**参考文献记录**

目录

[《软件故障定位关键技术研究综述》 1](#_Toc519776045)

[《The DStar Method for Effective Software Fault Localization》 1](#_Toc519776046)

[《基于变异的错误定位优化技术研究》 2](#_Toc519776047)

[《针对基于变异错误定位的一种动态变异执行策略》 2](#_Toc519776048)

《软件故障定位关键技术研究综述》

——黄小红、赵逢禹，上海理工大学。软件导刊（2017）。

主要内容：介绍了传统的故障定位技术以及2010年以来故障定位的相关成果，总结了常用的测试程序集。

故障定位常见技术有：

1. 基于覆盖的故障定位技术：采用可疑度描述来降低定位复杂度。
2. 基于程序切片的故障定位技术：将程序抽象化，通过删除不相关的部分将程序用切片表示，切片保留原始程序的行为规范。
3. 基于程序频谱的故障定位技术：根据测试用例集合运行完成后的记录信息，使用风险评估方法对程序中的每一行语句计算故障可疑度。

最新推广的故障定位技术：

1. 基于数据挖掘的故障定位技术：从大量程序运行信息数据中提取相关信息，产生一个可使用的数据模型，从而达到定位效果。
2. 基于模型检测的故障定位技术：通过推导期望的程序行为模型，检测失效执行对期望行为的违背情况，从而捕捉故障行为。
3. 使用覆盖率数据集成技术，通过创建可疑语句排名列表的技术进行定位。
4. 以信息检索技术为切入点的故障定位技术：使用基于源代码的搜索得到故障相关报告的降序等级排列。
5. 算法调试的故障定位技术：将复杂度计算分解为一系列子计算。
6. 公式故障定位技术：依赖于对编码失败的程序公式执行轨迹跟踪。

常用程序集：来源于SIR和SourceForge，最常用Siemens、grep、flex。

《The DStar Method for Effective Software Fault Localization》

——W. Eric Wong, Vidroha Debroy, Ruizhi Gao, and Yihao Li。IEEE Transactions on Reliability, 2014, 63(1):290-308

主要内容：

介绍了一种技术叫DStar（D\*），可以给自动NCF化的故障定位提供可疑的位置，每条语句的可疑度计算公式为：

(NCF)\*

(NUF+ NCS)

\*代表指数，并且D\*这项技术在单错误和多错误场景中的错误定位精度高于其他类型

比其他故障定位技术更有效，当指数增长，有效性增强，直到达到平衡为止。

《基于变异的错误定位优化技术研究》

1. 在Simens程序套件上的实验结果表明，Tarantula技术的错误定位效果优于并集、交集和最近邻等方法，Jaccard, Ochiai, Op2错误定位效果与Tarantula相当或者优于它。
2. 怀疑度计算公式是基于程序谱的错误定位方法的研究内容之一，有30多个不同的怀疑度计算公式。
3. 大量实验结果显示没有一个公式能在所有情况下都比其他所有公式错误定位精度好，最新Wong等人提出来的DStar公式单错误和多错误场景中的错误定位精度高于其他类型相似的CBFL公式。
4. 一种基于变异的错误定位技术MUSE，实验证明错误比现有的CBFL：Jaccard, Ochiai, Op2等更好，Metallaxis-fl定位精度比MUSE更好。
5. MBFL是现有错误定位技术中定位精度最好的，但是时间和资源开销大。

《针对基于变异错误定位的一种动态变异执行策略》

龚沛、耿楚瑶、郭俊霞、赵瑞莲，北京化工大学。计算机科学（2016）

提出一种动态变异执行策略：

1. 首先被测程序执行测试用例集，搜集被测程序语句覆盖信息及测试用例执行结果。
2. 筛选出被失败测试用例所覆盖的语句，对这些语句植入人工故障，生成变异体。
3. 对所有变异体执行失败的测试用例，得到akf和anf值，然后选择一个基于频谱的语句怀疑度计算公式计算所有变异体的怀疑度上界值，并按从大到小顺序排列。根据排列顺序逐个对变异体执行通过的测试用例，【执行通过的测试用例的执行顺序也通过一定规则进行排列】，计算语句上变异体怀疑度的最大值，就是语句的怀疑度。
4. 被测程序的所有语句按照怀疑度值从大到小排序，得到怀疑度表。

Metallaxis-FL: mutation-based fault localization

Mike Papadakis, Yves Le Traon. Software Testing, Verification & Reliability(2015)

本文是最早提出基于变异的故障定位技术，将变异体和故障程序的位置联系起来，利用人工注入故障与实际故障之间的相似性来定位软件中的错误实体。被失败执行的测试用例杀死越多的变异体，越能预测一个错误的位置。实验验证基于变异的方法比基于语句的方法具有更高的定位精度。