Automatically Finding Bug Locations

Program Spectra Visualization and Analysis for Fault Localization

NII

Abstract

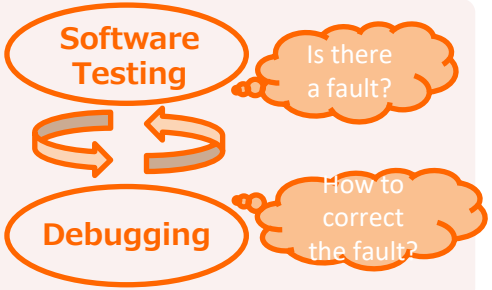
Spectrum-Based Fault Localization (SBFL) uses the information derived during testing to identify the faults existing in the subject program. However, because of the diversity of real-life programs and faults, current SBFL techniques cannot adapt all the debugging situations. Without knowing how SBFL works, we cannot get sufficient confidence to use it in practice. In our work, we propose a framework that illustrates the spectra distributions of various debugging instances and the performance of SBFL at these instances in a graphical way. Based on visualization, we can get a better analysis and understanding of the rationales of various instances in which SBFL is applied.

Preliminary

Software Debugging:

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

Automated Fault Localization



SBFL and Metrics

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

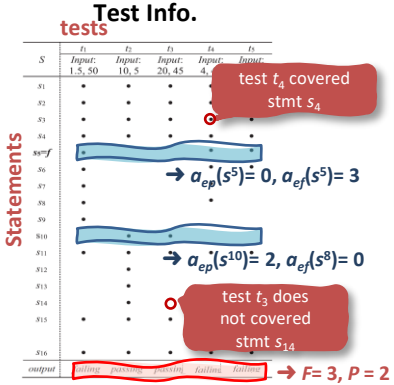
Subject Program

```
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;
```

Test Info.



Components' Spectra

$a_{ef}(s)$= | {$t \in T$ | $s \in C^t$, o^t = *fail* } |
#(Failure-revealing test cases that Execute s)

$a_{ep}(s)$= | {$t \in T$ | $s \in C^t$, o^t = *pass* } |
#(Passed test cases that Execute s)

Basic Intuitions

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

Metrics for Scoring Statements → R(s)

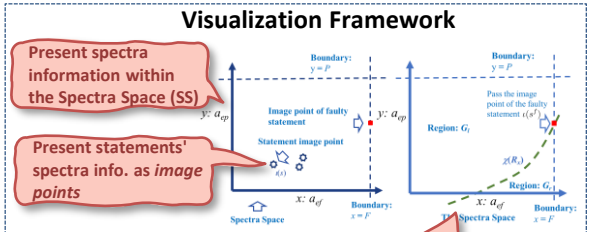
Op: $a_{ef} - a_{ep}/(P+1)$
Tarantula: $\frac{a_{ef}/F}{a_{ef}/F+a_{ep}/P}$
Ochiai: $\frac{a_{ef}}{\sqrt{F(a_{ef}+a_{ep})}}$
Other Metrics …

Larger R(s)
More likely to be faulty!

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

Research Contents

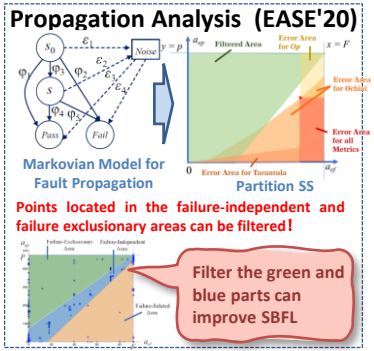
Visualization Framework



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

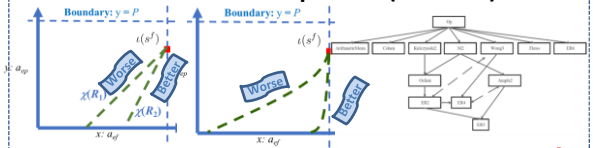
Propagation Analysis (EASE'20)



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

Filter the green and blue parts can improve SBFL

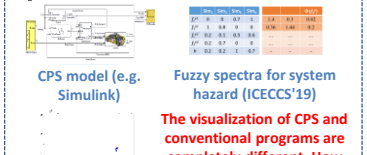
SBFL Metric Comparison (PRDC'19)



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

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

Spectra of CPS Components



CPS model (e.g. Simulink)

Fuzzy spectra for system hazard (ICECCS'19)

The visualization of CPS and conventional programs are completely different. How should we proceed?

Some Members of group G3 of ERATO MMSD project(X. Zhang, P. Arcaini, F. Ishikawa)