- 阅读论文，准备报告

1. 《动态 Web应用中的缺陷定位研究》—— WDL方法
2. 《一种基于遗传算法的多缺陷定位方法》—— GAMFal方法
3. 《MSFL: A Model for Fault Localization Using Mutation-Spectra Technique》——将mutation和spectrum结合
4. 《A Hybrid Approach to Fine-grained Automated Fault Localization》—— 提出了一种基于动态谱和静态谱的混合细粒度AFL方法。
5. 《Enhancing Spectrum-Based Fault Localization Using Fault Influence Propagation》——集成了基于频谱的故障定位和故障影响传播分析

# A Hybrid Approach to Fine-grained Automated Fault Localization

作者：Leping Li、Hui Liu

汇报人：陈冰婷　　导师：邹卫琴

# 目录

Contents

# Why use HybridAFL?

**Challenging**

**Automated**

**Tedious**

**Simple**

**Time-consuming**

**Intuitive**

# What is HybridAFL?

**Rationale:** *Some types of statements are significantly more/less error-prone than others, and thus statement types could be exploited for fault localization.*

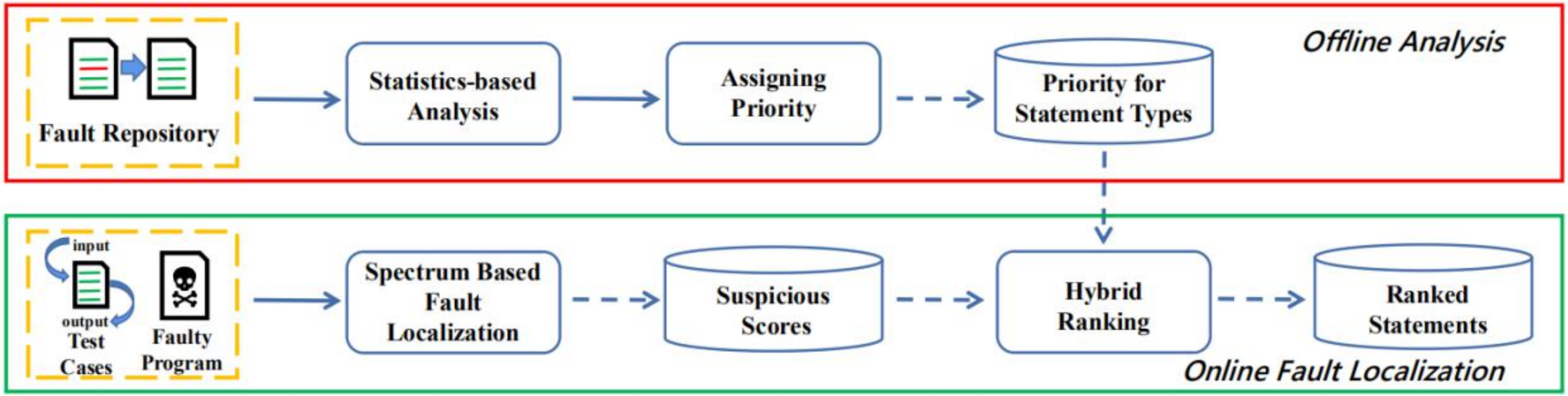
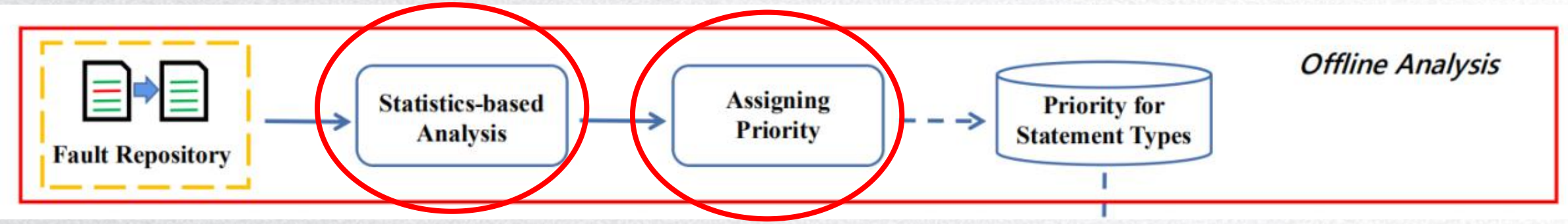


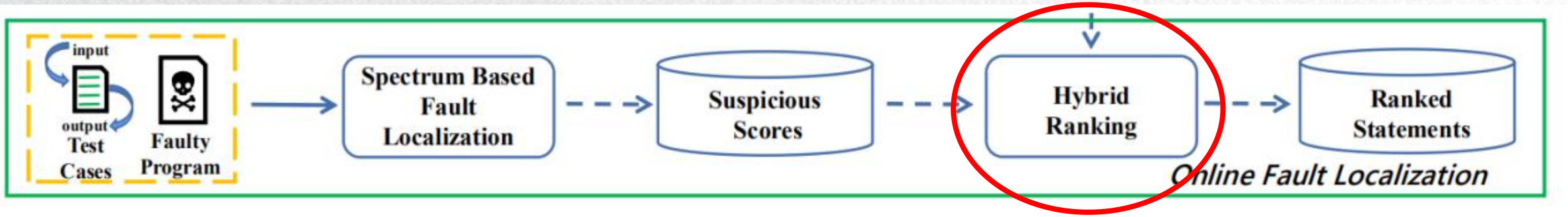

**Figure 1: Overview of HybridAFL**

# What is HybridAFL?




- HybridAFL applies statistics analysis to a fault repository to compute the error-proneness for each of the statement types involved in the repository.
- Based on the resulting error-proneness, HybridAFL identifies special statement types that are steadily more/less error-prone than others.
- HybridAFL assigns priorities to special statement types selected in the preceding step. The more error-prone a statement type is,the greater priority it receives.

# What is HybridAFL?




- For a faulty program to be debugged, HybridAFL collects suspicious statements from the faulty program and computes their suspicious scores by traditional SBFL.
- HybridAFL adjusts the suspicious scores according to statement types, i.e., multiplying the SBFL suspicious scores by the priorities assigned to the statement types.
- All of the suspicious statements are ranked by the updated suspicious scores in descending order. Statements on the top are more likely to be faulty and thus should be inspected first.

# 2.1 Statistics-based Analysis on Fault Repositories

**Key Rational:** *Different types of statements are not equally error-prone.*

$$TSS(pr_i) = \bigcup_{k=1}^{TV(pr_i)} SS(V_{i,k})$$

$$TFS(pr_i) = \bigcup_{k=1}^{TV(pr_i)} FS(V_{i,k})$$

$$AP(pr_i) = \frac{|TFS(pr_i)|}{|TSS(pr_i)|}$$

$$TSS(pr_i) = \bigcup_{k=1}^{N} TSS(pr_i, t_j)$$

$$TFS(pr_i) = \bigcup_{k=1}^{N} TFS(pr_i, t_j)$$

$$AP(pr_i, t_j) = \frac{|TFS(pr_i, t_j)|}{|TSS(pr_i, t_j)|}$$

# 2.2 Assigning Priorities to Special Statements

**Key Rational:** *Some types of suspicious statements are significantly more/less error-prone than others.*

1)
$$RP(pr_i, t_j) = AP(pr_i, t_j)/AP(pr_i)$$

$$RP(t_j) = \{RP(pr_1, t_j), RP(pr_2, t_j), \ldots, RP(pr_n, t_j)\}$$

2)
$$nor\_RP(t_j) = \{lg(RP(pr_1, t_j)), \ldots, lg(RP(pr_n, t_j))\}$$

3)
$$W(t_j) = lg^{-1} avg(nor\_RP(t_j))$$
$$= 10^{avg(nor\_RP(t_j))}$$

# 2.3 Hybrid Ranking

**Key Rational:** *Different typesof statements are not equally error-prone.*

$$sus'(ss_j) = sus(ss_j) \times W(T(ss_j))$$

# Evaluation

| | |
|---|---|
| RQ01 | Are different types of statements equally error-prone? |
| RQ02 | Does the proposed approach outperform the state-of-the-art fine-grained fault localization techniques? |
| RQ03 | How does the performance of the proposed approach varyin locating different types of faulty statements? |
| RQ04 | Can we further improve the proposed approach by assigning priorities to all statements besides the special statementsselected in Section 3.3? |
| RQ05 | Can we further improve the proposed approach by leveraging the median instead of the average in Formula 17? |
| RQ06 | Should we apply multi-level sorting to fault localization byleveraging both SBFL scores and priorities of statement types? |
| RQ07 | Can we improve other SBFL algorithms in the same wayas we improve $Ochiai$? |

# RQ1: Some Types of Statements are Significantly More Error-Prone Than Others

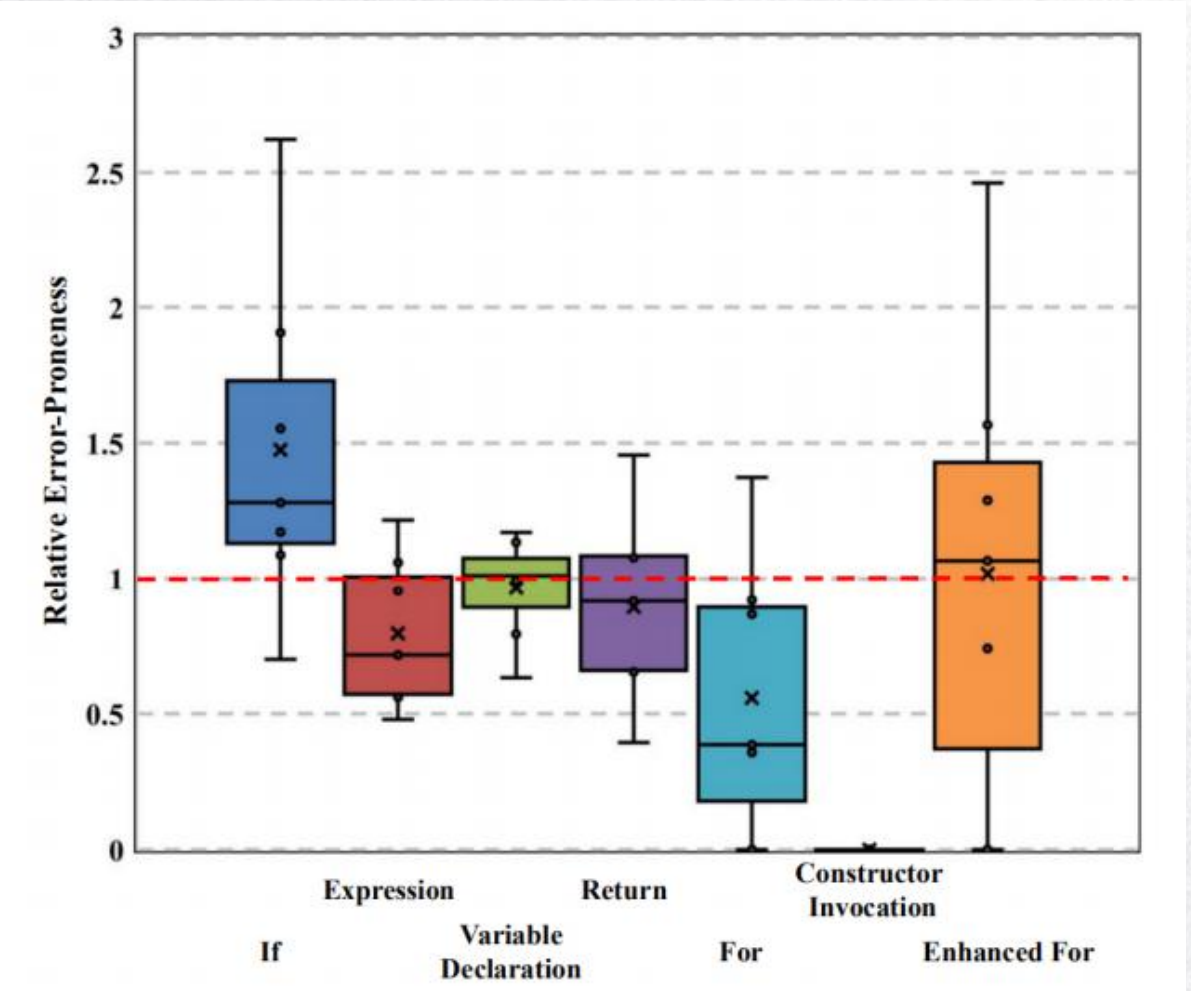



**Figure 2: Relative Error-Proneness Varying among Programs**

# RQ2: HybridAFL Improves the State of the Art

**Table 3: Absolute Waste Effort (AWE)**

| Approaches \ Projects | Math | Closure | Lang | Cli | JSoup | Databind | Compress | **Overall** |
|---|---|---|---|---|---|---|---|---|
| Ochiai ($x$) | 7,227 | 41,306 | 1,220 | 1,260 | 3,468 | 7,327 | 2,425 | 64,223 |
| HybridAFL ($y$) | 5,524 | 37,448 | 907 | 1,103 | 3,786 | 7,445 | 2,066 | 58,279 |
| Relative Reduction ($1 - y/x$) | 23.6% | 9.3% | 25.7% | 12.5% | -9.2% | -1.6% | 14.8% | 9.3% |

# RQ3: Good at Locating High-Priority Statements

**Table 4: Effect on Different Types of Statements**

| | If | Variable Declaration | Return | Expression | For | Constructor Invocation | Others |
|---|---|---|---|---|---|---|---|
| Priorities | 1.36 | 1 | 1 | 0.75 | 0.4 | 0 | 1 |
| #Improved Cases ($x$) | 66 | 15 | 20 | 1 | 0 | 0 | 6 |
| #Decreased Cases($y$) | 0 | 11 | 19 | 50 | 3 | 0 | 0 |
| #Draw Cases($z$) | 3 | 4 | 6 | 22 | 0 | 0 | 5 |
| Chance of Improvement ($x/(x+y+z)$) | 95.7% | 50.0% | 44.4% | 1.4% | 0% | - | 54.5% |
| Chance of Reduction ($y/(x+y+z)$) | 0% | 36.7% | 42.2% | 68.5% | 100% | - | 45.5% |

# RQ4: Selection of Special Statement Types

**Table 5: Selection of Special Statement Types Reduces AWE**

| Setting \ Programs | Math | Closure | Lang | Cli | JSoup | Databind | Compress | **Overall** |
|---|---|---|---|---|---|---|---|---|
| Selection Enabled ($x$) | 5,524 | 37,448 | 907 | 1,103 | 3,786 | 7,445 | 2,066 | 58,279 |
| Selection Disabled ($y$) | 6,420 | 36,933 | 1,140 | 1,108 | 3,917 | 8,240 | 2,003 | 59,643 |
| Relative Reduction in AWE ($1 - y/x$) | -16.2% | 1.4% | -25.7% | -0.5% | -3.5% | -10.7% | 3.0% | -2.3% |

# RQ5: Average VS. Median

**Table 6: Derivation of Priorities Has Minor Effect on AWE**

| Priorities \ Projects | Math | Closure | Lang | Cli | JSoup | Databind | Compress | **Overall** |
|---|---|---|---|---|---|---|---|---|
| Average ($x$) | 5,524 | 37,448 | 907 | 1,103 | 3,786 | 7,445 | 2,066 | 58,279 |
| Median ($y$) | 5,666 | 37,523 | 986 | 1,131 | 3,648 | 7,585 | 2,047 | 58,586 |
| Relative Reduction in AWE ($1 - y/x$) | -2.6% | -0.2% | -8.7% | -2.5% | 3.6% | -1.9% | -0.9% | -0.53% |

# RQ6: Multi-level Sorting Does Not Work

**Table 7: Multi-level Sorting Increases AWE**

| Sorting \ Projects | Math | Closure | Lang | Cli | JSoup | Databind | Compress | **Overall** |
|---|---|---|---|---|---|---|---|---|
| Hybrid | 5,524 | 37,448 | 907 | 1,103 | 3,786 | 7,445 | 2,066 | 58,279 |
| Multi-Level (SBFL First) | 6,716 | 39,762 | 1,177 | 1,196 | 3,385 | 7,357 | 2,238 | 61,931 |
| SBFL Only | 7,227 | 41,306 | 1,220 | 1,260 | 3,468 | 7,327 | 2,425 | 64,223 |
| Multi-Level (Priority First) | 4,701 | 52,101 | 1,068 | 1,518 | 9,965 | 36,144 | 2,419 | 107,916 |
| Priority Only | 6,459 | 77,752 | 1,267 | 1,965 | 13,405 | 53,571 | 2,938 | 157,357 |

# RQ7: Improving Various SBFL Algorithms

**Table 8: Relative Improvement on Various SBFL Algorithms**

| SBFL Algorithms \ Projects | Math | Closure | Lang | Cli | JSoup | Databind | Compress | **Overall** |
|---|---|---|---|---|---|---|---|---|
| Ochiai | 23.6% | 9.3% | 25.7% | 12.5% | -9.2% | -1.6% | 14.8% | 9.3% |
| Jaccard | 14.3% | 9.9% | 5.9% | 8.5% | -9.2% | 4.6% | 15.0% | 8.8 % |
| DStar | 15.3% | 10.1% | 14.0% | 8.1% | -9.2% | 3.4% | 12.2% | 9.0 % |
| Barinel | 14.3% | 9.9% | 9.0% | 8.4% | -9.2% | 7.2% | 15.1% | 9.2% |

THANK YOU FOR YOUR LISTENING.

# 谢谢您的聆听