

Improvements Towards the Release of the Pavilion 2.0 Test Harness



Kody J. Everson
Dakota State University
Kody.Everson@trojans.dsu.edu

Maria Francine Lapid
Los Alamos National Laboratory
Lapid@lanl.gov

Mentors (HPC-ENV PRE-team): Paul Ferrell, Nicholas Sly, and Jennifer Green

Overview

High performance computing production support entails thorough testing in order to evaluate the efficacy of a system for production-grade workloads. There are various phases of a system's life-cycle to evaluate, requiring different methods to effectively measure performance and correctness. Due to the unique and distributed nature of an HPC system, the necessity for sophisticated tools to automatically harness and assess test results, all while interacting with schedulers and programming environment software, requires a customizable, extensible, and lightweight system to manage concurrent testing.

Pavilion 2.0 is the next major release after the original Pavilion test harness, which is based on its predecessor, Gazebo.

Pavilion Usage and Underlying Process

How to use Pavilion

- Step 1: Write a test (specify build, run, scheduler, results specifications)
- Step 2: Run test(s) or suite(s) (optional: view status)
- Step 3: Get result, debug as needed

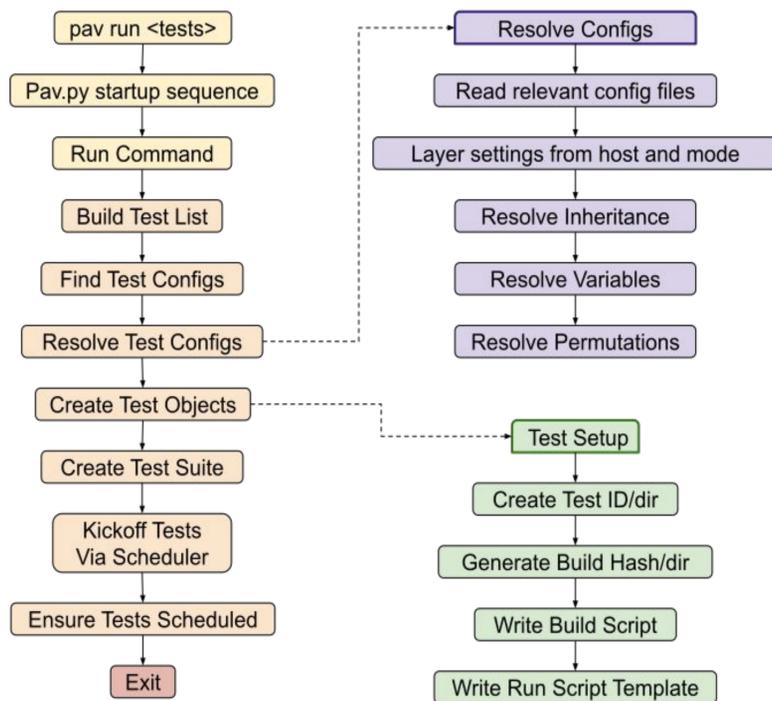
```
hello_world.yaml
hello_world:
  run:
    cmds:
      - 'echo hello'
```

A simple test config

```
bash $ pav run hello_world
Running 1 test in series s154.
id | name | status | note
-----
154 | hello_world | SCHEDULED |
bash $ pav result
id | name | result
-----
154 | hello_world | PASS
```

Output of pav run and pav result

Pavilion 2.0 Run Command and Test Setup



Commands

Commands are one way to add functionality to Pavilion and are the main way users interact with the system. The plugin system makes it simple for users to add their own commands, or overwrite existing ones, according to their preferences and machine specifics. We added the following commands:

- log - outputs the log file (build, kickoff, or run) of a given test[†]
- cancel - used to cancel a provided test, group of tests, or test series*
- clean - used to wipe out the Pavilion working directory (removes all tests, series, downloads, and build directories)*
- status --all - prints the last few tests run by a user[†]
- run --status - prints the status of the jobs started with the run command*

```
-bash-4.2$ pav status --all --limit 4
Test statuses
-----
Test id | Name | State | Time | Note
-----
472 | stream | RESULTS_ERROR | 15 Jul 2019 10:06:38 UTC-06:00 | The test...
473 | stream | COMPLETE | 18 Jul 2019 10:09:48 UTC-06:00 | The test...
474 | stream | RESULTS_ERROR | 18 Jul 2019 10:10:50 UTC-06:00 | The test...
475 | stream | COMPLETE | 18 Jul 2019 10:14:48 UTC-06:00 | The test...
```

Output of status --all command

```
bash $ pav log run 154
hello
bash $ pav log kickoff 154
The kickoff log is empty.
bash $ pav log build 154
The build log is empty.
```

Output of log command

```
bash $ pav clean -v
Removing Tests...
Removed test 0000001
Removed test 0000002
Skipped test 0000003
Skipped test 0000004
Removing Series...
Removed series 0000001
Removed series 0000002
Skipped series 0000003
Removing Downloads...
Removing Builds...
Removed build 58d90e966a0976e2
Removed build c306e89258705bb1
Skipped build 4fe2db555009a8f
```

Output of clean command

```
bash $ pav cancel 21 22 s22 s23
test 21 cancelled.
test 22 could not be cancelled has state: SCHED_CANCELLED.
test 24 cancelled.
test 25 cancelled.
```

Output of cancel command

Result Parsers

Result parsers look at the output of the benchmarks, determine what makes a test "pass", and can extract important data from the test's output.

Result Parser	What it does	Keys needed
Constant [†]	inserts a given constant into the results	constant
Command [†]	runs a given command	<ul style="list-style-type: none"> cmd success success_value stderr_out
Table [†]	extracts values from a table and puts the data in a nested dictionary	<ul style="list-style-type: none"> row_names col_names

```
bash $ pav log run 77
Output of Stream test:
-----
Function Best Rate MB/s Avg time Min time Max time
Copy: 6212.9 0.028593 0.025753 0.032994
Scale: 6000.8 0.029258 0.026663 0.032463
Add: 8469.2 0.031915 0.028338 0.034907
Triad: 8106.8 0.033490 0.029605 0.040039
```

```
stream.yaml
results:
  table:
    row_names: ['Copy', 'Scale', 'Add', 'Triad']
    col_names: ['Best Rate MB/s', 'Avg time', 'Min time', 'Max time']
```

Sample table that can be parsed by the table result parser and the yaml config for it

Tests

The following tests are already available on the LANL clusters:

hello_mpi	ior*	supermagic	magic_cookies	mounts_test [†]
imb*	license-check	slow_test*	stream ^{†1}	mpi-slowness [†]

Other Features

Miscellaneous features we implemented include:

- Variable-handling
- Allow users to add commands to their kickoff scripts*, regardless of scheduler
- Text wrapping*

```
-bash-4.2$ pav status
Test statuses
-----
Test id | Name | State | Time | Note
-----
22 | ior | COMPLETE | 18 Jul 2019 10:26:43 UTC-06:00 | The test completed with result: PASS
23 | ior | COMPLETE | 18 Jul 2019 10:27:04 UTC-06:00 | The test completed with result: PASS
24 | ior | COMPLETE | 18 Jul 2019 10:27:15 UTC-06:00 | The test completed with result: PASS
25 | ior | COMPLETE | 18 Jul 2019 10:27:37 UTC-06:00 | The test completed with result: PASS
```

Text-wrapping example for a table

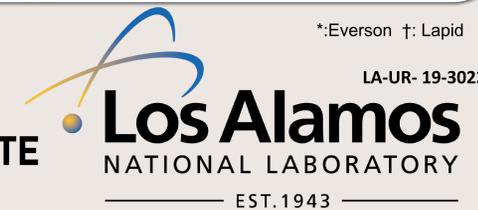
Future Work & Acknowledgments

Currently, the Pavilion 2.0 code is open to the public. Our team is working to get Pavilion adopted by other HPC centers. We would also like to introduce the following features in the future:

- conditional testing
- test series files
- additional results layer for analysis

We would like to thank our mentors and the rest of the Programming and Runtime Environments team (Dan Magee, Jordan Orgas, David Shrader, Calvin Seamons, and Trent Steen) for helping us throughout the summer.

1 McCalpin, John D., 1995: "Memory Bandwidth and Machine Balance in Current High Performance Computers", IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter, December 1995.



*:Everson †:Lapid

LA-UR-19-30223