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

Nicholas Tocci

April 3, 2019

Problem

How can we know if our test suites are effective?

👍

Coverage

Coverage

%

Def: % of a system that has been tested.

Calculation

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

High Coverage

% != 👍

Pseudo-tested Methods

Definition

What is a Pseudo-tested Method?

```
PASSED
```

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 = set([2,4,3,1])
    sortedNumbers = set([1,2,3,4])
    orderedNumbers = numberOrder(numbers)
    assert numbers == sortedNumbers
```

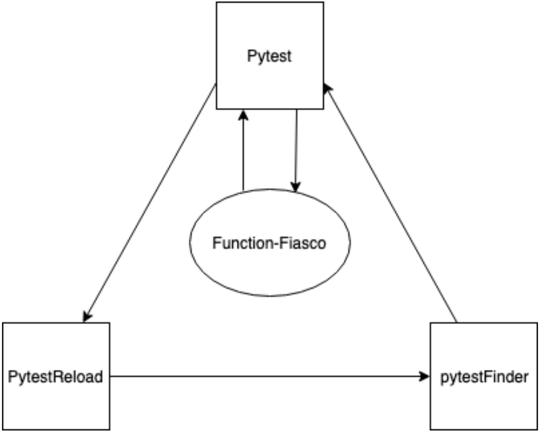
What is Function-Fiasco

A Pseudo-tested method detection tool

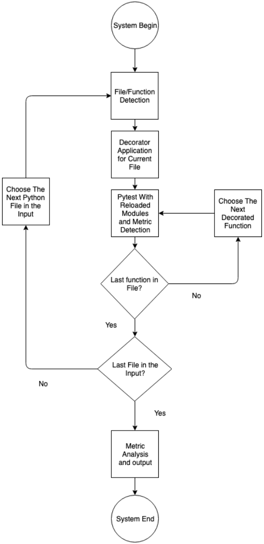
Decorator Function

```python
def skipper(func):
    functionsComplete = globs.functionsComplete
    checked = checkFunctionsComplete(func, functionsComplete)
    globs.checked = str(checked)
    if checked == True and globs.firstExe == True:
        def wrapper(*args, **kwargs):
            checkType(var, func.__name__)
            return var
        return wrapper
    elif checked == True and globs.firstExe == False:
        def doFunc(*args, **kwargs):
            var = func(*args, **kwargs)
            return checkType(var, func.__name__)
        return doFunc
    else:
        def doFunc(*args, **kwargs):
            var = func(*args, **kwargs)
            return var
        return doFunc
```

Execution Flow



Flow to system


System Begin
File/Function Detection
Decorator Application for Current File
Pytest With Reloaded Modules and Metric Detection
Choose The Next Decorated Function
Last function in File?
No
Yes
Last File in the Input?
No
Choose The Next Python File in the Input
Yes
Metric Analysis and output
System End

Coverage Calculation

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

Coverage Example

NUMM	NUMTM	Function 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**

$$NUMTTM = NUMTM - NUMPTM$$

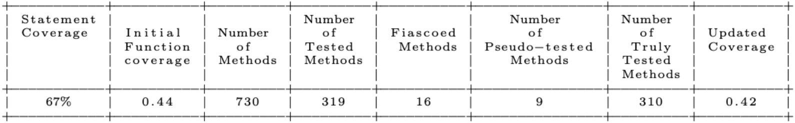
Truly-Tested-Method Example

NUMTM	NUMPTM	NUMTTM
25	3	22

Adequate-Coverage Calculation

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

Output



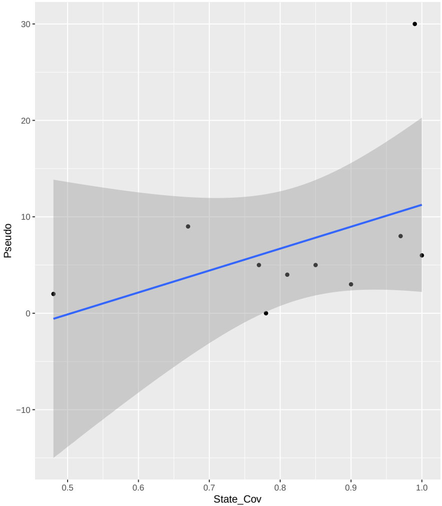
```
+------------+-----------+---------+---------+-----------+---------------+---------+-----------+
| Statement  |           |         | Number  |           |    Number     | Number  |           |
| Coverage   | Initial   | Number  |   of    | Fiascoed  |      of       |   of    | Updated   |
|            | Function  |   of    | Tested  | Methods   | Pseudo-tested |  Truly  | Coverage  |
|            | coverage  | Methods | Methods |           |    Methods    | Tested  |           |
|            |           |         |         |           |               | Methods |           |
+------------+-----------+---------+---------+-----------+---------------+---------+-----------+
|    67%     |   0.44    |   730   |   319   |    16     |       9       |   310   |   0.42    |
+------------+-----------+---------+---------+-----------+---------------+---------+-----------+
```

List of Systems

	System_Name	Num_Methods	Fiascoed	Num_Tests
1	Hashids-Python	16	10	59
2	Bleach	368	8	312
3	Pycco	22	6	17
4	Howdoi	20	2	18
5	Flashtext	42	7	23
6	Honcho	58	7	124
7	Maya	88	13	277
8	Gator	91	53	505
9	Hatch	134	14	339
10	Nikola	732	16	205

Table: List of systems used for testing.

Statement Coverage



Type Risk

	Type	Used	Found	ratio
1	Strings	69	33	47.8%
2	Booleans	49	35	73.5%
3	Ints	18	4	22.2%
4	Floats	1	0	0%

Table: Breakdown of the number of pseudo-tested method per type.

Field Connects

- Computer Science

Field Connects

- Computer Science
- Software Engineering

Field Connects

- Computer Science
- Software Engineering
- Software Testing

What is the impact of this research?

Coverage with fault detection

Better Understanding of Pseudo-tested Methods

Automatic Detection Tool

Future Research

- Different Types and Paramaterization
- Bug Fixes
- Documentation
- Further Testing

Demo

Function-Fiasco

Automatic Detection System for Pseudo-tested Methods