Moving beyond the single node with OpenFOAM & openMPI

Author: Bob Beattie
Rev: 0.2
Date: 27th Feb 2020

Quick background:

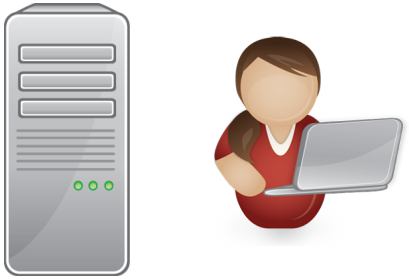
1980’ Introduced to BASIC.
1983’ Introduced to 6502 assembler.
1987’ Introduced to Z80 & 680x0 assembler.
1989’ Frontier Software - Amiga & Atari memory systems.
1990’ Harrogate College of Arts & Techn - Audio/Visual techn
1994’ Solid State Logic - Assembly & test of digital editing suites.
1997’ Nokia R&D - Lots of cool projects with lots of cool people !
2012’ The big Nokia breakup, started Southern Assemblies Ltd
2017’ Tricis Ltd - R&D hw/sw development
2019’ Taking a break.

Investigated using OpenFOAM in 2010 at Nokia as part of my ‘simulate it before you build it’ approach.

Tricis products with forced air cooling had air-flow path simulated to optimise baffles / diverters and remove circulating hot-spots.

The single node

$ <*solver*>



$ mpirun -np 4 <*solver*> -parallel

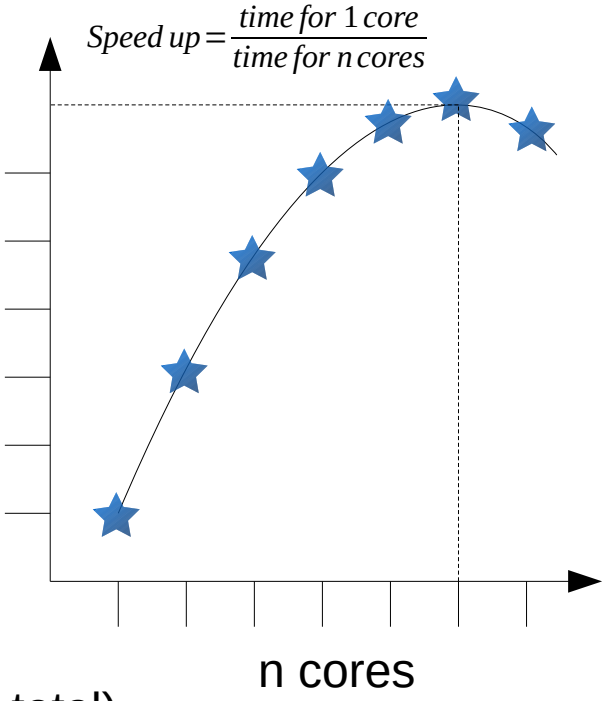
The single node

Determine the most efficient number of cores to use:

```
for n in {1..max_CPU};
 do
 { time mpirun -np ${n} <solver> -parallel ;} >> time.solver
done
```

1, 2, 3, … … #cores

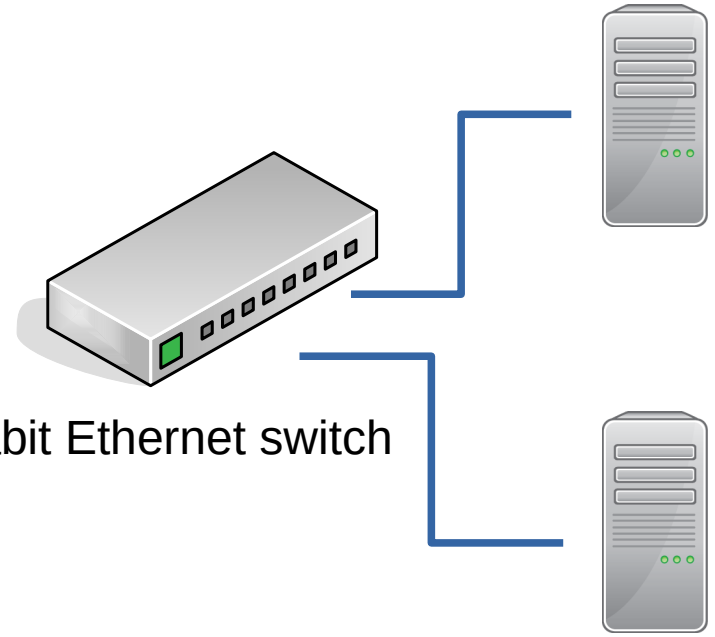
Speed up



HP ProLiant DL380 G6
2x Xeon X5650 6 core (12 total)
48GB RAM in 12x4GB 2R PC3R-10600(2 DIMMS per RAM channel)
Best performance @ 10 cores
Worse performance > 10 cores

Adding the next node

1. Keep the starting setup as simple as possible.
2. Ensure the added machine has the same user, login, OS and OpenFOAM version.
3. Assign static IP addresses to both nodes via netplan or, simpler, your local DHCP server.
4. Connect them via an Ethernet switch, Gigabit recommended.



Node 1 with static IP 192.168.1.11
User = foam
Pass = cfd

Node 2 with static IP 192.168.1.12
User = foam
Pass = cfd

Note: Node2 only needs to be a headless install as a GUI is not required.
Install OpenFOAM with “apt-get install --no-install-recommends ...” to prevent installation of the recommended ParaView and 3rd Party addons.

Setup password-less SSH

Each node needs to have password-less SSH login will all other nodes in the cluster.

Create ssh keys on node 1 and copy to node2:
$ ssh-keygen -t rsa -b 2048
$ sudo ssh-copy-id -i foam@192.168.1.12

Confirm we can log into node 2:
$ ssh foam@192.168.1.12

Create ssh keys on node 2 and copy to node1:
$ ssh-keygen -t rsa -b 2048
$ sudo ssh-copy-id -i foam@192.168.1.11

Confirm we can log in to node 1:
$ssh foam@192.168.1.11

$exit → (back to node2) exit → (back to node1)

Setup machine hostnames & hosts

In node 1:

edit /etc/hostname and name the machine “foam1” or similar.

edit /etc/hosts and add an entry for each machine in the cluster:

```
127.0.0.1       localhost
192.168.1.11    foam1
192.168.1.12    foam2
```

In node 2:

edit /etc/hostname and name the machine “foam2”

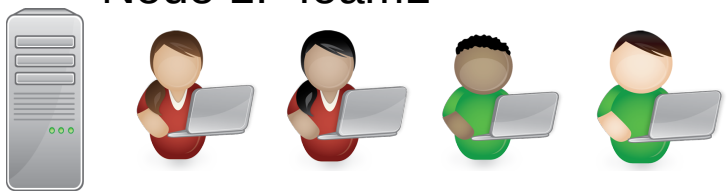
edit /etc/hosts to be the same as node 1.

Reboot both machines to get to a known default configuration.

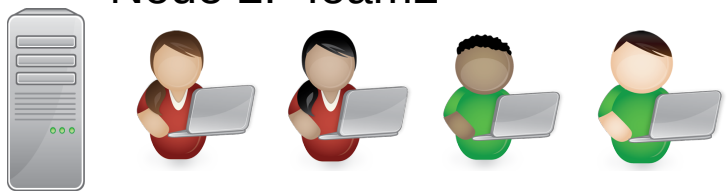
Changes to the case file

In this example setup we have two nodes, each with four processor cores.

Node 1: “foam1”



Node 2: “foam2”



We need to provide a machine capability text file to openMPI so it understands how it should share the parallel tasks between the various nodes.
The format of the file takes the following scheme:

<hostname> slots = <ideal number of processes> max_slots = <maximum processes>

Our example setup would then be:

foam1 slots = 4 max_slots = 4
foam2 slots = 4 max_slots = 4

Across the two machines we have a total of eight processor cores.
(‘slots’ is chosen from the number of processors that make the most efficient use of the computer - see slide 4)
I call my machine capability file “machines” and keep it under system/ in the case.

decomposeParDict (hierarchical)

numberOfSubdomains and n(x y z) should reflect the total processor cores assigned.
Performing decomposePar will result in the usual set of processor* dirs.

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  5                                     |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      decomposeParDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

numberOfSubdomains 8;
method          hierarchical;
hierarchicalCoeffs
{
    n           (8 1 1);
    delta       0.001;
    order       xyz;
}

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

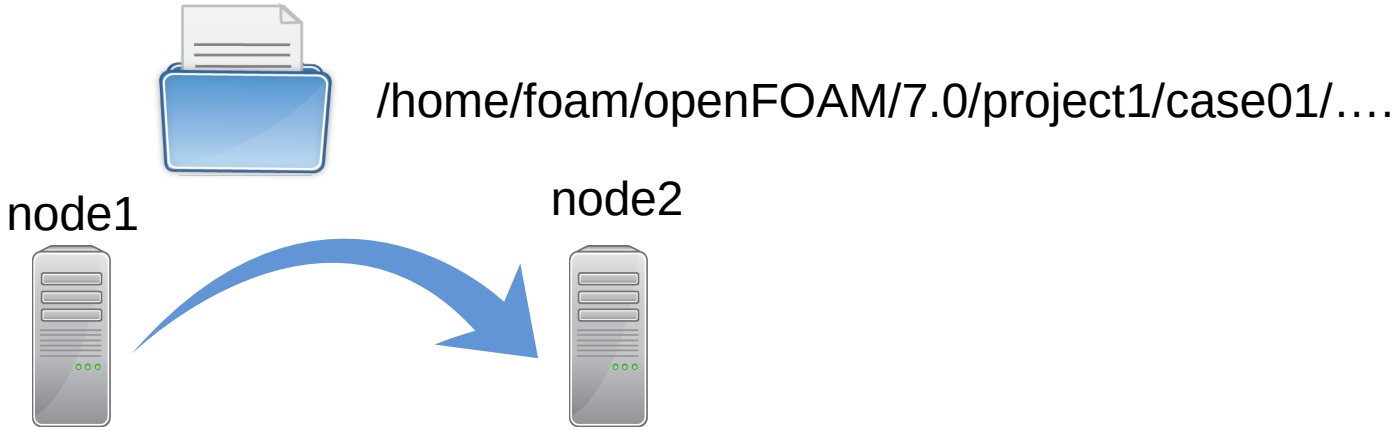
distribute the processor dirs

In our example setup, directories “processor0” through “processor7” will be created from decomposePar; 0→3 for node1 and 4→7 for node2.

processor4..7 dirs can be copied to node2 with scp, but the directory structure must remain the same:

```
casepath=$(pwd)
remote='foam2'
mpirun -H ${remote} mkdir -p ${casepath}
scp -r * foam@${remote}:${casepath}
```

1. Creates an exact copy of the current working directory path on the remote machine.
2. Copies the complete case file to the remote machine. (not suitable for large designs)



issue openMPI to remote execute

Execute snappyHexMesh with 8 processes:

```
mpirun -np 8 --mca btl_tcp_if_exclude lo,virbr0 \
       -hostfile system/machines snappyHexMesh -overwrite -parallel
```

The 8 processes will be shared between machines based on the entries in the system/machines file.

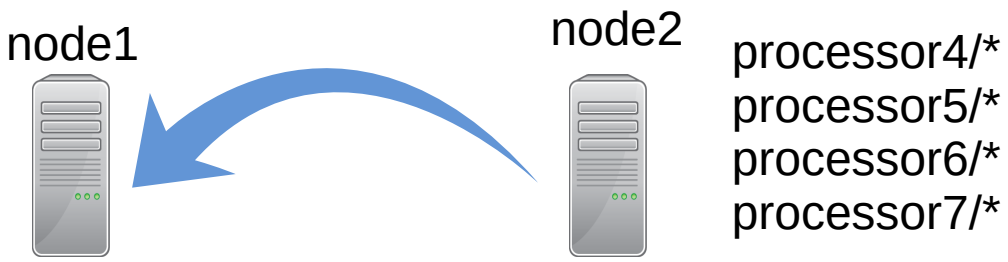
NOTE:
If openMPI fails to run, it may be because the top of ~/.bashrc contains a check if the shell is running interactively. These SSH logins are not interactive so the script would exit at that point, thus not executing the OpenFOAM setup script.
In this case, ". /opt/openfoam7/etc/bashrc" needs to be moved from the end of ~/.bashrc to the top before the interactive test:

```
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

. /opt/openfoam7/etc/bashrc

# If not running interactively, don't do anything
case $- in
    *i*) ;;
      *) return;;
esac
```

post operation reconstruct

Once openMPI has completed the assigned task, the results for the remote node need to be copied back.

```
casepatch=$(pwd)
remote='foam2'
for i in {4..7};
  do
   scp -r foam@${remote}:${casepath}/processor${i} .
done
```



...and reconstruct the separate processor* directories:

```
reconstructParMesh -constant && rm -r processor*
```

View the mesh with ParaView.

a 40 core example case

https://drive.google.com/file/d/1mz78anNSo9cYZ0uAeLjdb3CpEfbbylxa/view?usp=sharing

NOTE:
The scripts may not be the best example of how to do things.