

AI 驱动 软件研发 全面进入数字化时代

中国·北京 08.18-19

AI+
software
Development
Digital
summit



“Bug” 的寻根溯源之旅 自动化软件缺陷定位技术

谢晓园 武汉大学
xxie@whu.edu.cn

科技生态圈峰会 + 深度研习 —— 1000+ 技术团队的选择



2023K+
全球软件研发行业创新峰会
上海站

会议时间 | 06.09-10



2023K+
全球软件研发行业创新峰会
北京站

会议时间 | 07.21-22



2024K+
全球软件研发行业创新峰会
深圳站

会议时间 | 05.17-18



K+峰会详情



会议时间 | 08.18-19

AiDD AI+软件研发数字峰会
北京站



会议时间 | 11.17-18

AiDD AI+软件研发数字峰会
深圳站



AiDD峰会详情

▶ 演讲嘉宾



谢晓园

武汉大学计算机学院教授 博士生导师

武汉大学特色化示范性软件学院副院长

武汉大学珞珈青年学者、基金委外国优秀青年学者研究基金获得者。主要研究方向为软件测试、软件缺陷定位与修复、程序分析与切片、智能软件工程等。曾解决了软件缺陷定位领域公式性能评估的瓶颈难题，被软件工程顶级期刊IEEE TSE评为全球软件蜕变测试领域十大代表性研究者之一。录用发表包括软件工程CCF A类在内的论文50余篇。以第一及通讯作者连续两年获国际软件工程顶级会议 (CCF A类) 杰出论文奖。获计算机学会NASAC青年软件创新奖、大陆首个ACM SigEvo HUMIES奖、湖北省科技进步一等奖等学术奖励。担任SCI期刊编委、客座编辑，历任ICSE/MET蜕变测试研讨会PC Chair。担任包括CCF A类会议ASE、ICSE在内的多个国际会议PC members，以及包括CCF A类期刊TSE、TOSEM 和ACM Computing Survey等在内的多个国际知名期刊审稿人。

目录

CONTENTS

1. 背景与简介
2. 实践中的关键性挑战
3. 解决思路及效果
4. 其他辅助技术
5. 总结与展望

PART 1

背景与简介

Background and Introduction



背景

现代社会软件无处不在，关键系统的软件缺陷将造成灾难性损失

2018 - 2019

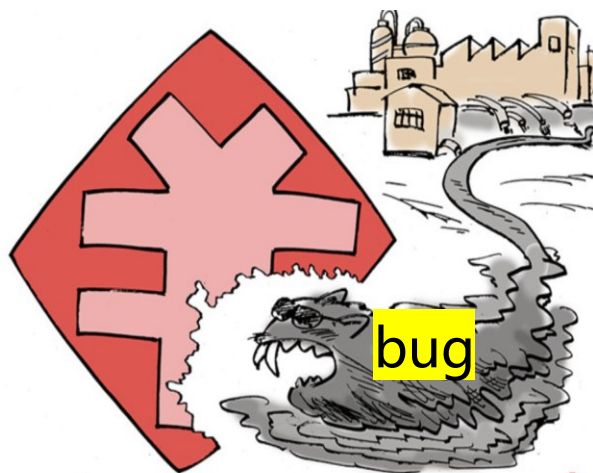
致命性航空事故



机载MCAS软件缺陷造成两起空难，共造成**346人死亡**

2022. 12

巨额的经济损失



权威机构发布报告称，2022年软件缺陷问题可能使美国经济损失**2.41万亿美元**

2023. 2

大规模车辆召回



因自动驾驶软件缺陷，特斯拉在全球范围召回**近36.3万辆汽车**



背景

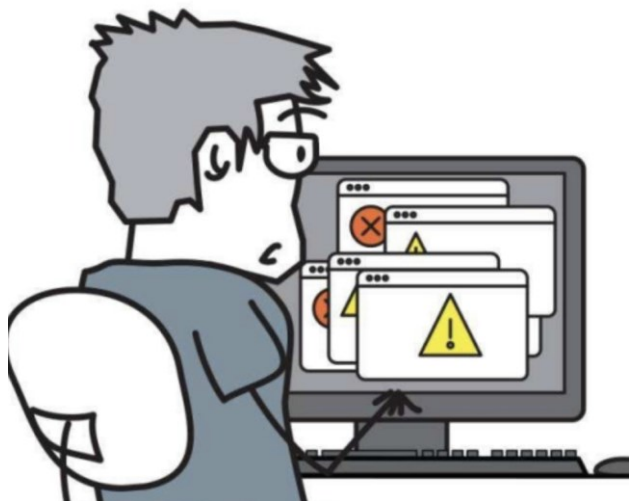
高质量软件已成为经济社会发展的迫切需求

高效的**软件缺陷定位**

是构筑高质量软件的**重要一环**



背景



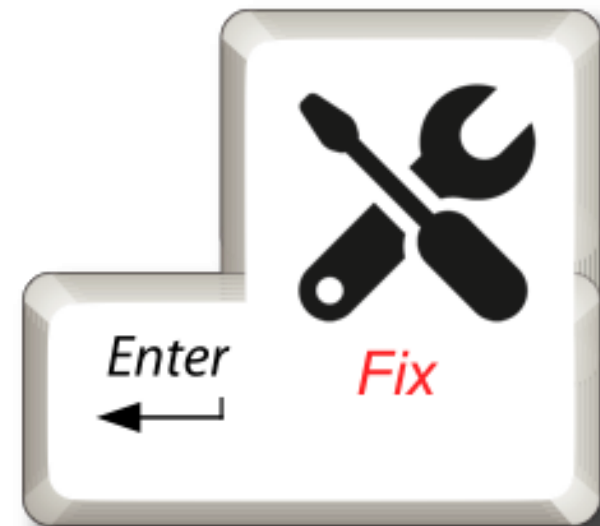
发现软件程序错误

软件测试
(Software Testing)



判断导致程序错误的底层
缺陷位置或原因
(缺陷定位)

软件调试
(Software Debugging)



对定位到的缺陷进行修复



背景

软件缺陷的产生不可避免

缺陷定位贯穿整个软件生命周期



人工缺陷定位

- ✗ 费时费力
- ✗ 效果不稳定

自动化缺陷定位

- ✓ 智能高效便捷
- ✓ 效果稳定

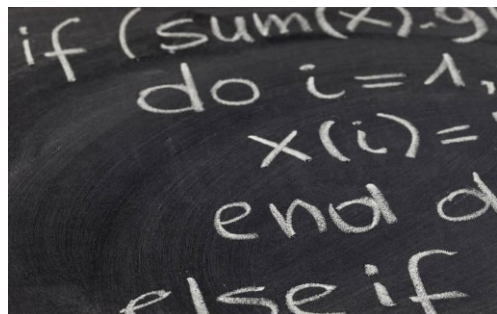
存在一系列
关键性挑战





自动化代码缺陷定位

当前主流**代码级**自动化缺陷定位技术



基于统计学的定位技术



基于程序状态的定位技术



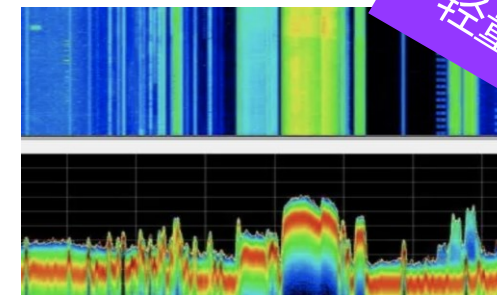
基于数据挖掘的定位技术



基于切片的定位技术



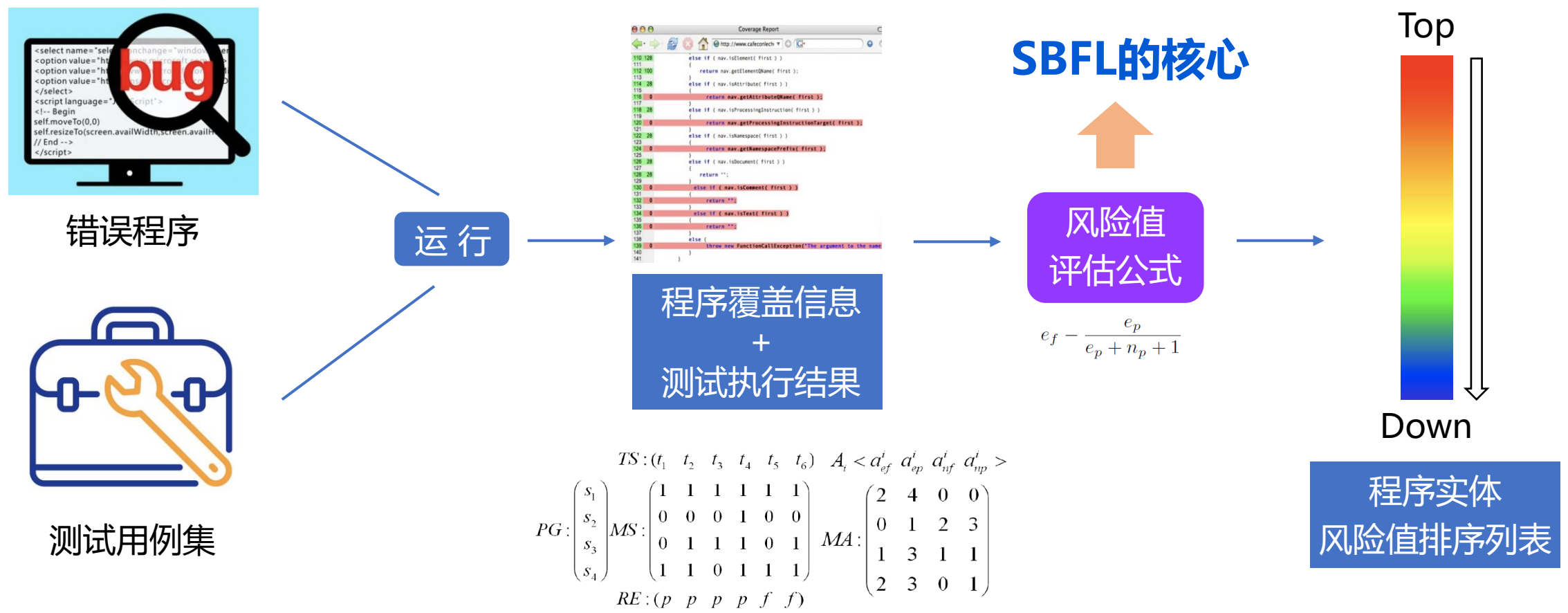
基于信息检索的定位技术



基于程序频谱的定位技术

自动化代码缺陷定位

Spectrum-based Fault Localization (SBFL) 流程图



$$TS: (t_1 \ t_2 \ t_3 \ t_4 \ t_5 \ t_6) \quad A_i < a_{ef}^i \ a_{ep}^i \ a_{nf}^i \ a_{np}^i >$$

$$PG: \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} \quad MS: \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \quad MA: \begin{pmatrix} 2 & 4 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 1 & 3 & 1 & 1 \\ 2 & 3 & 0 & 1 \end{pmatrix}$$

$$RE: (p \ p \ p \ p \ f \ f)$$

PART 02

实践中的关键性挑战

Key Challenges in Practice

挑战一：最优风险评估公式构造

挑战①

如何不通过大量实验来设计最优SBFL公式？

挑战②

如何面向Non-testable程序展开缺陷定位？



程序员面临的真实调试场景

挑战③

如何在多缺陷环境下更好地定位缺陷？

挑战④

如何在更细粒度下进行缺陷定位？



挑战一：最优风险评估公式构造

数十年来，大量SBFL公式被陆续提出，“最优公式”是目标

$$\begin{cases} -1 & \text{if } N_{CF} < N_F \\ N_S - N_{CS} & \text{if } N_{CF} = N_F \end{cases}$$

$$\frac{2N_{CF}N_{US} - 2N_{UF}N_{CS}}{N_CN_S + N_F N_U}$$

$$\frac{N_{CF}}{N_F + N_{CS}}$$

$$\frac{2N_{CF}}{2N_{CF} + N_{UF} + N_{CS}}$$

$$\frac{N_{CF}}{N_C}$$

$$\frac{N_{CF}}{N}$$

$$\frac{N_{CF} + N_{US}}{N}$$

$$N_{CF} - N_{CS}$$

$$\frac{N_{CF} + N_{US}}{N_{CF} + N_{US} + 2(N_{UF} + N_{CS})}$$

$$\frac{1}{2} \left(\frac{N_{CF}}{N_F} + \frac{N_{CF}}{N_C} \right)$$

$$N_{CF} - \frac{N_{CS}}{N_S + 1}$$

$$\frac{N_{CF}}{N_{CF} + 2(N_{UF} + N_{CS})}$$

$$\chi^2 = \frac{(N_{CF} - E_{CF})^2}{E_{CF}} + \frac{(N_{CS} - E_{CS})^2}{E_{CS}} + \frac{(N_{UF} - E_{UF})^2}{E_{UF}} + \frac{(N_{US} - E_{US})^2}{E_{US}}$$

$$\frac{2N_{CF} - N_{UF} - N_{CS}}{2N_{CF} + N_{UF} + N_{CS}}$$

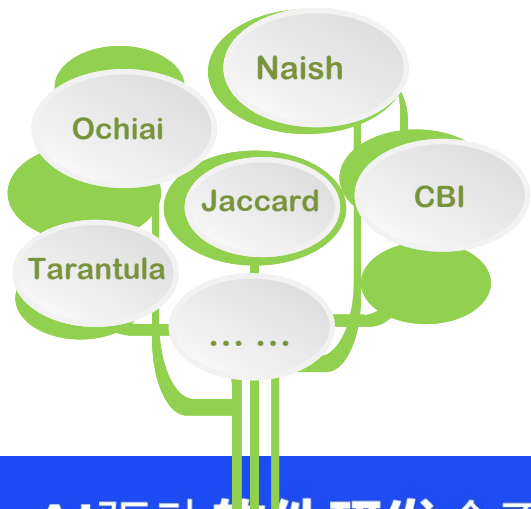
$$\frac{4N_{CF}N_{US} - 4N_{UF}N_{CS} - (N_{UF} - N_{CS})^2}{(2N_{CF} + N_{UF} + N_{CS})(2N_{US} + N_{UF} + N_{CS})}$$

$$\frac{N_{CF} (2(N_{CF} + \sqrt{N_{US}}) + \sqrt{N_{CS}})}{N_{CF} + N_{US} + 2(N_{UF} + N_{CS})}$$

$$N_{CF} \sqrt{|N_{CS} - N_{CF} + N_{UF} - N_{US}|}$$

$$\sqrt{N_{CF} + N_{US}}$$

$$N_{CF} - h, \text{ where } h = \begin{cases} N_{CS} & \text{if } N_{CS} \leq 2 \\ 2 + 0.1(N_{CS} - 2) & \text{if } 2 < N_{CS} \leq 10 \\ 2.8 + 0.001(N_{CS} - 10) & \text{if } N_{CS} > 10 \end{cases}$$



挑战一：最优风险评估公式构造

采用实验方法确定公式效果：成本高昂，随机性强且泛化性无法保证

这样的背景下，无法确定真正的“最优公式”

大量的实验程序

缺陷样例收集困难
占用大量存储空间

复杂的度量指标

度量角度不一
部分指标合理性不足

较高的硬件门槛

实验高度依赖算力
造成不必要的资源浪费

繁琐的数据分析

结论仍需人工分析
加重实验人员负担

挑战二：对Non-testable程序的定位

挑战①

如何不通过大量实验来设计最优SBFL公式？

挑战②

如何面向Non-testable程序展开缺陷定位？

```
{  
    JButton hello = new JButton( "Hello, wor  
    hello.addActionListener( new HelloBtnList  
  
    // use the JFrame type until support for t  
    // new component is finished  
    JFrame frame = new JFrame( "Hello Button"  
    Container pane = frame.getContentPane();  
    pane.add( hello );  
    frame.pack();  
    frame.show();  
    display the fra
```



程序员面临的真实调试场景

挑战③

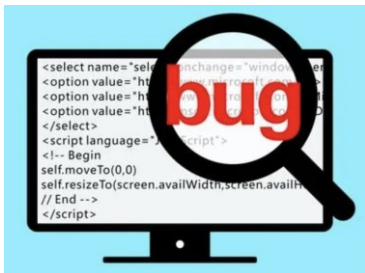
如何在多缺陷环境下更好地定位缺陷？

挑战④

如何在更细粒度下进行缺陷定位？

挑战二：对Non-testable程序的定位

测试执行结果是动态缺陷定位的必要信息



错误程序



测试用例

运行

缺陷定位
后续流程



测试输入
预期结果

有时不可得

$$PG: \begin{pmatrix} s_1 \\ s_2 \\ \cdot \\ \cdot \\ \cdot \\ s_n \end{pmatrix} \quad MS: \begin{pmatrix} 1/0 & 1/0 & \dots & 1/0 \\ 1/0 & 1/0 & \dots & 1/0 \\ & & \cdot & \\ & & & \cdot \\ & & & & \cdot \\ 1/0 & 1/0 & \dots & 1/0 \end{pmatrix} \quad TS: (t_1 \quad t_2 \quad \dots \quad t_m)$$

~~$RE: (p/f \quad p/f \quad \dots \quad p/f)$~~

进而造成定位所需信息缺失

▶ 挑战三：多缺陷环境下的定位

挑战①

如何不通过大量实验来设计最优SBFL公式？

挑战②

如何面向Non-testable程序展开缺陷定位？



程序员面临的真实调试场景

挑战③

如何在多缺陷环境下更好地定位缺陷？

挑战④

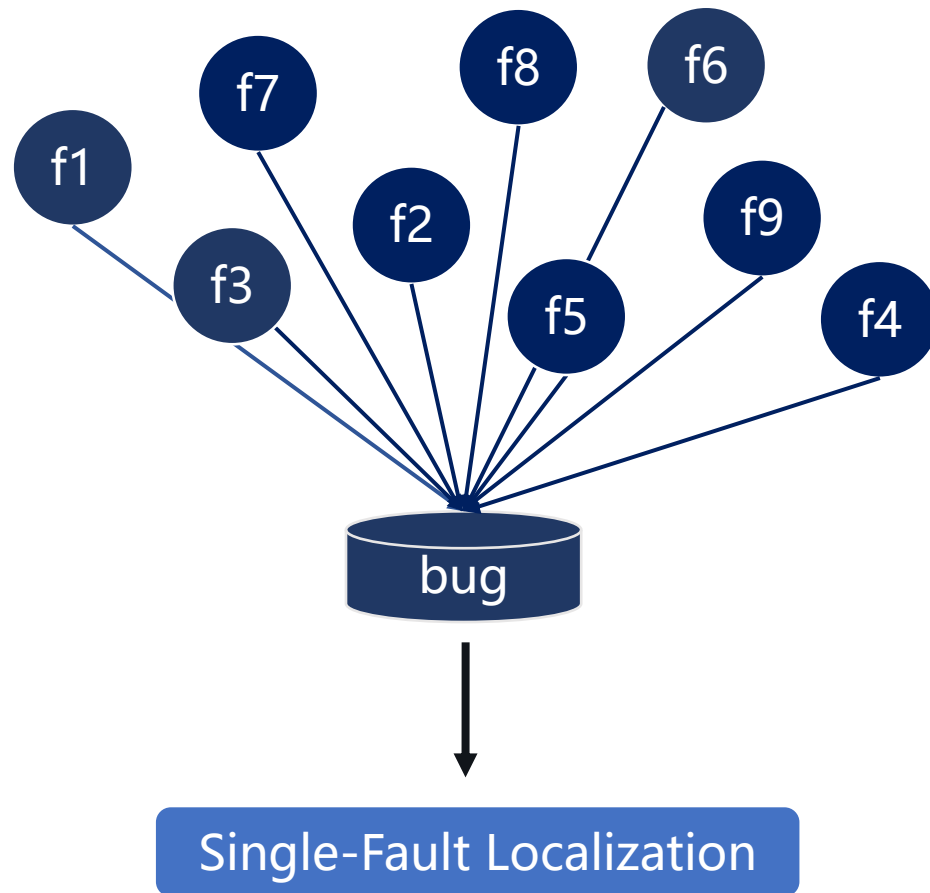
如何在更细粒度下进行缺陷定位？

挑战三：多缺陷环境下的定位

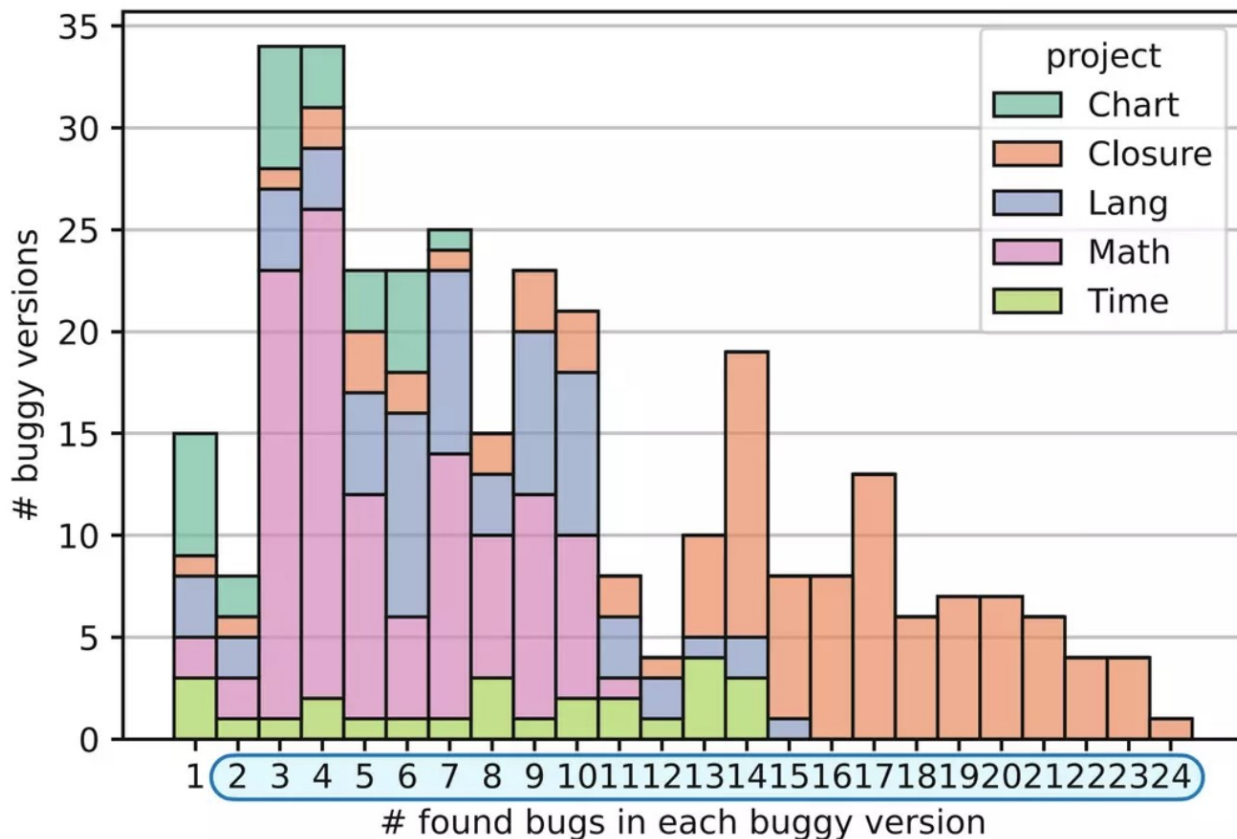
过于理想
实验中常见
单缺陷环境下

所有的失败测试用例均对应
同一个缺陷，指向清晰明显。

缺陷定位具有良好环境



挑战三：多缺陷环境下的定位



Multi-Fault Subjects

对现实开源工程的
调研表明：

真实编程实践中，
95.4%的错误程序都
含有不止一处缺陷^[1]。

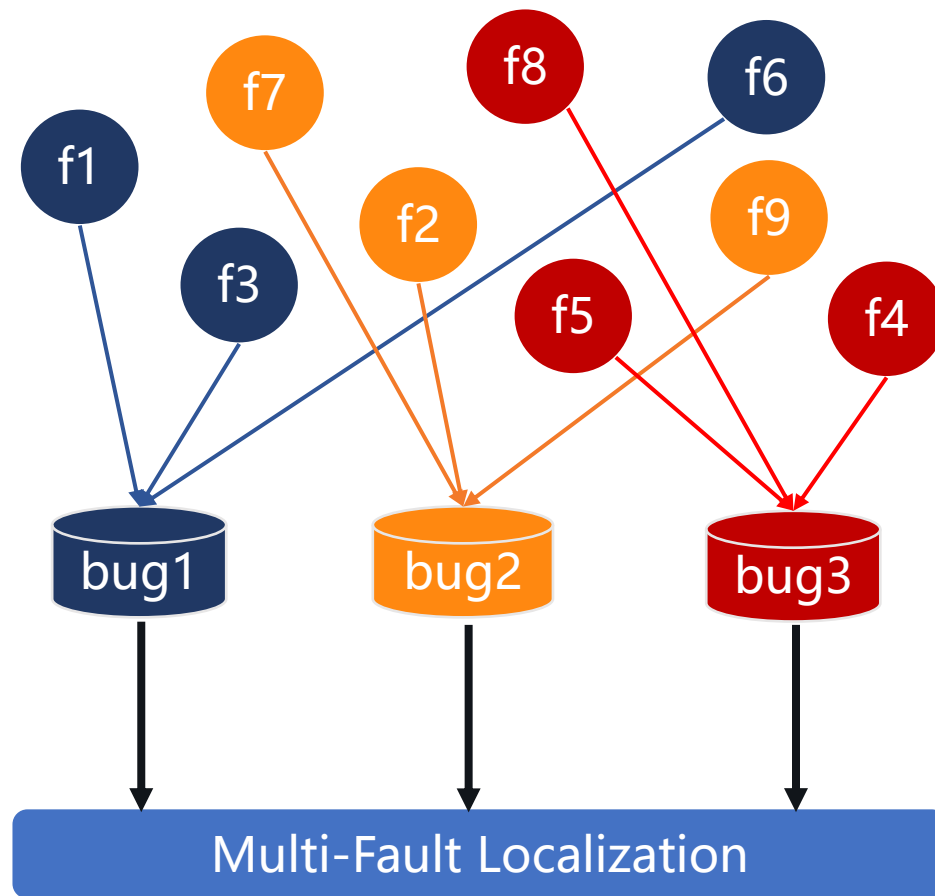
挑战三：多缺陷环境下的定位

普遍存在
工业环境常态

多缺陷环境下

各个失败测试用例对应不同
缺陷，指向混乱复杂。

给定位造成很大干扰



挑战四：细粒度缺陷定位

挑战①

如何不通过大量实验来设计最优SBFL公式？

挑战②

如何面向Non-testable程序展开缺陷定位？

```
{  
    JButton hello = new JButton( "Hello, wor  
    hello.addActionListener( new HelloBtnList  
  
    // use the JFrame type until support for t  
    // new component is finished  
    JFrame frame = new JFrame( "Hello Button"  
    Container pane = frame.getContentPane();  
    pane.add( hello );  
    frame.pack();  
    frame.show();  
    display the fra
```



程序员面临的真实调试场景

挑战③

如何在多缺陷环境下更好地定位缺陷？

挑战④

如何在更细粒度下进行缺陷定位？



挑战四：细粒度缺陷定位

Why it matters ?

较粗粒度定位结果

文件：范围过大

函数：范围仍较大

语句：难以理解缺陷根因



Faulty Position

≠

Root Cause

即便准确定位到缺陷位置，程序员仍需理解缺陷的根因

细粒度定位结果

例如变量：

直接锚定错误根因

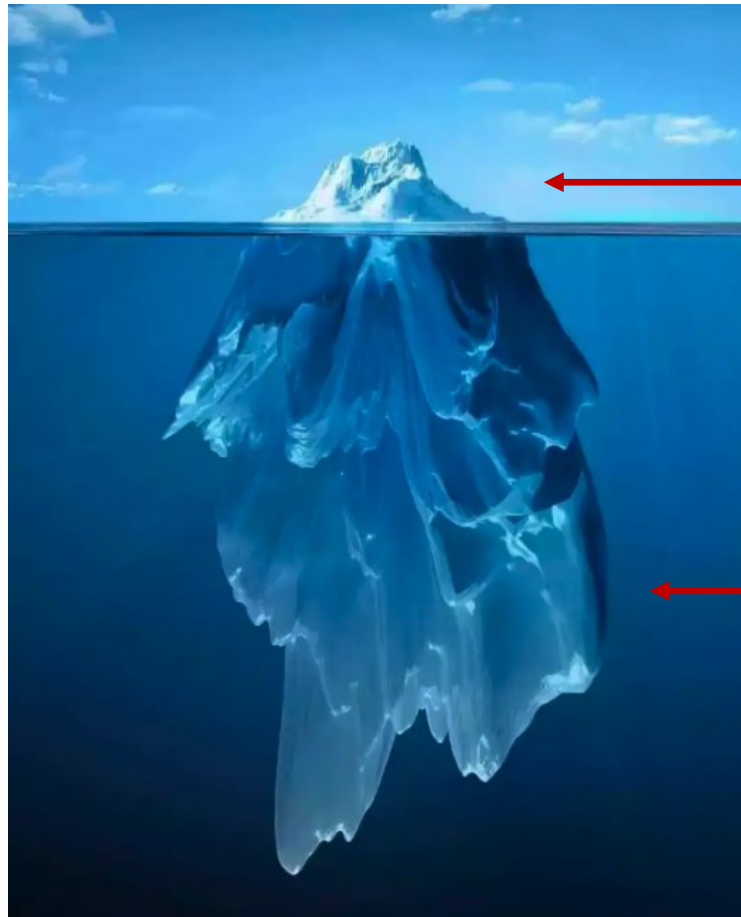
进一步减少人工排查成本

有效引导缺陷修复



▶ 挑战四：细粒度缺陷定位

Why is it difficult?



程序运行态信息

较为浅表的覆盖信息
(Coverage)

易于获取
目前应用最为广泛

更丰富、**更深层**的信息
对缺陷定位有极大增益

难以挖掘
尚未得到足够应用

PART 03

解决思路及效果

Solutions and Efficacy

解决思路

思路①

SBFL公式理论
分析框架及最优
公式生成技术

挑战②

如何面向Non-
testable程序
展开缺陷定位?

```
{  
    JButton hello = new JButton( "Hello, wor  
    hello.addActionListener( new HelloBtnList  
  
    // use the JFrame type until support for t  
    // new component is finished  
    JFrame frame = new JFrame( "Hello Button"  
    Container pane = frame.getContentPane();  
    pane.add( hello );  
    frame.pack();  
    frame.show();  
    display the fra
```



程序员面临的真实调试场景

挑战③

如何在多缺陷
环境下更好地
定位缺陷?

挑战④

如何在更细
粒度下进行
缺陷定位?

思路一：SBFL公式理论分析框架及最优公式生成技术

风险评估公式 (risk evaluation formula R)

mapping A_i into the risk value of statement s_i

SBFL最核心的组成部分

$$\begin{matrix}
 TS: (t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & A_i < a_{ef}^i & a_{ep}^i & a_{nf}^i & a_{np}^i > \\
 PG: \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} MS: \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} MA: \begin{pmatrix} 2 & 4 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 1 & 3 & 1 & 1 \\ 2 & 3 & 0 & 1 \end{pmatrix} \\
 RE: (p & p & p & p & f & f)
 \end{matrix}$$

$$\text{Tarantula} = \frac{\frac{e_f}{e_f+n_f}}{\frac{e_p}{e_p+n_p} + \frac{e_f}{e_f+n_f}}$$

所有语句风险值降序排列，依次排查程序缺陷

- **Expense** metric (E): 须被检查语句的比例, The lower, the better

$$s_{i1}, s_{i2}, \dots, s_{ik}, s_f, s_{k+2}, \dots, s_n$$

$$\text{Expense} = (k+1) / n$$

思路一：SBFL公式理论分析框架及最优公式生成技术

定义：环境无关的任意两个公式之间的关系

- 公式 R_1 优于 公式 R_2 ($R_1 \rightarrow R_2$)
 - 如果在任意程序、任意错误语句、任意测试执行情况下，公式 R_1 给出的Expense值永远小于等于公式 R_2 给出的Expense值，则称 R_1 优于 R_2
- 公式 R_1 与公式 R_2 等价 ($R_1 \leftrightarrow R_2$)
 - 如果在任意程序、任意错误语句、任意测试执行情况下，公式 R_1 给出的Expense值永远等于公式 R_2 给出的Expense值，则称 R_1 等价于 R_2

定义：最优公式 Maximality

- F 是所有公式的全集。对于一个公式 $R_1 \in F$ ，如果任意一个与其不相同的公式 $R_2 \in F$ 均满足“如果 $R_2 \rightarrow R_1$ 则有 $R_2 \leftrightarrow R_1$ ”，那么 R_1 称为 F 中的maximal。



思路一：SBFL公式理论分析框架及最优公式生成技术

给定一套测试用例集和一个公式 R ，待测程序 $PG = \{s_1, s_2, \dots, s_n\}$ 可以被划分为三个子集：

- S_B^R : 风险值比错误语句 s_f 的要高，排序在 s_f 之前的语句
- S_F^R : 风险值与错误语句 s_f 的相同，与 s_f 排序相同的语句
- S_A^R : 风险值比错误语句 s_f 的要低，排序在 s_f 之后的语句

$$S_B^R = \{s_i \in S \mid R(s_i) > R(s_f), 1 \leq i \leq n\}$$

$$S_F^R = \{s_i \in S \mid R(s_i) = R(s_f), 1 \leq i \leq n\}$$

$$S_A^R = \{s_i \in S \mid R(s_i) < R(s_f), 1 \leq i \leq n\}$$



只与R的定义有关

公式之间优劣关系：集合包含关系，是真正决定定位效果的关键因素

思路一：SBFL公式理论分析框架及最优公式生成技术

Theorem 1: 给定两个公式, 如果其对应的程序语句子集存在关系
 $S_B^{R_1} = S_B^{R_2}, S_F^{R_1} = S_F^{R_2}, S_A^{R_1} = S_A^{R_2}$, 则 $R_1 \leftrightarrow R_2$, 也即
 $R_2 \rightarrow R_1$ 且 $R_1 \rightarrow R_2$



思路一：SBFL公式理论分析框架及最优公式生成技术

Theorem 2: 给定两个公式，如果其对应的程序语句子集存在关系 $S_B^{R_1} \subseteq S_B^{R_2}$ ，
 $S_A^{R_2} \subseteq S_A^{R_1}$ ，则 $R_1 \rightarrow R_2$

Theorem 3: 给定两个公式，如果其对应的程序语句子集存在关系 $S_B^{R_1} \subsetneq S_B^{R_2}$ ，
 $S_A^{R_2} \subsetneq S_A^{R_1}$ ，则 $R_1 \rightarrow R_2$ ，且 $R_2 \not\rightarrow R_1$ ，即 R_1 严格优于 R_2



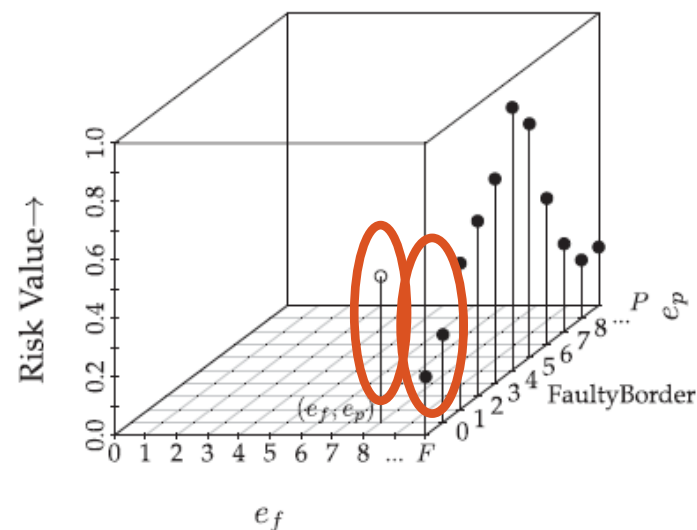
Theorem 4: 给定两个公式，如果其对应的程序语句子集存在反例使得相对应的子集间无确定包含关系，则 $R_1 \not\rightarrow R_2$ ，且 $R_2 \not\rightarrow R_1$

思路一：SBFL公式理论分析框架及最优公式生成技术

Theorem 5: 对于任意公式 $R \in F$, 当且仅当 $U_R = \emptyset$ 时, R 为 maximal

- $U_R = \emptyset$ 是 maximality 的充要条件
- 性质: 如果 $U_{R_1} = U_{R_2} = \emptyset$, 并且 $P_{R_1} = P_{R_2}$, 那么 $R_2 \leftrightarrow R_1$
- 性质: 如果 $U_{R_1} = U_{R_2} = \emptyset$, 并且 $P_{R_1} \neq P_{R_2}$, 那么 $R_2 \not\leftrightarrow R_1$

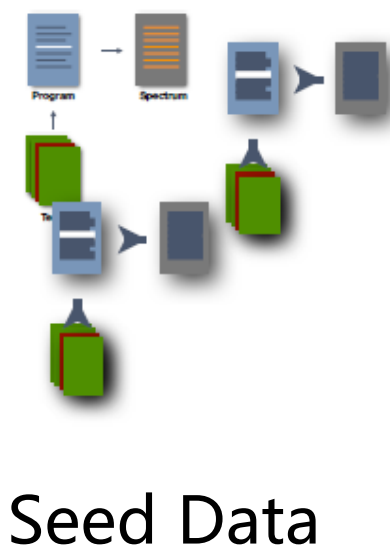
将平面上每个 $\langle e_{if}, e_{ip} \rangle$ 点映射到三维空间 Risk Value 轴的相应高度



思路一：SBFL公式理论分析框架及最优公式生成技术

使用GP来自动生成最优风险值评估公式

| Operator Node | Definition |
|---------------|--|
| gp_add(a, b) | $a + b$ |
| gp_sub(a, b) | $a - b$ |
| gp_mul(a, b) | ab |
| gp_div(a, b) | 1 if $b = 0$, $\frac{a}{b}$ otherwise |
| gp_sqrt(a) | $\sqrt{ a }$ |



Fitness
(minimise)

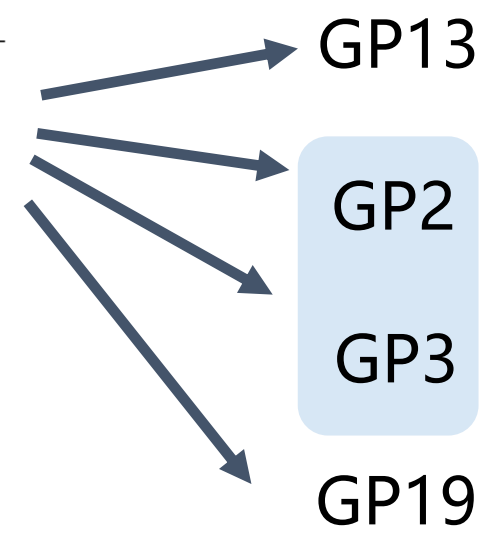
$$e_f^2(2e_p + 2e_f + 3n_p)$$

$$e_f^2(e_f^2 + \sqrt{n_p})$$

...

$$\text{fitness}(\tau, B, P) = \frac{1}{n} \sum_{i=1}^n E(\tau, p_i, b_i) \text{ (to be minimised)}$$

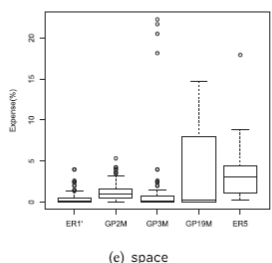
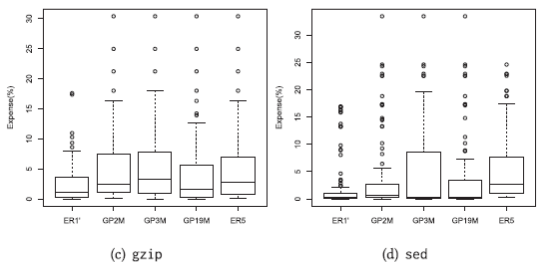
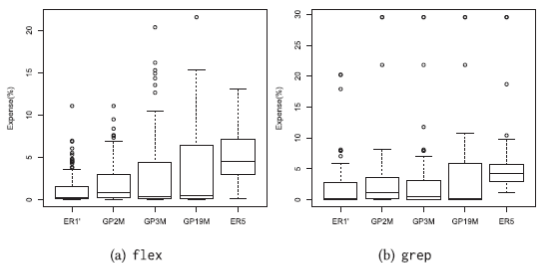
$$E(\tau, p, b) = \frac{\text{Ranking of } b \text{ according to } \tau}{\text{Number of statements in } p} * 100$$



思路一：SBFL公式理论分析框架及最优公式生成技术

公式最优化效果 (实践):

GP生成的最优公式
效果显著优于现有公式



Annual "Humies" Awards
For Human-Competitive Results
Produced By Genetic And Evolutionary Computation

14th Annual ACM SIGEVO
HUMIES Awards



GP13 是 "practical greatest" formula

GP生成

自动化
无人工干预
一次生成
多个公式



人工设计



需大量
人工参与
一次设计
一个公式

解决思路

思路①

SBFL公式理论
分析框架及最优
公式生成技术

思路②

Test oracle缺失
环境下的缺陷
定位技术

```
{  
    JButton hello = new JButton( "Hello, wor  
    hello.addActionListener( new HelloBtnList  
  
    // use the JFrame type until support for t  
    // new component is finished  
    JFrame frame = new JFrame( "Hello Button"  
    Container pane = frame.getContentPane();  
    pane.add( hello );  
    frame.pack();  
    frame.show();  
    display the fra
```



程序员面临的真实调试场景

挑战③

如何在多缺陷
环境下更好地
定位缺陷?

挑战④

如何在更细
粒度下进行
缺陷定位?



思路二：Test Oracle缺失环境下的缺陷定位技术

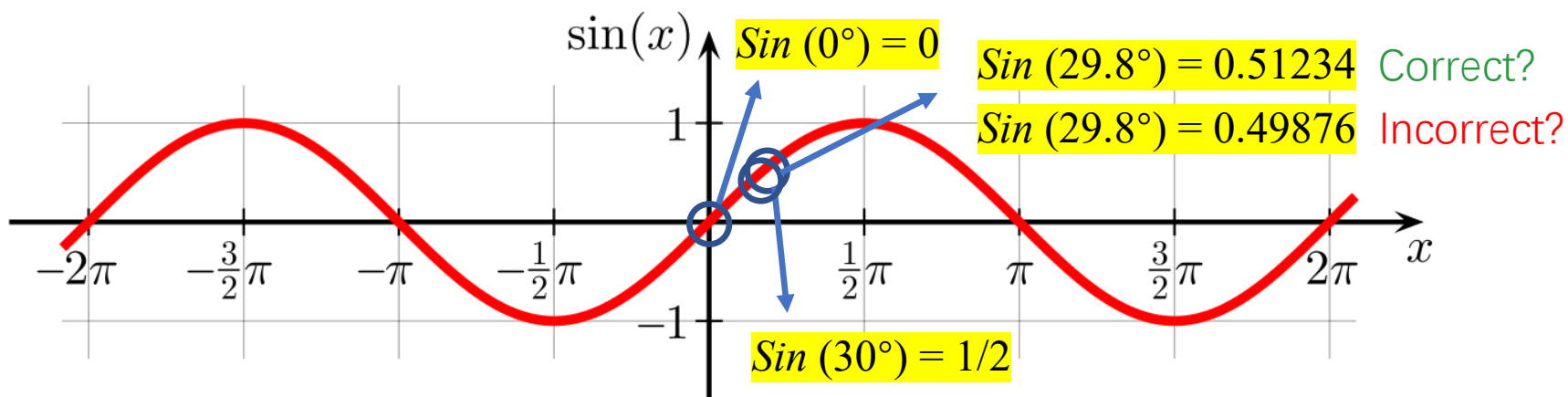
Test Oracle与Oracle问题

Test Oracle

验证输出结果**正确性**的机制

Oracle Problem

Oracle**不可得**或应用Oracle**成本高昂**



难以判断

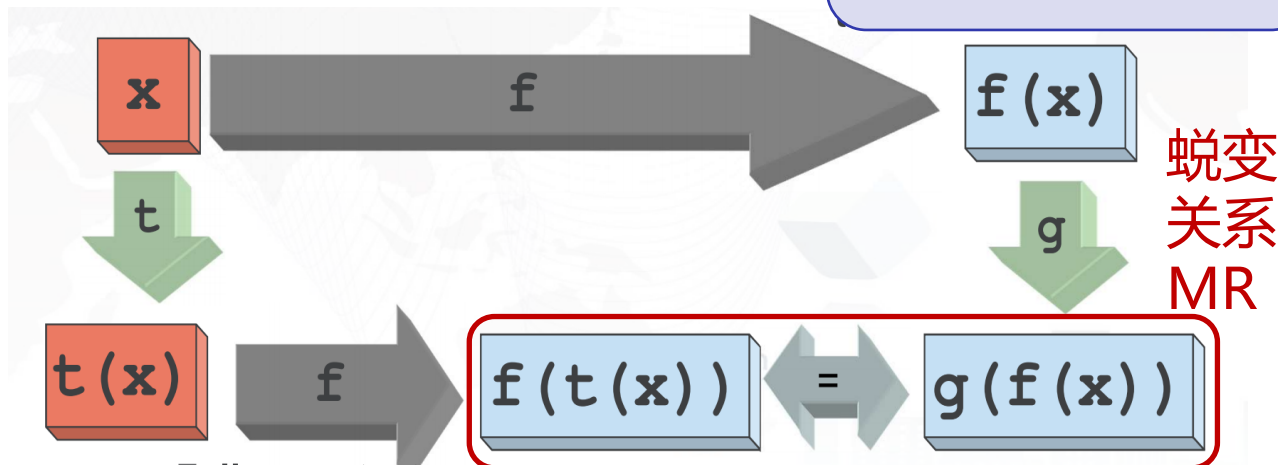
思路二：Test Oracle缺失环境下的缺陷定位技术

解决Test oracle缺失的技术：
蜕变测试 (Metamorphic Testing)

29.8°

初始测试用例
(source test case):
 $\{x, f(x)\}$

根据**现有策略**生成
初始测试用例
如Random testing



待测程序的一种内在属性
涉及多个<输入, 输出>对

$$\sin(x) = \sin(x + 360^\circ)$$

389.8°

根据初始测试用例,
按照**MR**来构造后续
测试用例

后续测试用例
(follow-up test case):
 $\{t(x), f(t(x))\}$

分别执行初始测试用例和后续
测试用例, 检查它们的输出是

否符合MR的定义

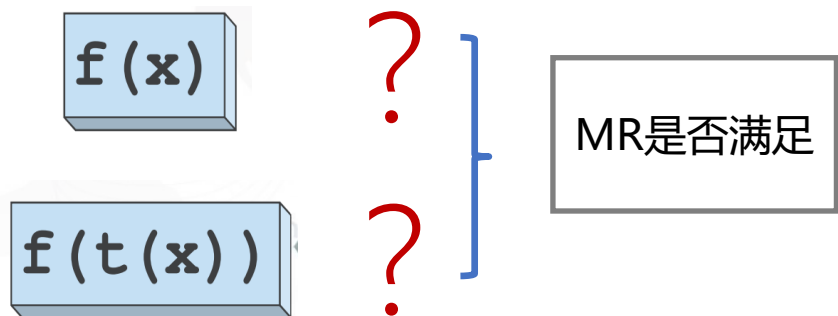


思路二：Test Oracle缺失环境下的缺陷定位技术

如何利用MT思想解决Oracle缺失环境下的缺陷定位？

~~传统SBFL：一条失效测试用例的执行轨迹中一定包含了faulty statement~~

改进SBFL：绑定于同一个MR的一组测试用例，如果其执行结果不满足MR定义，则至少其中一条测试执行轨迹包含了faulty statement



解决方案：将蜕变关系和传统的程序切片结合，构造蜕变切片：Metamorphic slice (MSlice)

$$e_mslice(MR, T^S) = \left(\bigcup_{i=1}^{ks} e_slice(t_i^S) \right) \cup \left(\bigcup_{i=1}^{kf} e_slice(t_i^F) \right)$$

$$d_mslice(v, MR, T^S) = \left(\bigcup_{i=1}^{ks} d_slice(v, t_i^S) \right) \cup \left(\bigcup_{i=1}^{kf} d_slice(v, t_i^F) \right)$$



思路二：Test Oracle缺失环境下的缺陷定位技术

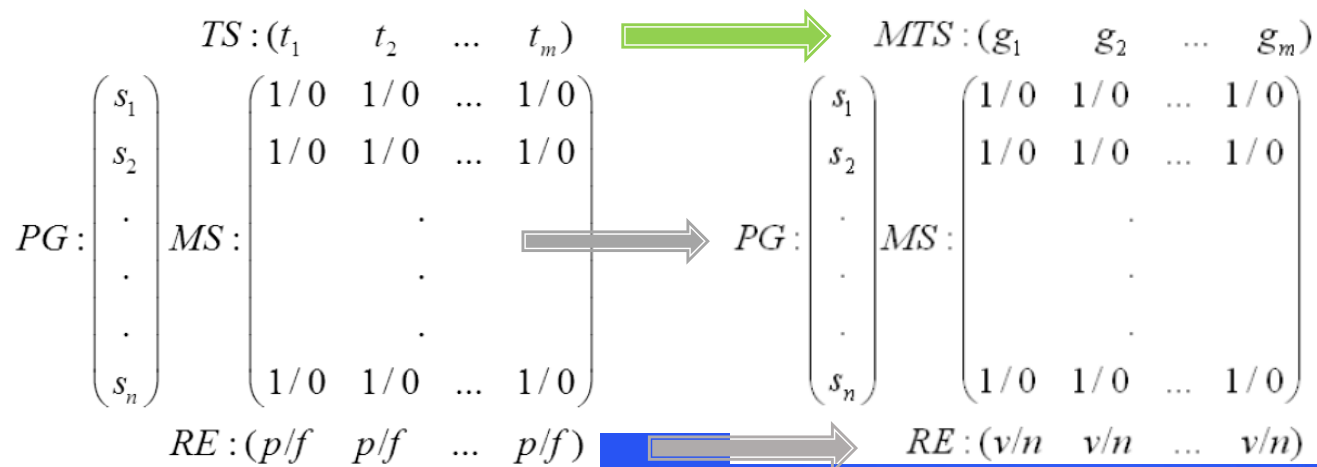
如何利用MT思想解决Oracle缺失环境下的缺陷定位？

三步替换，摆脱了对Test Oracle的依赖，使得输出结果得以补全

每一个单个测试用例 t_i \rightarrow 蜕变测试用例组 g_i

t_i 的执行切片 $\rightarrow g_i$ 的执行蜕变切片

单个测试用例的通过或失败 \rightarrow 蜕变测试用例组对MR的遵循或违反





思路二：Test Oracle缺失环境下的缺陷定位技术

MSlice成功实现了Oracle缺失条件下的高效缺陷定位

补全了缺失的频谱信息，与高度依赖Oracle的传统SBFL技术相比，实现了**几乎无损**的缺陷定位效果。

SBFL without Oracle

≈

SBFL with Oracle

Summarized results. (The paired Wilcoxon-Signed-Rank Test)

| Comparison | | BETTER | WORSE | SIMILAR |
|-------------|---|--------|-------|---------|
| MS vs. S-ST | T | 4 | 2 | 3 |
| | O | 0 | 4 | 5 |
| | J | 0 | 4 | 5 |
| MS vs. S-FT | T | 3 | 1 | 5 |
| | O | 1 | 5 | 3 |
| | J | 1 | 4 | 4 |
| MS vs. S-AT | T | 3 | 2 | 4 |
| | O | 0 | 4 | 5 |
| | J | 0 | 4 | 5 |



思路二：Test Oracle缺失环境下的缺陷定位技术

MSlice大幅突破了传统SBFL的使用瓶颈

- **彻底摆脱**Test Oracle的限制
- 将SBFL的应用环境**扩大到**被Oracle问题困扰的测试调试中
- **极大拓展**传统缺陷定位技术的灵活性和可用性



几乎所有应用领域





解决思路

思路①

SBFL公式理论
分析框架及最优
公式生成技术

思路②

Test oracle缺失
环境下的缺陷
定位技术

```
{  
    JButton hello = new JButton( "Hello, wor  
    hello.addActionListener( new HelloBtnList  
  
    // use the JFrame type until support for t  
    // new component is finished  
    JFrame frame = new JFrame( "Hello Button"  
    Container pane = frame.getContentPane();  
    pane.add( hello );  
    frame.pack();  
    frame.show();  
    display the fra
```



程序员面临的真实调试场景

思路③

基于失效聚类的
软件多缺陷隔离
技术

挑战④

如何在更细
粒度下进行
缺陷定位?



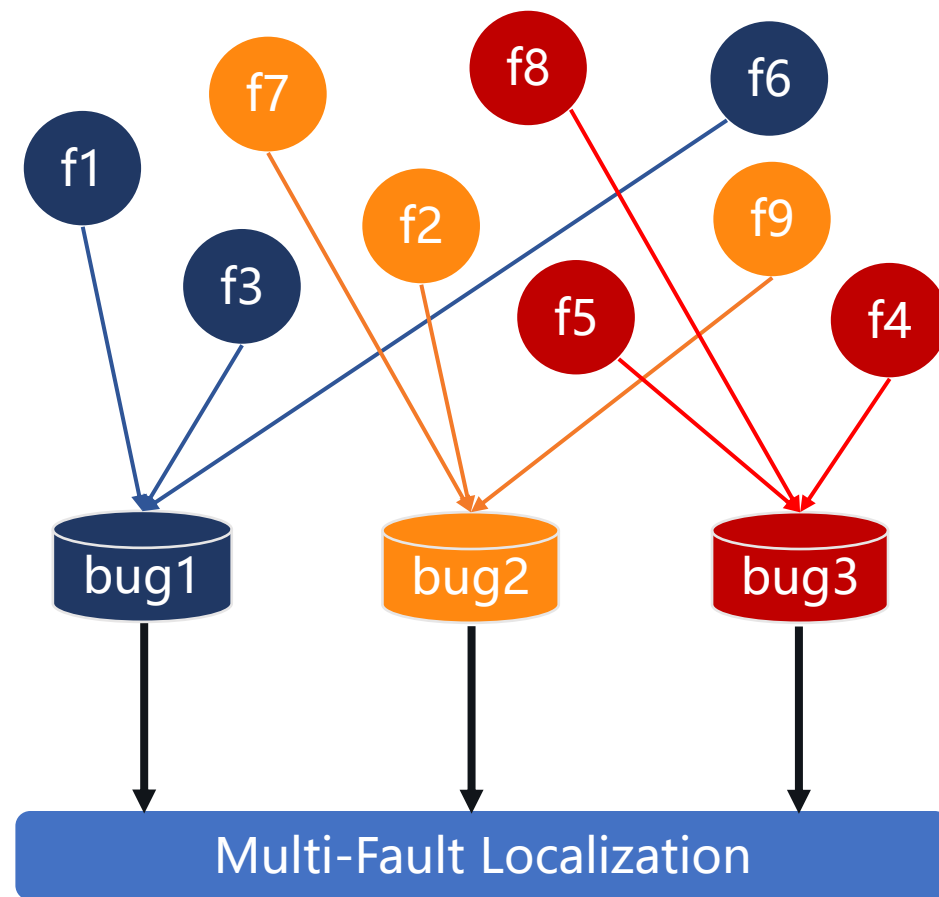
思路三：基于失效聚类的软件多缺陷隔离技术

多缺陷问题普遍存在于工业环境软件开发中

多缺陷环境下

各个失败测试用例对应不同缺陷，指向混乱复杂。

造成传统SBFL的有效性下降 [1-4]



[1] Wong W E, Gao R, Li Y, et al. A survey on software fault localization[J]. IEEE Transactions on Software Engineering, 2016, 42(8): 707-740.

[2] Gao R, Wong W E. MSeer - An Advanced Technique for Locating Multiple Bugs in Parallel[J]. IEEE Transactions on Software Engineering, 2019, 45(03): 301-318.

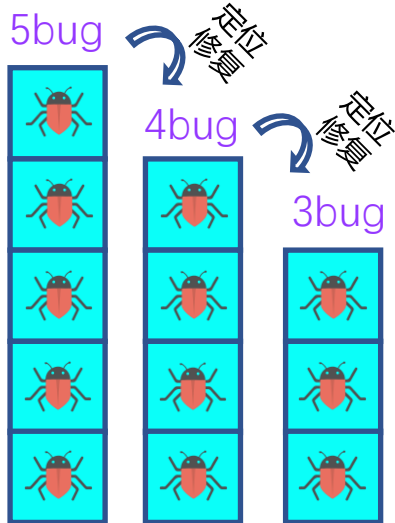
[3] Yi Song, Xiaoyuan Xie, Xihao Zhang, et al. *Evolving Ranking-Based Failure Proximities for Better Clustering in Fault Isolation*. ASE' 22

[4] Yi Song, Xiaoyuan Xie, Quanming Liu, Xihao Zhang and Xi Wu. A Comprehensive Empirical Investigation on Failure Clustering in Parallel Debugging. *Journal of Systems and Software* [JSS], 2022, 193: 111452

思路三：基于失效聚类的软件多缺陷隔离技术

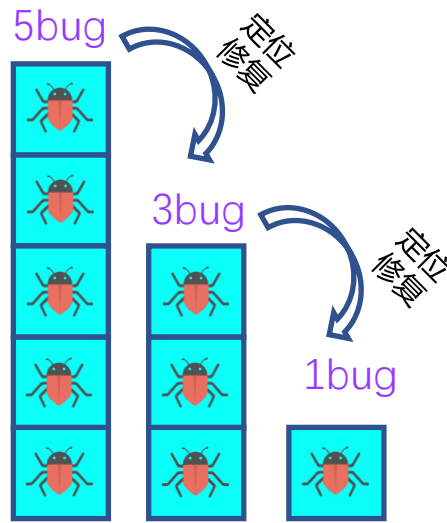
解决多故障定位问题的三种基本思路：

◆ One-bug at a time



一次处理一个缺陷

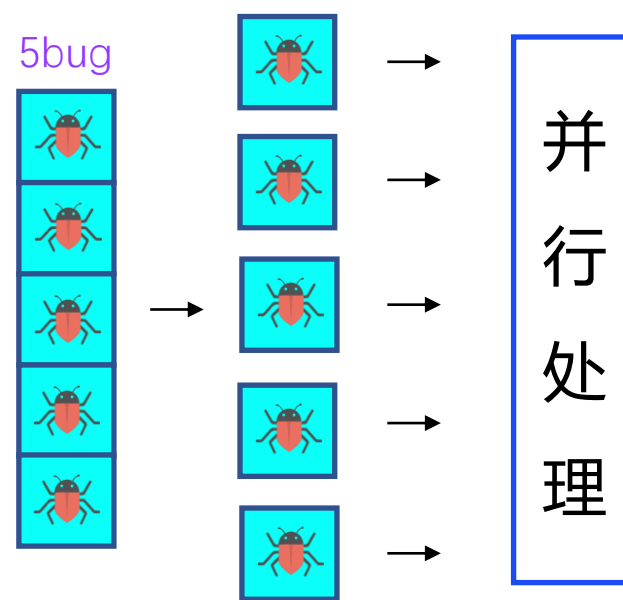
◆ Multiple-bug at a time



一次处理多个缺陷

忽略失败测试用例和缺陷之间复杂的映射关系
重复定位修复过程，直到所有测试用例均通过

◆ Parallel debugging

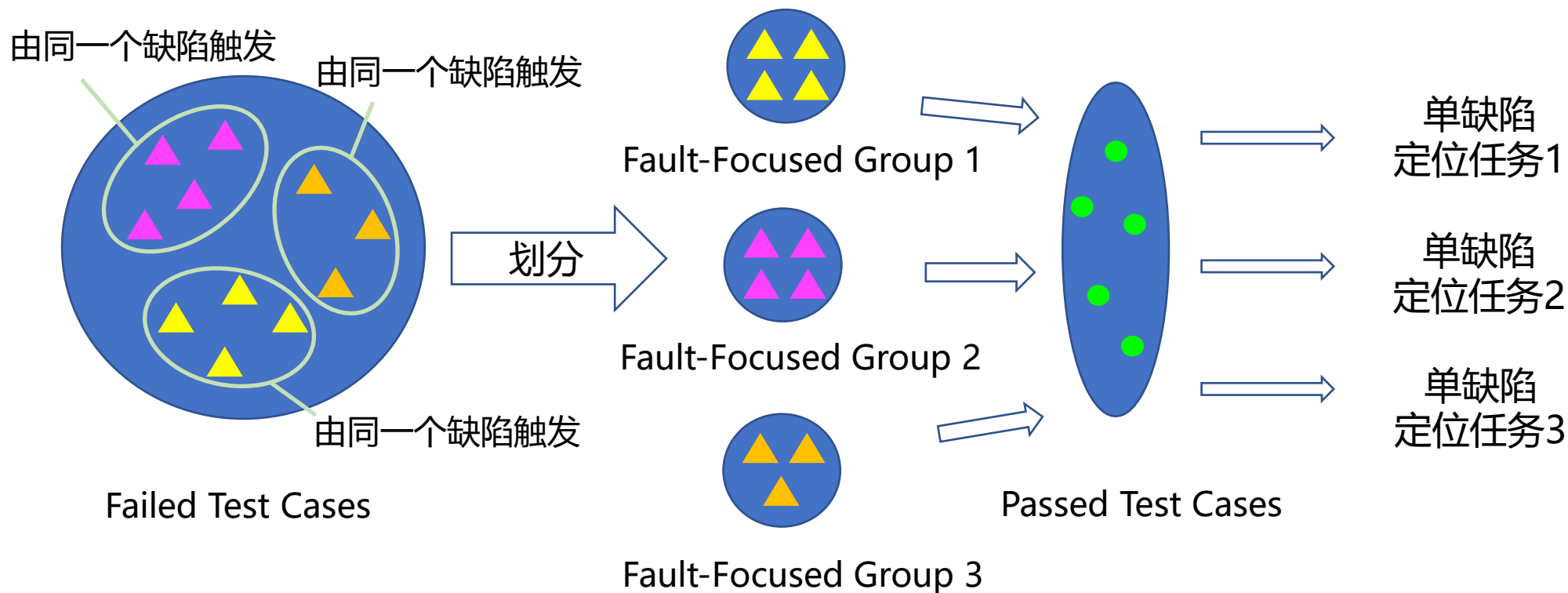


分析失败测试用例和缺陷之间复杂的映射关系
将一整个多缺陷定位任务隔离成多个单缺陷定位任务
不同程序员分别处理不同子任务

思路三：基于失效聚类的软件多缺陷隔离技术

Parallel Debugging

并行调试的核心在于如何恰当地对软件异常行为 (通常为失败测试用例) 根据其缺陷根因进行划分。这一过程也称为**Failure Indexing (失效索引)**。



思路三：基于失效聚类的软件多缺陷隔离技术

失效索引过程



如何准确提取失效行为的特征，从而对其进行数字化、形式化的建模和表征？

如何对已经被二次表征的失效行为进行相似性度量，以判断其是否由同一个缺陷触发？

如何基于上游距离信息，对所有失效行为进行聚类，以实现缺陷的隔离？



思路三：基于失效聚类的软件多缺陷隔离技术

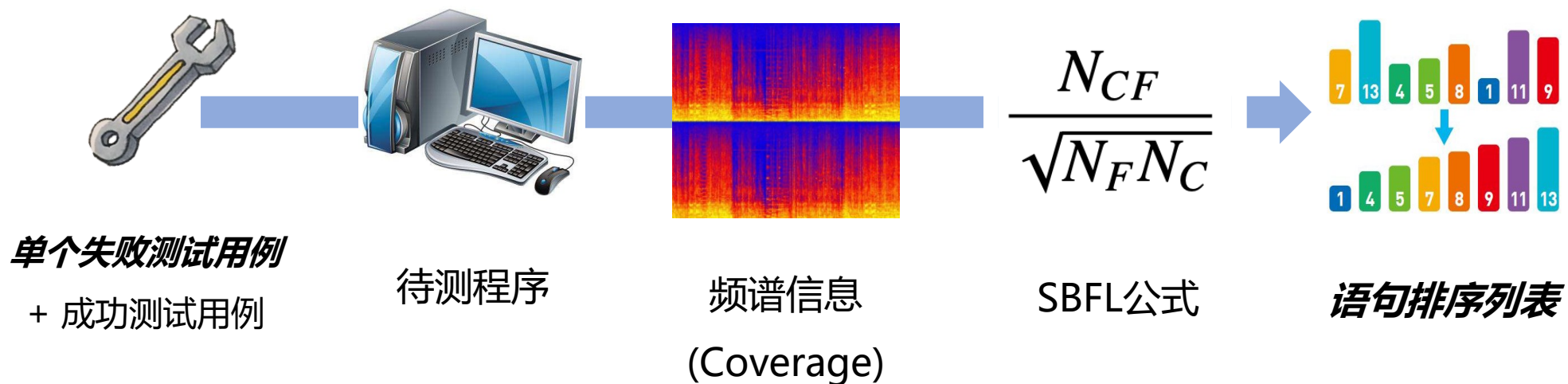
当前六种主流的Failure Proximity

| Fingerprinting Function | Distance Metric |
|---|--------------------------------|
| Failure Point-based Failure Proximity (FP) | 0-1 distance |
| Stack Trace-based Failure Proximity (ST) | 0-1 distance or edit distance |
| Code Coverage-based Failure Proximity (CC) | Jaccard, Euclid distance, etc. |
| Predicate Evaluation-based Failure Proximity (PE) | Euclidean distance |
| Dynamic Slicing-based Failure Proximity (DS) | Jaccard distance |
| Statistical Debugging-based Failure Proximity (SD) | Kendall tau distance |

已被证明
效果最优

思路三：基于失效聚类的软件多缺陷隔离技术

Statistical Debugging (SD)-based Failure Proximity将一个失效行为表征为一个**语句风险值排序列表**



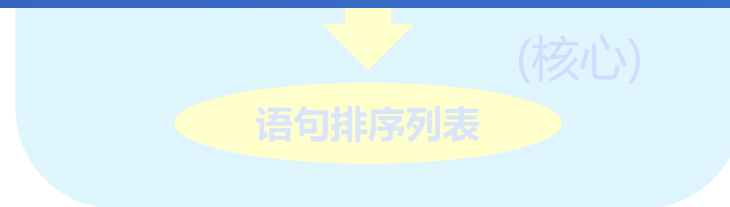
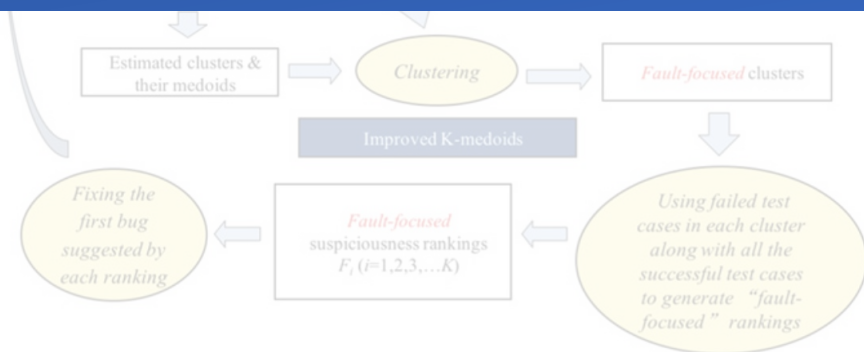


思路三：基于失效聚类的软件多缺陷隔离技术

当前SD-based Failure Proximity的代表性技术 MSeer



现有众多风险值评估公式几乎都是为缺陷定位设计，
在失效索引环境下，这些公式的效果和其缺陷定位效果一致吗？





思路三：基于失效聚类的软件多缺陷隔离技术

发现SBFL公式的失效索引能力和缺陷定位能力并不一致

| Versus R_1 | Group1 | Group2 | Group3 | Group4 | Group5 | Group6 | Group7 | Group8 | Group9 | Group10 | Group11 | Group12 |
|--------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------|---------|---------|
| Group1 | | ✓✓✓✓ | | | ✓✓✓✓ | ✓✓✓✓ | | | ✓✓✓✓ | | ✓✓✓✓ | ✓✓✓✓ |
| Group2 | | | | | ✓✓✓✓ | ✓✓✓✓ | | | ✓✓✓✓ | | ✓✓✓✓ | ✓✓✓✓ |
| Group3 | ✓✓✓✓ | ✓✓✓✓ | | | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ |
| Group4 | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ |
| Group5 | | | | | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | ✓✓✓✓ | | ✓✓✓✓ | ✓✓✓✓ |
| Group6 | | | | | ✓✓✓✓ | | | | ✓✓✓✓ | | ✓✓✓✓ | ✓✓✓✓ |

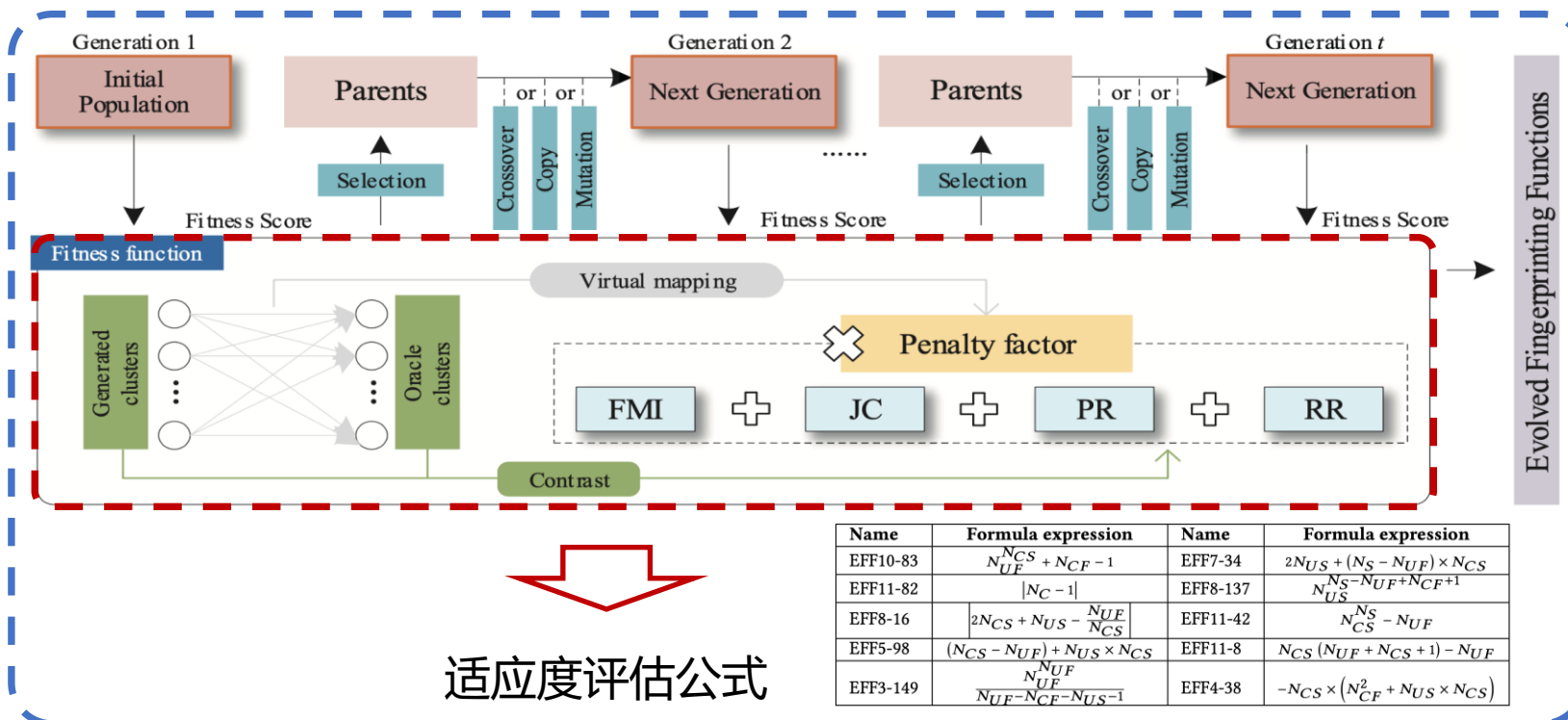
如何获得失效索引环境下适用的风险评估公式 进一步提升MSeer能力？

| R | FMI | JC | PR | RR |
|---------|---------------|---------------|---------------|---------------|
| Group1 | 116.79 | 96.96 | 117.77 | 75.23 |
| Group2 | 131.3 | 110.28 | 134.2 | 88.81 |
| Group3 | 80.33 | 68.71 | 74.97 | 60.98 |
| Group4 | 53.12 | 45.68 | 52.83 | 41.44 |
| Group5 | 148.82 | 125.06 | 137.75 | 100.96 |
| Group6 | 135.83 | 112.8 | 134.65 | 93.75 |
| Group7 | 107.87 | 89.29 | 108.51 | 68.92 |
| Group8 | 93.32 | 77.02 | 88.58 | 60.39 |
| Group9 | 138.59 | 114.87 | 137.33 | 95 |
| Group10 | 83.33 | 66.85 | 81.36 | 51.4 |
| Group11 | 155.55 | 130.36 | 154.95 | 120.39 |
| Group12 | 164.33 | 139.62 | 170.15 | 120.47 |

在缺陷定位任务中表现良好的公式，
在失效索引任务中并不一定同样好。

思路三：基于失效聚类的软件多缺陷隔离技术

基于演化的多缺陷隔离评估公式自动生成算法



适应度评估公式

| Name | Formula expression | Name | Formula expression |
|----------|---|----------|---|
| EFF10-83 | $\frac{N_{UF}^{N_{CS}} + N_{CF} - 1}{ N_{C} - 1 }$ | EFF7-34 | $\frac{2N_{US} + (N_{S} - N_{UF}) \times N_{CS}}{N_{US}^{N_{S} - N_{UF} + N_{CF} + 1}}$ |
| EFF11-82 | $\frac{2N_{CS} + N_{US} - \frac{N_{UF}}{N_{CS}}}{(N_{CS} - N_{UF}) + N_{US} \times N_{CS}}$ | EFF8-137 | $\frac{N_{CS}^{N_{S}} - N_{UF}}{N_{CS} \times (N_{UF} + N_{CS} + 1) - N_{UF}}$ |
| EFF8-16 | $\frac{N_{UF}^{N_{UF}}}{N_{UF} - N_{CF} - N_{US} - 1}$ | EFF11-42 | $-N_{CS} \times (N_{CF}^2 + N_{US} \times N_{CS})$ |
| EFF5-98 | | EFF11-8 | |
| EFF3-149 | | EFF4-38 | |

Algorithm 1 Evolution Framework

Input:

the size of population n , the maximum tree depth of EFF d , fitness function FF , the size of selected parents p , crossover rate c , copy rate o , mutation rate m , stopping criterion SC

Output:

evolved fingerprinting function

- 1: $IP \leftarrow n$ randomly initialized EFFs within the constraint of d
- 2: $CP \leftarrow IP$
- 3: **repeat**
- 4: $FS \leftarrow FF(CP)$
- 5: $PA \leftarrow$ Selecting p individuals in CP according to FS
- 6: $NG \leftarrow$ crossover, copy, or mutate individuals in PA with c , o , and m , respectively
- 7: $CP \leftarrow NG$
- 8: **until** $SC == True$

Algorithm 2 Fitness Function

Input:

an r -bug faulty version, an evolved fingerprinting function EFF , distance metric DM , the faults number estimation strategy EN , the medoids initialization strategy IM , clustering algorithm CA

Output:

fitness score

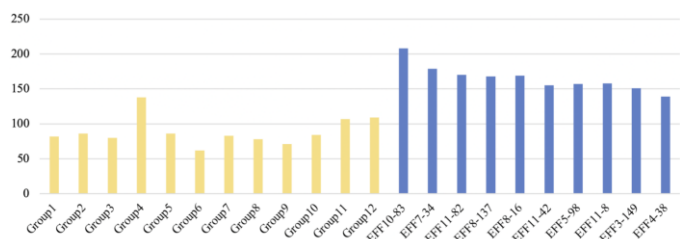
- 1: Utilize EFF to convert failed test cases into ranking lists
- 2: Calculate distances between ranking lists using DM
- 3: Employ EN to get the predicted number of faults k
- 4: **if** $k \neq r$:
- 5: set FS to be zero
- 6: **elif** $k == r$:
- 7: Leverage IM to initialize medoids and running CA
- 8: Obtain $Total_metrics$ and $PenaltyFactor$ by calculating and analyzing four external metrics
- 9: set FS to be $Total_metrics$ times $PenaltyFactor$

Yi Song, Xiaoyuan Xie, Xihao Zhang, et al. *Evolving Ranking-Based Failure Proximities for Better Clustering in Fault Isolation*, ASE'22

思路三：基于失效聚类的软件多缺陷隔离技术

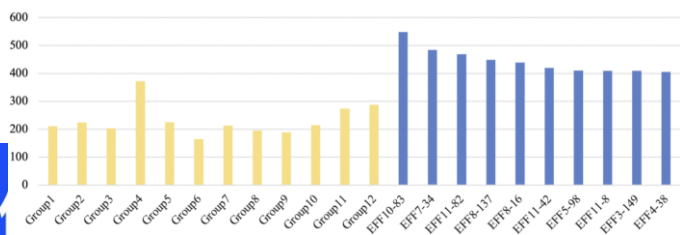
基于演化的多缺陷隔离评估公式自动生成算法

| | | EFF10-83 | EFF7-34 | EFF11-82 | EFF8-137 | EFF8-16 | EFF11-42 | EFF5-98 | EFF11-8 | EFF3-149 | EFF4-38 |
|---------------|-------|----------------------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| 按缺陷数量 分子情况 | 2-bug | FV_e 60 (22.45%) | 58 (18%) | 68 (38.78%) | 64 (30.61%) | 27 | 45 | 23 | 32 | 45 | 66 (34.69%) |
| | | Sum_{FS} 182.82 (21.75%) | 180.15 (19.97%) | 214.3 (42.71%) | 186.88 (24.45%) | 85.7 | 139.07 | 75.27 | 96.37 | 137.46 | 210.8 (40.38%) |
| 按缺陷类型 分子情况 | 3-bug | FV_e 60 (50.00%) | 45 (12.50%) | 48 (20.00%) | 44 (10.00%) | 48 (20.00%) | 46 (15.00%) | 37 | 37 | 47 (17.50%) | 43 (7.50%) |
| | | Sum_{FS} 154.69 (43.64%) | 124.43 (15.54%) | 128.43 (19.26%) | 120.27 (11.68%) | 123.14 (14.35%) | 121.93 (13.22%) | 99.85 | 98.53 | 125.11 (16.18%) | 119.21 (10.70%) |
| 按缺陷数量 分子情况 | 4-bug | FV_e 45 (2.27%) | 44 | 30 | 34 | 45 (2.27%) | 31 | 61 (38.64%) | 33 | 33 | 19 |
| | | Sum_{FS} 108.16 (2.43%) | 104.3 | 71.9 | 82.03 | 111.95 (6.02%) | 78.05 | 147.46 (39.65%) | 80.59 | 82.63 | 48.55 |
| 按缺陷数量 分子情况 | 5-bug | FV_e 43 (30.30%) | 32 | 24 | 26 | 49 (48.48%) | 33 | 36 (9.09%) | 56 (69.70%) | 26 | 11 |
| | | Sum_{FS} 103.05 (32.86%) | 74.99 | 54.37 | 59.4 | 118.42 (52.68%) | 81.2 (4.69%) | 87.54 (12.87%) | 133.92 (72.67%) | 63.98 | 26.46 |
| 按缺陷类型 分子情况 | TypeA | FV_e 73 (87.18%) | 62 (58.97%) | 54 (38.46%) | 55 (41.03%) | 62 (58.97%) | 43 (10.26%) | 57 (46.15%) | 58 (48.72%) | 46 (17.95%) | 39 |
| | | Sum_{FS} 191.64 (94.74%) | 171.41 (74.18%) | 155.58 (58.09%) | 153.72 (56.20%) | 162.3 (64.92%) | 116.26 (18.14%) | 148.79 (51.19%) | 149.95 (52.37%) | 127 (29.05%) | 116.32 (18.20%) |
| 按缺陷类型 分子情况 | TypeP | FV_e 69 (50.00%) | 52 (13.04%) | 57 (23.91%) | 53 (15.22%) | 57 (23.91%) | 65 (41.30%) | 57 (23.91%) | 51 (10.87%) | 43 | 46 |
| | | Sum_{FS} 179.34 (40.03%) | 138.12 (7.85%) | 150.89 (17.82%) | 143.81 (12.29%) | 148.62 (16.05%) | 177.19 (38.35%) | 149.79 (16.96%) | 135.99 (6.18%) | 118.67 | 134.78 (5.24%) |
| 按缺陷类型 分子情况 | TypeH | FV_e 66 (17.86%) | 65 (16.07%) | 59 (5.36%) | 60 (7.14%) | 50 | 47 | 43 | 49 | 62 (10.71%) | 54 |
| | | Sum_{FS} 177.74 (19.57%) | 174.33 (17.28%) | 162.53 (9.34%) | 151.05 (1.61%) | 128.29 | 126.8 | 111.54 | 123.47 | 163.51 (10.00%) | 153.93 (3.55%) |
| 总情况 | All | FV_e 208 (50.72%) | 179 (29.71%) | 170 (23.19%) | 168 (21.74%) | 169 (22.46%) | 155 (12.32%) | 157 (13.77%) | 158 (14.49%) | 151 (9.42%) | 139 (0.72%) |
| | | Sum_{FS} 548.72 (47.41%) | 483.86 (29.99%) | 468.99 (25.99%) | 448.58 (20.51%) | 439.21 (17.99%) | 420.25 (12.90%) | 410.12 (10.18%) | 409.41 (9.99%) | 409.18 (9.93%) | 405.02 (8.81%) |



关于 FV_e (公式准确预估缺陷数量的能力):

相比SOTA, 增长幅度从0.72%到**50.72%**不等



关于 SUM_{FS} (公式将失效按照缺陷根因进行归类的能力):

相比SOTA, 增长幅度从8.81%到**47.41%**不等



解决思路

思路①

SBFL公式理论
分析框架及最优
公式生成技术

思路②

Test oracle缺失
环境下的缺陷
定位技术

```
{  
    JButton hello = new JButton( "Hello, wor  
    hello.addActionListener( new HelloBtnList  
  
    // use the JFrame type until support for t  
    // new component is finished  
    JFrame frame = new JFrame( "Hello Button"  
    Container pane = frame.getContentPane();  
    pane.add( hello );  
    frame.pack();  
    frame.show();  
    display the fra
```



程序员面临的真实调试场景

思路③

基于失效聚类的
软件多缺陷隔离
技术

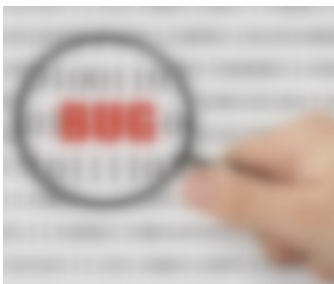
思路④

面向变量的
细粒度软件缺陷
定位技术



思路四：面向变量的细粒度软件缺陷定位技术

现有缺陷定位技术大多面临**粒度粗糙问题**



文件/函数粒度：给出File/Function的含错风险排序，程序员需自行进入File/Function内部**手动定位**缺陷位置。



语句粒度：给出语句的含错风险排序。尽管排查空间大大缩小，但即便给定错误语句，**程序员也难以理解缺陷根因。**

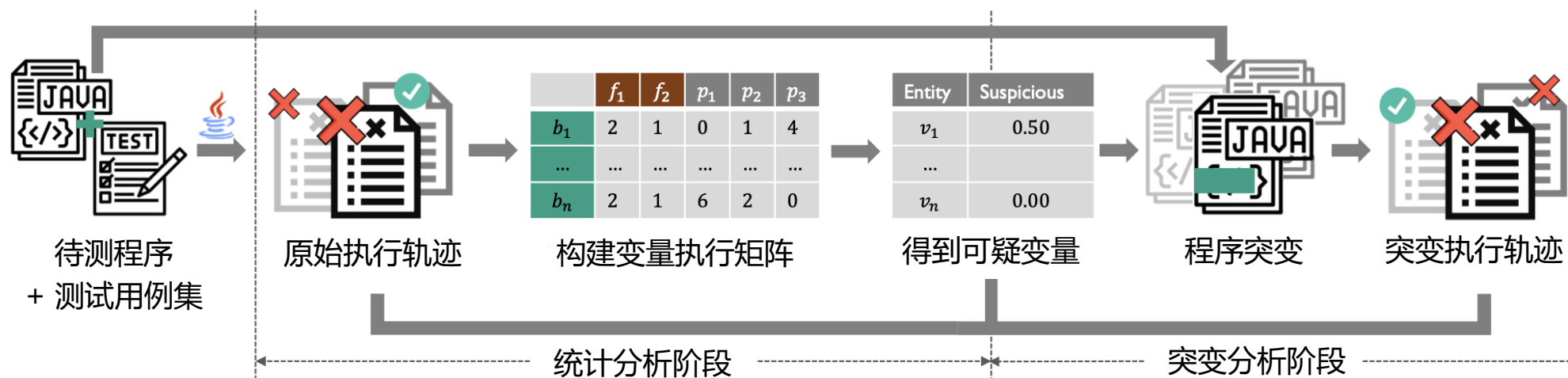


变量粒度：给出程序变量的含错风险排序。**直接锚定**缺陷根因所在，所给结果具有良好的**解释能力。**



思路四：面向变量的细粒度软件缺陷定位技术

IsoVar – 变量粒度级别的缺陷定位代表性技术



IsoVar方法流程图



思路四：面向变量的细粒度软件缺陷定位技术

IsoVar统计分析阶段：根据覆盖信息初步计算变量与失败执行的关联

| 程序块 | 测试用例 | | | |
|-------|-------|-------|-------|-------|
| | f_1 | f_2 | p_1 | p_2 |
| b_1 | 1 | 2 | 1 | 2 |
| ... | ... | ... | ... | ... |
| b_n | 0 | 0 | 2 | 0 |

程序块 b_1 在测试用例 f_2 中执行了 2 次

统计

变量在失败测试用例中出现的平均频率

变量在通过测试用例中出现的平均频率

| | f_1 | f_2 | p_1 | p_2 |
|-------|-------|-------|-------|-------|
| b_1 | 1 | 2 | 1 | 2 |
| ... | ... | ... | ... | ... |
| b_n | 0 | 0 | 2 | 0 |

计算执行路径相似度

$$\text{Cos}(V_{f_1}, V_{p_1}) \quad \text{Cos}(V_{f_1}, V_{p_2}) \quad \text{Cos}(V_{f_2}, V_{p_1}) \quad \text{Cos}(V_{f_2}, V_{p_2})$$

$$\text{Sim}_{\langle f, p \rangle} = \frac{\sum_{i=1 \rightarrow n}^{j=1 \rightarrow m} \text{Cos}(V_{f_i}, V_{p_j})}{mn}$$

初步计算得到：

各个变量是缺陷根因的可能性大小



思路四：面向变量的细粒度软件缺陷定位技术

IsoVar突变分析阶段：进一步隔离出真正导致缺陷的变量

Assumption：对错误相关变量进行突变会对失败执行产生较大影响，而对通过执行的影响较小。

| | f_1 | f_2 | p_1 | p_2 |
|-------|-------|-------|-------|-------|
| b_1 | 1 | 2 | 1 | 2 |
| ... | ... | ... | ... | ... |
| b_n | 0 | 0 | 2 | 0 |

突变操作

| | f_1 | f_2 | p_1 | p_2 |
|-------|-------|-------|-------|-------|
| b_1 | 2 | 1 | 1 | 2 |
| ... | ... | ... | ... | ... |
| b_n | 2 | 0 | 1 | 0 |

$\bar{I}_f(m)$ ：突变 m 对失败测试用例造成的平均影响

$\bar{I}_p(m)$ ：突变 m 对通过测试用例造成的平均影响

$fitness(v)$ ：突变 m 对执行路径造成的综合影响

$$\bar{I}_f(m) = \frac{\sum_{i=1 \rightarrow n} 1 - \text{Cos}(V_{f_i}, V_{f_{i'}})}{n}$$

$$\bar{I}_p(m) = \frac{\sum_{i=1 \rightarrow m} 1 - \text{Cos}(V_{p_i}, V_{p_{i'}})}{m}$$

$$fitness(v) = \bar{I}_f(m) - \beta \bar{I}_p(m), m \in \mathcal{M}(v)$$



思路四：面向变量的细粒度软件缺陷定位技术

IsoVar结果汇总阶段

数据分析阶段结果：

$$Sus(v) = \frac{Freq_f}{Freq_f + Freq_p} - Sim_{\langle f, p \rangle}$$

+

突变分析阶段结果：

$$\overline{fitness(v)} = 1 - \frac{\sum_{m \in \mathcal{M}(v)} fitness(v)}{|\mathcal{M}(v)|}$$

||

综合结果：

$$isolation(v) = Sus(v) + \gamma \overline{fitness(v)}$$



思路四：面向变量的细粒度软件缺陷定位技术

IsoVar效果

| Projects | MAP | | | | | | MRR | | | | | | Top-1 | | Top-5 | | Top-10 | |
|-----------|---------------------|--------------------|--------------------|---------------------|----------------------|-------|---------------------|--------------------|--------------------|---------------------|----------------------|-------|--------------------|----------------------|--------------------|----------------------|--------------------|----------------------|
| | VFL _{ochi} | VFL _{tar} | VFL _{bar} | IsoVar _s | IsoVar _{sm} | Imp% | VFL _{ochi} | VFL _{tar} | VFL _{bar} | IsoVar _s | IsoVar _{sm} | Imp% | VFL _{och} | IsoVar _{sm} | VFL _{och} | IsoVar _{sm} | VFL _{och} | IsoVar _{sm} |
| Time | 0.238 | 0.268 | 0.265 | 0.251 | 0.267 | 12.2% | 0.252 | 0.252 | 0.251 | 0.321 | 0.330 | 31.0% | 3 | 4 | 8 | 13 | 15 | 15 |
| Chart | 0.319 | 0.295 | 0.337 | 0.366 | 0.377 | 18.2% | 0.317 | 0.254 | 0.295 | 0.380 | 0.416 | 31.2% | 4 | 6 | 13 | 14 | 14 | 15 |
| Lang | 0.422 | 0.387 | 0.410 | 0.447 | 0.451 | 6.9% | 0.415 | 0.359 | 0.375 | 0.485 | 0.486 | 17.1% | 15 | 18 | 33 | 38 | 38 | 43 |
| Math | 0.265 | 0.254 | 0.272 | 0.270 | 0.274 | 3.4% | 0.258 | 0.234 | 0.251 | 0.291 | 0.295 | 14.3% | 12 | 17 | 39 | 40 | 50 | 48 |
| Closure | 0.154 | 0.121 | 0.152 | 0.162 | 0.160 | 3.9% | 0.180 | 0.146 | 0.171 | 0.210 | 0.201 | 11.7% | 12 | 14 | 25 | 34 | 39 | 42 |
| Mockito | 0.269 | 0.246 | 0.250 | 0.250 | 0.273 | 1.5% | 0.271 | 0.239 | 0.242 | 0.275 | 0.276 | 1.8% | 7 | 8 | 10 | 10 | 14 | 14 |
| FasterXML | 0.140 | 0.105 | 0.204 | 0.241 | 0.263 | 87.9% | 0.181 | 0.162 | 0.252 | 0.321 | 0.344 | 90.1% | 3 | 5 | 8 | 11 | 12 | 15 |
| Spoon | 0.151 | 0.103 | 0.148 | 0.184 | 0.187 | 23.8% | 0.238 | 0.159 | 0.212 | 0.242 | 0.249 | 4.6% | 7 | 10 | 19 | 18 | 22 | 23 |
| Traccar | 0.273 | 0.200 | 0.296 | 0.382 | 0.384 | 40.7% | 0.316 | 0.271 | 0.298 | 0.385 | 0.427 | 35.1% | 5 | 11 | 16 | 20 | 25 | 28 |
| Summary | 0.239 | 0.211 | 0.244 | 0.264 | 0.270 | 13.0% | 0.259 | 0.220 | 0.247 | 0.302 | 0.309 | 19.3% | 68 | 93 | 171 | 198 | 229 | 243 |

VFL_{ochi}, VFL_{tar} and VFL_{bar} denote the variants based on Ochiai, Tarantula and Barinel. IsoVar_s: results of statistical analyses; IsoVar_{sm}: results combining two analysis. The MAP and MRR in the Summary row indicates the weighted average, with the weight being the number of bugs per project.

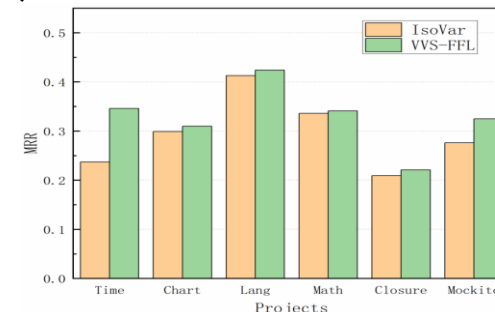
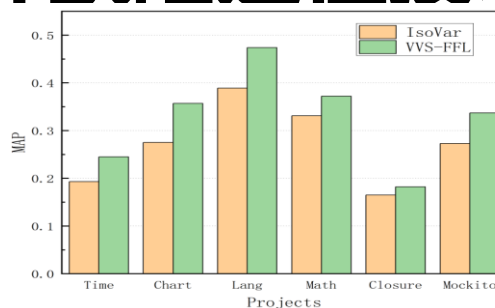
MAP在 0.16-0.45 之间, MRR在 0.25-0.49 之间

思路四：面向变量的细粒度软件缺陷定位技术

变量级缺陷定位结果可视化工具VVS-FFL:

使用变量值序列作为runtime profile进行定位, 在MAP和MRR上平均优于IsoVar 20.8%和10.7%

将定位结果进行可视化展示, 在IDE中进行代码高亮, 直接提示开发者需要检查的位置 (可配置)



```
Partial(Partial partial, int[] values) {
    this.iChronology = partial.iChronology;
    this.iTypes = partial.iTypes;
    this.iValues = values;
}

Partial(Chronology chronology, DateTimeFieldType[] types, int[] values) {
    this.iChronology = chronology;
    this.iTypes = types;
    this.iValues = values;
}

public int size() { return this.iTypes.length; }

public Chronology getChronology() { return this.iChronology; }

protected DateTimeField getField(int index, Chronology chrono) { return this.iTypes[index].getField(chrono); }

public DateTimeFieldType getField(int index) { return this.iTypes[index]; }

public DateTimeFieldType[] getFieldTypes() {
    return (DateTimeFieldType[])this.iTypes.clone();
}
```

```
public void validate(ReadablePartial partial, int[] values) {
    int size = partial.size();

    int i;
    int value;
    DateTimeField field;
    for(i = 0; i < size; ++i) {
        value = values[i];
        field = partial.getField(i);
        if (value < field.getMinimumValue()) {
            throw new IllegalArgumentException(field.getType(), value, field.getMinimumValue(), (Number)null);
        }
        if (value > field.getMaximumValue()) {
            throw new IllegalArgumentException(field.getType(), value, (Number)null, field.getMaximumValue());
        }
    }

    for(i = 0; i < size; ++i) {
        value = values[i];
        field = partial.getField(i);
        if (value < field.getMinimumValue(partial, values)) {
            throw new IllegalArgumentException(field.getType(), value, field.getMinimumValue(partial, values), (Number)null);
        }
    }
}
```

```
Time_1_buggy - Partial.java
Time_1_buggy | src | main | java | org | joda | time | Partial | with | Add Configuration... | Git: | Event Log
Project | README.md | Partial.java | DurationFieldType.java | DurationField.java | Maven
211 | }
212 | DurationField lastUnitField = null;
213 | for (int i = 0; i < types.length; i++) {
214 |     DateTimeFieldType loopType = types[i];
215 |     DurationField loopUnitField = loopType.getDurationType().getField(iChronology);
216 |     if (i > 0) {
217 |         int compare = lastUnitField.compareTo(loopUnitField);
218 |         if (compare < 0) {
219 |             throw new IllegalArgumentException("Types array must be in order largest-smallest: " +
220 |                 types[i - 1].getName() + " < " + loopType.getName());
221 |         } else if (compare == 0) {
222 |             if (types[i - 1].getRangeDurationType() == null) {
223 |                 if (loopType.getRangeDurationType() == null) {
224 |                     throw new IllegalArgumentException("Types array must not contain duplicate: " +
225 |                         types[i - 1].getName() + " and " + loopType.getName());
226 |                 }
227 |             } else {
228 |                 if (loopType.getRangeDurationType() == null) {
229 |                     throw new IllegalArgumentException("Types array must be in order largest-smallest: " +
230 |                         types[i - 1].getName() + " and " + loopType.getName());
231 |                 }
232 |             }
233 |         }
234 |     }
235 | }
```

PART 04

其他辅助技术

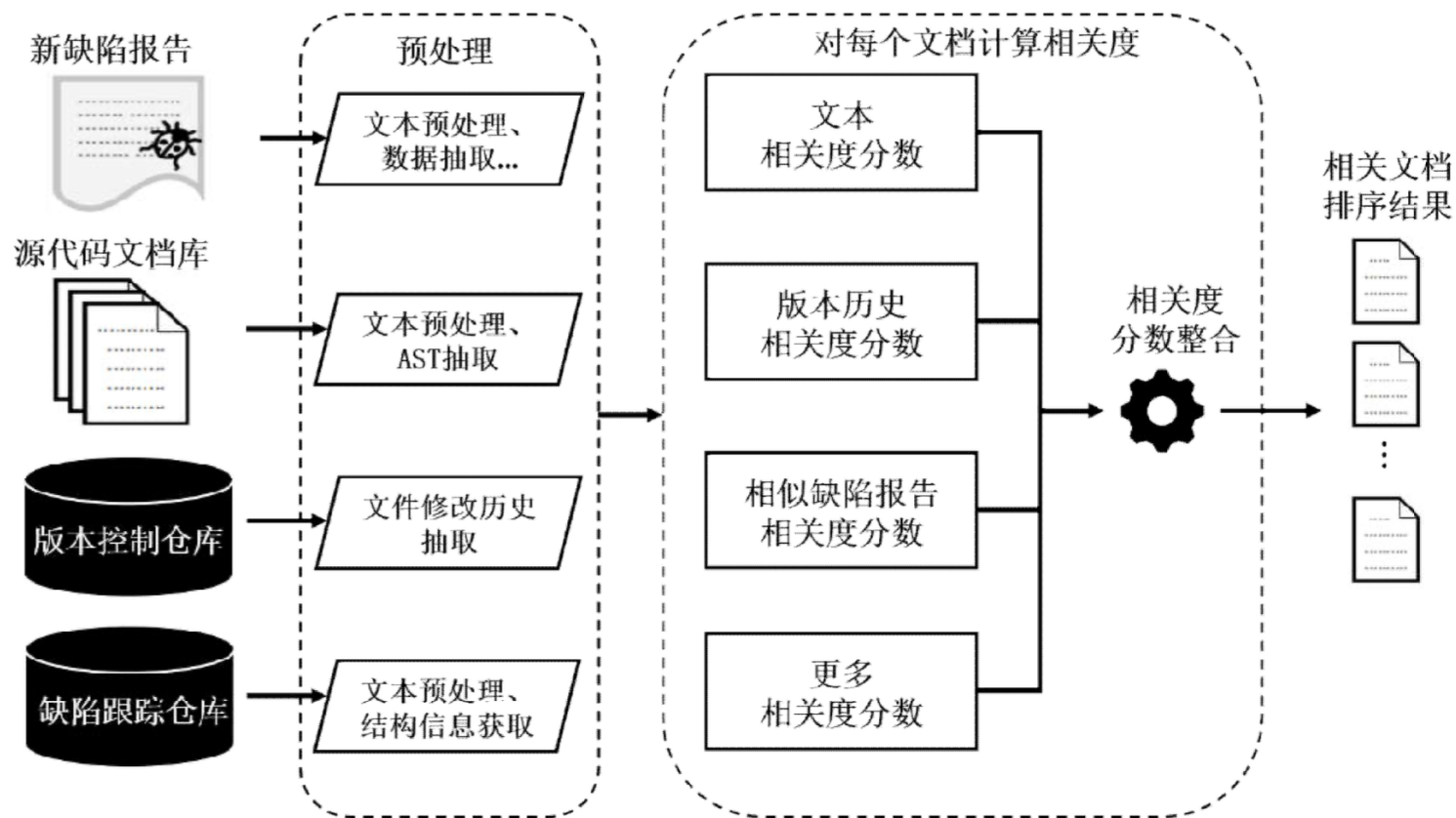
Other Related Techniques



其他辅助技术 (1): 基于信息检索的缺陷定位

Information Retrieval-based Fault Localization (IRFL)

IRFL
一般研究框架

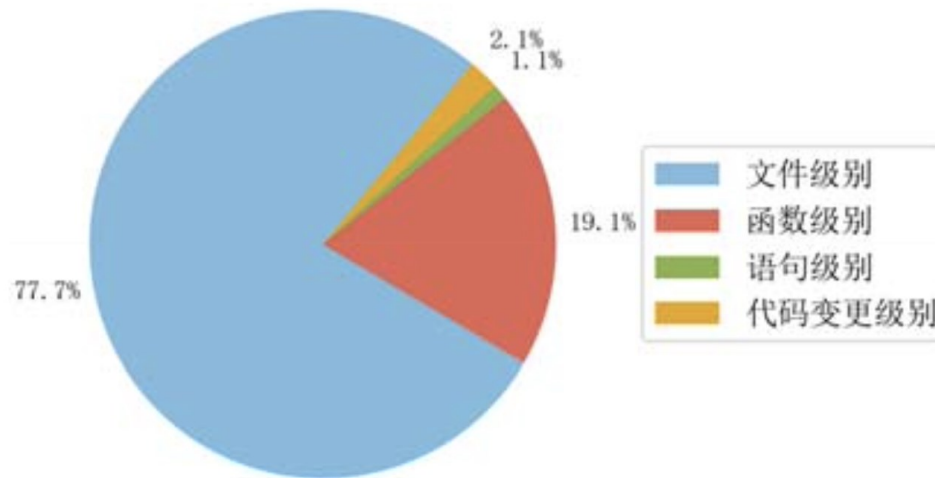




其他辅助技术 (1): 基于信息检索的缺陷定位

Information Retrieval-based Fault Localization (IRFL)

IRFL技术的
不同定位粒度



和SOTA相比, 我们的新方法:

在Accuracy@1、MAP、MRR 上分别提升 21.0%、21.3%、17.10%

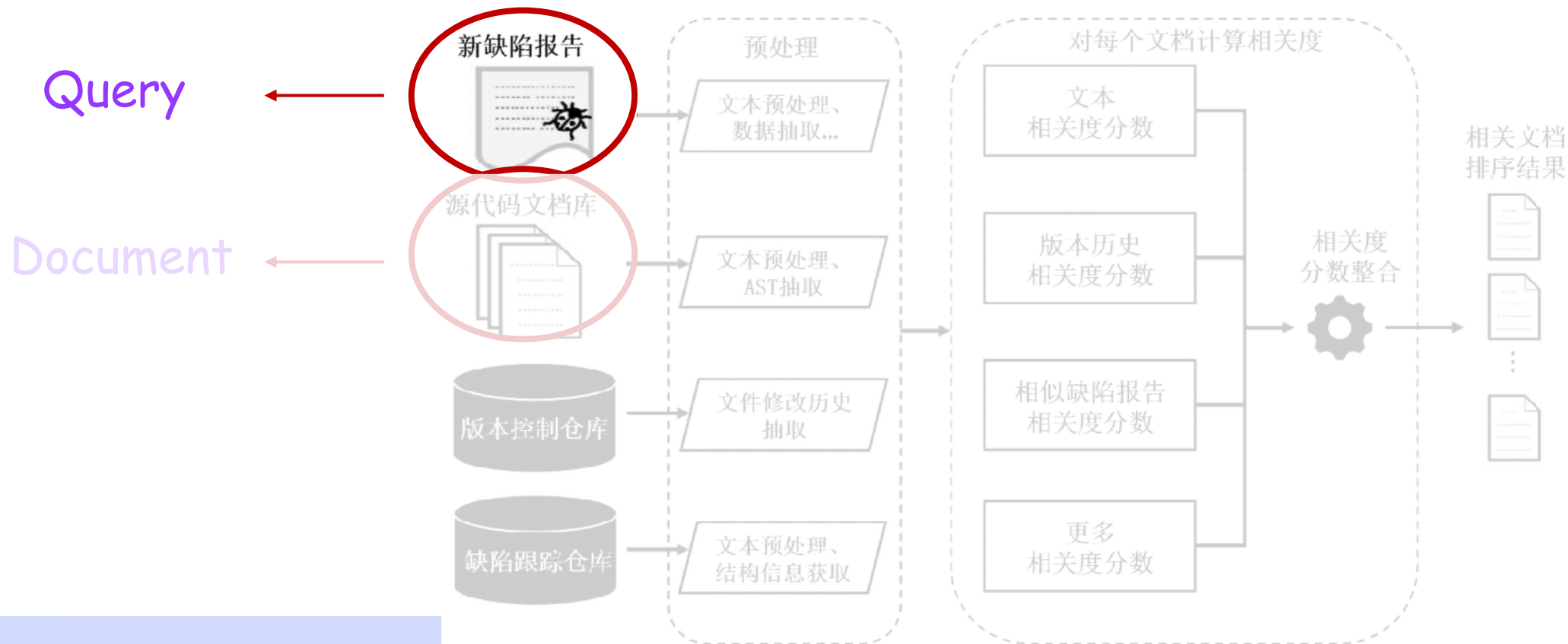
最高分别为 0.851、0.862、0.898

李政亮, 陈翔, 蒋智威, 顾庆. 基于信息检索的软件缺陷定位方法综述[J]. 软件学报, 2021, 32(2): 247-276.



其他辅助技术 (1): 基于信息检索的缺陷定位

准确理解缺陷报告十分重要





其他辅助技术 (2): 系统级缺陷报告理解 - 标题

Integrated terminal links not recognized when starting with workspace folder
colelawrence opened this issue on May 12 · 4 comments

列表视图下的标题

VSCode Version: 1.46.0-insider
Commit: 2b472d1
Date: 2020-05-11T09:54:42.339Z (7 hrs ago)

OS Version: Darwin x64 19.4.0

Steps to Reproduce:

1. Run a command that prints a path like `workspaceFolderA/readme.md`
2. Notice that a direct link is not found (bug)
3. Try opening "Find in Project" (Cmd + P) and paste path `workspaceFolderA/readme.md`
4. See no results (screenshot below)

No one assigned

Labels

***as-design**

Projects

None yet

Milestone

No milestone

Linked pull requests

Successfully

Usability and aesthetic problems with the table view UI

列表视图下的标题

Details

| | | | |
|--------------------|----------------|----------------|-------------|
| Type: | Bug | Status: | OPEN |
| Priority: | Blocker | Resolution: | Unresolved |
| Affects Version/s: | 1.4.0, 2.0.0 | Fix Version/s: | None |
| Component/s: | None | | |
| Labels: | None | | |

Description

HBASE-15675 changed the display and formatting of the table view in the master UI. After this change by default regions are sorted lexicographically by encoded region name, which has no correspondence with ordering in the key space. Previously entries adjacent in the region list would be adjacent in key space too. This made it easy to locate boundaries for split and merge requests. This is a big step backwards for usability. For anyone who wants to manage splits and merges (after **HBASE-47609**), unless they have memorized what encoded names correspond to the adjacent regions of interest, they first have to check the box named "ShowDetailName&Start/End Key" and resort the lists.

- Typescript module resolution fails in files matched by config.exclude**
javascript
#97531 by beck was closed yesterday
- Extension API: command for editor/context is visible in outputChannel context**
*caused-by-extension
#97530 by Surendrajat was closed yesterday
- improving diff ui / diff optical illusions**
#97529 opened yesterday by aleksandarbos
- Integrated terminal links not recognized when starting with workspace folder**
*as-designed
#97528 by colelawrence was closed yesterday
- Up, down, close buttons in Peek Problems are not visible when file name is long**
bug help wanted languages-diagnostics
#97526 opened yesterday by shreedhart
- Error files output on "ng serve" does not allow control-click to navigate to file.**
needs more info
#97524 opened yesterday by waigon
- HBASE-14498**
Master stuck in infinite loop when all Zookeeper servers are unreachable (...)
- HBASE-13740**
Stop using Hadoop private interfaces
- HBASE-17621**
Usability and aesthetic problems with the table view UI
- HBASE-20572**
HBase2 does not compile against Hadoop3 after HADOOP-10768
- HBASE-7386**
Investigate providing some supervisor support for znode deletion
- HBASE-20255**
provide documentation on licensing for dependencies
- HBASE-17204**
Make Off heap Bucket Cache default ON



阅读标题

帮助用户在不阅读主体的前提下快速理解缺陷报告: bug分配、初步缺陷定位、根因分析等



其他辅助技术 (2): 系统级缺陷报告理解 – 标题

缺陷报告标题
撰写的常见原则

- Your title should serve as a **concise summary** of what the bug is.
- How would you describe the bug **using approximately 10 words**? This is the first part of your bug report a triager or developer will see. A good summary **should quickly and uniquely identify** a bug report.

— Testlio

— Mozilla

🔔 The TypeScript language service died 5 times right after it got started. ***caused-by-extension** **new release**
#105563 by Azarchaniel was closed yesterday

🔔 **Feature request** ***question**
#105561 by Nayerah-Aboubreak was closed 2 days ago

🔔 **Bug report** ***question**
#105560 by Nayerah-Aboubreak was closed 2 days ago

🔔 TS Server fatal error: path.replace is not a function **invalid**
#105554 by Terahpatrick was closed 2 days ago

🔔 **Cuda 3.0 not working**
#1440 by alex-pardo was closed on Jun 7, 2016

🔔 Tensorflow missing symbol in compilation
#1437 by daveh86 was closed on Mar 9, 2016

Too short

🔔 Uncomment HTML: Typo in the first sentence of the given description
#17900 by manankalra was closed on Jul 25, 2018

🔔 Feature Proposal: Users can enter a "class code" to join a classroom **status: blocked**
#17899 by utsab was closed on Apr 11

🔔 ES6: Use getters and setters to Control Access to an Object - can pass without proper validation
#17898 by TheNewStyles was closed on Oct 12, 2018

🔔 Basic CSS: Use CSS Variables to change several elements at once - issue
#17897 by Puritanic was closed on Jul 27, 2018

🔔 **Feature Request: Combined/Complete Guide of A Certificate (without the challenge part) for people with slow or restricted internet connection** **type: feature request**
#17895 by zrfrank was closed on May 7, 2019

Too many words

🔔 Unexpected value 'undefined' declared by the module 'DynamicModule' **addon: notes** **app: angular** **question / support**
#7065 by maxigimenez was closed on Jun 13, 2019 5.1.x

🔔 Dependabot can't resolve your JavaScript dependency files
#7062 by dependabot-preview **bot** was closed on Jun 13, 2019

🔔 **To support nuxt-optimized-images module - https://www.bazzite.com/docs/nuxt-optimized-images/** **app: vue** **compatibility with other tools** **inactive** **question / support**
#7060 by p-moreira was closed on Sep 6, 2019

🔔 addDecorator doesn't work on config.js **babel / webpack** **cra** **has workaround** **high priority** **question / support**
#7058 opened on Jun 12, 2019 by chingchinglcc 5.3.x

🔔 Storybook server will rebuild when a storyshots-puppeteer test case fail **addon: storyshots** **inactive** **question / support**
#7056 by chris-fran was closed on Aug 3, 2019

With URLs



其他辅助技术 (2): 系统级缺陷报告理解 – 标签

Remove "Force SSL for all site pages" setting or update its hint #2834

Closed

AndreiMaz opened this issue on 18 Jan 2018 · 0 comments



AndreiMaz commented on 18 Jan 2018 • edited ▾

Member

In 4.00 we enabled "Force SSL for all site pages" by default. It is no longer possible to set "secure" cookies over insecure (e.g. HTTP) origins on Firefox and Chrome after they implemented the Strict Secure Cookies specification

But actually it still works fine in 3.90. So it means that something was implemented wrong way in 4.00
I think should investigate it one more time.

Assignees



RomanovM

Labels

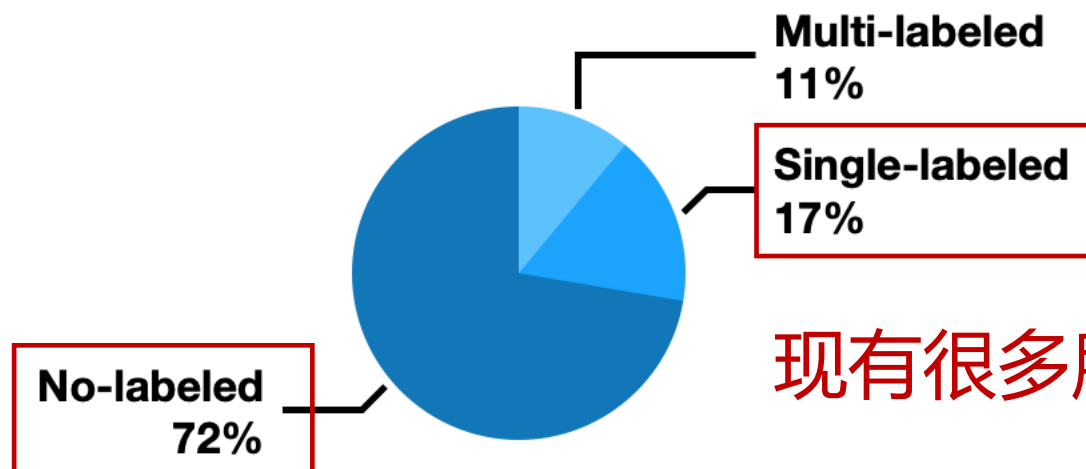
functionality / feature

方便对bug进行分类、对属性进行标记等；
可以用于检索等操作



其他辅助技术 (2): 系统级缺陷报告理解 – 标签

现有技术大多是为单标签生成服务



现有很多所谓的单标签其实是不完整的

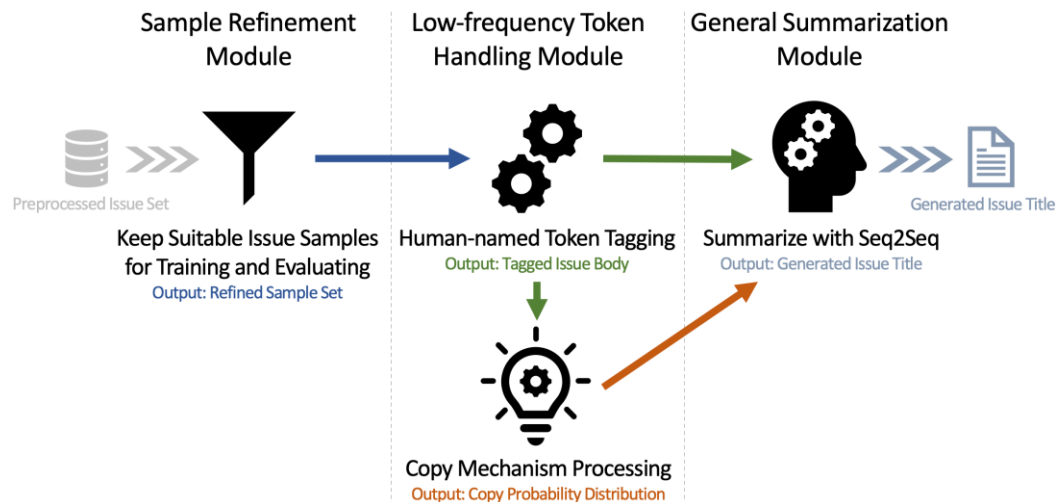
In year 2018, the amounts of **no-labeled** issue reports, **single-labeled** issue reports, and **multi-labeled** issue reports are **10,862,127**, **2,497,953**, and **1,656,937**, respectively.



其他辅助技术 (2): 系统级缺陷报告理解

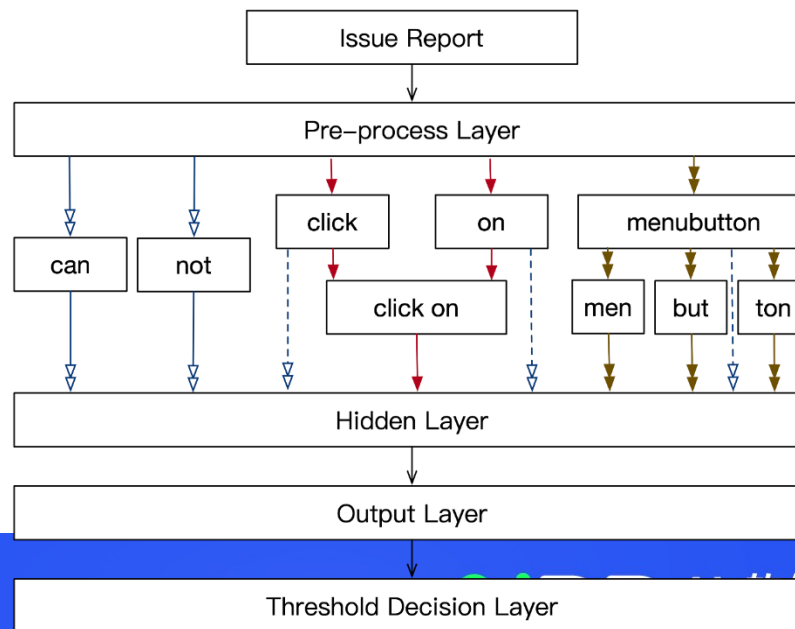
标题生成: iTAPE & iTAPE+

- Input: 缺陷报告主体
- Output: 符合当前项目风格的标题



标签生成: MULA & PLPI

- Input: 缺陷报告主体
- Output: 符合当前项目风格的多标签



PART 5

总结与展望 Prospects



总结与展望：自动化代码缺陷定位当前研究趋势

所基于信息

- 程序覆盖信息
- 动/静态切片信息
- 程序频谱信息
- 缺陷报告
- 版本历史信息
- 运行时变量信息
-

所采用技术

- 程序插桩
- 日志/断言/断点
- 切片技术
- 数据挖掘
- 深度学习
-

所面向粒度

- 文件
- 函数
- 代码提交记录
- 基本块
- 语句
- 变量
-



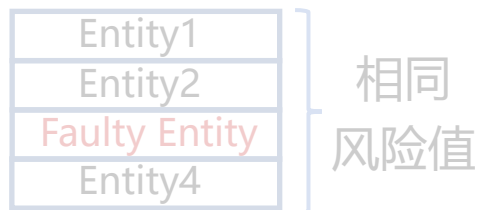
总结与展望：自动化代码缺陷定位当前面临挑战



多缺陷问题



Tie难题



代码缺失缺陷

```
switch(a[0]){
  case "connect":
    if(a[1]){
      if(clients.has(a[1])){
        ws.send("connected");
      }
      MISSING
    }else{
      ws.id = a[1]
      clients.set(a[1], {client: {}});
      ws.send("connected");
    }
  }
}
```

对上述问题的解决

将极大赋能自动化代码缺陷定位技术的落地应用
使其在真实工业环境软件开发中**迎来春天**

Indexing技术效能仍有待提升。

大幅降低缺陷定位效率。

针对既有代码，难以处理此类情况。

谢谢!

