ORACLE

CFD Ready Cluster Runbook

How to run CFD Software on OCI

May 07, 2020 | Version 1.00

Confidential: Public

TABLE OF CONTENTS

INTRODUCTION

This runbook has been developed for customers who want to run Computational Fluid Dynamics (CFD) workloads using HPC RDMA Cluster Networking on OCI. It provides the steps to deploy a CFD ready cluster through Marketplace with GlusterFS / NFS file systems and OpenFOAM CFD software installed. The CFD cluster also installs prerequisite libraries for other commonly used commercial CFD applications, including Star-CCM+, Ansys Fluent and Converge. This guide will walk through setup of those applications on OCI HPC Cluster Networking infrastructure.

CFD READY CLUSTER

A CFD Ready Cluster stack is available on Marketplace for easy deployment of CFD libraries on top of OCI HPC Cluster Networking. This stack currently supports Oracle Linux OS and uses the HPC CN image to launch the cluster.

When the Cluster is provisioned, prerequisite libraries for some of the most commonly used CFD applications are also installed, including StarCCM, Ansys Fluent and Converge. Setup of these applications requires only configuration of the main commercial software and license. In addition to prerequisite libraries, the cluster is also installed with a GlusterFS file system for high data throughput and OpenFOAM-7 CFD software in /mnt/gluster-share. See the sections below to setup the cluster through Marketplace and run through examples of each CFD application on our OCI HPC Cluster Network.

CFD CLUSTER ARCHITECTURE

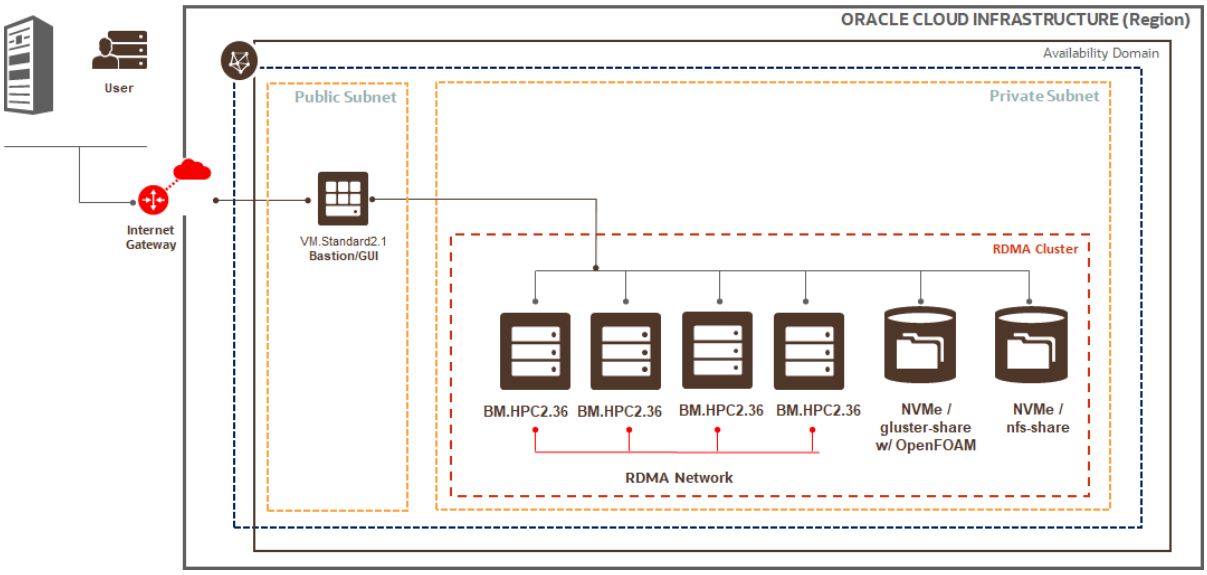
The CFD Ready Cluster solution in Marketplace deploys the following base architecture:

- VCN
  - Public Subnet, Security List, Route Table
  - Private Subnet, Security List, Route Table
  - Internet Gateway
  - NAT Gateway
- Compute Nodes
  - Bastion Host in a Public Subnet
  - HPC Compute Nodes in Private Subnet

A bastion is provisioned in the public subnet and acts as the login node. It is accessible through SSH by means of the private ssh key used to launch the infrastructure. The HPC compute nodes are on the private network and interconnected using a RDMA RoCE v2. Compute nodes are only accessible through the bastion inside the network.

On top of the base architecture, NFS share and GlusterFS file systems are installed to provide file storage across all nodes. If a different file system is desired, Ansible is available on the cluster to easily deploy new fileservers through playbooks. Within the GlusterFS file share, OpenFOAM-7 software is provided to run CFD simulations.

For visualization, vncserver and GNOME desktop can be installed directly on the bastion. This guide will cover installation of these tools and connection via a VNC server.



LAUNCH CFD READY CLUSTER

The CFD Ready Cluster stack is available in Marketplace and can be deployed easily using the following steps:

1. Within marketplace, select **Get App** at the top right.
2. Select the OCI Region then click **Sign In**.
3. Verify the version of the HPC Cluster image and then select the **Compartment** where the cluster will be launched. Accept the terms and conditions, then Launch Stack.
4. Fill out the remaing details of the stack:
    a. Select the desired AD for the compute shapes and the bastion.
    b. Copy-paste your public ssh key
    c. Type in the number of Compute instances for the cluster
5. Click **Create**.
6. Navigate to **Terraform Actions** then click **Apply**. This will launch the CN provisioning.
7. Wait until the job shows 'Succeeded' then navigate to **Outputs** to obtain the bastion and compute node private IP's.

ACCESS THE CLUSTER

There are a couple ways to access the cluster. You can either ssh into the bastion host by using the private key associated with the cluster network and the public IP for the bastion. If you want to access the cluster using VNC, you can also do so either by ssh tunneling or by opening the VNC listening port in the public subnet security list.

SSH Into the Bastion

To ssh into the bastion, specify the public IP of the bastion and use the default username opc (for Oracle Linux instances) in the below command. Ensure you are using the private key associated with the public key that was used to launch the cluster network.

```
ssh -i /home/user/.ssh/<private key> opc@ipaddress
```

Login using VNC

By default, the only access to the Oracle Linux machine is through SSH in a console mode. If you want to see the graphical interface, you will need to set up a VNC connection. The following script can be executed on the bastion for the default user opc.

1. Install the GNOME GUI and tigervnc server via the yum repository:

```
sudo yum -y groupinstall "Server with GUI"
sudo yum -y install tigervnc-server mesa-libGL
```

2. Set the graphical target to default:

```
sudo systemctl set-default graphical.target
```

3. Configure the vncserver service:

```
sudo cp /usr/lib/systemd/system/vncserver@.service
/etc/systemd/system/vncserver@:1.service


sudo sed -i 's/<USER>/opc/g' /etc/systemd/system/vncserver@:1.service


sudo sed -ie '/^ExecStart=/a PIDFile=/home/opc/.vnc/%H%i.pid'
/etc/systemd/system/vncserver@:1.service


sudo mkdir /home/opc/.vnc/
sudo chown opc:opc /home/opc/.vnc
echo "password" | vncpasswd -f > /home/opc/.vnc/passwd
chown opc:opc /home/opc/.vnc/passwd
chmod 600 /home/opc/.vnc/passwd
```

4. Start up the vnc service and change the vnc password:

```
sudo systemctl start vncserver@:1.service
```

```
sudo systemctl enable vncserver@:1.service

vncpasswd
```

5. You can connect through VNC by using SSH tunneling and TCP port 5900+N, where N is the display number used to setup the vncservice (for example, 5901 for :1, 5902 for :2). On your local machine, create an ssh tunnel using the following command:

   ```
   ssh -L <port number>:127.0.0.1:<port number> opc@<public ip>
   ```

   Example:

   ```
   ssh -L 5901:127.0.0.1:5901 opc@<public ip>
   ```

6. If you want to access the VNC server directly without SSH forwarding, ensure that your security list allows connections on the VNC listening port (e.g., 5901).
   a. In the Console, opennavigate to **Networking** then **Virtual Cloud Networks**.
   b. Select **Subnets** and then the public subnet.
   c. In the default security list, add an Ingress Rule with the following details:
      - Stateless: **No**
      - Source Type: **CIDR**
      - Source CIDR: **0.0.0.0/0**
      - IP Protocol: **TCP**
      - Source Port Range: **All**
      - Destination Port Range: **5900-5901**

7. Allow access in local firewall settings, as follows:

   ```
   sudo firewall-cmd --zone=public --permanent --add-port=5901/tcp

   sudo firewall-cmd --reload
   ```

8. Install your preferred VNC client on your local machine, such as the following:
   Windows - TigerVNC
   MacOS/Windows - RealVNC
   MacOS/Windows - TurboVNC

9. Open the VNC client and enter the IP address connection as <public ip>:1

RUNNING OPENFOAM

OpenFOAM is a C++ CFD software that using various methods to break down models into 3 dimensional meshes and apply engineering theorems or approximations across the meshes. It provides a wide range of solvers for engineering applications, combustion analysis, heat transfer, fluid flow & turbulence, solid mchanics and electromagnetics.

OpenFOAM has a large user base because it is free, open source software and is contributed to extensively by the Acadamia community. Many commercial software applications do not have the range of solvers that OpenFOAM does simply because they are not contributed to by a community of users. Because of its popularity, we distribute OpenFOAM on the CFD Ready Image in Marketplace.

Installation

During the provisioning process, OpenFOAM is installed via the Ansible openfoam.yml playbook within the GlusterFS file system. You can access OpenFOAM at the following path: `/mnt/gluster-share/install/OpenFOAM`.

The installation includes both OpenFOAM-7 and ThirdParty-7 folders. Within OpenFOAM-7, an applications folder is provided that include engineering solvers and utilities for performing tasks that involve data manipulation. There is also a tutorials folder that contains baseline meshes for training and validation. The ThirdParty-7 folder contains all the third-party tools needed to run OpenFOAM including gcc compilers, scotch, cmake and ParaView for post processing.

To run an OpenFOAM simulation, there are several folders that need to be present to describe the initial conditions, execution scripts and mathematical solvers. The following table describes the typical directories within a simulation model. Note that OpenFOAM also allows you to create your own custom directories – read more in the documentation: https://www.openfoam.com/documentation/user-guide/index.php .

COMPONENT	SIMULATION SOFTWARE
Constant Folder	Contains the mesh files that specify the physical properties for the model, such as transport properties in the case of CFD or rheology properties for solid mechanics.
System Folder	Contains scripts that set the parameters for the solver, such as decomposeParDict to break the model into cells and allocate to each processor for parallel processing.
0 Folder	Folder that holds the initial conditions for running the simulation.
Allrun	Run file for a solver using a specific model
All clean	Wipes out all data from previous runs

Operation

In this example, we will use the motorbike tutorial which is available in *`/mnt/gluster-share/work`*.

1. First, log into your bastion host and then into one of the compute nodes.
2. Change drive to the *`/mnt/gluster-share/work`* folder.
3. To run parallel processing, the *`decomposeParDict`* utility is used to specify the following:
   - *`numberOfSubdomains`* for the total number of processes that will be used for the simulation. Typically, the number of processes used in the simulation is equal to the total number of cores available in the cluster.
   - *`hierarchicalCoeffs`* for defining how the model is split (e.g, first in x direction, then y direction then z).

The Allrun file for motorbike has already been modified with switch-case logic to specify the *`numberOfSubdomains`* and *`hierarchicalCoeffs`* parameters. These parameters are set based on the number of processes declared when the Allrun file is executed (up to a 20-node cluster). You can modify the cases in the Allrun file if you prefer a different hierarchal breakdown.

4. Run the Allrun file and the number of processes that will be used in the motorbike simulation. For a 2-node cluster, this would be:

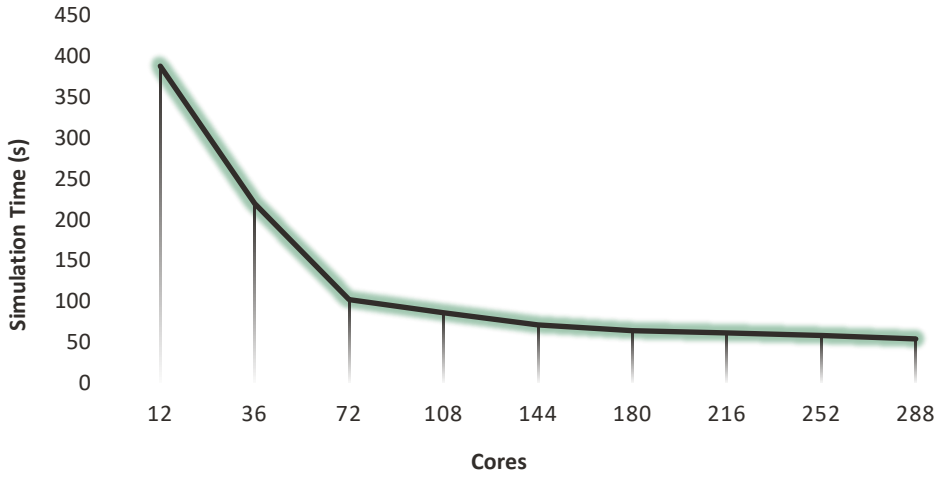
```
./Allrun 72
```

5. The output of the simulation documents the execution time in seconds from log.simpleFoam. You can capture this time and then run a new simulation on a different number of processes to compare the performance.

6. Run the Allclean script in between simulations to clear out the previous data.

```
./Allclean
```

At the end of the run, you can plot the simulation time against the number of cores and determine the optimal number of nodes to run at for the model.

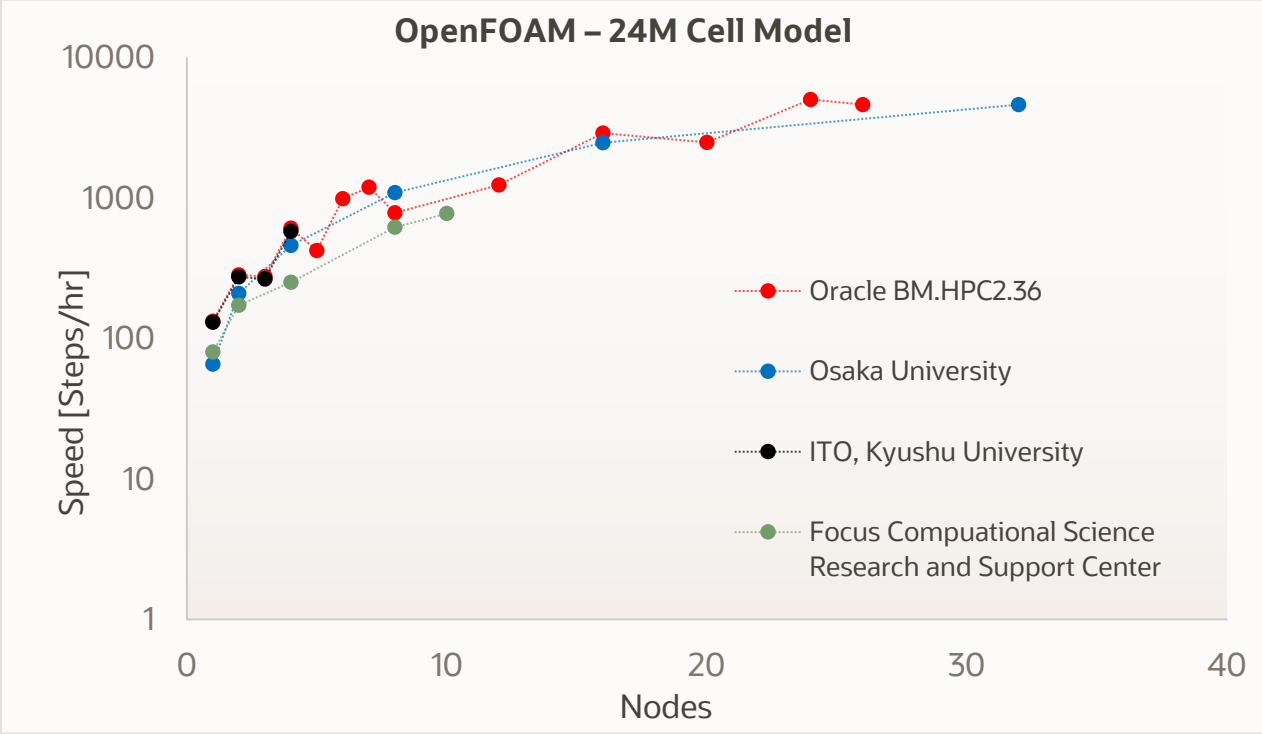


Performance Comparison

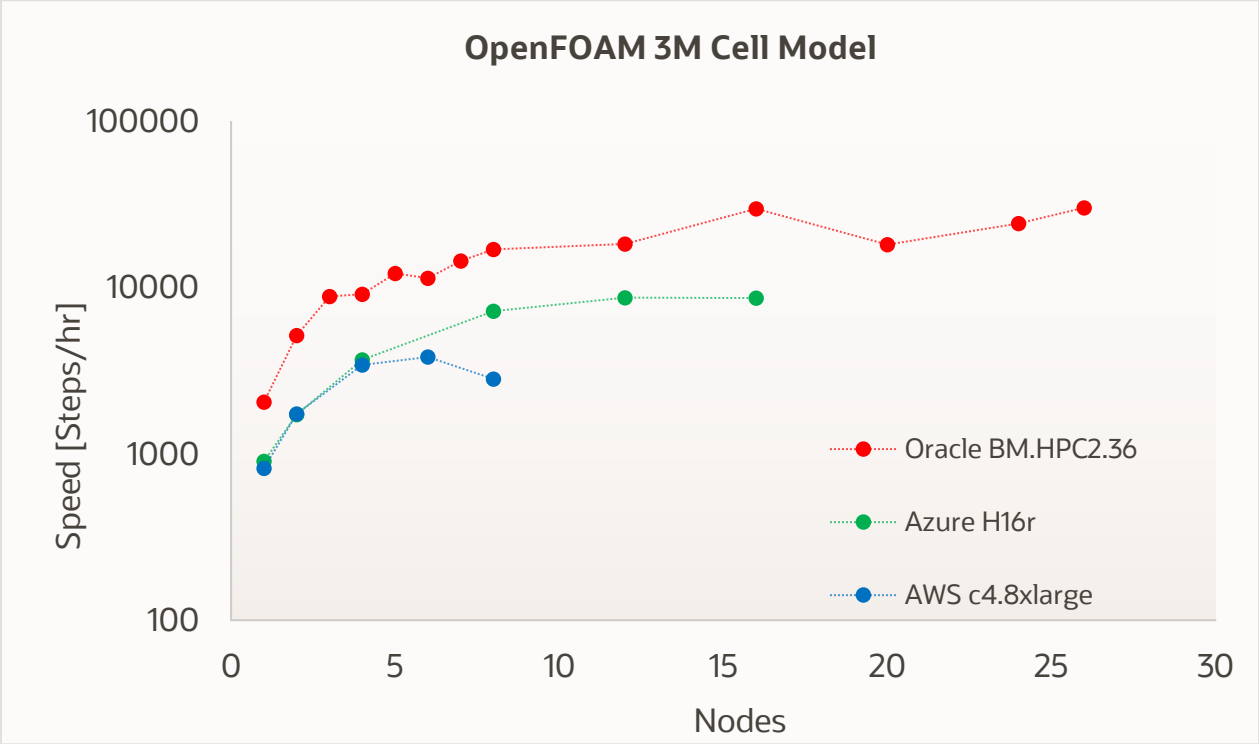
We partnered with the open CAE Society of Japan to conduct an OpenFOAM CFD benchmark test. The benchmark compared our HPC Cluster Network offering against Japan's largest public, on-premis HPC environments and against other HPC cloud providers. A 24M cell Model and 3M cell Model were used in the evaluation, and speedup of OpenFOAM was calculated to analyze the performance of each environment.

The results of the 24M cell benchmark showed that Oracle HPC provides consistent performance with some of Japan's most powerful supercomputers. The infrastructure between ITO and Oracle's HPC shape were identical – both contained the same Intel 6154 processor, same memory per core, etc. As shown in the figure, the performance of ITO (black line) is exactly the same as OCI HPC Cluster Networking. We also had comparable results to Osaka University's on-premises environment.

The key takeaway from this test is that Oracle's list price per BM.HPC2.36 node is $2.70 per hour, versus on-premises environments, which cost millions of dollars to setup and maintain.



The results of the 3M cell Model showed that Oracle HPC Cluster Networking outperformed every other cloud provider. This includes outperforming AWS and Azure HPC shapes.



Full results can be obtained here: Open CAE OpenFOAM Gitlab 3M-20190222 Comparison and Open CAE OpenFOAM Gitlab 24M-20190701 Comparison for more analysis.

RUNNING STAR-CCM+

StarCCM+ is a commercial CFD software that using physics approximations to model fluid flow around engineering models. Like OpenFOAM, it also uses meshes to define the boundary conditions, break the models into manageable pieces, and distributes these pieces across multiple CPU's to achieve more efficient simulations. StarCCM+ is developed by Siemens and is widely used in Automotive, Aerospace and Oil&Gas industries to model fluid flow and combustion.

Installation

The prerequisite libraries needed to run Star-CCM+ on OCI HPC Cluster Networking are installed during the provisioning process as part of the Ansible playbook cfd.yml. These include the following libraries:

- `libSM`
- `libX11`
- `libXext`
- `libXt`

The commercial Star-CCM+ software must be downloaded and configured on the cluster after provisioning. You can download the STAR-CCM+ installer from the Siemens PLM website or push it to your machine using scp:

```
scp /path/own/machine/STAR-CCM_version.zip opc@<public ip>:/mnt/gluster-share
```

Alternatively, you can upload Star-CCM+ into object storage, create a preauthenticated request, and then and download it onto the cluster:

1. Login to the OCI Console. In the main menu, select **Object Storage**.
2. Choose the correct **Region** on the top right.
3. Select the correct **Compartment** on the left-hand side.
4. Create a bucket if you do not have one already created.
5. In the bucket, select upload object and specify the path of the installer.
6. Click the 3 dots on the right-hand side of the installer object and select **Create Pre-Authenticated Request**.
7. If you lose the URL, you cannot get it back, but you can regenerate a new Pre-Authenticated Request (PAR) URL.
8. Download the installer form object storage with:

   ```
   wget PAR_URL
   ```

9. Untar the installer in a shared location, such as /mnt/gluster-share/install or /mnt/nfs-share/install.

   ```
   tar -xf <star-ccm package>.tar.gz
   ```

10. Installing Star-CCM+ is straightforward: You can either start the GUI if you have a VNC session or else run the below command to start the installer:

    ```
    sudo bash /mnt/<file-share path>/install/starccm+_<version>/STAR-
    CCM+<version><platform>.sh
    ```

11. To specify the host you need to run on, you need to create a machinefile. You can generate it using the following command, or manually create it using *hostname:corenumber*.

    ```
    sed 's/$/:36/' /mnt/nfs-share/etc/hostfile > machinefile
    ```

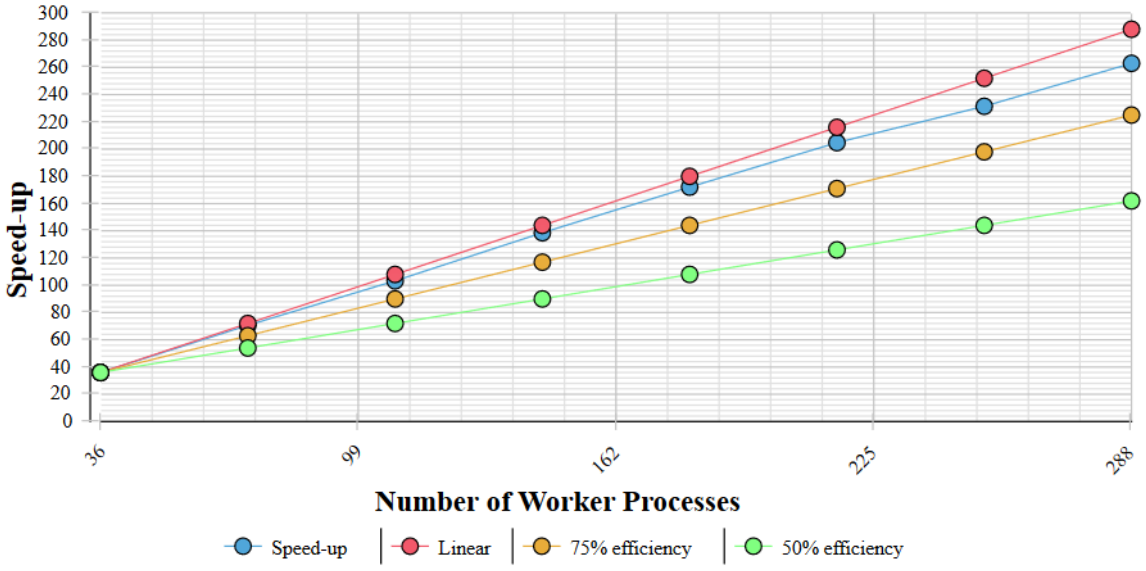
Operation

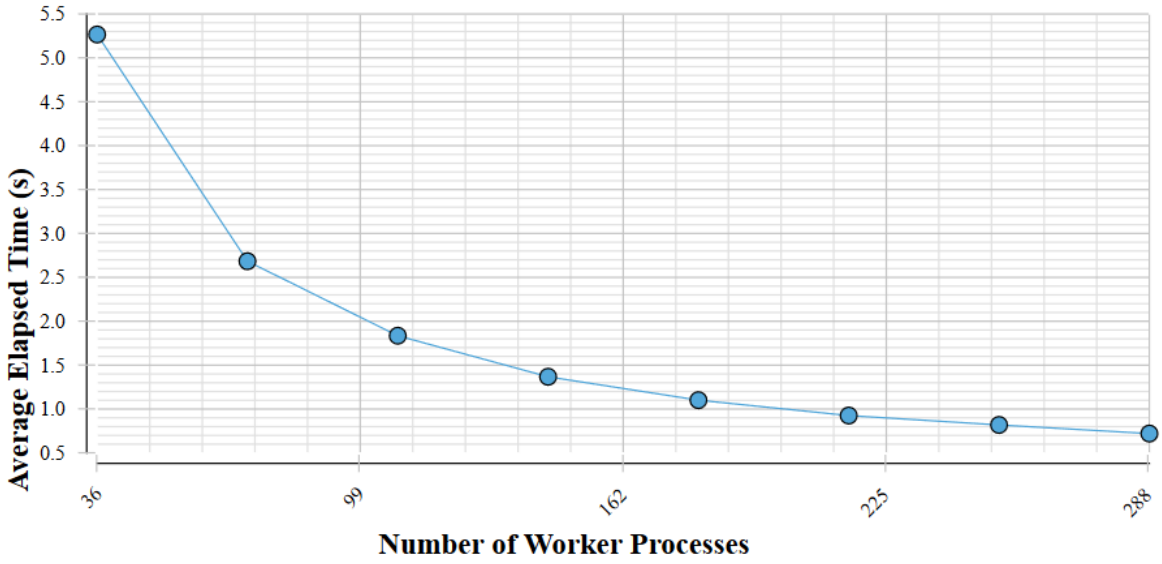
1. To run on multiple nodes, place your model.sim file on the gluster-share drive /mnt/gluster-share or nfs-share drive /mnt/nfs-share.
2. Create the following bash script in the install folder. Be sure to replace values for the following variables in accordance with your environment:
   a. CORES – the number of cores that will be used in the benchmark
   b. benchITS – the number of benchmark tests (e.g, 36,72,108,144,180,216,252,288)
   c. MPINAME – the MPI version that will be used for running the processes in parallel
   d. INSTALLPATH – the install directory of the starccm binary
   e. POD – your Power On Demand Key for running StarCCM
   f. MODELNAME – the model.sim file that you will run in the benchmark
   g. MACHINEFILE – the hostfile pointing to all the compute nodes that will be used in the benchmark

```
#!/bin/bash

#VARIABLES:
export DATE=`date '+%Y%m%d%H%M'`
CORES=<number of cores>
benchITS=<number of benchmark tests>
MPINAME=openmpi3
INSTALLPATH=/mnt/gluster-share/install/<version>/STAR-CCM+<version>/
POD=<pod key>
MODELNAME=/mnt/gluster-share/work/<model.sim>
MACHINEFILE=/mnt/gluster-share/machinefile


#LOG EVENT
echo `date` | tee -a ${MPINAME}.$CORES.${DATE}.log

#RUN SIMULATION
if [ $MPINAME == intel ]; then
echo "running Intel"
$INSTALLPATH/star/bin/starccm+ -v -power -licpath 1999@flex.cd-adapco.com -podkey $POD
-np $CORES -benchmark "-preclear -preits 40 -nits 20 -nps $benchITS" -machinefile
$MACHINEFILE -rsh ssh -mpi $MPINAME -cpubind bandwidth,v -mppflags "-iface enp94s0f0 -
genv I_MPI_DAPL_PROVIDER ofa-v2-cma-roe-enp94s0f0 -genv I_MPI_DAPL_UD 0 -genv
I_MPI_FALLBACK 0 -genv I_MPI_DYNAMIC_CONNECTION 0 -genv I_MPI_FABRICS shm:dapl -genv
I_MPI_DAT_LIBRARY /usr/lib64/libdat2.so -genv I_MPI_DEBUG 6" -load $MODELNAME | tee -a
${MPINAME}.$CORES.${DATE}.log

elif [ $MPINAME == openmpi3 ]; then
echo "running OpenMPI"
$INSTALLPATH/star/bin/starccm+ -v -power -licpath 1999@flex.cd-adapco.com -podkey $POD
-np $CORES -benchmark "-preclear -preits 40 -nits 20 -nps $benchITS" -machinefile
$MACHINEFILE -rsh ssh -mpi $MPINAME -cpubind bandwidth,v -mppflags "--display-map -mca
btl self -mca UCX_TLS rc,self,sm -mca HCOLL_ENABLE_MCAST_ALL 0 -mca coll_hcoll_enable
0 -mca UCX_IB_TRAFFIC_CLASS 105 -mca UCX_IB_GID_INDEX 3 --cpu-set
0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,
32,33,34,35" -load $MODELNAME | tee -a ${MPINAME}.$CORES.${DATE}.log;

elif [ $MPINAME == platform ]; then
echo "running platform"
$INSTALLPATH/star/bin/starccm+ -v -power -licpath 1999@flex.cd-adapco.com -podkey $POD
-np $CORES -nosuite -machinefile $MACHINEFILE -rsh ssh -mpi $MPINAME -mppflags "-
intra=shm -e MPI_HASIC_UDAPL=ofa-v2-cma-roe-enp94s0f0 -UDAPL -
aff=automatic:bandwidth:core -affopt=v -prot" -benchmark "-preclear -preits 40 -nits
20 -nps $benchITS" $MODELNAME | tee -a ${MPINAME}.$CORES.${DATE}.log

fi

#LOG EVENT
echo `date` | tee -a ${MPINAME}.$CORES.${DATE}.log
```

3. Run the bash script created in step 2.
4. Once the benchmark is complete, a benchmark.html file will be created which shows the time to solution, speedup and parallel efficiency at each iteration. You can also determine the optimal number of nodes to run at for the model.

   The following graphs show the the speedup, parallel efficiency and time to solution for a lemans polymesh 17M model using platform mpi.



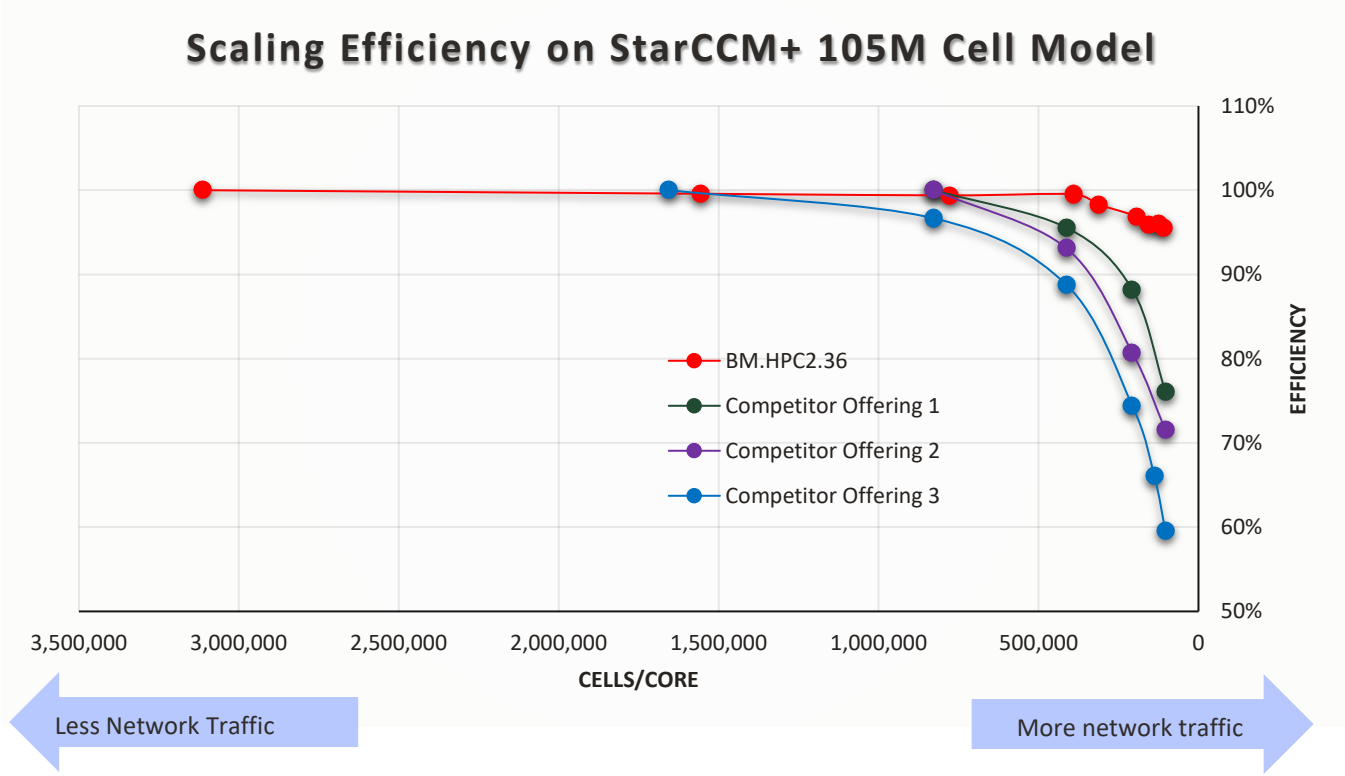


Performance Comparison

When running simulations across an HPC cluster, the ability to efficiently scale at high node counts is important. It guarantees predictability of the simulation and increases the return-on-investment for expensive application licenses.

To evaluate scaling, we ran a StarCCM+ benchmark using LeMans 100M Cell Model on our BM.HPC2.36 HPC Cluster Network and compared with a cloud competitors HPC shape. All environments contained RDMA networking, with Mellanox FDR InfiniBand associated with the competitors' shape.

The results showed that BM.HPC2.36 scales over 100% from 450,000 cells per core to below 6,000 cells per core, consistently, using the same performance that you see with on-premises clusters.



RUNNING ANSYS FLUENT

Ansys Fluent is another commercial CFD software that uses meshes and engineering solvers to model fluid flow around objects. It is popular because of the complex functions and meshing capabilities it provides, as well as the consistency and precision it offers. Ansys Fluent is developed by Ansys, Inc and is widely used in Automotive, Electronic and Industrial industries to model fluid flow.

Installation

The prerequisite libraries needed to run Ansys Fluent on OCI HPC Cluster Networking are installed during the provisioning process as part of the Ansible playbook cfd.yml. These include the following libraries:

- libGLU
- libXrender.x86_64
- libXtst.x86_64
- motif-2.3.4-14.el7_5.x86_64
- mesa-libGLU-9.0.0-4.el7.x86_64
- mesa-libGLU
- mesa-libGL
- motif
- axel
- fontconfig
- freetype
- freetype-devel
- fontconfig-devel
- libXext
- libXrender-devel.x86_64
- libXrender.x86_64
- mesa-libGL.x86_64

The commercial Ansys Fluent software must be downloaded and configured on the cluster after provisioning. You can download the FLUENT installer from the ANSYS website and push it to your machine using scp.

```
scp /path/own/machine/FLUID_version.tar opc@<public ip>:/mnt/gluster-share
```

Alternatively, you can upload Ansys Fluent into object storage, create a preauthenticated request, and then and download it onto the cluster:

1. Login to the OCI Console. In the main menu, select **Object Storage**.
2. Choose the correct **Region** on the top right.
3. Select the correct **Compartment** on the left-hand side.
4. Create a bucket if you do not have one already created.
5. In the bucket, select upload object and specify the path of the installer.
6. Click the 3 dots on the right-hand side of the installer object and select **Create Pre-Authenticated Request**.
7. If you lose the URL, you cannot get it back, but you can regenerate a new Pre-Authenticated Request (PAR) URL.
8. Download the installer form object storage with:

   ```
   wget PAR_URL
   ```

9. Untar the installer in a shared location, such as /mnt/gluster-share/install or /mnt/nfs-share/install.

   ```
   tar -xf <ansys fluent package>.tar.gz
   ```

10. Create a folder called fluent if not already present after untarring the installer.

    ```
    mkdir /mnt/gluster-share/install/fluent
    ```

11. Launch a silent install via the following command, where <license server> is the IP of your license server, 2325 is the ANSYS License Interconnect port, and 1055 is the FlexNET port.

    ```
    sudo bash INSTALL -silent -install_dir "/mnt/gluster-share/install/fluent" -fluent -
    licserverinfo 2325:1055:<license server>
    ```

12. Wait for the install to complete. There is a known problem when running the ANSYS installer that it can hang. When you see `RSS is disabled`, feel free to exit using CTRL-C.

13. If you want to add Ansys Fluent to your environmental variables, run the following commands:

```
echo export PATH=/mnt/gluster-share/install/fluent/v190/fluent/bin/:$PATH | sudo tee -a ~/.bashrc

source ~/.bashrc
```

14. If you have a VNC session installed, you can launch Ansys Fluent by running the following command:

```
/mnt/gluster-share/install/fluent/v190/fluent/bin/fluent
```

15. To specify the host you need to run on, you need to create a machinefile. You can generate it using the following command, or manually create it using *hostname:corenumber*.

```
sed 's/$/:36/' /mnt/nfs-share/etc/hostfile > machinefile
```

Operation

1. To run on multiple nodes, place your model file on the gluster-share drive (*e.g, /mnt/gluster-share/work/*). The following example show setup of an f1 racecar 140M cell model:
   a. Change drives into the work folder.

      ```
      cd /mnt/gluster-share/work/
      ```

   b. Install the model from object storage and untar in the work folder. Example:

      ```
      wget <PAR f1_racecar_140m.tar>  -O - | tar x
      ```

   c. Create a directory in the work folder called 'f1_racecar_140M' and then move the cas_dat folder contents into this newly created folder.

      ```
      mkdir f1_racecar_140m

      mv bench/fluent/v6/f1_racecar_140m/cas_dat/* f1_racecar_140m/
      ```

   d. Gunzip the f1_racecar_140m folder and remove the bench folder.

      ```
      gunzip f1_racecar_140m/*

      rm -rf bench/
      ```

2. You can set the the mpi flags for either Intel MPI or Platform MPI in the mpirun.fl files.
   a. To set your own flags, modify each mpirun.fl file in the fluent folder with your own flags.

      Replace:

      ```
      FS_MPIRUN_FLAGS="$FS_MPIRUN_FLAGS -genv I_MPI_ADJUST_REDUCE 2  -genv
      I_MPI_ADJUST_ALLREDUCE 2 -genv I_MPI_ADJUST_BCAST 1"
      ```

      To:

      ```
      FS_MPIRUN_FLAGS="$FLUENT_INTEL_MPIRUN_FLAGS -genv I_MPI_ADJUST_REDUCE 2 -genv
      I_MPI_ADJUST_ALLREDUCE 2 -genv I_MPI_ADJUST_BCAST 1"
      ```

   b. Export an environmental variable for you flags.

      ```
      echo export FLUENT_INTEL_MPIRUN_FLAGS="<<< your flags >>>"
      ```

      Example using Intel MPI 2018 Flags:

      ```
      echo export FLUENT_INTEL_MPIRUN_FLAGS='"-iface enp94s0f0 -genv
      I_MPI_FABRICS=shm:dapl -genv DAT_OVERRIDE=/etc/dat.conf -genv
      I_MPI_DAT_LIBRARY=/usr/lib64/libdat2.so -genv I_MPI_DAPL_PROVIDER=ofa-v2-cma-
      roe-enp94s0f0 -genv I_MPI_FALLBACK=0 -genv I_MPI_PIN_PROCESSOR_LIST=0-35 -genv
      I_MPI_PROCESSOR_EXCLUDE_LIST=36-71"' | sudo tee -a ~/.bashrc
      ```

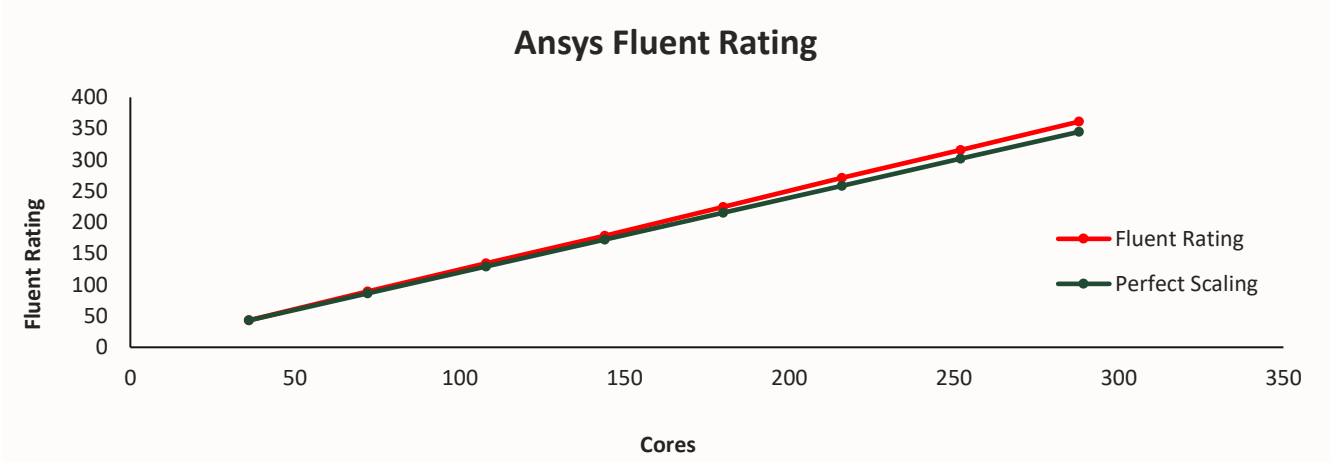
      Refer to the runbook and blog: https://blogs.oracle.com/cloud-infrastructure/running-applications-on-oracle-cloud-using-cluster-networking for more flags and guidance.

3. Within the work folder, set the following variables and run fluent on the desired number of cores. Note that this script uses Intel MPI – it is recommended to use Intel MPI for Ansys Fluent on OCI HPC for best performance

```
modelname=f1_racecar_140m

N=<number of cores>


fluentbench.pl -ssh -noloadchk -casdat=$modelname -t$N -cnf=/mnt/gluster-
share/machinefile -mpi=intel
```

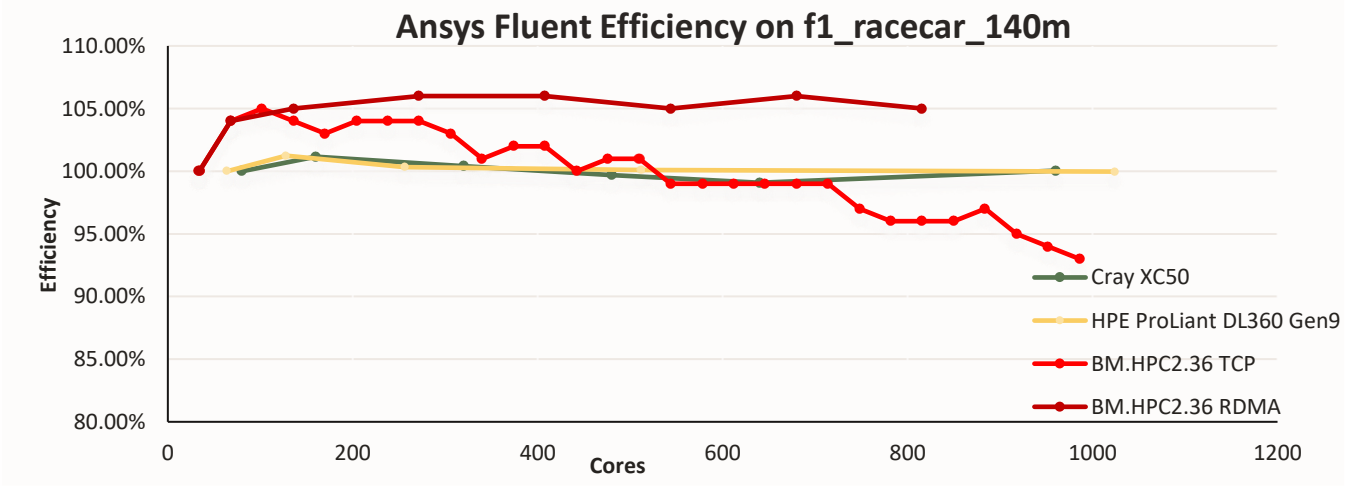
4. The output of the simulation produces the following files:
   a. .out file – shows the output of the run, including the solver rating, number of seconds per iteration and speed
   b. .log file – shows the history of the benchmark run
   c. .trn – shows the transcript of the run

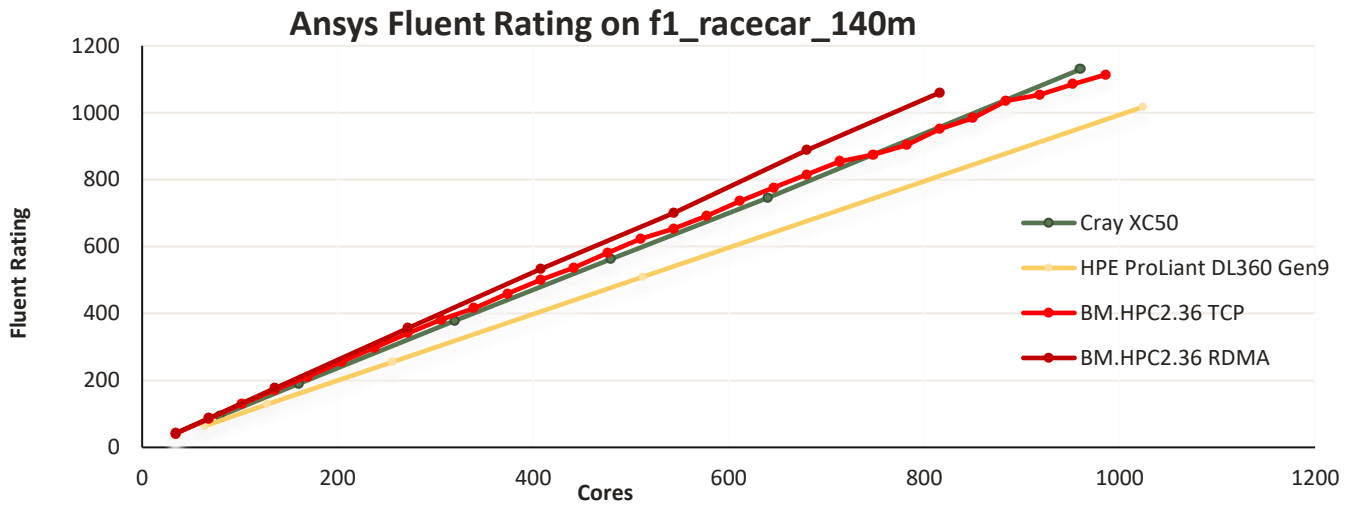
   You can plot the solver rating, speed and calculate the efficiency at each core count you run, and then plot to determine the optimal parameters to run at. The following is an example from the formula 1 racecar model:



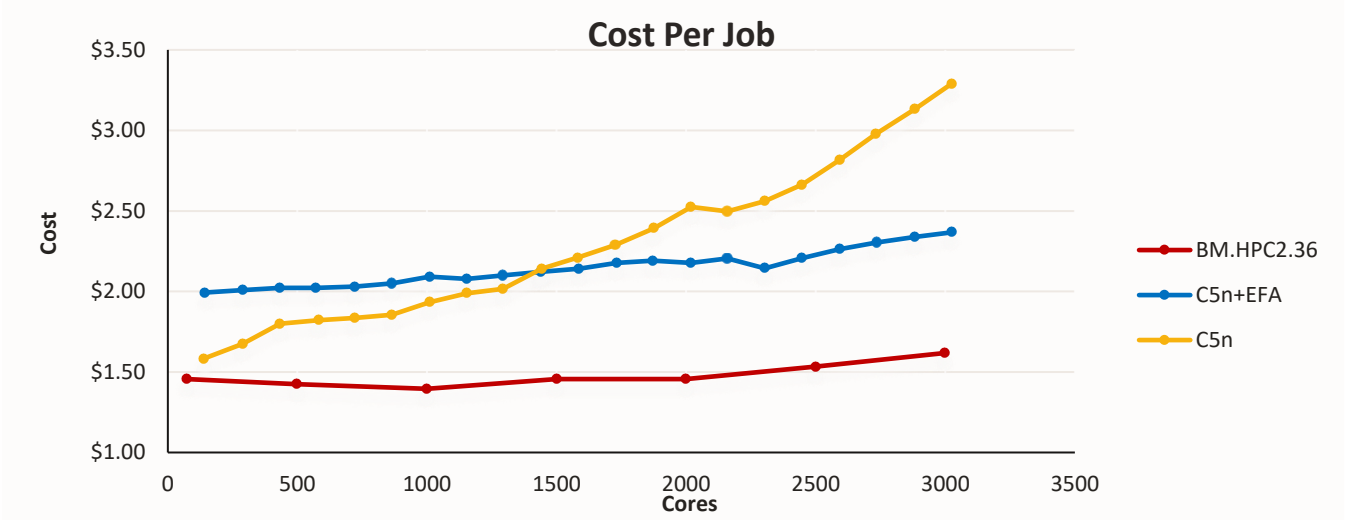
Performance Comparison

We ran an Ansys Fluent benchmark using a 140M Cell Formula-1 Racecar Model on our BM.HPC2.36 HPC Instances, both with and without Cluster Networking, and compared it with the benchmarking done on-premises by Ansys Inc. The environments Ansys used were *Cray XC50 with Intel Skylake Gold 6148 processor* and *HPE ProLiant DL360 Gen9 with Intel E5-2697A v4 processor*. The Ansys Fluent performance on OCI HPC Bare Metal instances with Cluster Networking (RDMA) exceeded the performance of the on-premis environments. The charts below illustrate the efficiency and fluent rating associated with the model on each environment.





The Formula 1 Racecar Model cost per job was also compared with AWS Cost per Job on their c5n.18xlarge HPC shapes. Our cost per job is consistently +30% less than the C5n shape with comparable performance.



RUNNING CONVERGE

Converge is a commercial CFD software for simulating multi-phase flows and turbulent combustion. Converge provides autonomous meshing and robust solvers that allow simulation of complex geometries. It is developed by Convergent Science and is widely used in Automotive industries.

Installation

The prerequisite libraries needed to run Converge on OCI HPC Cluster Networking are installed during the provisioning process as part of the Ansible playbook cfd.yml. Only the main Converge software needs installation, and can be downloaded / configured on the cluster after provisioning. You can download Converge from the Convergent Science website and push it to your machine using scp.

```
scp /path/own/machine/Convergent_Science-<version>.tar opc@<public ip>:/mnt/gluster-share
```

Alternatively, you can upload Converge into object storage, create a preauthenticated request, and then and download it onto the cluster:

1. Login to the OCI Console. In the main menu, select **Object Storage**.
2. Choose the correct **Region** on the top right.
3. Select the correct **Compartment** on the left-hand side.
4. Create a bucket if you do not have one already created.
5. In the bucket, select upload object and specify the path of the installer.
6. Click the 3 dots on the right-hand side of the installer object and select **Create Pre-Authenticated Request**.
7. If you lose the URL, you cannot get it back, but you can regenerate a new Pre-Authenticated Request (PAR) URL.
8. Download the installer form object storage with:

   ```
   wget PAR_URL
   ```

9. Untar the installer in a shared location, such as /mnt/gluster-share/install or /mnt/nfs-share/install.

   ```
   tar -xf <Convergent_Science package>.tar.gz
   ```

10. To specify the host you need to run on, you need to create a machinefile. You can generate it using the following command, or manually create it using *hostname:corenumber*.

    ```
    sed 's/$/:36/' /mnt/nfs-share/etc/hostfile > machinefile
    ```

11. Navigate into the Convergent_Science-<version> folder and run the installer. Ensure the install path is set to either /mnt/gluster-share/install or /mnt/nfs-share/install to take advantage of the high throughput file systems.

    ```
    ./INSTALL
    ```

12. Once installation is complete, move into the newly created Convergent_Science folder with the installed modues. Run the following commands to set the parameters:

    ```
    export CORES=<number of cores>

    export RLM_LICENSE=<license server>
    ```

13. It is recommended to use intel mpi for best performance on OCI HPC. Load the intelmpi module, as follows:

    ```
    module load /mnt/gluster-
    share/install/Convergent_Science/Environment/modulefiles/CONVERGE/CONVERGE-
    IntelMPI/<version>
    ```

14. Run the following mpi command, and specify the following intel mpi flags:

    ```
    mpirun -v -n $CORES -ppn 36 \

    -iface enp94s0f0 -genv I_MPI_FABRICS=shm:dapl \

    -genv DAT_OVERRIDE=/etc/dat.conf \

    -genv I_MPI_DAT_LIBRARY=/usr/lib64/libdat2.so \

    -genv I_MPI_DAPL_PROVIDER=ofa-v2-cma-roe-enp94s0f0 \

    -genv I_MPI_FALLBACK=0 -genv I_MPI_PIN=yes \

    -f /mnt/gluster-share/machinefile -genv I_MPI_DEBUG=6 converge-intelmpi -c -S | tee
    intel.$CORES.$DATE.out
    ```

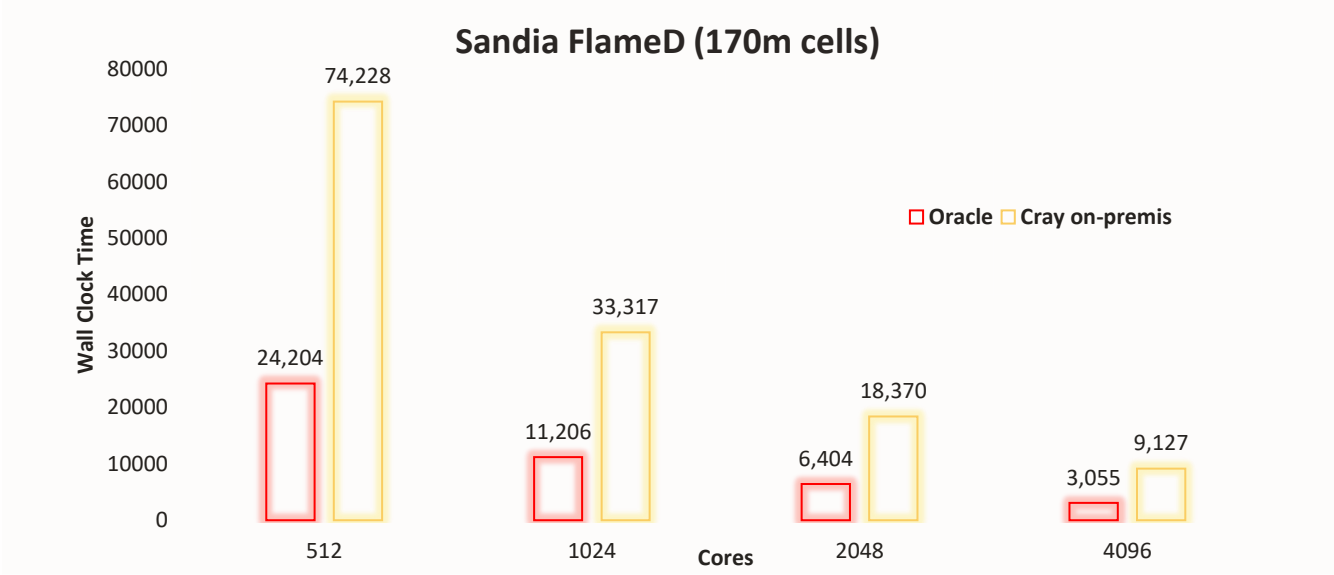
15. The output of the simulation produces the following files:
    a. Converge.log – shows the entire simulation output as a log and the total simulation run time.
    b. Time.out – shows the time per cycle

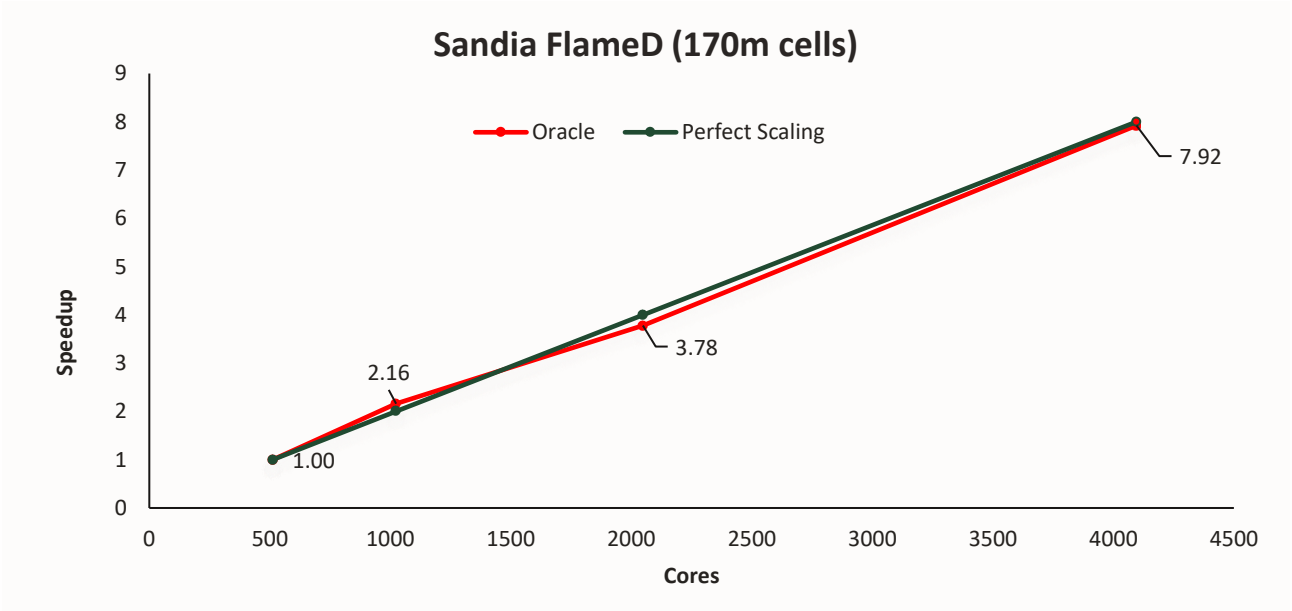
    You can calculate the speedup, scaling and efficiency from this data at each core count you run, and then plot to determine the optimal parameters to run at.

Performance Comparison

Oracle partnered with Converge Science to compare the BM.HPC2.36 Cluster Network shape with an on-premix Cray XE/XK supercomputer. The simulation model used was Sandia Flame D 170M cell model, and the core count was varied from 512 to 4,000.

The results showed that Converge scales very well on OCI HPC shapes, demonstrating near perfect scaling up to 4,000 cores. In comparison with the on-premis environment, Oracle was over **65%** faster at all core counts.





CONNECT WITH US

Call +1.800.ORACLE1 or visit oracle.com.
Outside North America, find your local office at oracle.com/contact.

blogs.oracle.com facebook.com/oracle twitter.com/oracle

Simcenter Star-CCM+ Runbook
June, 2020
Author: Arnaud Froidmont
Contributing Authors: Kristen Yang