**清华大学博士毕业论文评审意见**

该论文以改进基于“生成-检验”框架的代码故障自动修复技术的代码修复正确率和执行效率为目的展开研究，提出了一种测试准则纠错算法、一种代码修改方案搜索空间的压缩算法及两种“生成-检验”框架的扩展方案，设计实现了支持工具SmartDebug，并通过规模适当的实验验证了论文所提算法及扩展方案的有效性及效率。

论文分析了测试准则错误对基于频谱的故障定位（SFL）算法精度的影响，并发现“测试准则错误率越高，SFL算法的准确率受损度越高”这一问题。为解决这一问题，论文提出了一种测试准则纠错算法。该算法在“覆盖相似代码行的测试用例通常同时通过或不通过”假设的基础上，利用测试用例对程序代码的覆盖情况识别和修改测试准则中的错误。论文通过实验研究了纠错后测试准则的正确率，得出了纠错算法能够识别大多数测试准则错误的结论。论文进一步通过西门子测试集和grep研究了四种SFL算法在使用测试准则纠错算法前后故障定位精度的差异，得出四种SFL算法的定位精度均有恢复的结论。该算法在时间复杂度和空间复杂度可以接受的情况下，能够有效识别和修改测试准则中的错误，改进了框架的故障定位精度。

论文针对代码修改方案的搜索空间较大的问题，提出了一种称为“预过滤算法”的搜索空间压缩算法。该算法通过利用测试用例集中已通过和未通过测试用例的执行过程来压缩代码修改方案的搜索空间，将无效的修改方案剔除。论文以预过滤算法为基础，实现了一个完整的“生成-检验”系统PFDebug，并将该系统在Defects4J程序测试集上进行实验，得出的结论是：预过滤算法能够大幅度压缩代码修改方案的搜索空间，并且支持工具的修复正确率超过现有的同类工具，但存在可能将合理修改方案过滤掉的问题。

论文提出了两种“生成-检验”框架的扩展方案，即“交互式调试”框架扩展方案和针对特定类别故障修复算法的框架扩展方案xDebug。“交互式调试”扩展方案能够向开发人员提供交互式接口，方便其描述对程序运行状态的判断，提高故障定位的准确度。针对特定类别故障修复算法的扩展方案能够方便测试人员在该框架上开发针对特定故障类型的修复系统。论文在PFDebug的基础上实现了“交互式调试”，形成了新的系统SmartDebug。论文以多个实际程序为调试对象比较了SmartDebug和人工调试在调试效率方面的差异，得出的结论是SmartDebug能够加速调试过程。论文在xDebug的基础上实现了对Java空指针异常的修复系统NPEDebug，并通过一个空指针测试集验证了xDebug的可用性。

论文的结构安排清晰合理。论文的第一章从研究背景出发，阐述了基于“生成-检验”框架的故障定位及修复技术的研究现状，并提出了针对该框架的研究思路。论文的第二章至第四章分别针对框架的三个不同问题，提出了解决问题的核心想法及解决方案，阐述了实验的设计思路及评价方法，验证了解决方案的有效性及效率。第五章阐述了支持工具SmartDebug的设计思路及实现方案，并以一个调试任务展示了工具的使用方式。论文的第六章总结了工作，阐述了研究的不足之处及改进方向。论文的语言逻辑性强、文笔流畅、表达清晰，符合学术规范，反映出作者具有较强的科研能力。

不足之处：

1. 第二章对于测试准则纠错算法的研究中，有时间与空间复杂度的理论分析，但在修复测试准则错误的实验部分没有给出该算法使用的时间代价。
2. 第四章将“交互式调试”与人工调试进行比较的实验中，没有尽最大限度地消除实验参与人员带来的影响。可以设置10人先使用SmartDebug，后使用人工调试；另外10人先使用人工调试，后使用SmartDebug。另外，实验对象仅有一个，建议使用多个实验对象进行实验。

该论文表明作者掌握了故障定位及修复的基本理论和关键技术，达到了博士学位论文水平及毕业要求。