Impediments to regular testing of scientific software

A white paper for the 2019 Collegeville Workshop on Sustainable Scientific Software (CW3S19)

Robert Jacob, Argonne National Laboratory, jacob@anl.gov

One of the most important practices for keeping scientific software sustainable is regular testing. Software tests can be divided in to two categories: functional and system. A functional test will execute a small portion of the code, ideally a single subroutine, and compare the results against known or analytic solutions. A system test will execute larger sections of code or even the entire code base and test expected behaviors like "run to completion" or "write and re-start from a checkpoint" or "run with debugging flags". Multiple system and functional tests are grouped in to test suites.

If the scientific software under test is a high-performance computing application, the test suite for that software will itself be a high-performance computing application in that it will require similar node counts/memory as a full run of the application in order to test all features. The only difference from a production run of the application is that each test can typically be run in a short amount of wallclock time.

Despite the importance of testing for sustaining the nation's scientific software, there are few dedicated resources for testing at the nation's compute centers.

In a code under active development, test suites should be run on each change before it is integrated into the code and on the resulting integrated code before it is approved for further development. Typically, the integration test suite has to be run overnight on all changes from that day so that on the following day development can continue (or mistakes can be fixed). The need for continuous testing presents particular challenges when conducted within available compute infrastructure. In addition to a nightly test suite, good testing practice should provide developers with a shorter test suite they can run on their changes as they develop.

To further illustrate the problem, we'll consider the test suites developed for the DOE Energy Exascale Earth System Model (E3SM) and how every decision in the design of those suites has been shaped by the limits of testing resources.

While functional tests are still in development, E3SM has a rich set of system tests. There are separate tests for running to completion (the smoke test) and for writing and then reading a checkpoint while comparing with a simulation that ran straight through. There are tests that vary the number of MPI tasks between two runs and compares output for a bit-for-bit match (a requirement for many climate models). E3SM is a highly configurable model that allows different resolutions of the numerical grids and different combinations of component models (the atmosphere, ocean, land surface, etc.) Furthermore, a given configuration can have different behavior depending on runtime switches and parameter values. The combinations of

test type, resolution, component set and runtime settings leads to a large testing space to cover even when limited only to the cases of direct scientific interest.

When the project started, we had allocations at the 3 DOE computing centers: NERSC and both leadership computing centers, OLCF and ALCF. These were the target platforms for development and production. When we assembled our first integration test suite, which we thought covered enough but not all of the configuration space, we immediately had a problem: none of these platforms could run the suite overnight. The biggest impediment was that they all had queues that were to busy to guarantee a start every evening. If a debugging queue with short-turnaround was available, it did not allow enough nodes or enough wallclock time or had a limit on how many jobs could be run at once (each test in a suite is submitted by the test system as a separate job). Using a premium queue for testing to boost the priority would use up time that was allocated for production runs and still not guarantee a start every evening.

Fortunately, one of the 8 laboratories involved in E3SM, Sandia National Lab, had a general use CPU-only compute cluster that was uncrowded enough that it could reliably start the suite every evening. But the suite was to expensive to complete by the next morning if all tests were run at the standard resolution of 1 degree in the atmosphere. Several tests were converted to run at either 3 degrees or 7.5 degrees with corresponding low resolutions in the ocean for coupled or ocean-only tests. 7.5 degrees is too coarse for scientifically reliable climate simulations but since the system tests are mostly testing code logic for checkpoints and invariance under MPI task counts it can be used for those. We also lowered the length of integration from typically 5 days to just 5 or 9 timesteps in some cases to make sure the suite will finish overnight. We add tests as new features are added to the model and remove some that are no longer relevant, always making sure the suite can finish overnight.

E3SM has currently settled on an integration test suite with 71 tests, most run at coarse resolution and all run overnight on two clusters at Sandia with about 1,500 CPU-only nodes each. We also use a 600-nodes cluster at Argonne that provides overnight turnaround for the integration suite. Multiple machines are needed to provide coverage for different compilers and to test both the integration and master branches of our repository. We still can not count on overnight turnaround every night. Sometimes the clusters will fill up with other jobs and delay test suite completion.

We do not ignore the target platforms for our testing. We have developed a “production” test suite which has 2 tests: one for each of the 1-degree resolution cases used in our production runs. This can reliably run overnight on one of our production platforms: Cori-KNL at NERSC.. We also run the developer test suite on Cori-KNL every other evening. On Theta at ALCF we run the integration test suite once a week. Theta and Cori-KNL also run a “high resolution” suite once a week that uses up to 400 nodes. We have not yet started regular testing on Summit at OLCF. The LCF’s make automatic testing using a service like Jenkins difficult because they require One Time Passwords. When we were still testing on Titan at OLCF, an E3SM project staff member had the job of logging in once a week to start the test suite.

The developer test suite has been constructed, using coarse resolution configurations of E3SM, so that it can run in a couple of hours on a moderately sized workstation with 2 or more multi-core CPUs. That is still to long for testing every commit a developer might make in a day. But even if it returned in minutes, workstation testing is not enough for keeping high-performance computing scientific software sustainable. It must be tested at the scales it will be run in production and on the hardware on which it will be run. This is the only way to make sure that any problems specific to the CPU-network-compiler-OS-libraries on production platforms are found. Such testing will become more important as production platforms become more diverse and node architectures become more complex. In E3SM we are assuming that weekly or every-other-day testing of small suites on production platforms can be enough to catch problems when combined with nightly testing of larger suites on conventional clusters. Ultimately, changes to how production resources at national centers are allocated will be needed to make adequate testing of scientific software possible.

The dashboard for the E3SM test suites is visible at
https://my.cdash.org/index.php?project=ACME_Climate