南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

CS304 Software Engineering

Lab 6 Fault-localization

Outline

- Motivating example
- SBFL
- Tarantula
- Assignment

Motivating Example

```
1 class Code{
2     static int m1(int x){
3         if (x >= 0)
4             return x;
5         else
6             return -x;
7     }
8     static int m2(int x){
9         if (x > 1)//buggy
10            return x;
11        else
12            return 0;
13    }
14    static int m3(int x){
15        return x * x;
16    }
17 }
```

```
public void t1() {
    int a = Code.m1(-2);
    int b = Code.m2(a);
    int c = Code.m3(b);
    assertEquals(0, c);
}
```

×

```
public void t2() {
    int a = Code.m2(5);
    assertEquals(0, a);
}
```

×

```
public void t3() {
    int a = Code.m2(15);
    int b = Code.m3(a);
    int c = Code.m1(b);
    assertEquals(225, c);
}
```

√

```
public void t4() {
    int a = Code.m2(30);
    assertEquals(30, a);
}
```

√

Any thoughts?

Could we find the faulty method?

How to quantify these info?

What we have?

- e_f: the number of **failed tests executing** the program entity *e* (method, function, etc)
- e_p: the number of **passed tests executing** the program entity *e*
- n_f: the number of **failed tests** that **do not execute** the program entity *e*
- n_p: the number of **passed tests** that **do not execute** the program entity *e*

Rankings

- $e_f/(e_f+e_p)$ *"Kulczynski"*
- $e_f/(e_f+e_p+n_f)$ *"Jaccard"*
- $1-e_p/(e_f+e_p)$ *"SBI"*
- $e_f\text{^}2/(e_f+e_p)$ *"DStar2"*

Spectrum-based Fault Localization (SBFL)

- The ranking formulas reflect the information between test cases and methods, which is defined as “spectrum”.

- In spectrum-based fault localization, the methods that **rank top are recommended** to testers for efficient debugging process.

How about SBFL performance (on Defects4J)

ID	Program	#Faults	LoC	#Tests
Chart	JFreeChart	26	96K	2205
Closure	Closure Compiler	133	90K	7927
Lang	Commons Lang	65	22K	2245
Math	Commons Math	106	85K	3602
Time	Joda-Time	27	28K	4130
Real-Bug Total	5 projects	357	321K	20109

SBFL	Top-1	Top-3	Top-5
Tarantula	66	172	215
SBI	66	172	215
Jaccard	70	169	213
Kulczynski	70	169	213
Dstar2	73	171	209
Op2	77	166	206

Tarantula

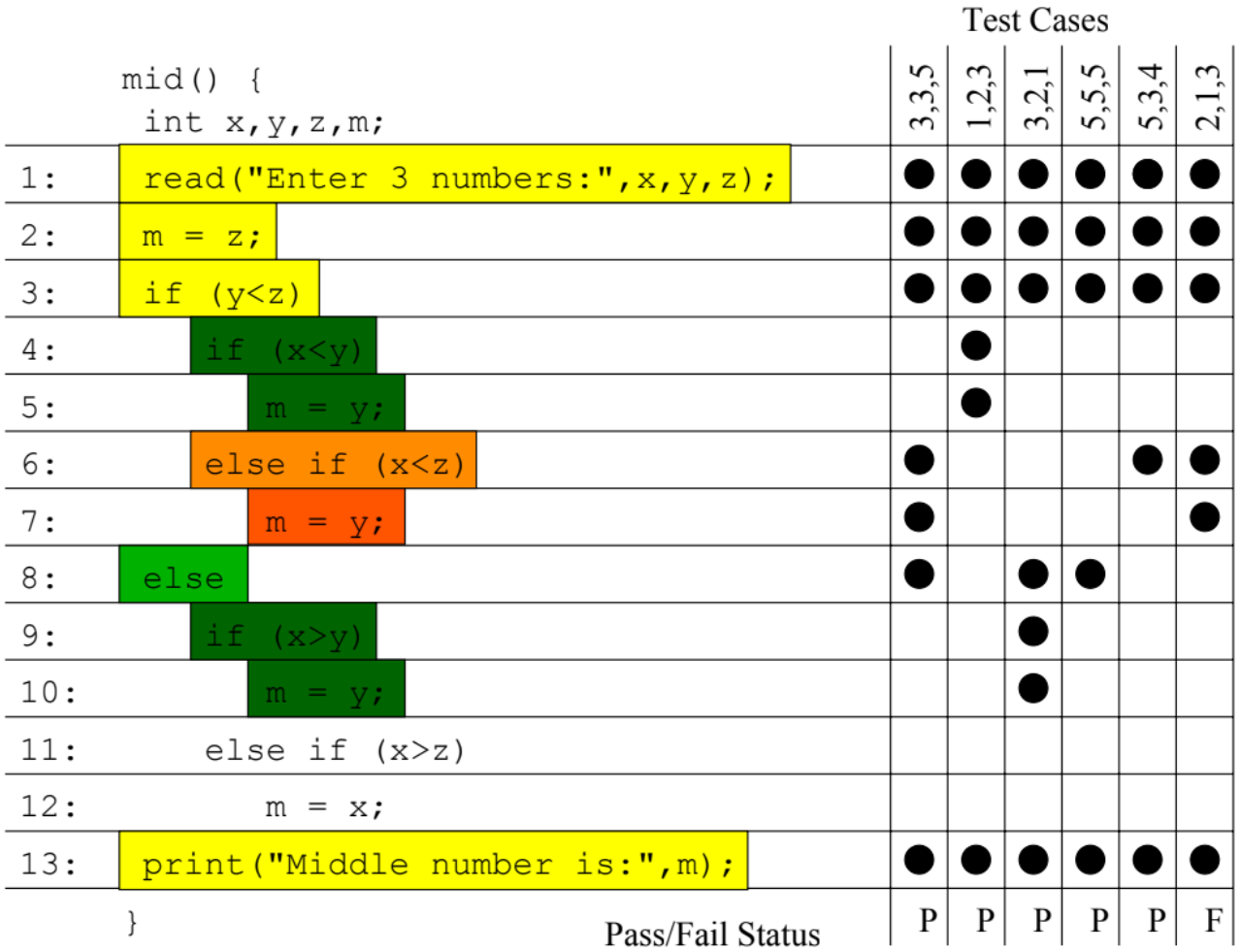
- Visualize the result of fault-localization with colors.

$$\text{color(s)} = \text{low color (red)} + \frac{\%\text{passed(s)}}{\%\text{passed(s)} + \%\text{failed(s)}} * \text{color range} \quad (1)$$

$$\text{bright(s)} = \max(\%\ \text{passed(s)}, \%\ \text{failed(s)}) \quad (2)$$

James A. Jones, Mary Jean Harrold, and John Stasko. 2002. Visualization of test information to assist fault localization. In *Proceedings of the 24th International Conference on Software Engineering* (ICSE '02). ACM, New York, NY, USA, 467-477. DOI=http://dx.doi.org/10.1145/581339.581397

Example


Test Cases
3,3,5
1,2,3
3,2,1
5,5,5
5,3,4
2,1,3
mid() {
int x,y,z,m;
1: read("Enter 3 numbers:",x,y,z);
2: m = z;
3: if (y<z)
4: if (x<y)
5: m = y;
6: else if (x<z)
7: m = y;
8: else
9: if (x>y)
10: m = y;
11: else if (x>z)
12: m = x;
13: print("Middle number is:",m);
}
Pass/Fail Status
P P P P P F

Assignment 1 (mandatory): due 23:59 pm, May 2

- PDF: Lab 6 Instruction

- Install Maven and Ant
  - **https://maven.apache.org/install.html**
  - **https://ant.apache.org/manual/index.html**
- Follow the instruction to **install a Tarantula tool**

Assignment 2 (mandatory): due 23:59 pm, May 2

- PDF: Lab 6 Instruction

- Run a demo (Triangle test suite)
- Write **Python** code to visualize the result (**PLEASE mark** the lines that are not covered by the test suite, specified with *-1 -1* output)

Assignment 3 (bonus): due 23:59 pm, May 7

- Change the code to compare **at least 4 (including the algorithm for Tarantula)** different ranking algorithms in total.
- You will need to **read the source code** of the tool in Java not only Python code to visualize. You may need to change code in *src/tarantula/TarantulaSuspiciousnessCalculation.java*

- If you want to run the program **on other** test suite, please contact me.

- **To submit:** code, report (algorithms, results including *statistics, tables and plots)* and analysis if possible.
- You will receive at most **double points (2*2)** of this lab

Tips

- Manage research papers: Mendeley (portable, sync, notes…)
- What is a research postgraduate life?
  - Reading: The PhD Grind (by Philip Guo)
- Should I apply for a Ph.D program? How to apply?
  - Reading: Applying to Ph.D. Programs in Computer Science (by CMU)