第37卷 第16期 计 算 机 工 程 2011年8月
Vol.37 No.16 Computer Engineering August 2011

·软件技术与数据库· 文章编号：1000—3428(2011)16—0082—02 文献标识码：A 中图分类号：TP311

一种改进的软件自适应随机测试策略

郑成文[1,2]，韩 柯[3]，张海粟[4]

(1. 总参通信训练基地，河北 宣化 075100；2. 通信指挥学院研究生大队，武汉 430010；
3. 中国电子系统设备工程公司研究所，北京 100141；4. 解放军理工大学指挥自动化学院，南京 210007)

摘 要：针对自适应随机测试(ART)存在的高维和距离度量问题，提出一种改进的软件自适应随机测试策略。分析 ART 的局限性，扩展 ART 的前提假设，基于测试用例特征相似性度量选择最佳的测试用例。实验结果表明，与传统随机测试策略相比，改进策略能更多地触发软件失效，测试效果更优。

关键词：软件测试；自适应随机测试；测试用例；测试序列；相似性

Improved Strategy for Software Adaptive Random Testing

ZHENG Cheng-wen[1,2], HAN Ke[3], ZHANG Hai-su[4]

(1. General Staff Communications Training Base, Xuanhua 075100, China;
2. Graduate Battalion, Commanding Communications Academy, Wuhan 430010, China;
3. Institute of China Electronic System Equipment Engineering Company, Beijing 100141, China;
4. Institute of Command Automation, PLA University of Science & Technology, Nanjing 210007, China)

【Abstract】Adaptive Random Testing(ART) is an enhanced version of Random Testing(RT). There are two factors that restrict the performance of ART, high-dimension data and distance metric. In this paper, an advanced strategy of ART based on test case similarity is presented by analyzing the limitation of ART and enlarging the hypothesis. Experimental result shows that the advanced ART, to some extent, can compensate for the limitations of existing ART and performs better than random test.

【Key words】software testing; Adaptive Random Testing(ART); testing case; testing sequences; similarity

DOI: 10.3969/j.issn.1000-3428.2011.16.028

1 概述

随机测试是一种有效的测试策略，该策略简单易行、成本低，不需要过多了解软件的内部结构信息，只是按一定的规则在整个输入域中随机产生测试用例，因而被广泛地用于软件测试和可靠性评估。

软件的失效区域常表现出一定的紧致性，即导致失效的输入点是紧密相连的，并且相邻的测试输入将导致相似的失效行为。基于这样的前提假设，文献[1]在随机测试策略的基础上提出了自适应随机测试(Adaptive Random Testing, ART)策略，具有代表性的 ART 算法有 FSCS-ART、RRT、MART 等；该文献还研究了影响 ART 性能的有利条件和不利条件。在 ART 策略中，测试用例不仅可以随机的产生，同时又根据测试用例执行结果的历史信息来选择新的测试用例。根据 ART 的前提假设和相应的算法，新的测试用例应该是触发软件失效可能性最大的。

文献[2]对 ART 的性能进行分析，并进行了改进；文献[3]提出了面向对象软件的自适应随机测试策略；文献[4]等也对自适应随机测试的样本分布情况提出了改进意见；文献[5]提出了一种各维独立递增的 FSCS 算法，从面向对象的角度给出了测试用例距离的一种定义，并开发了一个基于 Java 的自适应测试工具 JAT。此外，也有研究者将其他的自适应算法应用于测试用例的生成[6]。针对 ART 存在的局限性，本文对其前提假设进行扩展，提出一种基于测试用例相似性度量的软件自适应随机测试策略。

2 ART 的局限及原因分析

目前 ART 存在的局限主要有以下 2 种：(1)高维问题，如果输入域是二维空间，ART 表现出很好的性能，然而当输入域是三维以上空间时其有效性明显下降。事实上对于大多数软件，在操作使用时需要输入多个参数，即属于多维的情况，所以，ART 的应用效果受到了很大的限制。(2)距离度量，实际应用时参数大多是非数值型，即使是数值型的参数，也常常包含各种数据类型，即软件的输入空间不是欧氏空间，不容易找到标准的距离度量方法。即使是欧氏空间，由于不同参数的量纲不同，计算时也会出现大数“淹没”小数的问题。

ART 的前提假设是软件的失效区域呈现出一定的紧致性，即引起失效的输入点是聚集的，这里涉及到空间分布的范畴。当空间的维数较多时，空间本身就会变得稀疏，引起失效的输入点已不再紧密相连，距离会很大。所以高维问题从根本上打破了 ART 的前提假设。同时，随着维数的升高，距离的计算的复杂度会越来越高。

软件工程的领域知识表明，软件的缺陷分布常常是密集的，即 80%的软件缺陷集中于 20%的软件空间。这里的软件空间是指软件的结构，比如软件的功能点、执行路径或功能模块。经过多次映射后，缺陷的位置及分布同引起软件失效的输入及分布会有一定的差别，缺陷密集分布未必导致失效区域密集分布，尤其是对于大型复杂软件。然而对于小型软件，或是简单的应用程序，缺陷分布可以近似等同于失效区

基金项目：国家“973”计划基金资助项目(2007CB310800)

作者简介：郑成文(1979－)，男，博士研究生，主研方向：软件质量评测；韩 柯，研究员、博士生导师；张海粟，博士研究生

收稿日期：2011-02-17 **E-mail：**chengweninchina@sina.com

域的分布，此时符合 ART 的前提假设。而对于大型复杂软件的系统级测试，ART 的前提假设可能不成立。综上分析，需要对 ART 的前提假设做一些扩展。

3 ART 前提假设的扩展

软件失效常常表现出关联现象，不同的软件失效之间存在着一定的相关性，软件缺陷关联是软件失效关联的根源，只是在找出对应的缺陷之前，这种相关性难以准确地分析。比如，一个失效会引起或掩盖另一个失效，有时虽然纠正了源程序中的一处错误，相关的失效则都不再出现了。软件失效是由特定的测试用例触发产生的，可以从软件失效的相关性映射到测试用例特征的相似性，将 ART 的前提假设扩展为相似的测试用例可能导致相似的失效行为，即导致软件失效的测试用例在某一方面具有相似的特征。扩展后的前提假设可以涵盖先前的前提假设，因为测试输入之间距离的相近只是特征相似的一种情况。通过对前提假设的扩展，可以根据已执行测试用例的特征来预测新的失效可能发生的位置，利用测试用例特征相似性度量，选择最佳的测试用例。

4 测试用例的相似性度量

早期的 ART 研究中只是将测试输入用于测试用例的距离度量，这样一方面当输入参数较多即维数较高时，失效区域的紧致性就会明显下降；另一方面，当软件操作界面切换后，测试输入的参数也会发生变化，而界面切换前后的输入参数往往有着密切的关系。软件执行的功能序列反映了软件的行为，软件出现失效的关键原因是执行了一个或一些特定的功能序列，因此有必要拓展测试用例的相似性度量，将测试用例之间的距离由测试输入扩展到测试的执行序列。

对于大型软件项目的测试，测试过程中需要使用大量的、甚至是海量的测试用例。每个测试用例中包含着许多要素或属性，如测试场景、测试输入、执行序列、预期输出、实际输出和失效现象等，其中测试输入可能包含许多参数的取值。每一个属性都可以代表测试用例的某种特征，如果能根据软件的运行结果，恰当地选取测试用例特征并给出相应的度量方法，则可以使相似的测试用例在某些特征空间上表现为紧致性。根据已使用测试用例的执行结果，尤其是根据导致软件失效的测试用例的特征预测新的软件失效的可能位置。具体的说，如果某一测试用例触发了软件失效，生成的新测试用例要尽可能与该测试用例相似；反之，生成的新测试用例要尽可能与该测试用例相异。为此给出下列定义：

定义 1(测试序列) 测试序列是对测试用例的抽象。一个测试序列可以包含多个状态，一个状态可以包含多个参数，如 $S=\{c_1,c_1,\mathrm{L}\ ,c_n\}$，其中，$c_i$ 表示软件的不同状态，一个状态可以是一个人机交互界面，也可以是一个事件。c_i 可以有多个不同的输入变量 $Ic_i=\{ic_{i1},ic_{i2},\mathrm{L}\ ,ic_{im}\}$。不同的输入变量可以有不同的数值类型，可以是区间数值，也可以是离散数值等。每一个输入变量有相应的输入域。

测试用例特征的相似性包含 2 个层次，第 1 个层次为序列间的相似性。

定义 2(序列间的相似性) 对于给定的测试场景，不同序列的相似性度量为相同状态的数量。设 S_a 由 $c_1\ c_2\ c_3\ c_4$ 顺序组成，S_b 由 $c_1\ c_2\ c_3\ c_5$ 顺序组成，则两序列的相似度为 $Sims(S_a\ \ S_b)=3$。

第 2 个层次为状态间的相似性。同一状态由于参数取值不同而引起差异。不同的参数可能有不同的数值类型，即使是相同的数值类型，不同的参数可能有不同的量纲。也就是说，多数情况下状态空间不是欧氏空间。所以，用欧几里德距离来表征相似性的差异将可能失去意义，需要使用新的距离度量方法。可采用各参数分别计算距离再归一化求和的方法。由此得到以下定义：

定义 3(状态间的相似性) 对于给定的测试序列，其对应状态输入参数取值的相似性度量为 $Simc(c_a\ c_b)=\sum_{j=1}^{m}[1-dis(Ic_{ja}\ Ic_{jb})]$，某一状态的某一参数由于不同取值所引起差异用 $dis(Ic_{ja}\ Ic_{jb})$ 表示，由于软件输入空间可能很大，同一参数的距离值求出后需进行归一化处理，从而减小求和后大取值范围参数对小取值范围参数的影响。

5 测试步骤及实验

5.1 测试步骤

本文所提出策略的基本流程如图 1 所示，具体步骤如下：

(1)根据测试需求，确定一个测试场景，之后随机选择在此场景下的一个测试序列。对于序列中的每一个状态，为每一个参数随机选择一个输入值。一边选择测试输入，一边执行测试，并对实际输出与预期输出进行比较，观察软件的失效现象。

(2)若软件出现失效，记录失效信息。不改变测试序列，改变状态的参数值，并使状态间相似度最大。执行后，若失效现象相同，则重新选择一条执行序列作为新的测试用例，并使 2 条序列的相似度最大。

(3)若该测试序列未出现失效，重新选择一条执行序列作为新的测试用例，并使两条序列的相似度最小。

(4)在第(2)步中，改变状态的参数值后出现了不同的失效现象或无失效，则再次改变序列中状态的参数值，并使两状态间的相似度最小。

(5)测试序列如果经过多次更改，软件均未出现失效，并满足了相应的结束条件，则改变测试场景。结束条件的判断由测试需求和软件的复杂性决定。

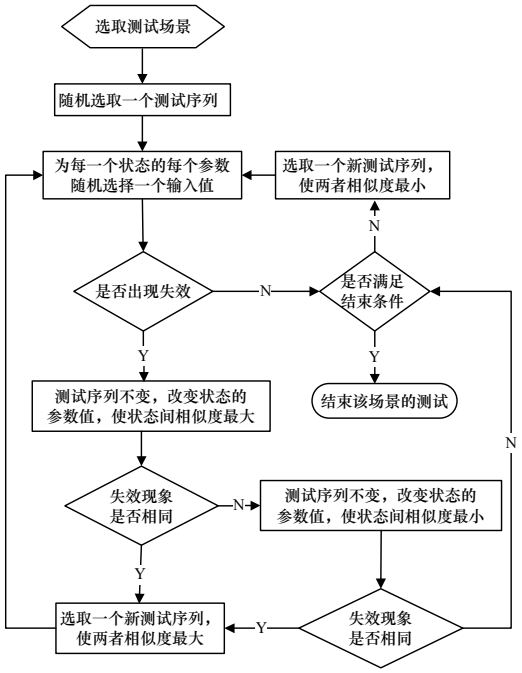

图 1 本文策略的基本流程

(下转第 100 页)