Accelerate slow pytests

Tom Rochette <tom.rochette@coreteks.org>

February 27, 2020 — 5bd14346

1 Problem

My pytests take a while to complete, how can I speed up the process?

2 Solution

A fairly cheap solution is to use parallelization to run your tests on multiple CPUs instead of the 1 cpu used by default. To do so, you can install `pytest-xdist`. Once the extension is specified, all you need to do is add `-n auto` when you call `pytest`.

Another thing you should do that requires more effort is to investigate which of your tests are consuming a lot of CPU time to execute. To do so, use the `--durations=0` flag when you call `pytest`. A report will be generated after your tests have run that lists how long setting up, running and tearing down each specific test took. The list is ordered from longest to shortest durations, meaning that the tests that have the most potential for being optimized will be at the top. You should focus on these tests because the longest one will determine how long it would take to run your tests even if you had an infinite amount of CPU cores.

Investigate why certain tests take a while to execute.

- Are some tests computing something that takes a while and is computed exactly the same way by multiple tests? Precompute this result once and share it between the different tests (think of it as a fixture).
- Are calls to a slow external API done? If you are not testing that the remote API is changing, store example responses and emulate receiving them.
- Is there a loop in the test that runs hundreds of thousands iterations while the same test could be executed with only a thousand iterations?

3 References

- https://github.com/pytest-dev/pytest-xdist