pytest-mpi Documentation

Release 0.2+0.g6c27015.dirty

James Tocknell

Oct 19, 2019

Contents:

pytest-mpi provides a number of things to assist with using pytest with MPI-using code, specifically:

- Displaying of the current MPI configuration (e.g. the MPI version, the number of processes)
- Sharing temporary files/folders across the MPI processes
- Markers which allow for skipping or xfailing tests based on whether the tests are being run under MPI

Further features will be added in the future, and contribution of features is very much welcomed.

CHAPTER 1

Usage

The important thing to remember is that *pytest-mpi* assists with running tests when *pytest* is run under MPI, rather than launching *pytest* under MPI. To actually run the tests under MPI, you will want to run something like:

```
$ mpirun -n 2 python -m pytest --with-mpi
```

Note that by default the MPI tests are not run—this makes it easy to run the non-MPI parts of a test suite without having to worry about installing MPI and mpi4py.

An simple test using the *mpi* marker managed by *pytest-mpi* is:

```
@pytest.mark.mpi
def test_size():
    from mpi4py import MPI
    comm = MPI.COMM_WORLD

    assert comm.size > 0
```

This test will be automatically be skipped unless *–with-mpi* is used. We can also specify a minimum number of processes required to run the test:

```
@pytest.mark.mpi(min_size=2)
def test_size():
    from mpi4py import MPI
    comm = MPI.COMM_WORLD

    assert comm.size >= 2
```

There are also *mpi_skip*, for when a test should not be run under MPI (e.g. it causes a lockup or segmentation fault), and *mpi_xfail*, for when a test should succeed when run normally, but fail when run under MPI.

CHAPTER 2

Markers

`pytest.mark.mpi(min_size=None)`

Mark that this test must be run under MPI.

Parameters **`min_size`** (*int*) – Specify that this test requires at least *min_size* processes to run. If there are insufficient processes, skip this test.

For example:

```
@pytest.mark.mpi(minsize=4)
def test_mpi_feature():
    ...
```

`pytest.mark.mpi_skip()`

Mark that this test should be skipped when run under MPI.

`pytest.mark.mpi_xfail()`

Mark that this test should be xfailed when run under MPI.

CHAPTER 3

Fixtures

`pytest_mpi.`**`mpi_file_name`**`(`*tmpdir*, *request*`)`

Provides a temporary file name which can be used under MPI from all MPI processes.

This function avoids the need to ensure that only one process handles the naming of temporary files.

`pytest_mpi.`**`mpi_tmp_path`**`(`*tmp_path*, *request*`)`

Wraps *pytest.tmp_path* so that it can be used under MPI from all MPI processes.

This function avoids the need to ensure that only one process handles the naming of temporary folders.

`pytest_mpi.`**`mpi_tmpdir`**`(`*tmpdir*, *request*`)`

Wraps *pytest.tmpdir* so that it can be used under MPI from all MPI processes.

This function avoids the need to ensure that only one process handles the naming of temporary folders.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index