Introduction

Introduction

Problem

Pseudo-teste

Function

Evaluation Strategy

Conclusion

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

Nicholas Tocci

November 19, 2018

Problem

Introduction

Problem

Pseudo-teste

Methods

Function Fiasco

Evaluatio Strategy

Strategy

Conclusio

How can we know if our test suites are adequate?



Coverage

Introduction

Problem Coverage

Calculation Coverage vs Adequate

Pseudo-tested Methods

Functior Fiasco

Strategy

Conclusion

Coverage

%

Def: % of a system that has been tested.

Calculation

Introduction

Problem

Coverage Calculation

Adequate Coverage

Pseudo-tested Methods

Function Fiasco

Evaluation Strategy

Conclusion

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

High Coverage

Introduction

Problem

overage

Coverage vs Adequate Coverage

Pseudo-tested

Function Fiasco

Evaluatio Strategy

% := 16

Pseudo-tested Methods

Introduction

Problem

Pseudo-tested Methods

Defintion

Functio

Fiasco

Evaluation Strategy

Conclusio

Pseudo-tested Methods

Definition

Introduction

roblem

Pseudo-teste

Defintion

Functio

Evaluation

Conclusio

What is a Pseudo-tested Method?

PASSED

Def: It will never fail.

Detection

Introduction

Problem

Pseudo-teste Methods

Defintion Detection

Function

Evaluation Strategy

Conclusio

How Can We Detect Pseudo-tested Methods

It is harder than you think!

Example of a Pseudo-tested method

Introduction

Problem

Pseudo-teste Methods

Defintion Detection

Function Fiasco

Evaluatio Strategy

```
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
```

What is Function-Fiasco

Introduction

Problem

Pseudo-teste

Fiasco
What is
Function-Fiasco

Evaluation

Conclusio

A Pseudo-tested method detection tool



Flow to system

Introduction

Problem

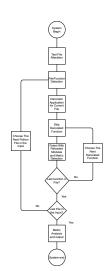
Pseudo-teste

unctior

What is

Function-Fias Flow

Evaluatio Strategy



Beginning

Introduction

Problem

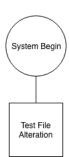
Pseudo-teste

Function

What is

Function-Fia Flow

Evaluation



Reloader Function

Introduction

Problem

Pseudo-teste

Function

Fiasco

What is Function-Fiasco

Evaluation

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

Next Steps

Introduction

Problem

Pseudo-teste

Function

Fiasco

Function-Fia

Evaluation



Decorator Function

Introduction

Problem

Pseudo-teste

Function Fiasco

What is Function-Fiasco

Evaluation Strategy

```
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

Introduction

Problem

Pseudo-tested

Function

What is Function-Fiase

Function-Fias Flow

Evaluation Strategy

Conclusion

Skip
Decorated
Function

Pytest With
Reloaded
Modules and
Metric
Detection

File and Function Checks

ntroduction

Problem

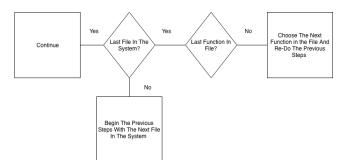
Pseudo-teste

Function

What is Function-Fias

Function-Fiasc Flow

Evaluation Strategy



System End

Introduction

Problem

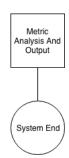
Pseudo-testeo

Function

What is

Function-Fias Flow

Evaluation Strategy



Coverage Calculation

Introduction

Problem

Pseudo-teste

Function

Evaluatio

Strategy

Coverage Calculation

Truly-Test

Calculatio Metrics

Conclusion

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

Coverage Example

Coverage Calculation

NUMM	NUMTM	Coverage
40	25	62.5%

Truly-Tested-Method Calculation

Introduction

Problem

Pseudo-teste

Function Fiasco

Evaluation Strategy Coverage

Truly-Tested-Method Calculation Metrics

Camaluaia

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

NUMTTM = NUMTM - NUMPTM

Truly-Tested-Method Example

Introduction

 $\mathsf{Problem}$

Pseudo-teste

Function

Fiasco

Evaluatio

Coverage

Calculation Truly-Tested-

Method Calculation

Metrics

NUMTM	NUMPTM	NUMTTM	
25	3	22	

Adequate-Coverage Calculation

Introduction

Problem

Pseudo-teste

Function

Evaluatio Strategy

Coverage Calculation

Truly-Tested-Method Calculation

Metrics Produced

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

Output

Introduction

Problem

Pseudo-teste

Functio

Evaluatio

Strategy

Coverage Calculation Truly-Tester Method

Metrics Produced

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

What is different?

Introduction

 $\mathsf{Problem}$

Pseudo-teste

Function

Fiasco

Strategy

Conclusion

Conclusion

What is different? Why Impact Feasibility

Feasibility
Future Research

FFiasco VS Mutation VS Fuzz VS Fault Injection VS Chaos

Why is a tool like this necessary?

Introduction

Problem

Pseudo-teste

Function Fiasco

Evaluation Strategy

Conclusion

What is different?

Impact
Feasibility
Future Research



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

What is the impact of this research?

Introduction

Problem

Pseudo-teste

Function

Evaluation Strategy

Conclusion

What is

Why

Impact Feasibility Future Research %

Coverage with fault detection



Better Understanding of Pseudo-tested Methods



Automatic Detection Tool

Feasibility

Introduction

 $\mathsf{Problem}$

Pseudo-teste

Function

Evaluation

Conclusion

Conclusion

differen Why

Why Impact

Feasibility
Future Resear

Dem

Github and GatorGrader

Future Research

ntroduction

Problem

Pseudo-teste Methods

Function Fiasco

Evaluatior Strategy

Conclusion

What is different? Why Impact Feasibility Future Research Return Inferencing

- Hypothesis
- Pytest Plugins



Demo

Introduction

 $\mathsf{Problem}$

Pseudo-teste

Function

Evaluation

Conclusion

What is different

Why Impact

Future Researc

Demo

DEMO