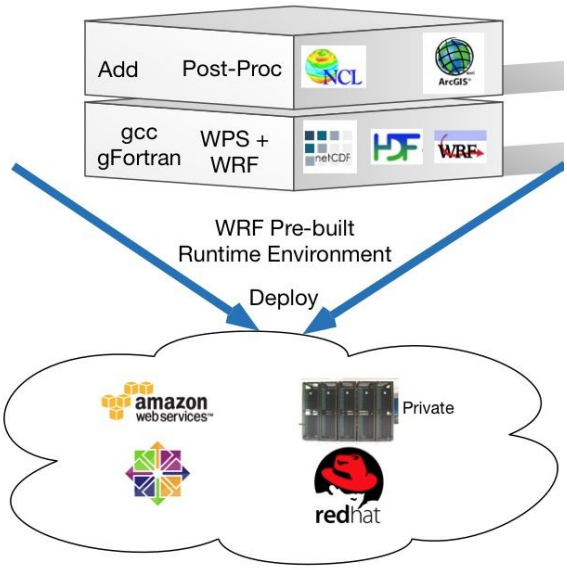
COFPS

WRF-CAWFE Modeling performance on AWS

Preliminary report and findings



Introduction

The primary goal was to verify that a software build of WRF-CAWFE can be constructed on local, on-premise servers along with required data sets (static geography, model input conditions, fuel map) to be copied to hosted Amazon Web Services EC2 (virtual machines) and execute the WRF-CAWFE model and data sets with minimal adjustments. Second, capturing the run times of WRF-CAWFE model between on-premise hardware and AWS EC2 instances was compared, as well as initial investigation in higher performance storage options and recommendations for future performance improvements.

The COFPS release of the WRF Preprocessing System (WPS) and WRF-CAWFE code and dependencies was compiled on a local server (cofps-int1) at NCAR/RAL, utilizing the free GNU Fortran compiler and related dependencies such as NetCDF and other libraries. In COFPS operations the commercial Intel Composer suite is used to deliver performance

improvements for the models, yet it's important to note the Intel Compiler was not used in these first comparisons described below.

The resulting WPS and WRF-CAWFE libraries, binaries, data sets and configuration files were copied (scp, secure copy linux command) onto solid state disk (SSD) storage volumes on AWS for use when attached to different EC2 instance types necessary to review potential performance gains across differing virtual machine specifications.

Initial WRF Configuration and Cloud Performance:

The WRF-CAWFE configuration was identical to the one used for operational forecasts. NCAR modelers configured WRF-CAWFE with two nested domains. The first one has a grid spacing of 1 km and the second one of 111 m. Both domains have the same number of grid points, 118 in the west-east direction and 118 in the north-south direction. A fire was ignited in the center of the domain at the beginning of the simulation. The fire spread was simulated for around 10 h until the fire reached the domain boundary and the simulation stopped. This allowed us to test the performance of WRF-CAWFE in a case with significant fire activity.

Level of effort to execute WRF-CAWFE model on AWS EC2 instances was minimal. Initial time to select a new EC2 instance type, apply OS patches and install additional runtime software (gfortran) was approximately 15 minutes, a process that need not be repeated for subsequent EC2 instances when utilizing virtual machine (VM) imaging tools.

Adjustment of the WRF-CAWFE model run-time script was not needed. The full process of WPS and WRF-CAWFE executed normally with NO adjustments required to libraries, environment variables or configuration files.

Summary: The "cut and paste" of the WRF-CAWFE modeling directory containing binaries and data from local server to AWS EC2 instance was a success.

Table 1 (below) summarizes results of our performance experiments. These are the main findings:

- Current hardware on site at NCAR is limited to running three concurrent fires with approximately 14 cores per forecast. Under normal circumstances AWS resources

will not be limited and could execute as many WRF-CAWFE forecasts as needed concurrently, accruing compute costs.

- Initial AWS single instances (Run #3 and Run #4) show 25%-33% slower compute times than the reference simulation in cofps-int1(Run #1). No optimization was attempted for these runs.
- Refinements to AWS (more cores, multiple instances) delivers performance times (Run #5) similar to on site virtual machines at NCAR (cofps-wrf2, Run #2). This simulation (Run #5) is within %15 of bare metal performance (Run #1).
- Maximizing all cores between two "compute optimized" AWS c4.8xlarge instances showed 10% better performance than on site bare metal server. (Run #6 vs. Run #1)
- * AWS charges for a "complete hour" when EC2 instances are powered on. Cost estimation is subject to rounding up to the nearest hour and may be dependent on top-of-the hour start/stop deltas.

Table 1: Summary of basic performance findings for 10 h forecast:

Run #	**Location**	**Type**	**Cores**	**Avg. output**	**Total WRF time**	**Compute Cost Approx***
1	cofps-int1	Local	24	21 min	195 min	x
2	cofps-wrf2	Local VM	24	24 min	232 min	x
3	AWS	c4.8xlarge	24	33 min	323 min	$9.54 ($1.59 per hour)
4	AWS	r4.8xlarge	32	28 min	266 min	$10.65 ($2.13 per hour)
5	AWS multinode	2x c4.8XL	48	24 min	225 min	$12.72 ($3.18 per hour)
6	AWS multinode	2x c4.8XL	64	19 min	176 min	$9.54 ($3.18 per hour)

Upcoming Optimization Experiments:

1. WPS is IO bound on most AWS SSD storage types, move to in-memory processing.
2. Workflow of pre-processing data sets for WRF can be streamlined for cloud compute and storage resources. This can reduce time to provide the first hourly forecast WRF-CAWFE output file.
3. WRF-CAWFE compilation rebuild utilizing Intel compiler, re-run tests.
4. Review WRF-CAWFE model configuration to match most optimal or affordable AWS compute configurations. Perhaps “right sizing” a WRF-CAWFE configuration (number of patches, number of tiles) to increase memory per MPI task to minimize message passing latency over Amazon networks.
5. Review AWS Science report (2015) about potential AWS configuration options, maximize compute parameters and core allocations.
6. Reconstruct multiple EC2 instances in closer network proximity to minimize MPI latency.
7. AWS Compute costs can be reduced by utilizing AWS Batch, Reserved Instance (20-30% cost savings), Spot instances.