



基于风险轨迹的软件 缺陷定位及自动化修复

指导教师：王曙燕教授
答 辩 人：韩雪



目录



▶ 研究目的和意义


▶ 国内外研究现状

▶ 主要研究内容和预期成果

▶ 难点分析与创新性

▶ 主要问题和技术关键

▶ 进度安排

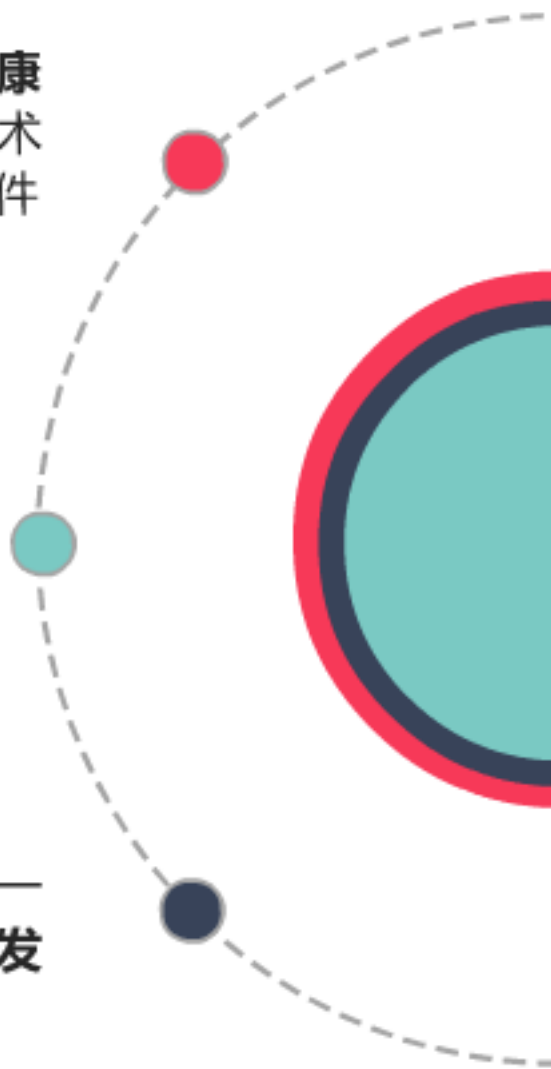


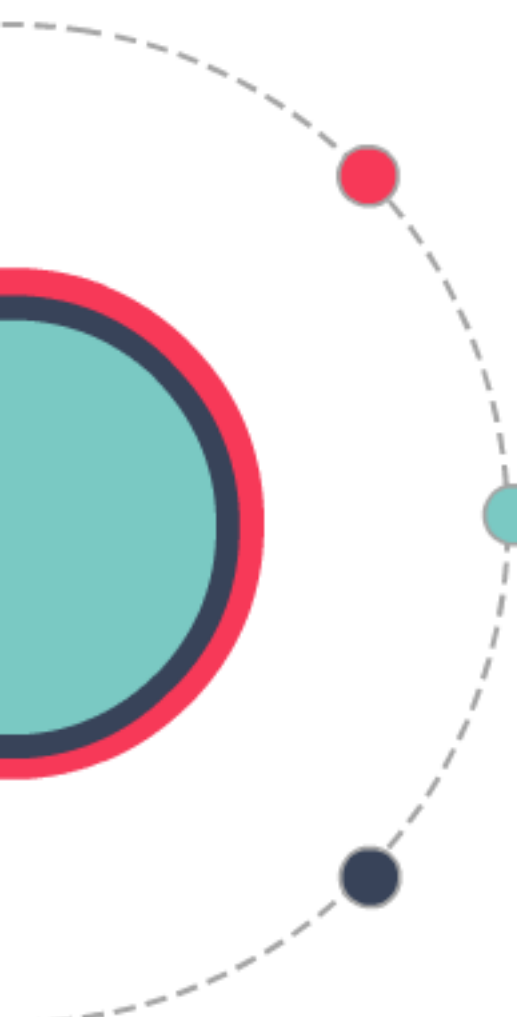
研究目的和意义

软件是**保证国家信息安全**和**软件产业健康发展**的重要因素。无论工业生产还是学术研究领域，定位和修复软件bug都是软件工程的核心问题。

随着**软件规模和复杂程度**的提高，软件缺陷出现的频率迅速上升，软件缺陷可能会给人类造成巨大损失或灾难。

开发人员面对大量缺陷报告无从入手时，自动程序修复可以成功完成其中一些缺陷的自动修复，从而有效**减少开发人员的程序调试时间**。






软件缺陷**定位阶段**是自动程序修复的**基础**。由于缺陷定位在程序调试过程中是繁琐且花费很高的活动，因此需要有效的软件缺陷定位技术来识别缺陷位置。

错误定位旨在找到隐含在程序源代码中的错误指令、过程或数据定义，其粒度包含**程序语句、基本块、分支、函数或类**。

在定位到缺陷语句位置时，能够给程序员提供**修复建议**可以显著降低软件开发成本。



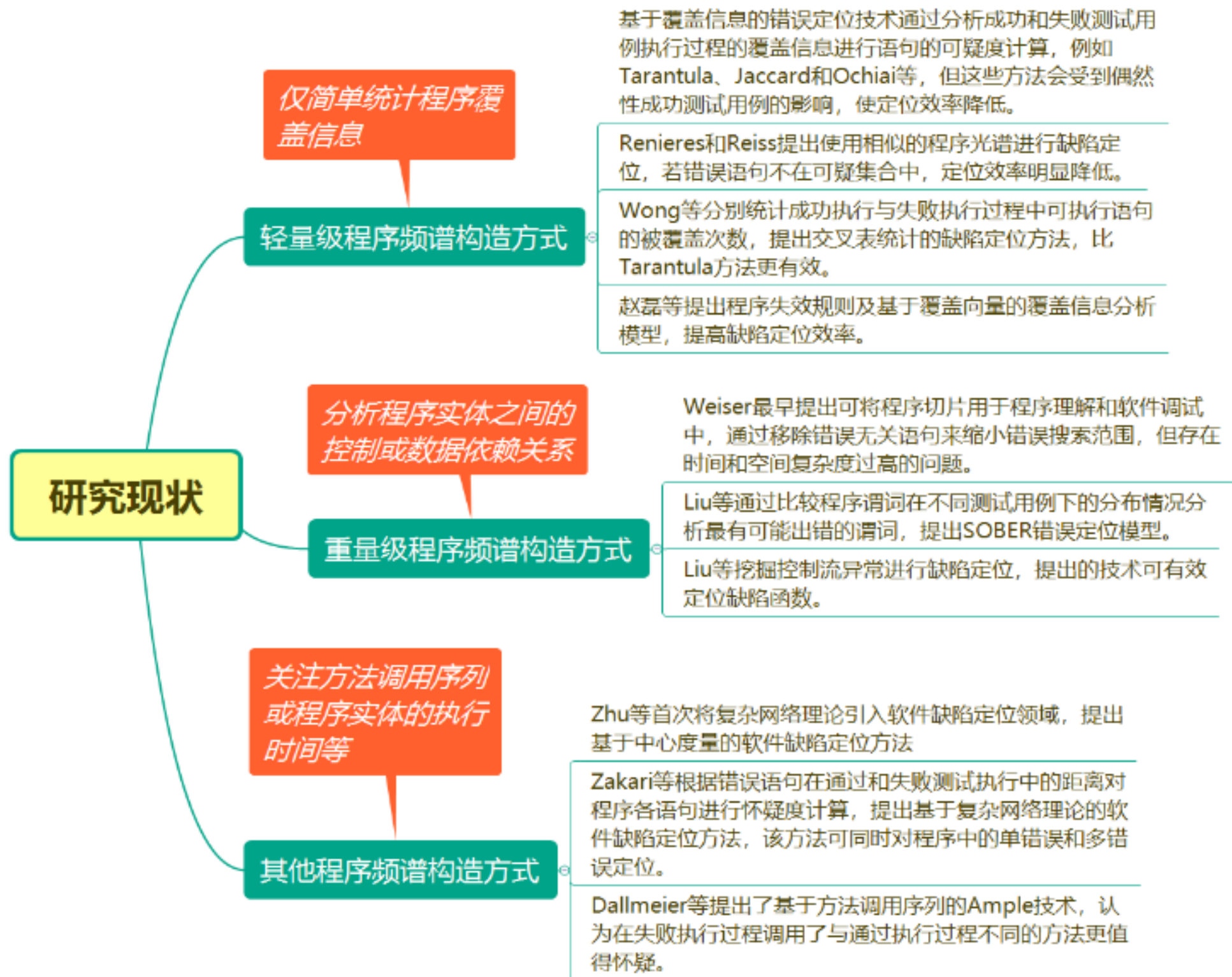
目录

- ▶ 研究目的和意义
 - ▶ 国内外研究现状
 - ▶ 主要研究内容和预期成果
 - ▶ 难点分析与创新性
 - ▶ 主要问题和技术关键
 - ▶ 进度安排
- 

目前软件缺陷定位问题：

- (1) 错误定位效率有待提高；
- (2) 大多关注软件某些特定属性而不是通用属性，使得缺陷定位方法难以通用；
- (3) 仅定位到缺陷语句位置未给出程序上下文信息，程序员据此无法快速修复缺陷；





目前自动程序修复问题：

- (1) 研究者尝试设计算法为程序自动生成补丁,而所生成的补丁与真实人工添加的补丁尚存较大差别;
- (2) 自动程序修复效率不高;



研究现状

基于搜索的方法

使用基于搜索的软件工程中的启发式方法、演化算法等方法寻找补丁

Weimer 和 Le Goues 等人作为该领域的先驱,设计了 GenProg 算法。将 C 程序看作抽象语法树,将代码段看作子树;基于遗传规划算法,GenProg 将被测程序的抽象语法树通过插入、删除或替换生成新的程序,并应用测试用例集验证是否被修复。

基于代码穷举的方法

通过策略穷举可能的代码修改,进而获得大量的潜在代码补丁。

Debroy 和 Wong 提出了基于程序变异的修复算法。程序变异(program mutation)源自变异测试,将程序进行一个小修改,如更改操作符或数值增加 1 等,进而得到修改后的程序。

基于约束求解的方法

将补丁生成转换为约束求解问题,应用求解器获得可行解并转换为最终补丁。

该类算法的先驱,Nguyen 等人提出 SemFix,一种 C 程序的基于约束的语义修复算法。将测试用例转换为约束,并应用 SMT 求解器求解,最终转换为补丁并输出。



目录



研究目的和意义



国内外研究现状



主要研究内容和预期成果



难点分析与创新性



主要问题和技术关键



进度安排



主要研究内容和预期成果

基于风险轨迹和复杂网络的软件缺陷定位

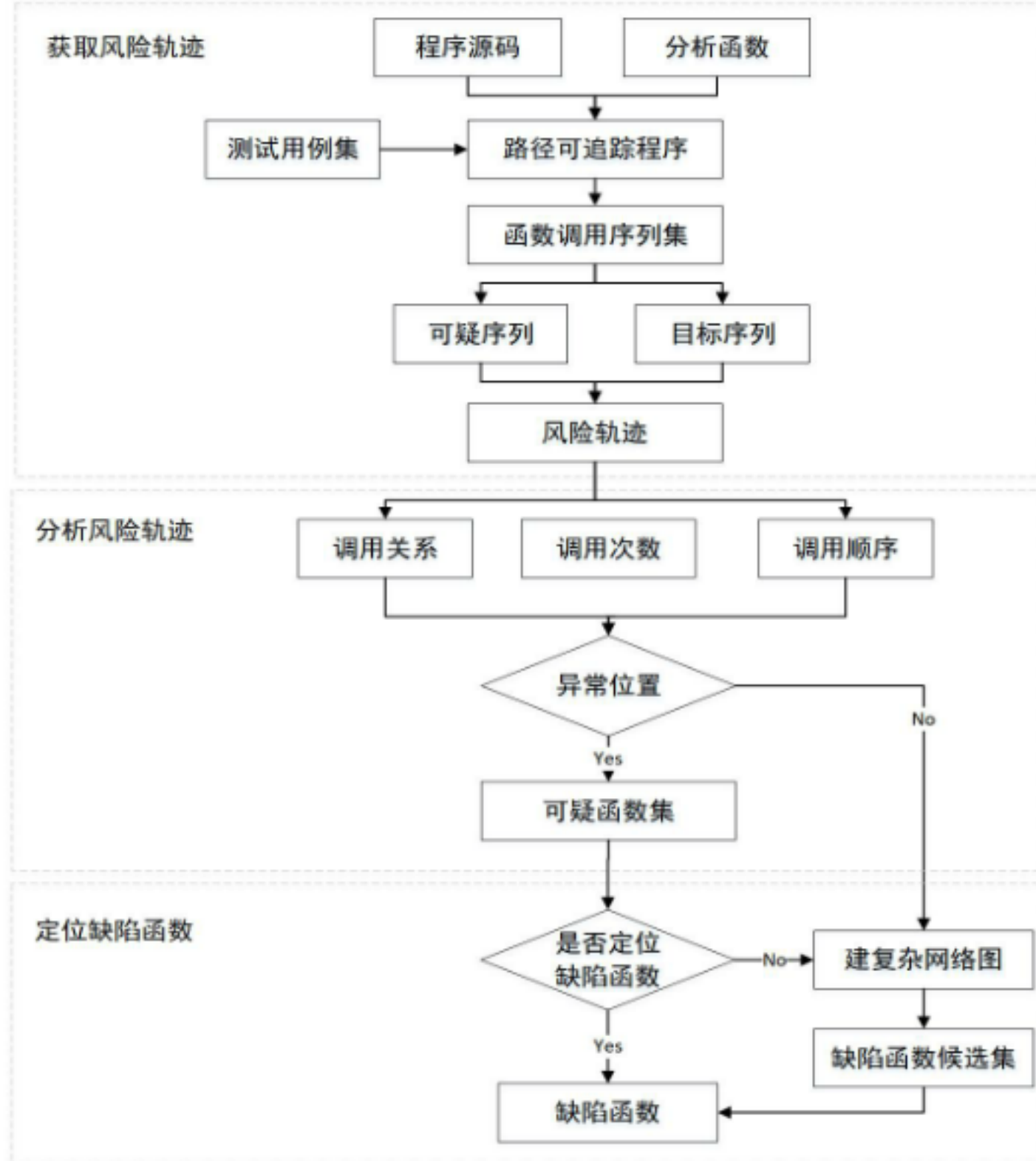
自动化程序修复的建议

基于关联规则和风险轨迹的软件缺陷定位

基于风险轨迹和复杂网络的软件缺陷定位

动态获取程序函数调用序列，根据测试用例在不同缺陷版本程序的执行结果，选取待测程序的**目标序列**和**可疑序列**，对比找出风险轨迹并提取可疑函数集。如果可疑函数集检查完还未发现缺陷函数，为待测程序**建复杂网络图**，根据函数节点出度值进行排序，去除上一步已检测过的函数，生成缺陷函数候选集并检查，最终定位缺陷函数。以函数为粒度进行分析，有助于程序员理解程序上下文并进行缺陷修复。





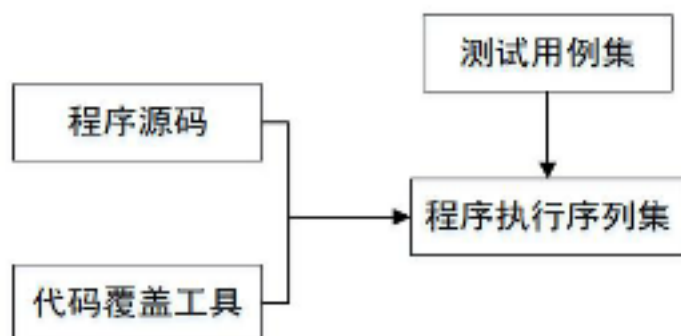
基于关联规则和风险 轨迹的软件缺陷定位

将每个测试用例的**执行轨迹及相应执行结果**抽象为一个事务，所有测试用例的执行轨迹及相应执行结果抽象为事务集，进行**关联分析**，计算语句**置信度和支持度**指标。因在错误定位场景中，至少存在一个失败的测试用例才能测试出程序执行失败，根据此设置支持度和置信度的最小阈值，筛选出符合条件的关联规则。根据测试用例在不同缺陷版本程序的执行结果，选取待测程序的目标序列和可疑序列，提取并**分析风险轨迹**。

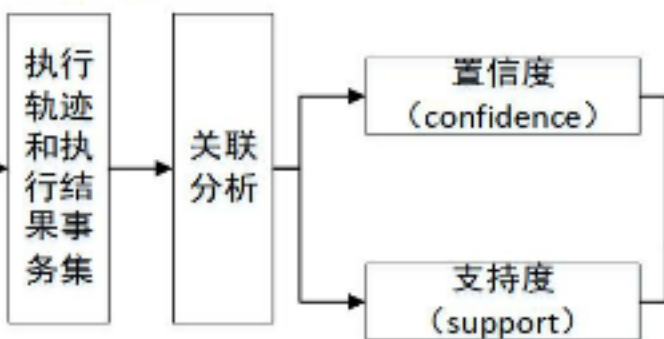
由于关联规则定位出的结果可能有多个语句可疑度排序相同，风险轨迹的分析根据异常出现的执行时间先后可对这些语句排序。



获取程序覆盖轨迹



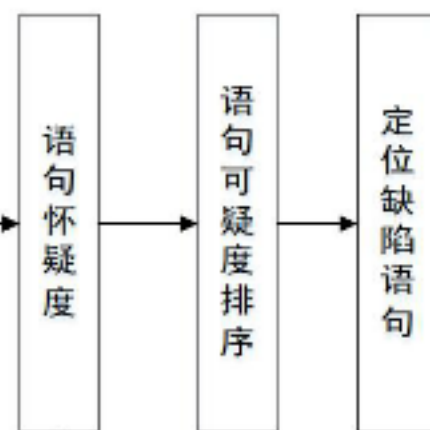
关联分析



风险轨迹分析



缺陷定位



自动化程序修复的建议

在对程序语句计算出可疑度之后，通过一定策略穷举可能的**代码修改**，进而获得大量的潜在代码补丁。利用**程序变异**算法按可疑语句的排序依次进行小修改，如**更改操作符或数值增加1等**，进而得到修改后的程序。应用变异获得可能的程序，将其作为修复后的程序并检验补丁的修复效果。

预期成果

根据目前的工作进度和实际情况，选题拟达到以下目标：

- 1** 利用复杂网络相关度量解决风险轨迹定位不足的缺点，完成对以函数为粒度的软件缺陷定位，并在评测数据集上证明所提方法的有效性和适用性。
- 2** 通过关联分析计算各程序语句置信度和支持度指标，据此对各语句进行可疑度排序，**利用分析风险轨迹的结果调整已排好序的语句**，以期提高定位效率；
- 3** 在上述定位到缺陷语句的基础上，通过一定**策略穷举可能的代码修改**，进而获得大量的潜在代码补丁。对一个出错的C程序进行变异，直到它既能满足需要的功能，又能不再产生之前的错误。



目录



研究目的和意义

国内外研究现状

主要研究内容和预期成果

难点分析与创新性

主要问题和技术关键

进度安排



难点分析

需要选取合适的技术**动态获取程序函数调用序列**，选取合适工具为待测程序**建立复杂网络图**，如何利用复杂网络相关度量解决风险轨迹定位不足是研究重难点；

选取合适工具**动态获取程序覆盖信息**，对程序各语句关联分析，计算支持度和置信度，对各语句进行可疑度排序，如何**利用分析风险轨迹的结果调整已排好序的语句**；

如何**改进程序变异算法**，以期提高程序修复效率。

创新点

1

程序频谱在构造时另辟蹊径，**关注函数调用序列**，将复杂网络引入软件缺陷定位中，并提高软件缺陷定位效率，将软件系统抽象成复杂网络，使提出的方法**不关注程序内部细节，更具通用性**；

2

在分析风险轨迹时，**提出有效的排序策略**对异常语句排序。将分析风险轨迹的结果用来调整基于关联分析的可疑语句排序；

3

改进程序变异的修复算法，使其生成补丁更有效性。



目录



研究目的和意义



国内外研究现状



主要研究内容和预期成果



难点分析与创新性



技术关键



进度安排

关键技术

- (1) 程序轨迹和覆盖信息获取技术：Pvtrace、GCOV、GCC
- (2) 数据挖掘算法：关联分析
- (3) 不同类型异常信息的排序策略
- (4) 建复杂网络图并分析其度量值
- (5) 程序变异算法



目录

- ▶ 研究目的和意义
- ▶ 国内外研究现状
- ▶ 主要研究内容和预期成果
- ▶ 难点分析与创新性
- ▶ 主要问题和技术关键
- ▶ 进度安排

研究计划表

2018年9月至2018年12月	查找国内外相关资料，并对获得的文献、资料进行整理并总结
2019年01月至2019年04月	提出一种基于风险轨迹和复杂网络的软件缺陷定位方法，并完成相关的有效性实验验证。
2019年05月至2019年06月	将风险轨迹分析的结果用于调整基于关联分析的可疑语句排序，完成对程序语句的缺陷定位。
2019年07月至2019年08月	搭建软件测试平台，对比以函数和语句为分析粒度缺陷定位方法的优劣。
2019年09月至2019年12月	对一个出错的C程序进行变异，直到它既能满足需要的功能，又能不再产生之前的错误。
2020年01月至2020年03月	撰写论文，毕业答辩。

谢 谢 指 导