面向对象软件回归测试技术研究*

方 菲, 孙家骕, 王立福, 杨芙清

(北京大学 计算机科学与技术系,北京 100871)

E-mail: sjs@cs.pku.edu.cn

http://www.cs.pku.edu.cn

摘要: 回归测试的目标之一是在程序修改后,只对进行修改的部分重新测试,从而达到与完全测试相同的测试覆盖.利用数据流或部分数据流技术,对结构化程序进行回归测试的技术已相继提出.随着面向对象方法的逐渐成熟,对面向对象软件测试技术的研究有了迫切的需求.通过分析对象系统的特性,定义了对象之间的依赖关系.通过这个依赖关系,导出测试对象的方法序列,并应用程序切片技术,标识那些受到程序修改影响的测试用例,只有这些测试用例才需要在回归测试中重新执行.最后给出了一完整的对象系统的回归测试解决方案.

关键词: 面向对象;增量层次模型;程序切片;数据流;回归测试

中图法分类号: TP311 **文献标识码**: A

软件测试的任何阶段均涉及到回归测试问题.完全的回归测试(即重新执行所有的测试用例)不仅没有必要,而且从测试代价上讲也是不可能的.目前已提出的回归测试技术大都是针对结构化程序[1~3]的.随着面向对象(object-oriented,简称OO)方法的成熟,OO软件测试已成为目前的研究热点之一.OO软件测试主要是从测试充分性角度出发,研究类(如果不特别说明,本文中的类与对象均指对象实例的集合)及子类的测试技术(包括类集成技术),而对回归测试技术却研究得很少.

本文研究了OO程序中对象的数据依赖和对象方法的测试依赖,给出了测试依赖的计算算法,并在此基础上提出了基于类的OO软件回归测试策略.

1 对象系统中的依赖

1.1 对象之间的数据依赖

面向对象方法的两大特点是封装和继承.一个对象(如C++中的类)的定义可能是嵌套的,因为类能封装另一个类的一个(或多个)实例作为它的属性;另一方面,对象的定义是分散的,因为类能作为子类继承其父类的属性和方法,即子类的某些属性和方法是在父类中定义的.通常,称包含其他对象的对象为复杂对象,不包含任何其他对象的对象为简单对象.虽然封装和继承给程序设计带来巨大的好处,但却极大地增加了程序的测试难度.

为了研究由封装构成的依赖,本文采用数据依赖分析的方法进行讨论.对象之间的依赖关系用

* **收稿日期**: 1999-11-15; **修改日期**: 2000-01-13

基金项目: 国家"九五"重点科技攻关项目资助(98-780-01-02)

作者简介: 方菲(1963-),女,湖北武汉人,博士生,主要研究领域为软件测试,软件可靠性工程;孙家骕(1946-),男,吉林扶余人,副教授,主要研究领域为程序设计语言,编译系统;王立福(1945-),男,河北保定人,博士,教授,博士生导师,主要研究领域为软件工程,软件质量保证;杨芙清(1932-),女,江苏无锡人,教授,博士生导师,中国科学院院士,主要研究领域为操作系统,软件工程,软件工程环境,面向对象方法.

符号 $\mathcal{R}$ 表示.

设 Os 为简单对象集,Oc 为复杂对象集.在给出对象之间的依赖关系之前,首先给出如下定义.

定义 1. 给定两对象 α 和 β,且 $\alpha\in Os$,$\beta\in Oc$,$\exists\ b$,b 是 β 的非对象属性,如果 b 值的改变会导致 α 中某些属性值的改变,则称 α 数据依赖于 b,表示为 $\alpha\mathcal{R}b$.

定义 2. 给定两对象 α 和 β,且 $\alpha\in Oc$,$\beta\in Os$,$\exists\ a$,a 是 α 的非对象属性,如果 β 中某些属性值的改变会导致 a 值的改变,则称 a 数据依赖于 β,表示为 $a\mathcal{R}\beta$.

定义 3. 给定两对象 α 和 β,且 $\alpha\in Os$,$\beta\in Oc$,$\exists\ b$,b 是 β 的对象属性,如果 b 的属性值的改变会导致 α 中某些属性值的改变,则称 α 数据依赖于 b,表示为 $\alpha\mathcal{R}b$.

定义 4. 给定两对象 α 和 β,且 $\alpha\in Oc$,$\beta\in Os$,$\exists\ a$,a 是 α 的对象属性,如果 β 中某些属性值的改变会导致 a 的属性值的改变,则称 a 数据依赖于 β,表示为 $a\mathcal{R}\beta$.

由定义 1~4,给出如下定义.

定义 5. 给定两对象 α 和 β,如果它们之间满足下列条件之一:

(1) $\alpha,\beta\in Os$,β 中某些属性值的改变可能会导致 α 中某些属性值的改变;

(2) $\alpha\in Os$,$\beta\in Oc$,$\exists\ b$,b 是 β 的属性,$\alpha\mathcal{R}b$;

(3) $\alpha\in Oc$,$\beta\in Os$,$\exists\ a$,a 是 α 的属性,$a\mathcal{R}\beta$;

(4) $\alpha,\beta\in Oc$,$\exists\ a$,a 是 α 的属性,$\exists\ b$,b 是 β 的属性,$a\mathcal{R}b$,

则称 α 数据依赖于 β,表示为 $\alpha\mathcal{R}\beta$.

为了计算系统中一个对象的依赖集,可以有两种方法.一种方法是,首先对复杂对象进行分解,直至分解到简单对象;然后,从简单对象到复杂对象自底向上地计算每个对象的依赖集.第 2 种方法是,如果用顶点表示对象,用有向边表示对象之间的依赖,则系统中对象之间的依赖关系是一种有向图.因此,求一个对象 P 的依赖集,等价于把图中的有向边改向后,求从顶点 P 的可达结点集.

1.2 方法的测试依赖

对象的交互,使得 OO 系统中任一方法存在着一个方法调用序列.为了研究对象交互引起的方法之间的测试依赖关系(即测试次序关系),本文借助于类图[2]表示,通过把类图转换成 AOV[4]图,并进行拓扑排序,从而得到对象方法的测试依赖关系.

在类图中,结点表示类中的一个方法或属性成员,边表示消息.类图可以是非连通的,图中每一个连通子图表示类中互相交互的属性.如图 1 所示.

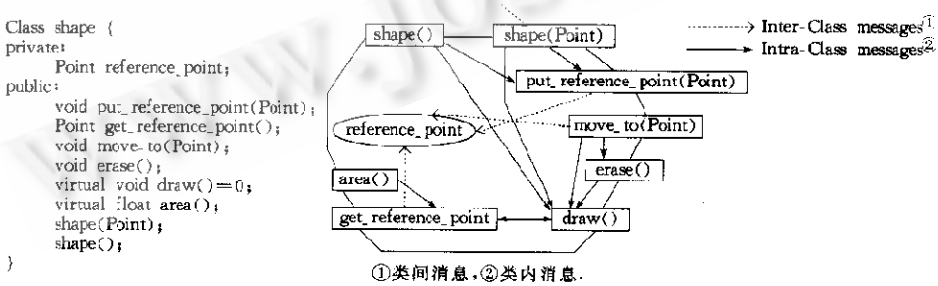

Fig. 1 The definition and class-graph of class Shape

图1 类Shape的定义及其类图

在图 1 中,从任意结点出发进行图的深度遍历可得到某一方法的调用序列,如 move_to(Point)→erase()→draw()→get_reference_point().测试时,方法调用序列实际上也构成了一个测试序列.即为了使测试更有效,一个好的测试过程应该是按方法调用序列自底向上进行的.

通过类图可以用如下算法构造出测试依赖图：

(1) 在类图中，方法结点为测试依赖图的结点；

(2) 对类图中的方法结点，如果方法 A 调用方法 B，则在测试依赖图中，有从 B 到 A 的有向边；

(3) 重复步骤(2)，直至类图中所有方法结点之间的边全部转换至测试依赖图中.

由此，得到如图 2 所示的测试依赖图.

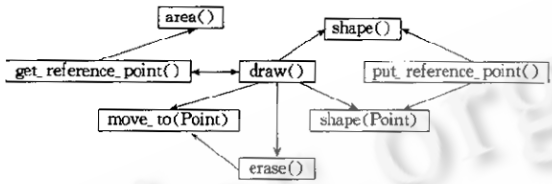

Fig. 2 The test dependency graph of class Shape
图2 类Shape的测试依赖图

如果将箭头终端结点作为优先测试的方法，将箭头始端作为其后测试的方法，那么，图 2 即为一个有向 AOV 网，类 Shape 方法的测试序列即为此 AOV 网的一个拓扑排序. 在构造 AOV 网时，可能会遇到有向环，这时，需打破环后再进行拓扑排序. 如图 2 所示，get_reference_point()与 draw()之间构成环，可去掉一个弧 draw()→get_reference()，得到其两个拓扑排序为 get_reference_point()-area()-draw()-erase()-move to(Point)-put_reference_point-shape()-shape(point)以及 put_reference_point()-get_reference_point()-area()-draw()-erase()-move_to(Point)-shape()-shape(point).

环的打破策略依赖于对具体应用程序的分析. 以上的拓扑排序表示了对象方法的一种测试优先次序，一方面，在测试时，可依据其进行构造桩和驱动；另一方面，在回归测试时，可依据其确定方法的回归测试次序.

2 面向对象软件回归测试

2.1 层次增量模型

OO 软件测试在技术上主要是沿用传统软件测试技术，如数据流技术. 而软件测试过程则是以层次增量方式进行的[1]. 层次增量测试，首先对类方法进行测试，将类方法集成为一个类，并进行测试，直至其成为安全部件(即满足测试充分性准则的部件)；然后将多个类集成为类簇或子系统，并进行集成测试；最后将多个类簇或子系统集成为最终系统，并进行系统测试.

在单个对象方法或方法的集成测试中，都需要确定对哪些测试用例进行回归测试. 可见，OO 回归测试与结构化回归测试最大的不同在于：OO 系统中的回归测试不再作为测试的一个独立的阶段(如单元或集成回归)，而是以增量方式进行的.

对象系统中的交互，既发生在类内的方法间又发生在多个类之间. 类 A 与类 B 的交互是通过：

(1) B 的实例变量作为参数传给类 A 的某方法. 这时，类 B 的改变必然导致对类 A 的方法的回归测试，如果类 A 的方法又与其他方法发生交互，这些方法及其类必须都进行回归测试.

(2) 类 A 的实例是类 B 表示的一部分. 这时，B 的所有对类 A 中变量进行引用的方法必须进行回归测试.

在这两种情况下，测试依赖集可用来确定情况(1)中所有与方法 A 的交互集；对象的数据依赖

集可用来确定情况(1)和(2)中所有必须重新测试的类和方法.

2.2 程序修改后的回归测试

从代码上看,对象系统中的修改主要有:

(1) 不修改系统中的对象属性,只对方法进行修改,包括对方法协议和方法体的修改;

(2) 修改系统中的对象属性.

通常,OO 系统中对象的属性是不会改变的.但由于设计的原因,可能会造成对系统中某个甚至多个对象的方法的修改,这种修改的影响是巨大的.它意味着,该对象以及系统中所有与该对象发生交互的对象都要进行完全新的测试.通常,这种测试代价是很高的.

协议的修改实质是对方法的重新定义,这时,回归测试应包含两方面的含义:一方面是必须构造足够的测试用例,按其测试准则,对该方法进行充分测试;另一方面是对所有与该方法有依赖关系的类,必须构造新的测试用例并对它们的交互进行充分测试.在方法体修改中,如果修改语句会对某一对象状态产生影响,那么,所有与此对象存在数据依赖的对象都必须进行回归测试.

因此,程序修改之后,为进行回归测试,需要重新计算程序修改后对象的数据依赖集和方法的测试依赖集.另外,应用动态程序切片技术[5,6],对系统中的所有方法进行下列计算:

执行切片:一个测试用例执行所包含的语句集.

动态切片:一个测试用例执行所包含的且能影响程序输出的语句集.

相关切片:一个测试用例执行所包含的且修改后可能会影响程序输出的语句集.

过程间动态切片:程序执行时,影响一个指定变量值的调用语句的集合.

这样,当对类的拓扑序列集中的某一方法 m 进行修改后,只需将执行切片、动态切片以及相关切片中包含修改语句的测试用例加入回归测试用例集中,并进行测试;然后,对于 m 的所有后继结点(即方法),将过程间动态切片中包含修改语句的测试用例,加入回归测试用例集中,并进行测试.

3 结 论

完全的回归测试不仅没有必要,而且由于测试成本等原因,完全的回归测试也是不可能的.本文分析了对象系统中类及方法之间的依赖关系,并通过这个依赖关系,应用程序切片技术给出了对象系统的回归测试解决方案.在 OO 程序中,单元和集成回归阶段通常是交叉进行的.类方法的改变所导致的影响范围通常是相当大的,也就是说,回归测试用例集会很大.另外,为了进行回归测试,必须计算数据依赖及测试依赖.如果系统中对象之间有着很强的依赖性(即对象之间数据耦合程度高),那么为了确定修改影响范围,其代价通常是非常高的,由此而引出的回归测试代价问题有待于进一步研究.

References:

[1] Harrold, M. J., Soffa, M. L. Interprocedural data flow testing. In: Proceedings of the Symposium on Software Testing, Analysis and Verification (TAV3-SIGSOFT89). Key West, Florida, 1989. 158~167. http://wedgwood.cs.umass.edu/cs521/readinglist.html.

[2] Rajiv, Gupta, Harrold, M. J., Soffa, M. L. An approach to regression testing using slicing. In: Proceedings of the Conference on Software Maintenance. IEEE Computer Society Press, 1992. 299~308. http://citeseer.nj.nec.com/check/140273.

[3] Bates, S., Horwitz, S. Incrementa. program testing using program dependence graphs. In: Conference Record of the 20th Annual ACM SIGPALAN-SIGACT Symposium on Principles of Programming Languages. ACM Press, 1993. 384~396.

http://www.cs.wise.edu/wpis/html/.
[4] Yan, Wei-min, Wu, Wei-min. Data Structure. Beijing: Tsinghua University Press, 1992 (in Chinese).
[5] Weiser, M. Porgram slicing. IEEE Transactions on Software Engineering, 1984,SE-10(4):352~357.
[6] Lueng, H. K. N., White, L. A study of regression testing. Technical Report, TR-88-15, Department of Computer Science, University of Alberta, Canada, 1988.

附中文参考文献:

[4] 严蔚敏,吴伟民. 数据结构. 北京:清华大学出版社,1992. 179~188.

An Approach to Object-Oriented Software Regression Testing*

FANG Fei, SUN Jia-su, WANG Li-fu, YANG Fu-qing

(*Department of Computer Science and Technology, Beijing University, Beijing* 100871, *China*)

E-mail: sjs@cs.pku.edu.cn

http://www.pku.edu.cn

Abstract: After changes are made to a previously tested program, a goal of regression testing is to retest the program based only on the modification while maintaining the same testing coverage as the complete retesting of the program. Several regression techniques for structural programs using data flow or partial data flow (also known as program slicing) have been proposed. With the object oriented (OO for short) method growing mature, there is an urgent need of testing techniques for OO programs. In this paper, based on the analysis of characteristics of OO system, firstly, the dependency relations among objects are defined, from which series of objects' methods are deduced. Secondly, program slicing techniques are used to identify test cases to be applied to modification. Finally, a complete regression approach to OO program is presented.

Key words: object oriented; incremental hierarchy model; program slicing; data flow; regression testing

* Received November 15, 1999; accepted January 13, 2000

Supported by the Key Sci-Tech Project of the National 'Ninth Five-Year-Plan' of China under Grant No. 98-780-01-02