最小测试用例集生成方法改进及应用

万松松[1],薛锦云[1,2],谢武平[1]
(1.江西师范大学 计算机信息与工程学院,江西 南昌 330027;
2.中国科学院 软件研究所,北京 100080)

摘 要:软件测试是保证软件质量和可靠性的重要手段,如何对软件进行全面且高效的测试一直是备受关注的问题。分析了白盒测试与黑盒测试的优缺点;具体分析了最小测试用例生成算法,接着对生成最小测试用例集的方法提出改进:首先消除掉测试需求中存在的冗余,再对由该测试需求生成的测试用例集使用简化算法,得到一组无冗余的测试用例集。这种先对测试需求进行精简的方法,使得测试用例集中测试用例的数量大为减少,提高了简化算法的使用效率。将上述最小测试用例集生成方法运用到Apla到delphi生成器系统的测试中,提高了测试效率。

关键词:软件测试;测试用例;测试需求;测试用例精简
中图分类号:TP306+.2 **文献标识码**:A **文章编号**:1673-629X(2008)10-0181-03

Improvement and Application of Minimal Test Suite Generation Technology

WAN Song-song[1], XUE Jin-yun[1,2], XIE Wu-ping[1]
(1.College of Computer Information Engineering, Jiangxi Normal Univ., Nanchang 330022, China;
2. Institute of Software, Chinese Academy of Sciences, Beijing 100080, China)

Abstract: Software testing is an important mean of ensuring software quality and reliability. How to conduct a comprehensive and efficient software testing has been a major concern. Analyzes the advantages and disadvantages of white box testing and black box testing, and then proposes some suggestions to improve the methods of generating the minimal test suite, which improves the efficiency in the use of simplified algorithm. At last, use the technology in our testing work, and this method increases testing efficiency.

Key words: software testing; test case; testing requirement; test suite reduction

0 引 言

软件测试是一个十分复杂的过程。测试用例的数量和质量决定软件测试的成本和有效性,如何得到高效的测试用例集是测试工作中最为重要的组成部分。而设计测试用例是一项细致并需要高度技巧的工作,稍有不慎就会顾此失彼而发生疏漏,导致影响测试效果。

因此人们一直在研究如何设计出一组有效、数量少又能充分满足测试需求的测试用例集,从而在保证和提高软件测试质量的同时,降低软件测试的成本。

收稿日期:2008-01-03
基金项目:国家自然科学基金项目(69783006)
作者简介:万松松(1982-),女,江西南昌人,硕士研究生,研究方向为软件形式化和自动化;薛锦云,教授,博士生导师,主要研究方向为软件形式化和自动化、并行分布式计算、智能教学软件、软件工程等。

1 测试技术研究

1.1 黑盒测试与白盒测试

目前人们已经提出许多测试方面的相关研究成果及理论,黑盒测试和白盒测试是两类广泛使用的软件测试方法。

黑盒测试以用户的观点,从输入数据与输出数据的对应关系出发进行测试。它不涉及到程序的内部结构。黑盒测试法注重于测试软件的功能需求,主要试图发现几类错误:功能不多或遗漏、界面错误、数据结构或外部数据库访问错误、性能错误、初始化和终止错误。具体的黑盒测试方法包括等价类划分、因果图、正交实验设计法、边界值分析、判定表驱动法、功能测试等。使用黑盒测试,测试人员无需深入烦琐的程序代码,但是如果程序实现了没有被描述的行为,黑盒测试永远也不会认识到。

白盒测试是一种按照程序内部的逻辑结构和编码结构设计并执行测试用例的测试方法。采用这种测试

方法，测试者需要掌握被测程序的内部结构。白盒测试通常根据覆盖准则设计测试用例，使程序中的每个语句、每个条件分支、每个控制路径都在程序测试中受到检验。具体的白盒测试方法包括控制流分析、数据流分析、逻辑覆盖、域测试、符号测试、路径分析、程序变异等。但是如果已描述行为没有被实现，白盒测试永远也不会揭示这一点。

黑盒测试与白盒测试各有优缺点，单独使用都不充分[1]。黑盒测试常常会有冗余和漏洞两方面的问题，如果黑盒测试结合白盒测试覆盖率指标执行，冗余和漏洞问题都会被发现并解决[1]。

1.2 最小测试用例集生成技术

提高软件测试的成本和有效性，受到测试用例的数量和质量的影响。因此，人们一直在研究如何设计出一组有效、数量少又能充分满足所有测试需求的测试用例集。为了得到最小测试用例集，目前一般的处理方法是：首先根据测试目标中的每个测试需求确定出相应的测试用例集，然后对这个测试用例集采用简化算法进行精简，去掉一些冗余的测试用例，最后得到一组最小测试用例集。现有的简化算法有：

(1)贪心法：每次从 T 中挑选一个测试用例，使之能最多地满足所有未被满足的测试需求，在测试需求集中去掉已经被满足的测试需求，直到所有测试需求都被满足。该算法的最坏时间复杂度是 $O(mn \cdot \min(m,n))$[2]；

(2)Harrold 等提出一种根据测试用例的重要性来选择测试用例的启发式方法[3]。该方法使用某种策略决定测试用例的“重要性”，按照测试用例的“重要性”依次挑选测试用例，然后将被这些测试用例满足的测试需求从 R 中去掉，直到所有测试需求都被满足。这种方法最坏的时间复杂度是 $O(m(m+n)l)$；

(3)Chen 与 Lau 提出的一种新算法，该算法不仅结合了贪心法和 Harrold 等提出的启发式算法的优点，同时还充分考虑了剔除 1－1 冗余策略的简化方法[4]。这种方法首先选出必不可少的重要测试用例，把这些测试用例所满足的测试需求从集合 R 中去掉；然后利用贪心法选出能最多地满足未被满足的测试需求的测试用例，并把相应的测试需求从 R 中去掉，直到所有的测试需求都被满足结束。算法最坏时间复杂度是 $O(mn+\min(m,n)\ nk)$。

上面所述的最小测试用例生成的方法都是对测试用例进行简化，简化算法的运行效率除了取决于算法本身外就是算法所处理的测试用例的数量，测试用例的多少将直接影响到简化算法的使用效率。然而在测试需求分析阶段就发现测试需求集中往往就已存在冗余，一个测试需求点通常对应一个数量较多的测试用例集，测试需求中的冗余会导致大量测试用例的冗余。在测试需求分析阶段消除测试需求中存在的冗余将大幅降低测试用例的数量，这将提高简化算法的使用效率。在不降低测试质量的情况下，可精简测试需求集。因此采用自顶向下逐步细化同时消除冗余的方法得到测试需求，该无冗余的测试需求对应一组测试用例集 T，最后对测试用例集 T 使用 Chen 和 Lau 提出的简化算法产生最小测试用例集。

由顶向下逐步细化同时消除冗余的方法步骤描述如下：

第一步，由测试目标得到若干个测试需求点 R_0，$R_1,R_2,\cdots,R_s$；

第二步，用图 1 所示算法消除测试需求 $R_0,R_1,R_2,\cdots,R_s$ 中的冗余，得到一组无冗余的测试需求 $R'_0,R'_1,R'_2,\cdots,R'_t$；

第三步，划分每个测试需求得到其子需求($R_{00},R_{01},R_{02},\cdots,R_{0k0}$)，($R_{10},R_{11},\cdots,R_{1k1}$),…，($R_{t0},R_{t1},\cdots,R_{tkt}$)；

第四步，使用如图 1 所示算法消除第三步所述子需求的冗余，得到一组新的无冗余的子需求；

第五步，划分第四步中得到的子需求；

第六步，消除冗余；……，依次类推，直至得到最终的测试需求集(如图 2 所示)。

```
for each ri in {r0,r1,r3,…}
  for each rrj in {rj+1,rj+2,rj+3,…}
    if ri=rrj
      将 rrj 从测试需求集 R 中删除
    end
  end
end
```

图 1 冗余消除算法

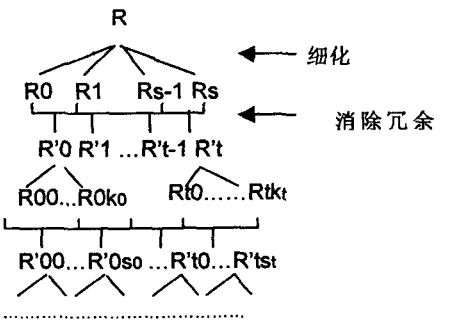

图 2 最小测试用例生成方法结构示意图

用这种方法来生成测试需求集不仅方法简单、符合程序员编写测试需求的过程，同时还会大幅度降低测试用例中的冗余度，在对测试用例集使用简化算法

时提高算法的使用效率。

2 测试技术的应用

2.1 待测系统分析

PAR 方法[5,6]是一种统一实用的算法程序设计和证明的方法,apla 是为实现算法程序形式化开发的 PAR 方法而定义的一种抽象程序设计语言。而 apla 语言→delphi 语言生成系统的实现是 PAR 方法的关键技术。笔者的测试工作就是基于 apla 语言→delphi 语言生成系统的测试。

待测问题是 apla 语言到 delphi 语言转换系统,转换系统要实现的功能是将 apla 语言准确无误地转换成 delphi 语言,其功能目的性非常明确[7]。

结合上述对白盒和黑盒测试的分析,使用黑盒测试,同时结合白盒测试覆盖率指标执行,用来解决黑盒测试中常出现的冗余和漏洞两方面的问题。

转换系统采用了模块化来实现,在生成测试需求点和测试用例集时,充分利用该系统模块化特性来减少测试用例的数量。生成测试需求的时,有些模块实现的功能必然会成为多个测试需求点中的测试目标,因此,在不降低测试质量的情况下,可精简测试需求集。

2.2 测试技术的运用

以笔者的工作为实例,采用以上方法来处理测试需求。根据《抽象程序设计语言 apla 报告》[8]对 apla 语言功能进行等价类划分,得到 8 个测试需求点分别为:程序结构测试点(R_0)、标志符说明部分测试点(R_1)、常量说明部分测试点(R_2)、类型申明部分测试点(R_3)、变量说明部分测试点(R_4)、表达式部分测试点(R_5)、过程与函数说明部分测试点(R_6)、语句序列测试点(R_7);根据等价类划分、边界值分析等方法划分每个测试需求得到其子需求($R_{00}, R_{01}, R_{02}, \cdots, R_{0k0}$),($R_{10}, R_{11}, \cdots, R_{1k1}$),…,($R_{70}, R_{71}, \cdots, R_{7k7}$);消除该子需求冗余,得到一组新的需求集;消除新需求集的冗余;依次类推。如在细分 8 个测试需求点后,关于"常量说明部分"根据常量说明格式的定义又得到了 8 个测试子需求,其中包括子需求:"标志符取名是否合法";而在"变量说明"中也包含子需求:"标志符取名是否合法",此时根据上述方法将"变量说明部分"的"标志符取名是否合法"消除掉。

最后得到一组测试需求集,该测试需求集对应的测试用例数目比未用该方法求解出来的测试用例集数目大幅减少。

测试需求集确定后,对该测试需求集所对应的测试用例集 T 使用 Chen 和 Lau 在文献[4]提出的的简化方法产生一组最小测试用例集。

3 结束语

比较了黑盒测试与白盒测试的优缺点,根据待测系统本身的特点选取测试方法。着重研究了最小测试用例集生成算法,指出在生成测试用例的过程中,可以使用自顶向下逐步细化同时消除冗余的方法求得测试需求集,尽量减少测试用例集数量。最后运用 Chen 和 Lau 提出的最小测试用例算法来生成最小测试用例集。

参考文献:

[1] Jorgensen P C. Software Testing: A Craftsman's Approach [M]. 2nd Edition. [s. l.]: CRC Press LLC, 2002.

[2] Johnson D S. Approximation algorithms for combinatorial problems[J]. Journal of Computer and System Sciences, 1974, 9(3): 256 - 278.

[3] Harrold M J, Gupta B, Soffa M L. A methodology for controlling the size of a test suite[J]. ACM Transactions on Software Engineering an Methodology, 1993, 2(3): 270 - 285.

[4] Chen T Y, Lau M F. A new heuristic for test suite reduction [J]. Information and Software Technology, 1998, 40(5/6): 347 - 354.

[5] Xue Jinyun, Ruth D. A simple program whose derivation and proof is also[C]//In Proceedings of The First IEEE International Conference on Formal Engineering Method (ICFEM' 97). [s. l.]: IEEE CS Press, 1997.

[6] 薛锦云,杨庆红,李云清. 程序设计方法学[M]. 北京:高等教育出版社,2001.

[7] 周　侃. 支持范型程序设计的 Apla - delphi 自动程序转换系统的研制[D]. 南昌:江西师范大学,2003.

[8] 薛锦云. 抽象程序设计语言 Apla 报告[R]. 南昌:江西师范大学计算机软件研究所,2002.

(上接第 180 页)

[4] International Telecommunication Union. Packet - based Multimedia Communications System, Recommendation H. 323. Telecommunication Standardization Sector of ITU[R]. Geneva, Switzerland: [s. n.], 1998.

[5] 北京飞漫软件技术有限公司. MiniGUI 用户手册[M]. 北京:北京飞漫软件技术有限公司,2003.