Automatic Detection of Pseudo-Tested Methods in a Test Suite Using Fault Injection

Nicholas Tocci

November 19, 2018

Problem

How can we know if our test suites are adequate?

Coverage

Def: % of a system that has been tested.

Calculation

$$Coverage = \frac{NumberofTestedMethods}{TotalNumberofMethods}$$

High Coverage

Pseudo-tested Methods

Definition

What is a Pseudo-tested Method?

Def: It will never fail.

Detection

How Can We Detect Pseudo-tested Methods

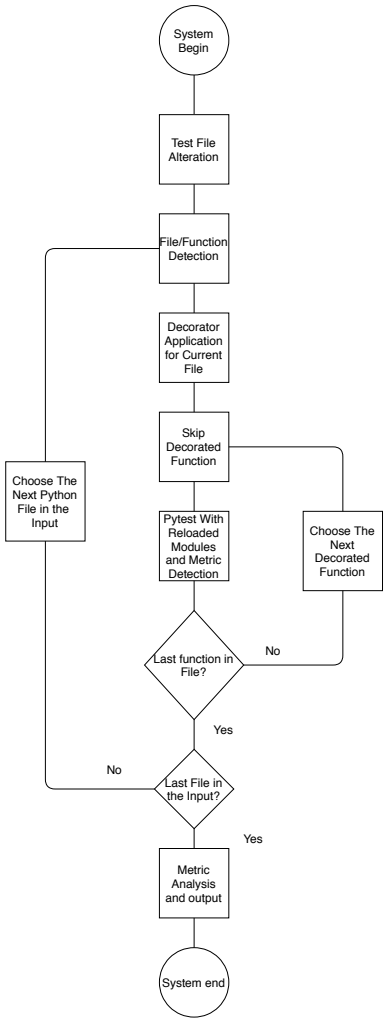
It is harder than you think!

Example of a Pseudo-tested method

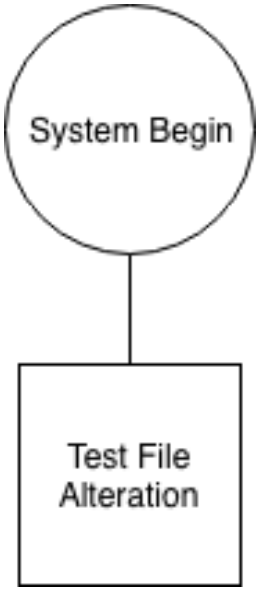

```
numbers.py:
def numberOrder(n):
    numbersSorted = sorted(n)
    return numbersSorted


test_numbers.py:
def test_numbers_ordered():
    numbers = {2,4,3,1}
    sortedNumbers = {1,2,3,4}
    orderedNumbers = numberOrder(numbers)
    assert numbers == sortedNumbers
```

A Pseudo-tested method detection tool

Flow to system



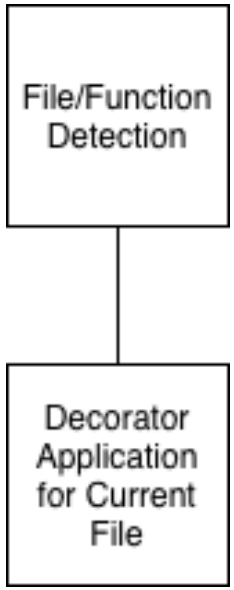
Beginning



Reloader Function


```
def reloader():
  for modules in listToReload:
    importlib.reload(modules)
```

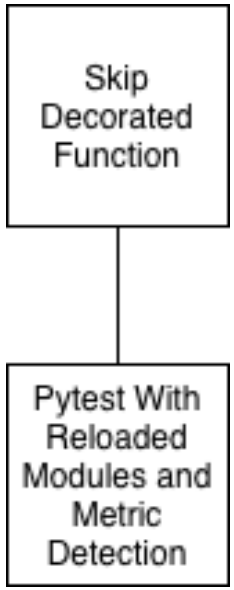
Next Steps



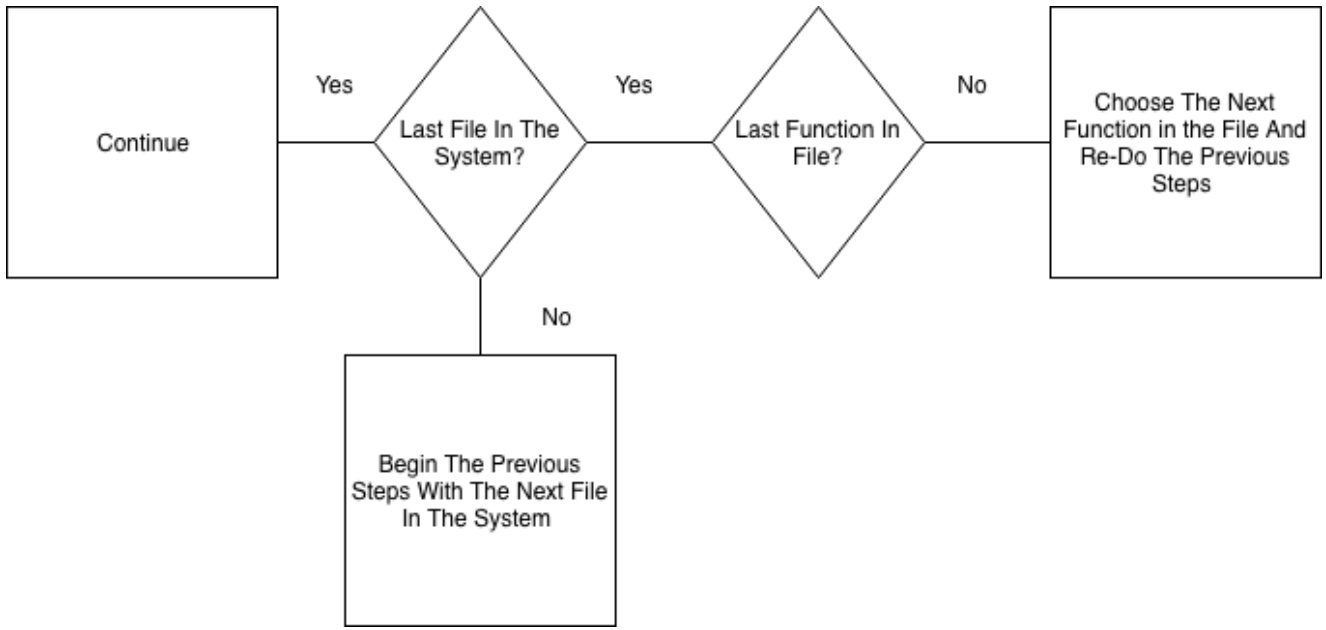
Decorator Function


```
def skipper(func):
  if(func.__name__ in functionsComplete):
    def doFunc(*args, **kwargs):
      return 5
    return doFunc

  else:
    def doFunc(*args, **kwargs):
      var = func(*args, **kwargs)
        return var
    return doFunc
```

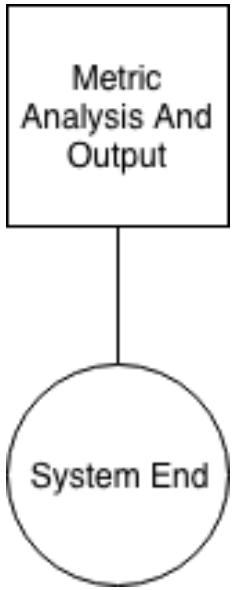
Pytesting



File and Function Checks



System End



Coverage Calculation

$$Coverage = \frac{NumberofTestedMethods}{TotalNumberofMethods}$$

Coverage Example

NUMM	NUMTM	Coverage
40	25	62.5%

Truly-Tested-Method Calculation

- **Number of Truly-Tested-Methods = NUMTTM**
- **Number of Tested Methods = NUMTM**
- **Number of Pseudo-tested Methods = NUMPTM**

$$NUMTTM = NUMTM - NUMPTM$$

Truly-Tested-Method Example

NUMTM	NUMPTM	NUMTTM
25	3	22

Adequate-Coverage Calculation

$$AC = \frac{NumberofTrulyTestedMethods}{TotalNumberofMethods}$$

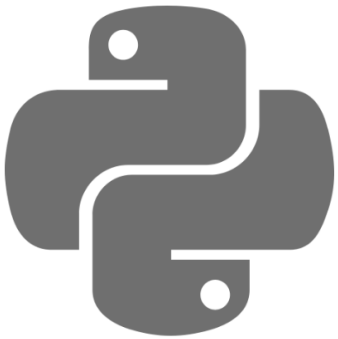
Output

NUMM	NUMTM	Coverage	NUMPTM	NUMTTM	AC
40	25	62.5%	3	22	55%

What is different?

FFiasco VS Mutation VS Fuzz VS Fault Injection VS Chaos

Why is a tool like this necessary?



- **Nothing like this exists for Python**
- **Coverage does not tell enough**

What is the impact of this research?

Coverage with fault detection

Better Understanding of Pseudo-tested Methods

Automatic Detection Tool

Github and GatorGrader

Future Research

- Return Inferencing
- Hypothesis
- Pytest Plugins

Demo

DEMO