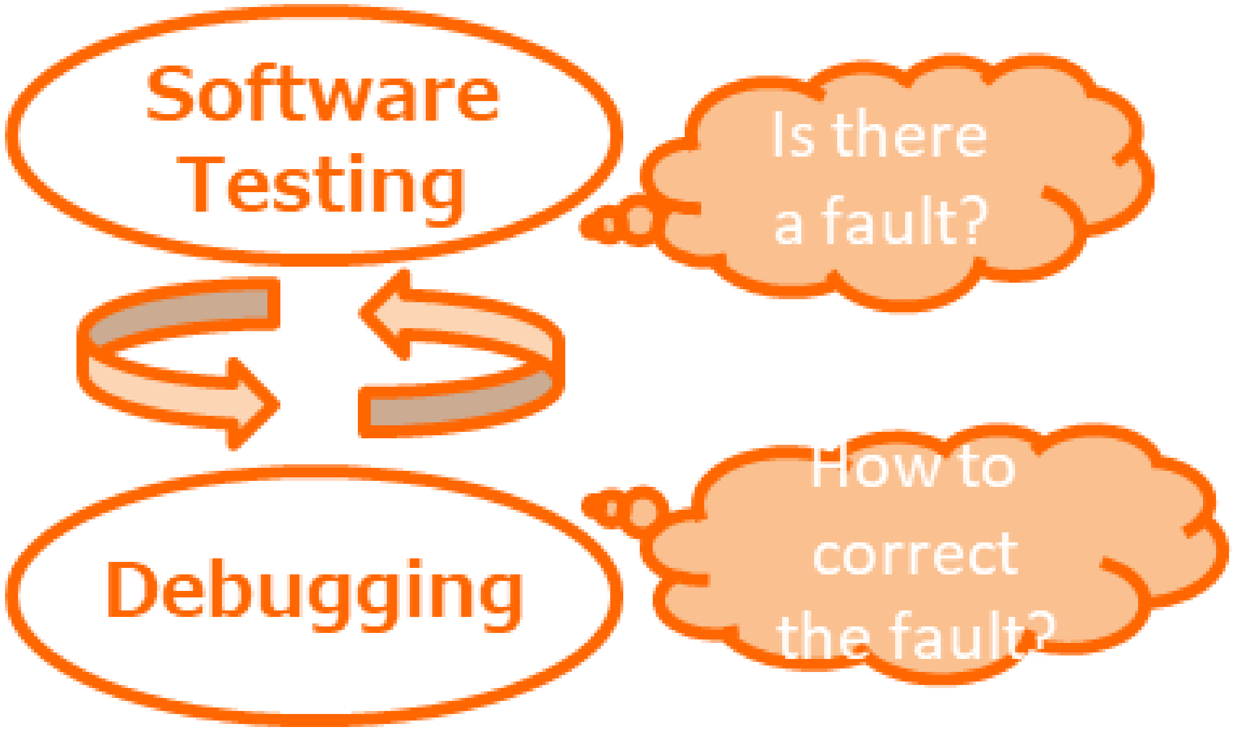
Program Spectra Visualization and Analysis for Fault Localization

Xiao-Yi Zhang, Paolo Arcaini, and Fuyuki Ishikawa

Background

Software Debugging:

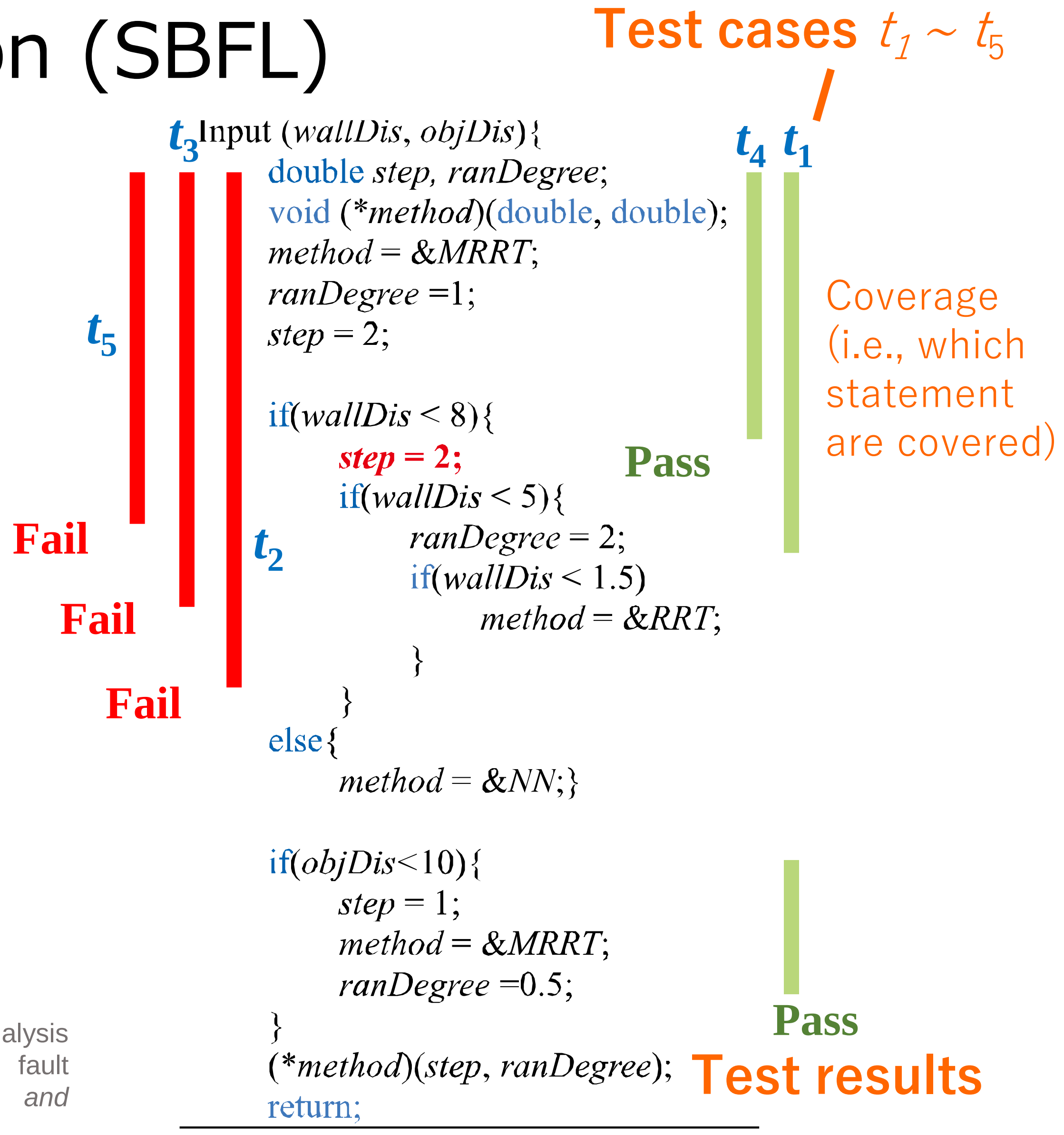
1. Fault localization:
   where is the fault?
2. Understanding:
   why is it a fault?
3. Repairing:
   how to remove the fault?



Our Topic: Automated Fault Localization

Baseline Technique -- Spectrum-Based Fault Localization (SBFL)

PG : the subject program;
$s \in S$:statements;
s^f : faulty statements;
$t \in T$: test cases;
$s \in C^t$: statements covered by t;
$o^t \in \{pass, fail\}$: the correctness of t



Xie, X., Chen, T. Y., Kuo, F. C., & Xu, B. (2013). A theoretical analysis of the risk evaluation formulas for spectrum-based fault localization. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *22*(4), 1-40.

SBFL (2)

Test information:

Test cases

Statements

S	t_1 Input: 1.5, 50	t_2 Input: 10, 5	t_3 Input: 20, 45	t_4 Input: 4, 45	t_5 Input: 7
s_1	•	•	•	•	
s_2	•	•	•	•	
s_3					•
s_4					•
$s_5=f$	•				•
s_6	•			•	•
s_7	•			•	
s_8	•			•	
s_9	•				
s_{10}		•	•		
s_{11}	•	•	•	•	•
s_{12}		•			
s_{13}		•			
s_{14}		•			
s_{15}	•	•	•	•	•
s_{16}	•	•	•	•	•
output	failing	passing	passing	failing	failing

"test t_5 covers stmtatemt s_3"

"test t_5 does not cover statement s_7"

Test Results (Pass or fail)

Test cases $t_1 \sim t_5$

```
Input (wallDis, objDis){
    double step, ranDegree;
    void (*method)(double, double);
    method = &MRRT;
    ranDegree =1;
    step = 2;

    if(wallDis < 8){
        step = 2;
        if(wallDis < 5){
            ranDegree = 2;
            if(wallDis < 1.5)
                method = &RRT;
            }
        }
    else{
        method = &NN;}

    if(objDis<10){
        step = 1;
        method = &MRRT;
        ranDegree =0.5;
    }
    (*method)(step, ranDegree);
    return;
```

t_3 t_5 t_2 t_4 t_1

Fail

Fail

Fail

Pass

Pass

Coverage (which statement are covered)

Test results

SBFL (3)

Test Info.

tests

S	t_1 Input: 1.5, 50	t_2 Input: 10, 5	t_3 Input: 20, 45	t_4 Input: 4, [illegible]	t_5 Input: [illegible]
s_1	•	•	•	•	[illegible]
s_2	•	•	•	•	[illegible]
s_3	•	•	•	•	•
s_4	•	•	•	•	•
s_5=f	•			•	•
s_6	•			•	•
s_7	•				
s_8	•			•	
s_9	•				
s_{10}		•	•		
s_{11}	•	•	•	•	•
s_{12}		•			
s_{13}		•			
s_{14}		•		[illegible]	[illegible]
s_{15}	•	•	•	[illegible]	[illegible]
s_{16}	•	•	•	•	•
output	failing	passing	passing	failing	failing

Statements

test t_4 covered stmt s_4

→ $a_{ep}(s^5)$= 0, $a_{ef}(s^5)$= 3

→ $a_{ep}(s^{10})$= 2, $a_{ef}(s^8)$= 0

test t_3 does not covered stmt s_{14}

→ F= 3, P = 2

Components' Spectra

$a_{ef}(s) = |\{t \in T \mid s \in C^t, o^t = fail\}|$

#(**F**ailure-revealing test cases that **E**xecute s)

$a_{ep}(s) = |\{t \in T \mid s \in C^t, o^t = pass\}|$

#(**P**assed test cases that **E**xecute s)

SBFL (4)

Basic Intuitions of SBFL

Basic Intuition 1： Statements covered by more failure-revealing test cases are more likely to be faulty
Basic Intuition 2： Statements covered by more passed test cases are less likely to be faulty

SBFL (5)

Basic Intuition 1： Statements covered by more failure-revealing test cases are more likely to be faulty
Basic Intuition 2： Statements covered by more passed test cases are less likely to be faulty

$a_{ef}(s) = |\{t \in T \mid s \in C^t, o^t = fail\}|$

#(**F**ailed test cases that **E**xecute s)

$a_{ep}(s) = |\{t \in T \mid s \in C^t, o^t = pass\}|$

#(**P**assed test cases that **E**xecute s)

Metrics for Scoring Statements → R(s)

Op: $$a_{ef} - a_{ep}/(P+1)$$

Tarantula: $$\frac{a_{ef}/F}{a_{ef}/F + a_{ep}/P}$$

Ochiai: $$\frac{a_{ef}}{\sqrt{F(a_{ef}+a_{ep})}}$$

Other Metrics ⋯

Statement *s* with higher R(s) value More likely to be faulty!

Above case: $R(s_5)$ is the largest. It is indeed the fault!

Challenges

- Diversity of Software Products and Faults

❖ Suitability and Applicability

What We need
➔ Understanding the Performance of SBFL
➔ Mining and Utilizing More Spectra Knowledge

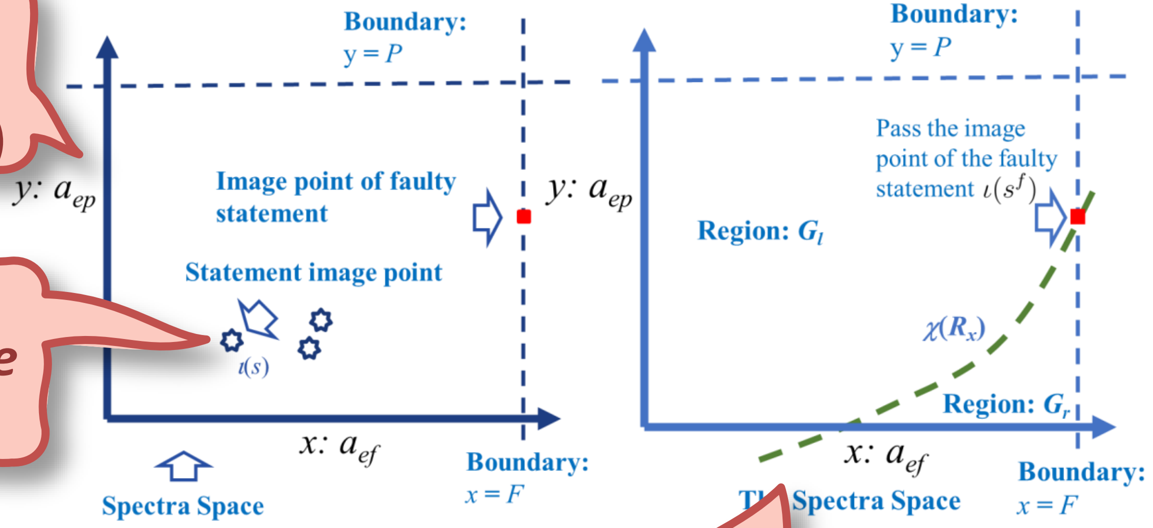
Challenges ➔ How to propose a framework to expose the characteristics during of fault propagation and fault localization

Our idea: Visualization

Visualization Framework

Present spectra information within the Spectra Space (SS)

Present statements' spectra info. as *image points*



Illustrate SBFL metrics as curves

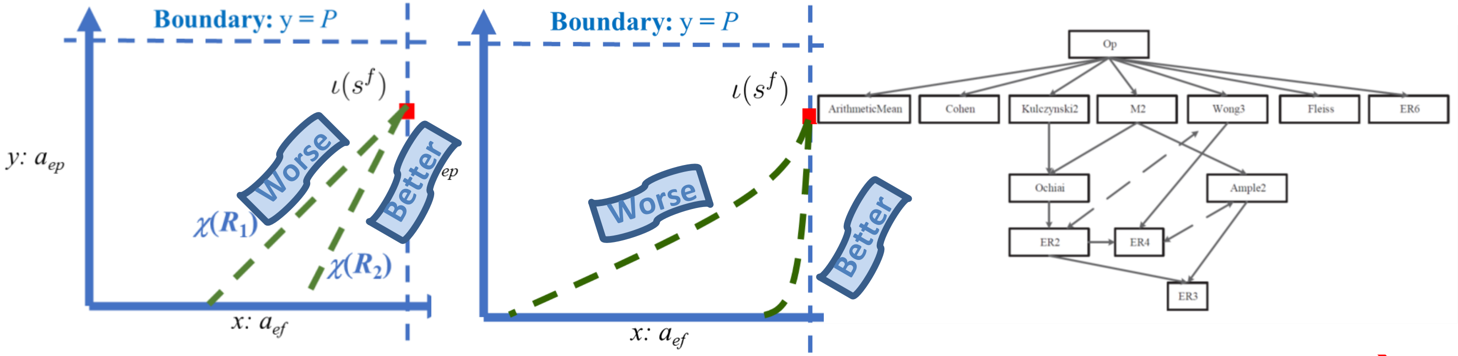
The graphical characteristics of 1) the distribution of image points, 2) the shape of Metric Performance Curves ➔

Analyzing the performances and rationales of various SBFL techniques

Metric Comparison (PRDC'19)

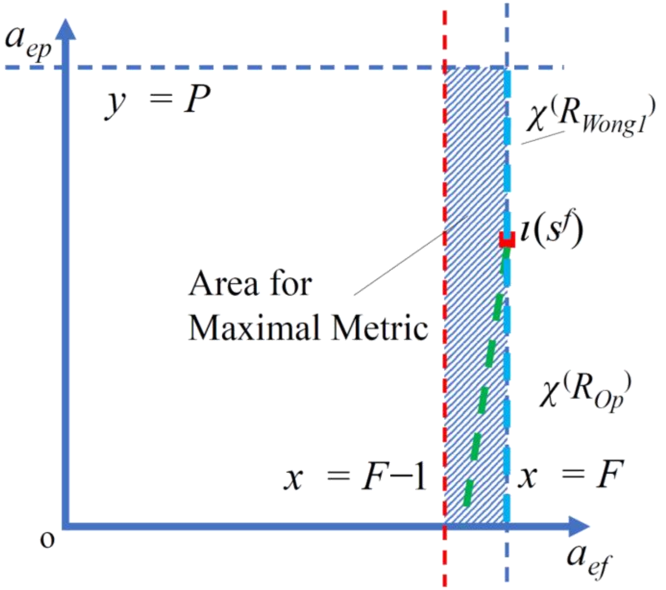
Research Problem:

1. How SBFL Performs During Fault Localisation? 2. Which Metrics are Better?



Which SBFL metric is better? ➔
The right and bottom MPCs are better!

Comparison Results (A➔B means A is better than B)



Observing (Comparison) Metrics' Performances

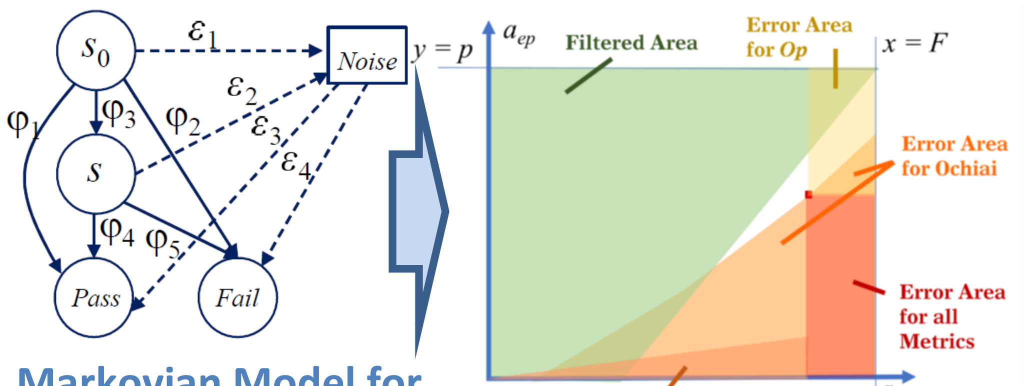
Analysis of "Maximal" Metrics

X. Zhang and Z. Zheng, "A Visualization Analytical Framework for Software Fault Localization Metrics," 2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC), Kyoto, Japan, 2019, pp. 148-14809, doi: 10.1109/PRDC47002.2019.00045.

Propagation Analysis (EASE'20)

Research Problems:

1. How the propagation of fault can affect the program spectra?
2. How the different spectra patterns can affect fault localization?

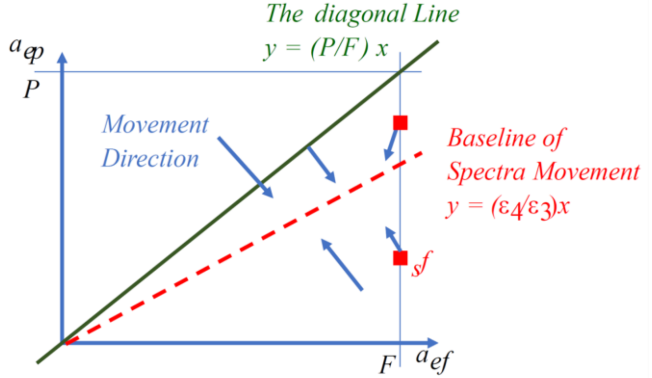


Markovian Model for Fault Propagation

Partition SS

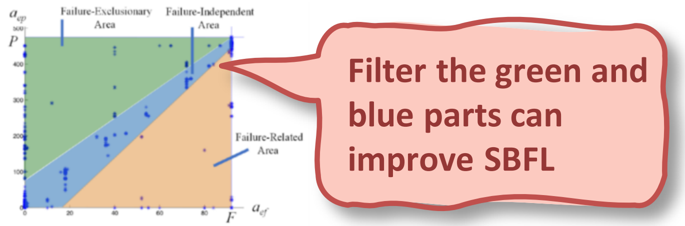
Different roles of components in fault propagation

How are they reflected in the Spectra Space



Observing the dynamics of spectra patterns under the fault interference

Points located in the failure-independent and failure exclusionary areas can be filtered!

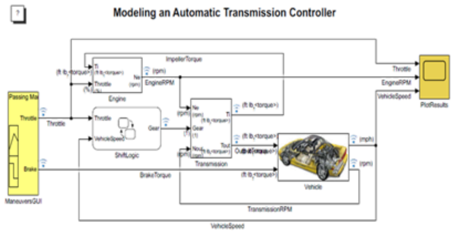


Refinement of SBFL Results

Xiao-Yi Zhang and Zheng Zheng. 2020. Exploring the Characteristics of Spectra Distribution and Their Impacts on Fault Localization. In Proceedings of the Evaluation and Assessment in Software Engineering (EASE '20). Association for Computing Machinery, New York, NY, USA, 100–109.

Future Direction

- Models of Cyber-Physical Systems (CPS), such as autonomous cars, with high safety requirement, have widely been addressed

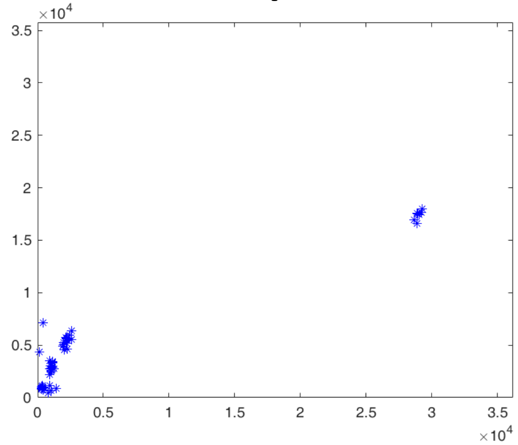


CPS model (e.g. Simulink)

	Sim_1	Sim_2	Sim_3	Sim_4			$\Phi(f_S^P)$
f_L^{p1}	0	0	0.7	1	1.4	0.3	0.82
f_S^{p1}	1	0.8	0	0	0.36	1.44	0.2
f_L^{p2}	0.2	0.1	0.3	0.6	…	…	…
f_S^{p2}	0.2	0.7	0	0	…	…	…
h	0.2	0.2	1	0.7	-	-	-

Spectra of system components for hazard

The visualization spectra of CPS components



Applying SBFL to analyse the components of CPS Systems in dealing with potential hazards (ICECCS'19)

The spectra pattern of CPS and classical program components are completely different. How should we proceed?

X. Zhang, P. Arcaini and F. Ishikawa, "Assessing the Relation Between Hazards and Variability in Automotive Systems," 2019 24th International Conference on Engineering of Complex Computer Systems (ICECCS), Guangzhou, China, 2019, pp. 190-199, doi: 10.1109/ICECCS.2019.00028.

Thank you!

ありがとうございました！

xiaoyi@nii.ac.jp
http://group-mmm.org/eratommsd