folder in parallel:

- mpirun –np XXX redistributePar –decompose –parallel
- mpirun –np XXX redistributePar -reconstruct -parallel

```
Usage: decomposePar [OPTIONS]
Options:
  -allRegions       Operate on all regions in regionProperties
  -case <dir>       Specify case directory to use (instead of cwd)
  -cellDist         Write cell distribution as a labelList - for use with
                    'manual' decomposition method and as a volScalarField for
                    visualization.
  -constant         Include the 'constant/' dir in the times list
  -copyUniform      Copy any uniform/ directories too
  -copyZero         Copy 0/ directory to processor*/ rather than decompose the
                    fields
  -decomposeParDict <file>
                    Use specified file for decomposePar dictionary
  -dry-run          Test without writing the decomposition. Changes -cellDist to
                    only write volScalarField.
  -fields           Use existing geometry decomposition and convert fields only
  -force            Remove existing processor*/ subdirs before decomposing the
                    geometry
  -ifRequired       Only decompose geometry if the number of domains has
                    changed
  -latestTime       Select the latest time
  -noSets           Skip decomposing cellSets, faceSets, pointSets
  -noZero           Exclude the '0/' dir from the times list
  -region <name>    Specify alternative mesh region
  -time <ranges>    List of ranges. Eg, ':10,20 40:70 1000:', 'none', etc
  -verbose          Additional verbosity
  -doc              Display documentation in browser
  -help             Display short help and exit
  -help-full        Display full help and exit

Decompose a mesh and fields of a case for parallel execution
```

Science and Technology Facilities Council

Hartree Centre

Decomposition Options

Decomposition Options

Specification in decomposeParDict:

numberOfSubdomains <int>;
method <word>;

- If a decomposition method requires any additional configuration controls, these are specified either within in a generic coeffs dictionary or a method-specific version.
- Note: Some of the decomposition methods are not included in the standard OF installation, need to download decomposition library and configure the Allwmake script in the ThirdParty Folder.
- Ideal to have equal work load on each processor, minimise the mesh interface between processors to reduce communication workload
- If one have a simple mesh, use simple/hierarchical decomposition. For complex mesh, try out kahip, scotch or metis (mesh dependent).

Name	Class
none	Foam::noDecomp
manual	Foam::manualDecomp
simple	Foam::simpleGeomDecomp
hierarchical	Foam::hierarchGeomDecomp
kahip	Foam::kahipDecomp
metis	Foam::metisDecomp
scotch	Foam::scotchDecomp
structured	Foam::structuredDecomp
multiLevel	Foam::multiLevelDecomp

```
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      decomposeParDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

numberOfSubdomains 4;

method          simple;

coeffs
{
    n           (2 2 1);
}

// ************************************************************************* //
```

```
method  hierarchical;

coeffs
{
    n   (4 2 3);
}

// -----

method  metis;

metisCoeffs
{
    method  k-way;
}
```

Science and Technology Facilities Council

Hartree Centre

Decomposition Options

1. Simple Method:

If domain is not split into equal no. of cells for each procs, will lead to load imbalance

Decomposes the domain in equal parts according in the directions specified in the corresponding subdictionary Coeffs

Method coefficients:

Property	Description	Required	Default
n	(nx ny nz)	yes	
order	order of operation (unused)	no	xyz
delta	delta (jitter) for rotation matrix	no	0.001
transform	cartesian coordinate transformation	no	

delta is the cell skew factor, default value is 0.001.

2. Hierarchical Method:

Works by first sorting the points in x direction into equal sized bins, then in y direction and finally in z direction. Uses single array to hold decomposition which is indexed as if it is a 3 dimensional array: finalDecomp[i,j,k] is indexed as i*n[0]*n[1] + j*n[1] + k. User can specify the order in which the decomposition is done.

Method coefficients:

Property	Description	Required	Default
n	(nx ny nz)	yes	
order	order of operation	no	xyz
delta	delta (jitter) for rotation matrix	no	0.001
transform	cartesian coordinate transformation	no	

Note: This can affect the no. of iteration required for convergence for an iterative solver.

Decomposition Options

3. KaHIP (Karlsruhe High Quality Partitioning) Method:

Multilevel graph partitioning programs: the graph partitioning problem asks for a division of a graph's node set into k equally sized blocks such that the number of edges that run between the blocks is minimized (minimize parallel communication cost).

```
numberOfSubdomains   N;
method               kahip;

kahipCoeffs
{
    config          fast;
    imbalance       0.01;
}
```

Method coefficients:

Property	Description	Required	Default
config	fast / eco / strong	no	fast
imbalance	imbalance on cells between domains	no	0.01
seed	initial value for random number generator	no	0

Strong should be used if quality is paramount.
eco should be used if you need a good tradeoff between partition quality and running time.
fast should be used if partitioning speed is your focus.

Specify the percentage of allowed work imbalance between processors

Specify an <int> for the random number generator

Decomposition Options

4. Metis Method (multilevel graph partitioning):

Decomposes the domain using the METIS algorithm, which tries to minimize the communication between processors. It is possible to specify weights for the different processors if the parallelised system is composed by machines with different performances.
If you want METIS, you must download the sources and unpack them in $WM_THIRD_PARTY_DIR, and run ./makeMETIS.

```
numberOfSubdomains   N;
method               metis;

metisCoeffs
{
    method           recursive;  // k-way
    options          ( ...);
    processorWeights ( ... );
}
```

```
metisCoeffs
{
    processorWeights
    (
        1
        1
        1
    );
}
```

A bisection method cuts the (hyper)graph in two, and recursive bisection repeatedly applies this strategy until the desired number of cuts have been made. Direct partitioning on the other hand tries to immediately divide up the graph

Method coefficients:

Property	Description	Required	Default
`method`	recursive / k-way	no	recursive
`options`	metis options	no	
`processorWeights`	list of weighting per partition	no	

Decomposition Options

5. Scotch Method (multilevel graph partitioning):

SCOTCH is automatically provided and build with OpenFOAM.
When run in parallel, will collect the whole graph on to the master, decompose and send back

```
scotchCoeffs
{
    //processorWeights ( 1 1 1 1 );
    //writeGraph  true;
    //strategy "b";
}
```

- **processorWeights**: List of weighting factors for allocation of cells to processors. The weights are normalized so they can be any range of values, optional entry
- **writeGraph**: <true>, writing out a file *.grf to be used as input for gpart command to obtain the decomposition strategy. (see https://www.openfoam.com/documentation/guides/v2112/api/classFoam_1_1scotchDecomp.html#details), optional entry
- **strategy**: Decomposition strategy, optional entry and complex,

Decomposition Options

6. Multi-Region Decomposition

When running multi-region simulations, it may be desirable to use different decomposition methods for one or more regions. For example, if the multi-region simulation contains a large region A and a very small region B, it can be advantageous to decompose region B onto fewer processors.

```
numberOfSubdomains  2048;
method  metis;

regions
{
    heater
    {
        numberOfSubdomains  2;
        method  hierarchical;
        coeffs
        {
            n   (2 1 1);
        }
    }

    "*.solid"
    {
        numberOfSubdomains  16;
        method  scotch;
    }
}
```

Required Entry, total number of subdomains in whole mesh

Optional Entry, required if value differs from total number of subdomain in whole mesh

Optional Entry, required if value differs from total number of subdomain in whole mesh

Decomposition Options

7. Multi-level Decomposition

A general way of decomposing at different level with different methods. For example:

```
numberOfSubdomains  2048;
method  multiLevel;

multiLevelCoeffs
{
    nodes
    {
        numberOfSubdomains 128;
        method  hierarchical;
        coeffs
        {
            n   (16 4 2);
        }
    }
    cpus
    {
        numberOfSubdomains 2;
        method  scotch;
    }
    cores
    {
        numberOfSubdomains 8;
        method  metis;
    }
}
```

2048 = 128 nodes x 2 cpus x 8 cores

Using different decomposition options
at different level

Science and Technology Facilities Council

Hartree Centre

OpenFOAM Parallel Input/Output

OpenFOAM Parallel Input/Output

Single or Multiple Processors Output file

- For standard input/output completion, one set of subdirectories for each processor is generated in the case directory. While this file structure is well organised for large parallel cases, it generates a large number of files.
- An alternative file format is the ‘collated’ approach, where the data of each decomposed field is collated into a single file that is written and read on the master processor.
- The control for file handling is specified in the OptimizationSwitches of the Case_Name/system/controlDict

```
OptimisationSwitches
{
    //- Parallel IO file handler
    //  uncollated (default), collated or masterUncollated
    fileHandler uncollated;
}
```

```
//- collated: thread buffer size for queued file writes.
//  If set to 0 or not sufficient for the file size, threading is not used.
//  Default: 1e9
maxThreadFileBufferSize 0;

//- masterUncollated: non-blocking buffer size.
//  If the file exceeds this buffer size scheduled transfer is used.
//  Default: 1e9
maxMasterFileBufferSize 1e9;
```

uncollated : This is the normal case where in parallel every processor writes its own processor directory.
masterUncollated : Special version of uncollated that performs all I/O on the master, and therefore does not require NFS (Network File System).
collated : For each output time, a single field file is assembled (single processor directory)

Multi-threading should be activated to avoid bottleneck in the processor writing out the data. So writing data can be assigned to some and not all threads.

OpenFOAM Parallel Input/Output

Parallel OpenFOAM runs on distributed storage

In that instance, the root path of the case directory may differ between machines. The path must then be specified in the decomposeParDict using the **distributed** and **roots** keyword. The distributed entry should read:

```
distributed  yes;
```

and the roots entry is a list of root paths, <root0>, <root1>, ..., for each node

```
roots
<nRoots>
(
   "<root0>"
   "<root1>"
   ...
);
```

where <nRoots> is the number of roots.

- Each of the processor directories should be placed in the case directories at each of the root paths specified in the decomposeParDict. Both the system and constant directories must also be present in each case directory.
- For example, a case running with 4 processors will have data of processor 0 on root0, data of processor 1 on root1 and etc..
- To perform reconstructPar, one will need to transfer all the data from different root directories back to master processor directory.

OpenFOAM Parallel Input/Output

Data Format

- Choice of data format can affect the disk space consumption and the speed of writing out data which affects actual simulation time.
- Output options are specified in the /system/controlDict
- writeFormat <ascii/binary>
- writeCompression <off/on>; (should be on for very large parallel case)
- writePrecision <int>
- writeControl

Controls the timing of write output to file.

- `timeStep`: Writes data every `writeInterval` time steps.
- `runTime`: Writes data every `writeInterval` seconds of simulated time.
- `adjustableRunTime`: Writes data every `writeInterval` seconds of simulated time, adjusting the time steps to coincide with the `writeInterval` if necessary — used in cases with automatic time step adjustment.
- `cpuTime`: Writes data every `writeInterval` seconds of CPU time.
- `clockTime`: Writes data out every `writeInterval` seconds of real time.

Science and Technology Facilities Council

Hartree Centre

Hands On Session

Hands On Session

Job submission on Archer 2 for Parallel Runs using sbatch scripts

Submit job: sbatch file_name.sh,

Check status of job: squeue –u username

```
#!/bin/bash
#SBATCH --job-name=MultiParallelOnCompute
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=128
#SBATCH --cpus-per-task=1
#SBATCH --time=12:00:00
#SBATCH --account=acct_name
#SBATCH --partition=standard
#SBATCH --qos=standard
#SBATCH --export=none

module load PrgEnv-gnu
module load gsl
export FOAM_INSTALL_PATH="/work/c01/c01/yiyuntan/OpenFOAM/OpenFOAM-v2212"
# source OpenFOAM bash
source ${FOAM_INSTALL_PATH}/etc/bashrc
export FOAM_USER_APPBIN=/work/c01/c01/yiyuntan/OpenFOAM/yiyuntan-v2212/platforms/linux64CrayDPInt32Opt/bin
export FOAM_USER_LIBBIN=/work/c01/c01/yiyuntan/OpenFOAM/yiyuntan-v2212/platforms/linux64CrayDPInt32Opt/lib
blockMesh
setFields
decomposePar
srun --nodes=1 --ntasks=128 --tasks-per-node=128 --exact --mem=80000M --cpu_bind=cores interFoam -parallel > log.interFoam
```

partition = <standard/highmem/serial>
qos = <standard/highmem/taskfarm/short/long/largescale/lowpriority/serial/reservation>

Module PrgEnv-gnu is necessary for openfoam installation, default environment used, gnu compiler
Module gsl is only needed when one want to use waveFoam solver from waves2Foam

Necessary step to source the OpenFOAM environment to use commands. Alternatively, It is possible to load installed version of openfoam on archer 2 from module load.

https://docs.archer2.ac.uk/user-guide/scheduler/

Hands On Session

Job submission on Cloud System for Parallel Runs

Applications -> Software -> OpenFOAM Sandbox

```
#source OpenFOAM bash
export FOAM_INSTALL_PATH="/mnt/ceph-training/course_materials/OpenFOAM/OpenFOAM-v2212"
source ${FOAM_INSTALL_PATH}/etc/bashrc
#copy openfoam tutorial to your home directory
cd $HOME
cp -r ${FOAM_INSTALL_PATH}/tutorials/multiphase/interFoam/damBreak .
cp -r ${FOAM_INSTALL_PATH}/tutorials/multiphase/interFoam/damBreakWithObstacle .
#set up and perform parallel run on the damBreak case
cd damBreak/damBreak
cp -r 0.orig 0
blockMesh
setFields
decomposePar or mpirun -np 4 redistributePar -decompose -parallel
mpirun -np 4 interFoam -parallel
touch fluid.foam #use decomposed files to visualize, not necessary to reconstruct and get VTK
reconstructPar or mpirun -np 4 redistributePar -reconstruct -parallel
foamToVTK
```

No. of processors used must be same as specified in decomposeParDict

Hands On Session

Modifying decomposeParDict for Parallel Runs

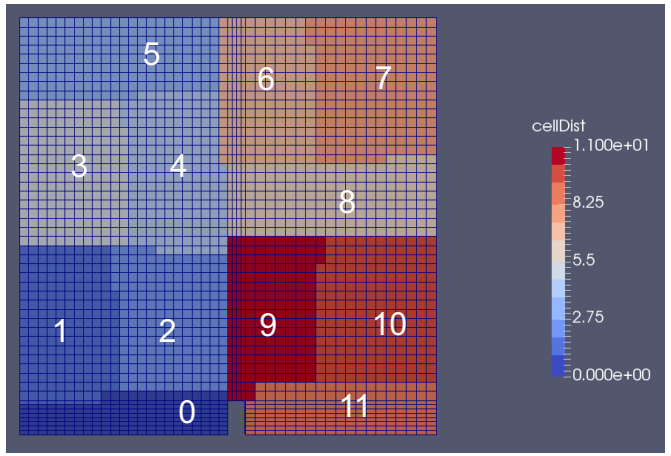
Modify decomposeParDict to use hierarchical method

```
/*--------------------------------*- C++ -*----------
| =========                 |
| \\      /  F ield         | OpenFOAM: The Open Source C
|  \\    /   O peration     | Version:  v2212
|   \\  /    A nd           | Website:  www.openfoam.com
|    \\/     M anipulation  |
\*---------------------------------------------------
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      decomposeParDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

numberOfSubdomains 12;

//method          simple;
method          hierarchical;

coeffs
{
    n           (4 3 1);
    order       xyz;
}


// ************************************************************
```

- Repeat the same steps as in previous slide, decompose the mesh
- ./Allclean, blockMesh, decomposePar
- Visualizing the decomposed mesh? decomposePar –cellDist
- cellDist now appears in the 0 directory, touch fluid.foam, view in paraview
- Interpreting the quality of the decomposition?

```
Number of processor faces = 245
Max number of cells = 189 (0% above average 189)
Max number of processor patches = 6 (63.6364% above average 3.66667)
Max number of faces between processors = 58 (42.0408% above average 40.8333)
```

- (0% above average 189), no work load imbalance
- Max number of processor patches, lesser processor patches, lower communication workload
- Lower max no. of faces between processor, lower communication imbalance.



Science and Technology Facilities Council

Hartree Centre

Hands On Session

Interpreting the quality of the decomposition

- Method simple, n (4 3 1)

```
Number of processor faces = 237
Max number of cells = 204 (7.93651% above average 189)
Max number of processor patches = 6 (89.4737% above average 3.16667)
Max number of faces between processors = 58 (46.8354% above average 39.5)
```

- Method hierarchical, n (4 3 1) , order yxz

```
Number of processor faces = 249
Max number of cells = 189 (0% above average 189)
Max number of processor patches = 6 (71.4286% above average 3.5)
Max number of faces between processors = 60 (44.5783% above average 41.5)
```

- Method hierarchical, n (4 3 1) , order xyz

```
Number of processor faces = 245
Max number of cells = 189 (0% above average 189)
Max number of processor patches = 6 (63.6364% above average 3.66667)
Max number of faces between processors = 58 (42.0408% above average 40.8333)
```

- Method scotch

```
Number of processor faces = 251
Max number of cells = 190 (0.529101% above average 189)
Max number of processor patches = 6 (56.5217% above average 3.83333)
Max number of faces between processors = 60 (43.4263% above average 41.8333)
```

Key Takeaway

Ideal to have equal workload on each processor, minimise the mesh interface between processors

Hands On Session

collated, masterUncollated, uncollated file output

Open up the controlDict of the case found in damBreak/system/
Add this into the controlDict:

```
OptimisationSwitches
{
fileHandler collated;
maxThreadFileBufferSize 1e9;
//fileHandler uncollated;
//fileHandler masterUncollated;
}

functions
{
    #sinclude   "sampling"
}
```

```
cp -r 0.orig 0
blockMesh
decomposePar or mpirun -np 12 redistributePar -decompose -parallel
mpirun -np 12 interFoam -parallel
```

you will notice that you will have only one processor folder call processorN

Hands On Session

Dynamically adapting meshes

Using Adaptive Mesh refinement at different region of the flow based on some criteria will lead to spatially varying mesh at each time step. This implies that the original domain decomposition might not be optimal at different time steps since the number of mesh cells for each processor will vary at each time step.

cd $HOME/damBreakWithObstacle/constant

Open up the dynamicMeshDict

How to overcome this problem?

Work load on each processor is recomputed every n timesteps

Perform re-decomposition of whole domain every n timesteps if work load is severely imbalance

Dynamic Load Balancing

ParMETIS is a parallel library for graph partitioning (for static load balancing) and repartitioning (for dynamic load balancing) developed at the University of Minnesota

```
dynamicFvMesh   dynamicRefineFvMesh;

// How often to refine
refineInterval  1;

// Field to be refinement on
field           alpha.water;

// Refine field inbetween lower..upper
lowerRefineLevel 0.001;
upperRefineLevel 0.999;

// If value < unrefineLevel unrefine
unrefineLevel   10;

// Have slower than 2:1 refinement
nBufferLayers   1;

// Refine cells only up to maxRefinement levels
maxRefinement   2;

// Stop refinement if maxCells reached
maxCells        200000;

// Flux field and corresponding velocity field. Fluxes on changed
// faces get recalculated by interpolating the velocity. Use 'none'
// on surfaceScalarFields that do not need to be reinterpolated.
correctFluxes
(
    (phi none)
    (nHatf none)
    (rhoPhi none)
    (alphaPhi0.water none)
    (ghf none)
    (alphaPhiUn none)
);

// Write the refinement level as a volScalarField
dumpLevel       true;
```

For $0.001 < \alpha < 0.999$, mesh will refine adaptively in space and time.

```
Courant Number mean: 0.000321848 max: 0.123196
Interface Courant Number mean: 0 max: 0
deltaT = 0.00144796
Time = 0.00262443

PIMPLE: iteration 1
Selected 5638 cells for refinement out of 42147.
Refined from 42147 to 81613 cells.
Selected 0 split points out of a possible 5638.
GAMG:  Solving for pcorr, Initial residual = 1, Final
time step continuity errors : sum local = 2.91633e-09
MULES: Solving for alpha.water
Phase-1 volume fraction = 0.0848214  Min(alpha.water)
MULES: Solving for alpha.water
Phase-1 volume fraction = 0.0848214  Min(alpha.water)
MULES: Solving for alpha.water
Phase-1 volume fraction = 0.0848214  Min(alpha.water)
GAMG:  Solving for p_rgh, Initial residual = 0.006229
time step continuity errors : sum local = 8.60189e-05
GAMG:  Solving for p_rgh, Initial residual = 7.29103e
time step continuity errors : sum local = 6.79905e-07
GAMG:  Solving for p_rgh, Initial residual = 1.82914e
time step continuity errors : sum local = 8.84097e-09
ExecutionTime = 1 s  ClockTime = 1 s
```

UKRI Science and Technology Facilities Council

Hartree Centre

Questions?

References

OpenFOAM: User Guide v2112
https://www.openfoam.com/documentation/guides/v2112/doc/openfoam-guide-parallel.html

Handling Parallelisation in OpenFOAM, Hrvoje Kasak
https://zhulianhua.github.io/documents/HandlingParallelisationOpenFOAM.pdf

KaHIP v0.6 – Karlsruhe High Quality Partitioning User Guide, Peter Sanders and Christian Schulz
http://algo2.iti.kit.edu/schulz/software_releases/kahipv0.6.pdf

Archer 2 User Guide, https://docs.archer2.ac.uk/user-guide/scheduler/

https://www.openfoam.com/news/main-news/openfoam-v1712/parallel

https://openfoamwiki.net/index.php/DecomposePar

Science and Technology Facilities Council

Hartree Centre

Thank you

hartree.stfc.ac.uk

@HartreeCentre

STFC Hartree Centre

yiyun.tan@stfc.ac.uk