WRF workload evaluation on IBM Cloud high performance computing cluster

Contents

02 Overview

03 Environment
- **–** Systems
- **–** Login system
- **–** LSF master
- – Worker nodes

04 WRF model elements
- **–** Dependencies
- **–** WRF compilation and configuration

05 WRF run scenarios

06 WRF results

07 Observations

07 Conclusion

08 References

Overview

High Performance Computing (HPC) workloads can use IBM Spectrum LSF scheduling software on a cluster of compute nodes which they can easily deploy on IBM Cloud® using the IBM Spectrum LSF offering. To demonstrate this capability as well as the performance and scalability of IBM Cloud, the Weather Research and Forecasting (WRF) model workload was selected for a set of benchmark runs.

The WRF model is a mesoscale numerical weather prediction system. The model is widely used for meteorological applications. The model's scale of resolution can range from tens of meters to hundreds of kilometers. For this effort, the Continental United States (CONUS) at 2.5km lateral resolution was chosen. This is representative of the current state of the art for deterministic forecast models, making it an interesting test case for cloud computing.

For weather forecast models, the simulation speed-up provides a useful figure of merit. This is the ratio of the forward integration time in forecast-hours, to the actual elapsed time required to complete the job. A simulation speed-up factor of ~24x or greater is desirable, because that would allow hourly updates for the next day's weather forecast, and a speed-up factor of ~48x which provides a two-day forecast is excellent.

The auto-scaling feature of IBM Spectrum LSF was leveraged to run the WRF workload using different numbers of compute nodes in the cluster. HPC users submit normal LSF job scripts, and a cluster that meets the job requirements can be provisioned and ready to start the job within a few minutes. If the cluster remains idle for more than a preset time, the cluster will automatically shrink to a predefined minimum size after job completion. Otherwise, the provisioned cluster remains available to immediately start jobs on the provisioned resources.

Environment

The Spectrum LSF offering on IBM Cloud was used to create all the necessary resources and to configure the HPC cluster for evaluating the WRF workload. Spectrum LSF makes use of virtual private cloud (VPC) infrastructure services including Schematics and Terraform capabilities. This enables simple cluster creation in any of the IBM Cloud VPC multi-zone regions (MZRs). 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 dynamically variable number of LSF worker nodes. For WRF measurements, we selected the IBM Cloud Sydney *au-syd-2* region. Once the base cluster was created, the WRF model software and its dependencies were installed, configured and compiled on the LSF master system.

The Spectrum LSF auto-scaling Resource Connector (RC) feature was used to dynamically provision worker nodes for the WRF model calculations. Upon completion of the WRF model run, the worker nodes would be automatically de-provisioned upon expiration of the LSF Resource Connector idle time. The number of worker nodes provisioned was varied to measure the scalability of the WRF workload on IBM Cloud.

Systems

The environment consisted of the following systems:

- Login system
- NFS server
- LSF master
- Dynamic worker nodes provisioned on demand

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

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

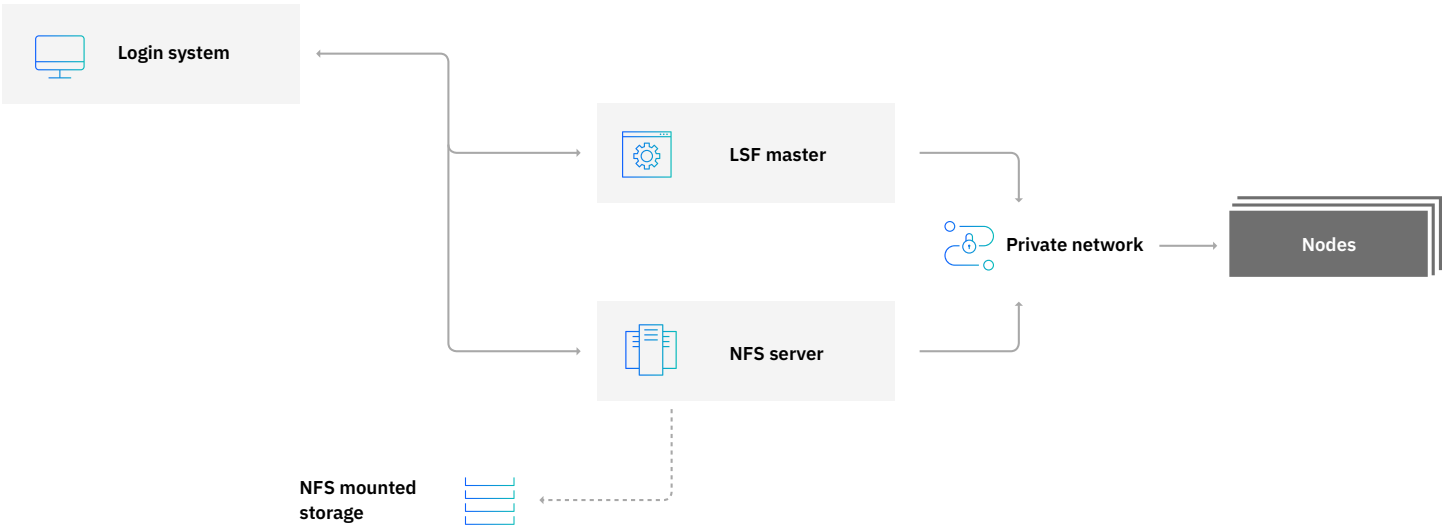


Figure 1: Cluster makeup

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
We selected a minimum worker count of zero, so all worker nodes are dynamically allocated and provisioned, based on LSF job requirements. When an LSF job is submitted, worker nodes are provisioned based upon demand. The nodes automatically join the cluster with the NFS shared storage mounted on /mnt/data. The WRF test case requires only a modest amount of memory per core, so we selected a "compute optimized" instance type for the worker nodes, with 16 virtual cpus and 32 GB memory.

– Custom image: hpcc-lsf10-cent77-jun0421-v5
– Provision profile: cx2-16x32

WRF model elements

The WRF model has several dependencies that were installed, compiled, and configured on the LSF master system. These dependencies were placed in the NFS shared storage for accessibility by the worker nodes during WRF forecast simulation run time.

Dependencies
The following dependent software and support libraries were used:

– netcdf-c-4.8.0
– netcdf-fortran-4.5.3
– openmpi-4.1.1
– Intel Parallel Studio XE Edition 2019
– gcc, g++ version 4.8.5, included with the base operating system

WRF compilation and configuration
WRF version 4.1.5 was built with the Intel Fortran compiler, Intel Parallel Studio XE Edition 2019, using the latest NetCDF software, and compiler options based on the distributed-memory plus shared-memory setup for Intel Haswell/Broadwell processors.

Minor changes were made to the resulting *configure.wrf* file in order to ensure proper architecture settings for the Intel Cascade Lake processors on IBM Cloud.

WRF run scenarios

WRF was used to carry out a 12-hour forecast for the continental US at 2.5km lateral resolution. Similar benchmarks have long been available for WRF version 3, but the version 3 inputs are not compatible with WRF version 4. As a result, WRF version 4 input data and boundary data were prepared using the WRF preprocessing system, WPS, as outlined in: github.com/Azure/azurehpc/tree/master/apps/wrf

The older WRF version 3 benchmarks had I/O turned off in the namelist.input file. In contrast, operational weather models write history files at regular intervals, and so we included history file writes at every forecast hour. Managing I/O is very important for the overall performance and scalability of WRF. We selected the asynchronous I/O capability that is built into WRF, using "quilting" with NetCDF outputs. In practice, this requires careful specification of the two-dimensional process grid and the I/O server configuration in the WRF *namelist.input* file. The forecast history files, 8 GB each, were written to the NFS shared storage.

The following table shows the breakdown of nodes and the underlying WRF *namelist.input* settings used to measure the scaling characteristics of the WRF model on IBM Cloud. The IBM Cloud VPC virtual server profile of cx2-16x32 was used for each worker node provisioned.

We used a mix of OpenMP and MPI for parallelization, with one OpenMP thread per physical core. Note that the worker virtual-machines have hyperthreading enabled, so there are two virtual CPUs per physical core. We made use of the "tiling" option in WRF to improve cache locality, where tiling is over grid points in the Y dimension. We launched the WRF executable with a helper script to explicitly control thread affinity by setting OMP_PLACES to a list of logical CPUs based on the local MPI rank within each worker node. The total number of MPI ranks is the number of I/O servers plus the product of the two-dimensional domain decomposition parameters, nproc_x * nproc_y.

We chose to integrate the model forward for 12 forecast hours, writing history files every forecast hour. This stresses the computation, communication, and I/O characteristics of the cloud. We used WRF version 4.1.5 with a 15 second time step interval was used, which results in 2,880 computational steps being performed in total. Radiation was computed every 10 forecast minutes.

Each scaling run was invoked from the LSF master with a normal LSF job script.

Worker nodes	vCPUs	Cores	nproc_x	nproc_y	I/O servers	Number of tiles	MPI ranks per node	OpenMP threads
33	528	264	10	12	12	20	4	2
66	1056	528	16	16	8	20	4	2
100	1600	800	19	20	20	16	4	2
135	2160	1080	26	20	20	16	4	2
169	2704	1352	25	26	26	12	4	2
198	3168	1584	32	24	24	12	4	2

WRF results

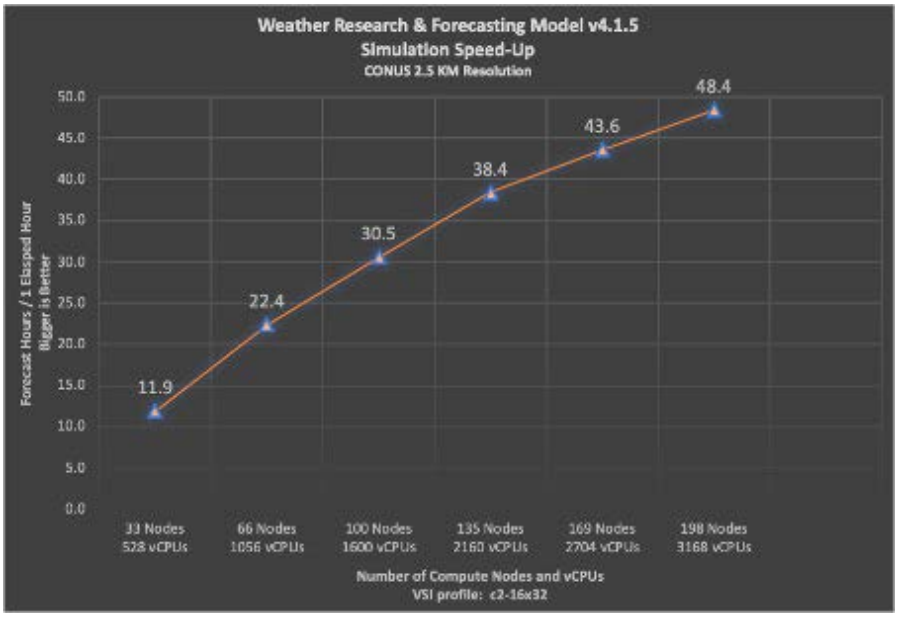
The simulation speed-up factor is taken to be the 15 second time-step divided by the average elapsed time per time-step for the 12-hour forecast, as reported by "Timing for main" values in the *rsl.error.0000* file. This is roughly equivalent to taking the forecast interval, 12 hours, divided by the elapsed time for the job. The speed-up factor indicates how many forecast hours can be computed in one wall-clock hour.

The testing method used was to submit a sequence of WRF jobs for a given number of nodes. The first job provisions the required worker nodes as specified in the LSF job script. Upon job completion, and within the LSF idle time, two additional runs were performed back-to-back, using the previously provisioned nodes.

Results shown are the average of the three runs:

Worker nodes	Provision time (minutes)	WRF calculation average total time (seconds)	Average time / step (seconds)	Average speed up: Forecast hours / 1 elapsed hour
33	2:26	3642.5	1.26	11.9
66	2:29	1932.6	0.67	22.4
100	2:08	1415.2	0.49	30.5
135	2:43	1124.2	0.39	38.4
169	3:03	991.0	0.34	43.6
198	2:28	892.8	0.31	48.4

The speed-up scaling curve is shown in the following graph.



Observations

Within one wall-clock hour, a 24-hour forecast can be completed using 100 nodes (1600 vCPUs), a 36-hour forecast can be completed using 135 nodes (2160 vCPUs), and a 48-hour forecast can be completed using 198 nodes (3168 vCPUs).

Analysis of timer outputs shows that the asynchronous I/O feature of WRF is very effective at hiding the cost of history file writes. The compute tasks continue with integrating the model forward in time, while the I/O tasks are writing history files.

We observed some performance variability for successive jobs, most likely caused by competition for compute and/or network resources. Variability tended to increase with an increasing number of worker nodes, with up to 8-9% variations at 198 workers. In any case, good scaling was observed up to ~3000 vCPUs, and overall performance is more than sufficient to achieve speed-up factors in the desirable range for operational forecasts.

Starting from a cluster with just the LSF master node, with no statically allocated worker nodes, it took typically 2-3 minutes to allocate and provision the worker nodes required for each job, spanning a range of 33 to 198 worker nodes. In operational use, it would be possible to provision the worker nodes just one time and run a sequence of forecast jobs at the same scale, reusing the allocated workers.

Conclusion

The IBM Cloud environment deployed using the Spectrum LSF offering provided good performance and scalability for the WRF weather model covering the continental US at 2.5 km lateral resolution. A 48-hour forecast can be completed in one elapsed hour using a cluster consisting of 198 cx2-16x32 IBM Cloud virtual server instances (VSIs). All 198 VSIs were provisioned and configured in the LSF cluster in under 3 minutes. This demonstrates that IBM Cloud has the performance and scalability for high resolution production weather forecasting.

By leveraging the auto-scaling capabilities of the Spectrum LSF offering, and the fast provisioning performance of IBM Cloud VSIs, HPC cluster setup is simple and efficient, and operational costs can be minimized by paying only for compute resources when they are needed.

Authors
Augie Mena
Robert Walkup
Paul D. Mazzurana

References

techcommunity.microsoft.com/t5/azure-global/run-wrf-v4-on-azure-hpc-virtual-machines/ba-p/1131097

github.com/Azure/azurehpc/tree/master/apps/wrf

weather.com/en-IN/india/news/news/2019-12-12-ibm-global-high-resolution-atmospheric-forecasting-system

mmm.ucar.edu/wrf/WG2/benchv3/

mmm.ucar.edu/weather-research-and-forecasting-model

en.wikipedia.org/wiki/Weather_Research_and_Forecasting_Model

IBM Corporation
IBM Cloud
Route 100
Somers, NY 10589

Produced in the United States of America
August 2021