Running OpenFOAM on IBM Cloud High Performance Computing Cluster

August 2021

Authors:
Robert Walkup
Greg Mewhinney

Overview

The IBM Spectrum LSF offering on IBM Cloud makes it easy to deploy and configure High Performance Computing (HPC) clusters to host HPC workloads on the virtual private cloud (VPC) infrastructure. From a graphical interface, you can choose the type and number of compute resources to be used in your cluster, and stand up your cluster in minutes. For users who prefer a more programmatic interface, the offering also supports use of the IBM Cloud Schematics CLI or API to deploy a cluster.

While some HPC workloads, e.g., for weather research and forecasting, use clusters containing 100s of nodes and 1000s of cores, some HPC workloads don't require a large-scale environment and will run well on a cluster with just one or a few nodes. This paper presents one such example.

OpenFOAM Elements and Characteristics

OpenFOAM is a widely used open-source toolkit for computational fluid dynamics (CFD). In this paper, we explore steady-state flow using the simpleFoam motorbike tutorial on IBM Cloud. This test case exercises key iterative solvers in OpenFOAM, including the geometric agglomerated algebraic multigrid (GAMG) solver. The motorbike tutorial also provides a mechanism for exploring OpenFOAM performance over a wide range of grid sizes. We examine the performance characteristics and discuss how to select cloud resources that are best suited for the motorbike simulations.

The key parameter for the motorbike tutorial, and for many CFD applications, is the number of grid cells used in the simulation. The number of grid cells determines the memory requirement and the compute resources needed for a timely solution. In practice, all of the OpenFOAM motorbike jobs require MPI (message-passing interface) to distribute the work over a substantial number of CPU cores. OpenFOAM uses domain decomposition for the parallelization strategy: each MPI rank works on a sub-domain, and messages are exchanged to provide data for the domain boundaries. For a simulation with a given number of grid cells, the local domains become smaller as the problem is scaled out to larger numbers of MPI ranks. Communication becomes increasingly important because the fraction of cells on local domain boundaries increases as one scales out, i.e., the surface to volume ratio increases. When the global domain becomes finely partitioned, some of the MPI ranks must communicate small messages with many neighbors, and so latency in the messaging software becomes an important consideration. When it comes to latency, communication through shared-memory is much faster than communication through any distributed-memory network device. For example, MPI ping-pong latency is just a few tenths of a microsecond with shared-memory, but can be a few tens of microseconds with the TCP / Ethernet networks on many cloud systems, including IBM Cloud.

For the simpleFoam motorbike tutorial, the number of grid cells is determined indirectly by setting the number of blocks in the “blockMeshDict” file in the system directory. This coarse grid is partitioned using the OpenFOAM decomposePar utility, where the decomposition method of choice is “scotch”, and one specifies the number of MPI ranks in the “decomposeParDict” file. The actual number of grid cells used in the simpleFoam simulation is determined by the snappyHexMesh job, which constructs grids that conform to the complex geometry of the motorbike and rider. The table below shows the number of blocks specified in “blockMeshDict”, the number of grid cells resulting from snappyHexMesh, and the memory requirement for a number of different grid sizes.

blockMesh	grid cells	memory GB
(30 12 12)	9.136E+05	5.6
(35 14 14)	1.387E+06	6.4
(45 18 18)	2.627E+06	8.8
(60 24 24)	5.413E+06	13.4
(80 32 32)	1.122E+07	24.4
(100 40 40)	2.073E+07	40.6
(125 50 50)	3.854E+07	68.8

Table 1. Grid sizes and the associated memory requirement for the simpleFoam motorbike tutorial.

The grids range from roughly ~1M cells to ~40M cells. The largest problem considered here, with blockMesh parameters (125 50 50), has ~38.5 M grid cells in the simpleFoam simulation, and requires about ~70 GB memory. With currently available CPU systems, each of these jobs can fit inside a single server or a large virtual machine, in which case communication will be very efficient using shared memory. Alternatively, one can choose some number of network-connected servers or virtual machines, and rely on the external network for scaling the problem out to a significant number of CPU cores.

Environment

On IBM Cloud, a cost-effective choice would be to use one large virtual machine. In particular, we recommend the "cx2-128x256" instance type. This is a "compute optimized" virtual machine with 128 virtual cpus (hyperthreading is enabled, and there are 128 logical cpus, 64 actual processor cores) and 256 GB memory. This instance type basically maps onto most of the CPU resources of a four-socket Intel Cascade Lake server. This choice provides ample memory, four sockets worth of memory bandwidth, and very efficient shared-memory communication, but the compute resources are limited to the 128 virtual cpus. To scale beyond one virtual machine requires use of an external network. For a core-to-core comparison, we measured the execution time for the simpleFoam motorbike benchmark on two IBM Cloud configurations, each using 128 virtual cpus: (1) a single instance of type cx2-128x256 with shared-memory communication, and (2) four instances of type cx2-32x64 with TCP / Ethernet networking between the four instances.

The Spectrum LSF offering on IBM Cloud was used to create all the necessary resources and to configure the HPC cluster for evaluating the OpenFOAM workload. The basic elements of the cluster are illustrated in Figure 1. There is a jump host (login system), one or more LSF master nodes, one NFS server node for storage, and a static number of LSF worker nodes. For OpenFOAM measurements, we selected the IBM Cloud Sydney *au-syd-2* region. Once the base cluster was created, the OpenFOAM software and its dependencies were installed, configured and compiled on the LSF master system.

Systems

The environment consisted of the following systems:

- Login System
- NFS Server
- LSF Master
- Static Worker Nodes

For all systems used, each underlying host had Cascade Lake processors.

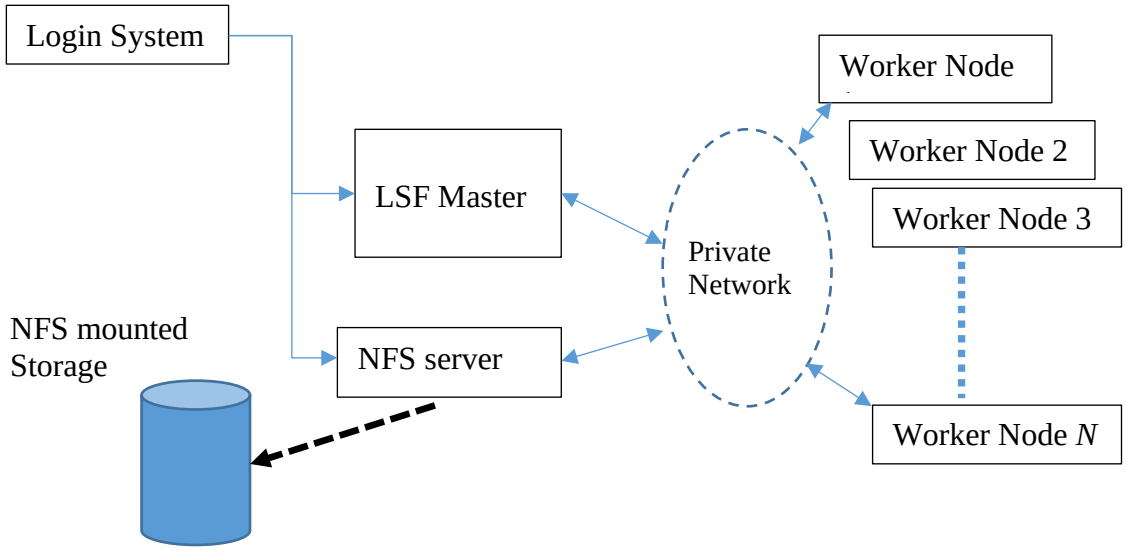

Figure 1. Cluster configuration.

Login System

The Login system is used as a jump-box into the HPC cluster systems. It is accessible by way of a public IP address.

- CentOS 7.6 (64 bit) : ibm-centos-7-6-minimal-amd64-2
- Provision profile: bx2-2x8

NFS Server

The NFS server has SAN storage attached to it, which is exported for NFS mounting by the LSF master and each worker node. The NFS server used out-of-the-box default settings.

- CentOS 7.6 (64 bit): ibm-centos-7-6-minimal-amd64-2
- Provision profile: cx2-64x128
- Storage: 2TB

LSF Master

The LSF master is provisioned from a custom image that has the LSF software installed. Upon startup, LSF is configured and started with the NFS shared storage mounted on /mnt/data. The default LSF configuration settings were used.

- Custom image: hpcc-lsf10-cent77-jun0421-v5
- Provision profile: bx2-32x128

Worker Nodes

Depending on requirements, worker nodes can be provisioned statically, generally for the life of the cluster, or dynamically in response to demand. OpenFOAM can be run easily in either environment. In either case, use the master node of your cluster to configure OpenFOAM on the cluster's shared /mnt/data filesystem. When the nodes are provisioned, they will automatically mount the /mnt/data/filesystem.

Use the following steps to create dynamic nodes.

- Select a worker node minimum count of zero
- Custom image: hpcc-lsf10-cent77-jun0421-v5
- To follow the examples in this paper, the provision profile will be one of the following depending on whether you plan to run OpenFOAM on a single large node, or multiple nodes.
    - Single node provision profile: cx2-128x256
    - Four node provision profile: cx2-32x64

OpenFOAM Results and Observations

The figure below shows the simpleFoam processing rate (higher is better) as a function of the size of the grid, where we define processing rate as the number of simpleFoam steps times the number of grid cells divided by the execution time.

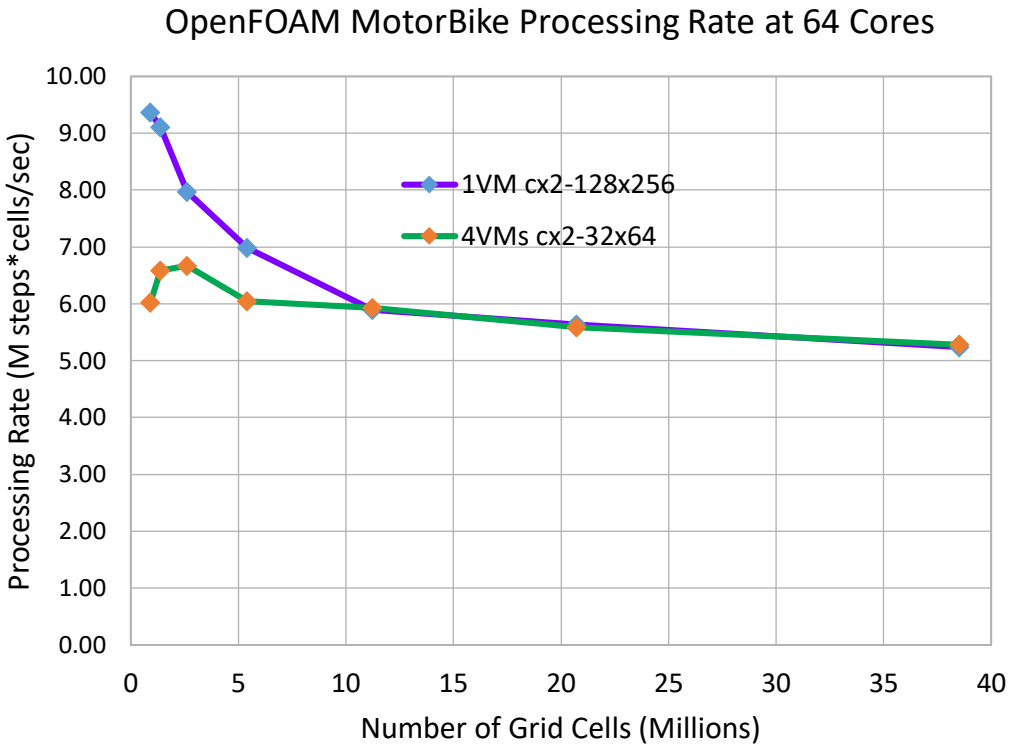

Figure 2. The simpleFoam processing rate is shown as a function of grid size for two different IBM Cloud configurations using 128 virtual cpus: (1) a single instance cx2-128x256 with shared-memory communication and (2) four instances of type cx2-32x64 with TCP / Ethernet interconnect.

For grids larger than ~10M cells, the two different virtual machine configurations provide almost identical performance. For these relatively large grids, communication performance is not very critical, and overall performance is dominated by computation, where bandwidth to memory is a major factor. For grids of ~5M cells or smaller, communication becomes more important, and the single instance cx2-128x256 provides better performance due to more efficient communication via shared memory. In fact, the overall processing rate for cx2-128x256 improves as the grid size decreases from ~5M to ~1M cells. This effect is due to a transition from computation that is limited by bandwidth to memory for large grids, to more efficient computation benefiting from caches as the number of grid cells and associated data structures become smaller.

Conclusion

For OpenFOAM problems with grids of ~10M cells or less, we recommend using the single virtual-machine approach, with one instance of type cx2-128x256. OpenFOAM problems with larger grids may fit on one cx2-128x256 instance, but time to solution will be limited by the number of cores (64) or the number of sockets (4). IBM Cloud offers 64-core, four-socket instance types with far more memory: bx2-128x512 (128 virtual cpus with 512 GB memory) and mx2-128x1024 (128 virtual cpus with 1024 GB memory), and these larger memory instance types can support much larger OpenFOAM problems. However, if time to solution is important, OpenFOAM problems with ~10M grid cells or more can be scaled out to engage more cores, using TCP / Ethernet connected virtual machines. With the current networking infrastructure on IBM Cloud, we recommend limiting scaling to >~ 2*10^5 grid cells per core, or >~ 3*10^6 grid cells per socket. This will ensure that OpenFOAM performance remains in the favorable compute-bound region of the scaling curve. Systems that provide a significantly lower latency interconnect can push OpenFOAM scaling out further and still remain in the favorable part of the scaling curve. However, once scaling becomes sub-linear, the end user must choose between somewhat reduced time to solution, and an increasing cost of the simulation. This kind of trade off occurs on all systems.

Trademarks

IBM®, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.
SoftLayer® is a registered trademark of SoftLayer, Inc., an IBM Company.
Other company, product, or service names may be trademarks or service marks of others.
The information contained in this document has not been submitted to any formal IBM test and is distributed on an “as is” basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer’s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will result elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.