文章编号：2096-1472(2017)-02-01-03

基于搜索的软件自动修复框架及其关键问题探讨

吴 川[1,2]
（1.中国矿业大学计算机学院，江苏 徐州 221116；
2.中国矿业大学徐海学院，江苏 徐州 221008）

摘 要：软件在开发和维护的过程中均可能产生软件缺陷，如果能够成功自动修复部分缺陷，则可以有效减少程序调试时间，避免损失。软件自动修复是一个新兴课题，尚存在很多需要解决的问题。本文首先介绍了软件自动修复的概念，并提出了基于搜索的软件自动修复的框架；接着，从缺陷定位、搜索策略、测试数据生成三个方面概括了基于搜索的软件自动修复面临的主要挑战，以及需要解决的一些关键问题；最后，总结全文并指出下一步的工作。所提框架及其关键问题的探讨，有助于软件自动修复技术的进一步研究和在工业生产中的推广应用。

关键词：基于搜索的软件自动修复；缺陷定位；测试数据生成
中图分类号：TP311 **文献标识码**：A

Search-Based Automatic Software Repair Framework and Discussion on Its Key Issues

WU Chuan[1,2]
(1.*School of Computer Science and Technology*,*China University of Mining and Technology*,*Xuzhou* 221116,*China*;
2.*Xuhai College*,*China University of Mining and Technology*,*Xuzhou* 221008,*China*)

Abstract: Faults occur in the process of software development and maintenance.If the faults are repaired successfully and automatically,the time of program debugging will be effectively reduced and the loss will be avoided.Automatic software repair is an emerging research area and there are still many problems to be solved.This paper introduces the concept of automatic software repair and puts forward a framework of automatic software repair based on search method.The major challenges and the key issues in the framework are summarized from three aspects,including fault localization,search strategy and test data generation.Finally,the summary of this paper and future work are given.The proposed framework and discussion on its key issues undoubtedly provide important basis for the further research and the application of automatic software repair in industry.

Keywords:search-based automatic software repair;fault localization;test data generation

1 引言(Introduction)

对于有缺陷的软件，一旦发生运行故障或其他软件错误，需要及时的对源程序进行修复。在发现软件缺陷后，软件自动修复是指，利用已有的程序补丁自动修复程序的过程[1]。具体而言，首先，识别可能存在缺陷的程序实体，其次，根据具有缺陷可能性的大小对可疑程序实体排序，并自动生成相关程序的补丁，接着，通过某种方法评估程序补丁的有效性，并选择正确的补丁。缺陷软件自动修复问题的研究具有极高的现实需求，可以有效减少程序调试时间，提高软件运行的效率，减少因缺陷带来的损失，因此，吸引了很多研究人员的关注[2-4]。

按照补丁生成的方式，软件自动修复的方法可以分为两大类：基于语义的修复方法和基于搜索的方法[4]。其中，基于搜索的方法是最早提出的一类软件程序修复方法[5]。该方法是一种“生成—确认”方法，将修复问题看成一个组合优化问题，运用元启发式搜索算法，在搜索空间内通过搜索生成候选补丁，并借助配套测试用例集对该补丁进行验证。基于搜索的软件自动修复能够利用已有的程序信息，并结合已有的软件维护技术，提供自动化的软件修复方案，易于部署和实施，在软件程序修复中占有重要地位。

已有的软件维护技术，例如缺陷定位，测试数据生成等，这些技术的应用降低了软件自动修复问题研究的难度，但是，如何更好地将上述技术应用到软件修复中，以及针对性的提高修复的效率，在该研究领域还需要解决很多问题。

在已有工作的基础上，提出和传统软件维护技术相结合的基于搜索的软件自动修复框架，指出其中需要解决的一些关键问题，并提出一些可行的解决方案，可以为从事软件自动修复研究提供参考。

2 基于搜索的软件自动修复框架(Framework of automatic software repair based on search method)

软件修复首先需要确定软件缺陷位于源程序的位置，其次，采用合适的方法生成补丁，最后，还要评估生成补丁的

基金项目：国际自然科学基金（项目编号：61203304，61502212）.

有效性。自动性的修复工作除了需要源程序外，还需要有配套的测试用例集，并且测试用例集中至少包含一个运行结果未达预期的测试用例。一方面，需要测试用例集收集程序的执行频谱，定位软件的缺陷；另一方面，若修复方法生成的补丁可以使得配套的所有测试用例均能执行通过，则可以认为该补丁正确。基于此，软件自动修复工作可以分为三个方面：(1)缺陷定位，(2)补丁生成，(3)补丁验证。进一步，如图1所示，本文提出基于搜索的软件修复框架可以分为五个阶段，期望在充分使用传统技术的基础上，例如缺陷定位、回归测试等等，致力提高软件修复的效率。

该框架依托于源程序和已有的测试用例集，首先进行缺陷定位，得到可疑的程序实体队列；然后，取怀疑度最大的程序实体，确定测试相关的范围，并对已有的测试用例集更新，接着，采用搜索的方法生成相关程序实体的补丁；对于生成的补丁，通过回归测试的方法对补丁进行验证，如果回归测试通过，则修复工作结束，否则，继续依次取怀疑度最大的实体，迭代进行补丁生成和验证。若上述步骤没有得到正确的程序补丁，则修复失败。

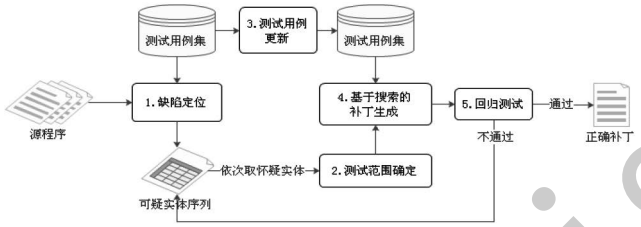

图1 基于搜索的软件自动修复框架

Fig.1 Framework of automatic software repair based on search method

3 基于搜索的软件自动修复关键问题(Key issues in search based automatic software repair)

基于搜索的软件自动修复框架涉及很多软件测试或维护的关键问题，目前尚未存在有效解决方案的如表1所列，在现有的软件测试和维护技术的研究基础上，根据采用技术的不同，这些关键问题，涵盖以下三个方面：(1)缺陷定位，(2)搜索策略，(3)测试用例的质量。表1所列关键问题的解决将有助于提高基于搜索的软件自动修复效率。

表1 基于搜索的软件自动修复关键问题列表

Tab.1 List of key issues in search based automatic software repair framework

编号	关键问题	阶段
1	大规模软件的缺陷定位	缺陷定位
2	缺陷的真实来源	测试范围确定
3	修复问题的模型建立	基于搜索的补丁生成
4	基于搜索的求解方法	基于搜索的补丁生成
5	高质量的测试用例生成	测试用例更新
6	测试用例的选择	回归测试

3.1 缺陷定位

所谓缺陷定位，是指基于程序的结构和测试用例执行的轨迹，确定软件缺陷的位置。基于缺陷定位的结果，能够有针对性的修复程序，从而降低程序修复的成本。可以看出，缺陷定位是软件自动修复中的先决问题，只有能够将存在缺陷的程序实体识别出来，才能够根据定位的结果生成程序补丁。在软件自动修复问题中，为了利用执行信息，便于部署实施，基于程序频谱的缺陷定位方法被广泛应用。按照程序频谱统计的数据，缺陷定位的结果一般是排好序的程序实体序列，依据缺陷定位方法的计算，序列首位的程序实体含有缺陷的可能性最大，序列末位的程序实体含有缺陷的可能性最小。如果序列首位的程序实体就是含有缺陷的程序实体，那么，只需要迭代一次补丁生成并验证的过程就可以完成软件自动修复过程。如果序列末位的程序实体是含有缺陷的程序实体，那么，软件自动修复的过程需要迭代多次才能找到正确的补丁。因此，缺陷定位结果的精确程度会极大的影响后续生成补丁的效率。

在缺陷定位中，一个待解决的关键问题是大规模软件的缺陷定位。在实际应用中，现有的基于频谱的缺陷定位方法需要能够准确提取每个程序实体被成功测试用例以及失败测试用例覆盖的次数，而软件的程序规模一般比实证研究中的测试对象规模大，这使得程序的频谱提取较难，缺陷定位的计算复杂度增加，且定位结果不准确，这就需要能够针对较大规模的软件提出针对性的缺陷定位方法。

另外，需要解决的关键问题是，在缺陷定位中，定位的程序实体往往是失效的程序实体，但该语句并不一定存在缺陷。具体而言，如果某个程序实体存在缺陷，那么，处于该实体之后的程序实体，由于缺陷的传播关系，都有可能发生失效。如果能够在缺陷定位中，每次都定位到导致程序失效的缺陷源头，将极大的提高软件自动修复的效率。

3.2 搜索策略

在软件工程中引入搜索的方法来解决问题，最早由Harman提出[6]，用来解决一些组合优化问题。在软件自动修复中，搜索的策略是影响修复效率的重要因素。用来修复程序的候选补丁，可以看成是对程序的一次修改行为，那么可以基于程序的修改，构造候选补丁，而生成候选补丁并进行选择的过程可以看成一个组合优化问题。基于该组合优化问题，可以尝试建立问题的数学模型并采用搜索的方法求解。

如前所述，基于搜索的软件自动修复技术是从候选空间中寻找补丁并进行评价的过程。针对该组合优化问题，如何建立一个有效的数学模型，学术界多次就此类问题研究[5]。在建立模型和求解的过程中，如果因补丁生成的操作算子较少，过小的解空间，容易导致无法找到最优解；而补丁生成的操作算子太多，过大的解空间将增加求解难度，典型的表现是，在求解过程中，易生成太多近似解，加大补丁验证的所耗费的时间。

另外，针对建立的数学模型，采取合适的求解方法也是修复技术的关键。已有的用来生成补丁和选择补丁的搜索方法如表2所列。这些方法，先构造程序对应的抽象语法树或测试用例集，组成进化种群，然后使用不同的搜索策略进化并搜索，以期望得到正确的程序补丁。其中，用遗传规划方法的GenProg[5]比较耗时，而采用随机算法，虽然能够减少方法的执行时间的，但需要更多的先验知识[9]。其他的搜索算法也存在效率过低，求解时间过长的问题。这意味着，对已有的搜索方法进一步研究，提出效率更高，更具有针对性的策略，是已有以后研究工作的重点。在不同的方法中，如何表示补丁，如何评价补丁，均是求解方法无可回避的难点。此外，参考有效的软件可靠性模型[10]，如何去除一些求解正确，但违背程序语义，不能应用于实际修复的补丁，也是以后研究工作需要解决的问题。

表2 已有的用于软件自动修复的搜索方法[5,7-9]

Tab.2 Existing search methods for automatic software repair[5,7-9]

搜索的方法	进化的种群
遗传规划方法	程序语法树
随机方法	程序语法树
自适应的进化算法	程序语法树
多种群进化算法	缺陷程序和测试用例集

3.3 测试用例的质量

对于修复后的程序，如果所有测试用例的运行结果均符合预期，这说明软件自动修复成功；反之，则修复失败。因此，测试用例质量影响软件修复过程的成败[11]。

首先，高质量的测试用例集是基于搜索的软件自动修复的方法必须提供的且测试用例应该随着程序的修复而不断迭代更新。这是因为，随着程序的修复，程序的控制结构、数据结构也可能发生改变，在这样的情况下，已有的测试用例可能存在充分性不够或部分失效。此时，如果按照现有测试用例修复软件的程序，那么，有可能存在过适应问题，即虽然修复的程序通过了测试用例的验证，但是修复的程序离预期的程序语义相差较大或仍然存在缺陷。在这种情况下，合理的做法是根据修复后的程序，对测试用例进行更新。鉴于软件修复的过程是一个迭代修复的过程，高效率的测试用例再生成方法有助于提高软件自动修复效率。

更为重要的是，软件初始配套的测试用例规模一般是比较大的，若所有测试用例均要运行，一次修复的若干补丁验证的过程就需要执行很长时间，这将极大的降低软件自动修复的效率。所以在软件修复中，需要合理的确定测试范围，并根据测试范围来选择测试用例对修复后的程序进行验证。该关键问题的解决不仅有利于提高回归测试的效率，也有利于进一步减少软件自动修复的时间。

需要提及的是，基于搜索的软件自动修复关键问题不仅仅是本文列出的六个问题，但在本文提出的基于搜索的软件自动修复框架下，上述六个问题的解决，应用于基于搜索的方法时，无疑将极大的提高软件自动修复的效率。

4 结论(Conclusion)

软件自动修复方法是目前软件维护领域的一个研究热点。本文结合传统的软件测试技术，提出基于搜索的软件自动修复框架，将软件修复看成一个不断使用测试用例对程序进行补丁的选择并评价的迭代过程。研究者需要注意缺陷定位、搜索策略，以及测试用例的质量，并提出了一些关键问题。因此，在已有研究工作的基础上，针对本文提出的关键问题，给出有效的具体解决方案将是下一步的研究工作。

参考文献(References)

[1] Kim D,et al.Automatic Patch Generation Learned from Human-Written Patches[C].International Conference on Software Engineering,2013:802-811.

[2] Goues CL,Forrest S,Weimer W.Current Challenges in Automatic Software Repair[J].Software Quality Journal,2013,21(3):421-443.

[3] Pei Y,et al.Automated Fixing of Programs with Contracts[J].IEEE Transactions on Software Engineering,2014,40(5):427-449.

[4] 玄跻峰,等.自动程序修复方法研究进展[J].软件学报,2016,27(4):771-784.

[5] Weimer W,et al.Automatically Finding Patches using Genetic Programming[C].International Conference on Software Engineering,2009:364-374.

[6] Harman M,Mansouri SA,Zhang YY.Search-Based Software Engineering:Trends,Techniques and Applications[J].ACM Computing Surveys,2012,45(1):1-6.

[7] Long F,Rinard M.An Analysis of the Search Spaces for Generate and Validate Patch Generation Systems[C].International Conference on Software Engineering,2016:702-713.

[8] Arcuri A,Yao X.A Novel Co-Evolutionary Approach to Automatic Software Bug Fixing[C].IEEE Congress on Evolutionary Computation(IEEE World Congress on Computational Intelligence),2008:162-168.

[9] Qi YH,et al.The Strength of Random Search on Automated Program Repair[C].Software Engineering,2014:254-265.

[10] 王二威.软件可靠性模型研究综述[J].软件工程,2016(02):1-2;57.

[11] Smith E K,et al.Is the Cure Worse than the Disease? Overfitting in Automated Program Repair[C].10th Joint Meeting on Foundations of Software Engineering,2015:532-543.

作者简介：

吴　川(1980-)，男，博士，讲师.研究领域：软件工程.