Class-Integration Testing Sequence Research Based on Dynamic Dependency *

CHEN Jianxun *, *XIAO Yiran*

(*College of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan* 430065, *China*)

Abstract: The cost of the class-integration-test depends largely on the testing sequence. Therefore, an approach based on dynamic dependency relation for class-integration-test order is proposed in order to obtain a suitable test sequence. Firstly, the class dependencies among those object relational graphs are analysed. Secondly, the loop is removed by applying the edge deletion rules. Lastly, the test order is achieved based on the topological sequence of a directed acycline graph. The simulation results show that 42% test stubs were reduced by applying the proposed method comparing to the Briand's method. It comes to a conclusion that this method meets the requirement of reducing the test stubs to the minimum. In addition, it improves test efficiency as well as reduces the test cost.

Key words: object relational graph; dynamic dependency; test stub; test sequence; directed acycline graph

EEACC: 7210A **doi**: 10.3969/j.issn.1004-1699.2014.01.012

基于动态依赖的类间测试顺序研究 *

陈建勋*, 肖亦然

(武汉科技大学计算机科学与技术学院, 武汉 430065)

摘 要: 类间集成测试顺序决定着测试成本的大小, 为了得到合适的测试顺序, 提出了一种基于动态依赖的类间测试顺序的方法。首先分析对象关系图中类间依赖关系, 然后运用边删除规则去除环路, 最后运用有向无环图的拓扑序列给出类的测试顺序。仿真结果表明, 本文的方法较 Briand 的方法减少了 42% 的测试桩。此方法满足最小化测试桩的需要, 提高了测试效率, 减少了测试成本。

关键词: 对象关系图; 动态依赖; 测试桩; 测试顺序; 有向无环图

中图分类号: TP311.5 **文献标识码**: A **文章编号**: 1004-1699(2014)01-0064-06

软件的集成测试在面向对象软件系统中是一个非常关键的过程, 与传统软件系统不同的是其对功能模块的测试由于对象的封装、继承和多态等特性, 变得十分复杂。在面向对象的程序中, 类间的联系通过消息传递, 一条消息引起连锁反应形成一条方法调用链, 称为依赖关系[1]。由于面向对象的程序设计的特性, 使得多个类构成的类簇中的依赖关系形成网状结构图, 因此从哪里开始测试以及如何安排类间测试顺序成为关键问题之一。测试桩数量是衡量测试代价的主要方法, 因此, 改进类间测试顺序以减少测试桩的开发, 对降低测试成本, 缩短测试周期, 提高测试效率是一个很有效的途径。

对于不存在环路的对象关系图 ORD(Object Related Diagram)[2], 类间测试顺序可以通过简单的逆向拓扑序列来解决; 对于存在环路的 ORD, 则需要删除某些依赖关系, 以打破其中的环路, 然后给出类间测试序列。因此, 确定类间测试顺序的核心问题就是打破环路。学者 Kung[2] 的方法是删除一条或多条关联边以断开环路, 没有考虑类间的复杂继承关系以及动态依赖关系。学者 Le Traon[3] 在 Tarjan[4] 算法基础上引入了强连通图, 但没有区分 3 种不同依赖类型, 影响了测试桩开发的复杂度。学者 Briand[5-7] 在 Tai[8] 和 Le Traon 算法的基础上使用权重计算的方法, 来确定移除哪些依赖关系。该方法即避免了因为移除继承、聚合关系引起的开发复杂测试桩的问题, 也避免了 Tai 等人的方法在某些场景下将产生多余测试桩的缺陷[9]。

在经过对多种方法的比较分析后, 本文在改进参考文献[10]的算法基础上结合了有向无环图算法分配测试顺序。该类方法使用有向图来表示系统中类的依赖关系, 并通过分析有向图的结构, 在保证测试桩的数目尽可能少的前提下, 利用边删除规则

项目来源: 国家自然科学基金项目(61100055, 61033003, 60974112, 91130034); 湖北省自然科学基金项目(2011CDB233)

收稿日期: 2013-10-21 **修改日期**: 2013-12-26

去除环路,在此基础上运用有向无环图的拓扑序列找到一个合适的测试顺序。

1　相关概念

1.1　对象的依赖关系

面向对象程序类间的依赖关系主要包括两类:一类是静态依赖关系,另一类是动态依赖关系。

1.1.1　静态依赖关系

静态依赖关系指的是整个程序代码静态结构中反映出来的类与类之间的关系。面向对象程序中,类间的静态关系主要有继承关系、聚合关系和关联关系。

(1)如果类 A 是类 B 的子类,则类 A、B 为继承关系,A 依赖于 B。

(2)如果类 A 的数据成员具有一个或多个类 B 的实例,则类 A、B 为聚合关系,称 A 依赖于 B。

(3)如果类 A 的成员方法使用了类 B 的实例,则类 A、B 为关联关系,称 A 依赖于 B。

在集成测试时若类 A 依赖于类 B,则先测试 B 再测试 A。

类簇以及它们之间的依赖关系可以抽象为对象关系图(ORD)。ORD 中每个节点代表着程序中的一个类,每条边代表类与类之间继承、聚合和关联关系中的一种,分别用 I,Ag,As 表示。

1.1.2　动态依赖关系

动态依赖关系是指类在程序运行时期形成的一种依赖关系。若类 A 是类 B 的子类,且重写了类 B 的虚方法,类 B 是类 C 的服务类,且调用了类 B 中被类 A 重写的虚方法,则在程序运行时,C 和 A 动态绑定,类 C 动态依赖于类 A[9]。本文是在 ORD 的基础上进行类间分析的,因此我们将可能存在的动态依赖关系都标记在 ORD 中。图 1 是扩展后的对象关系图 EORD(Extended Object Relation Graph)其中动态依赖用虚线有向边表示。

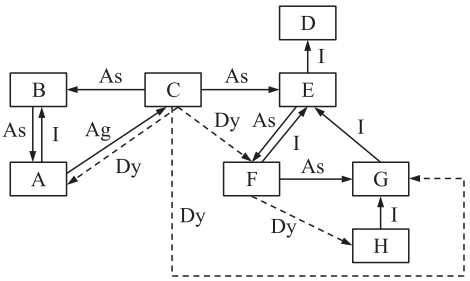

图 1　扩展后的对象关系图(EORD)

1.2　测试桩

定义:如果类 A 的一个组件使用一个或多个类 B 的服务组件,称为 A 依赖 B,在集成测试过程中,当 A 集成时,若 B 尚未被集成,我们不得不模拟 B 的服务组件,这个模拟组件通常被称为一个测试桩[10]。

在集成测试过程中,当需要对类 A 进行测试时,类 A 所依赖的另一个类 B 并没有经过测试,如果很难在短时间内构建类 B,则必定会影响到对类 A 的集成测试。此时需要构建模拟的对象来代替类 B。测试桩并不是真正的对象,但是能够为待测对象提供感兴趣的数据或状态,这样,待测对象便能够顺利使用依赖对象,或者模拟事件。故而集成测试中测试桩数目的多少决定了测试的成本。

2　改进的类间测试顺序算法

参考文献[10]中对算法进行了简单的描述,但是在一个强连通分量(SCC)中,当 Dy 和 As 边涉及环路数目相同时,没有明确的算法说明删除哪些边,并且在一次判断结束删除相应边以后,SCC 中有可能仍然存在环路,文献中没有相应的判断。根据这些不足点,再结合有向无环图计算的思想,提出本文的改进算法。

在依赖关系中,继承关系和聚合关系为强联系关系,动态依赖关系和关联关系均为弱联系关系[2]。为了避免删除强联系关系而导致 EORD 中依赖关系的不完整,故而只需要在弱联系关系中删除某些边去除环路。

为了减少测试代价,首先需要识别出 EORD 中由类以及它们之间的依赖关系形成的 SCC,然后查找每一个子强连通分量中所有的环路,统计强连通分量中每条弱关联关系所涉及的环路数目,删除涉及环路数目最多的依赖边,进而将一个有环图去除环路成为一个有向无环图。

2.1　EORD 中改进的环路消除算法

对于存在环路的 EORD,删除哪些边消除环路将直接影响到构造测试桩的数量。考虑动态依赖边对打破环路的影响,同时为了满足构造的测试桩最少,我们应该遵循删除最少的边打破尽量多的环路的原则,下面给出相关的删除规则。

规则:B 是 A 的父类,且是 C 的服务类。如果 C 在 A 和 B 之前进行测试,若 B 是非抽象类,则不需要为 A 构造测试桩,只需为 B 构造测试桩[11-13]。

在去除环路过程中,当 Dy 和 As 边涉及环路数目相同时,首先要判断该 SCC 中是否存在两个类有同向边,若同向边为 As 和 Dy,则删除这两条边;若同向边为 Ag、I 和 Dy,则删除 Dy 边。

根据上文提出的边删除规则以及算法的改进,下面给出相应的环路消除算法,算法流程图如图 2 所示。

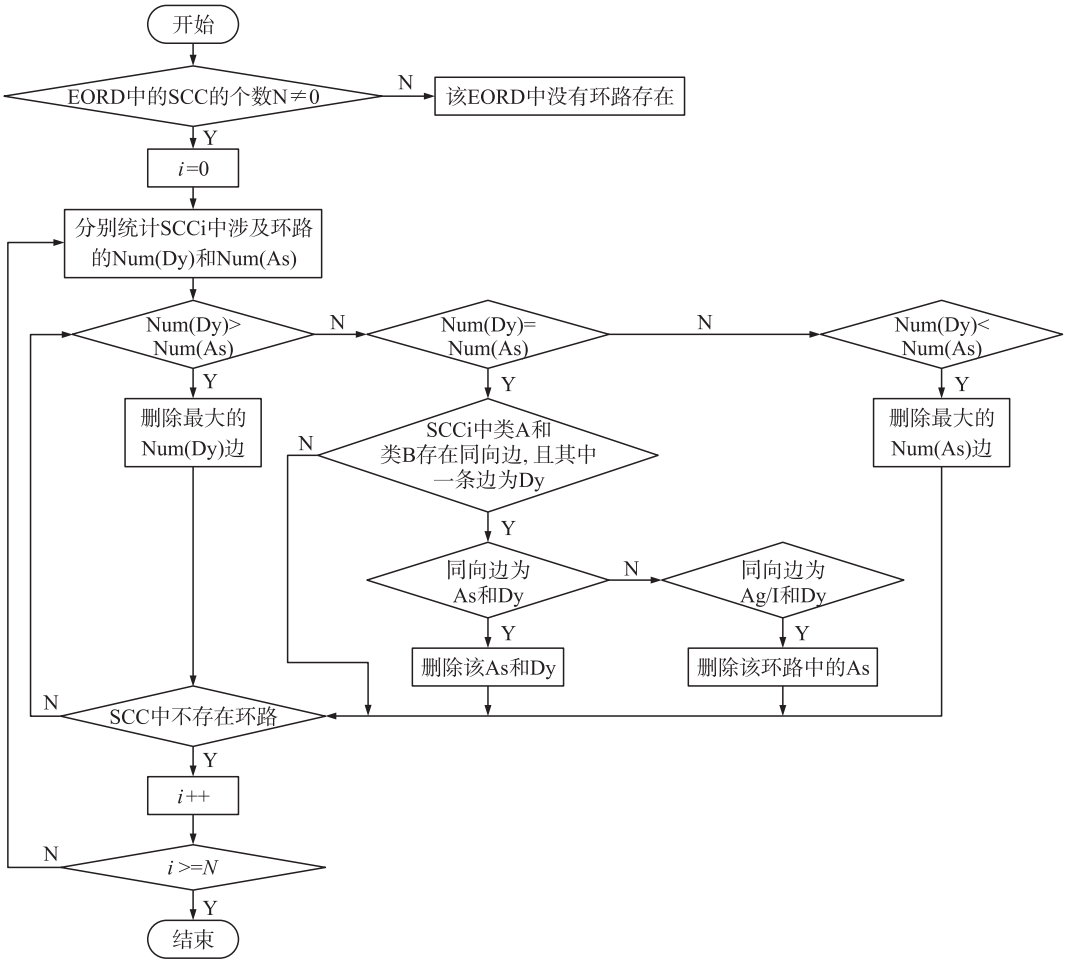

图 2 算法流程图

参考文献[10]算法复杂度为 $O(n^2)$,而本文改进的算法复杂度为 $O(n)$,较之前的算法较快速的找到需要删除的边。

下面把图 1 中所示用例应用到该算法中,对算法的具体步骤说明如下:

表 1 SCC{E,F,G,H} 中的环路

环路编号	环路
1	E→F→E
2	E→F→G→E
3	E→F→H→G→E

表 2 SCC{E,F,G,H} 弱关联关系中各关联边涉及的环路

边	边的类型	包含该边环路的集合	边涉及的环路数目
E→F	As	{1,2,3}	3
F→G	As	{2,3}	2
F→H	Dy	{3}	1

表 3 SCC{A,B,C} 中的环路

环路编号	环路
1	B→A→B
2	C→A→C
3	B→A→C→B

表 4 SCC{A,B,C} 弱关联关系中各关联边涉及的环路

边	边的类型	包含该边环路的集合	边涉及的环路数目
B→A	As	{1,3}	2
C→A	Dy	{2}	1
C→B	As	{3}	1

根据本节的算法,计算 SCC{E,F,G,H} 中各条关联边和动态依赖边涉及的环路数目,结果如表 2 所示。由算法得出删除 E→F 即可打破所有的环路。对于 SCC{A,B,C},根据算法,需要删除边 B→A 和边 C→A 打破环路。此时,EORD 成为了无环图,如图 3 所示。

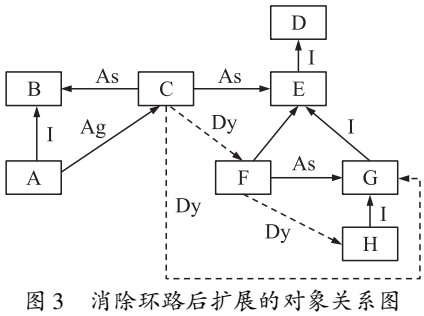

图 3 消除环路后扩展的对象关系图

打破 EORD 中所有环路需要删除 E→F、B→A 和 C→A 这三条边,分别为这三条边的源类 A,F 各自创建 1 个测试桩,共需要 2 个测试桩。因此图 1 所示的实例需要构建 2 个测试桩。

2.2　测试顺序分配

在程序的执行过程中,消除 EORD 中的环路以后,程序中仍存在动态依赖关系,由于动态依赖关系在程序运行时期才会存在,在测试一个类之前,该类所依赖的所有类都已经测试,而且在对一个类进行动态测试之前,所有的静态测试都已经测试完成。

定义:测试级 C=(C. goal,C. all,C. type)[14],其中 C. goal 为被测试类;C. all 为被测试类所依赖的类构成的并集;C. type 为测试的类型,静态测试用 S 表示,动态测试用 Dy 表示。对于 EORD 中的每一个类 X,为每个类定义一个静态测试级 $C=(\{X\},S(X),S)$;对于满足 $D(X)\neq\Phi$ 的类 X,定义一个动态测试级 $C=(\{X\},D(X),Dy)$。

以图 3 所示 EORD 为例,首先为每个类各自定义一个静态测试级,其中类 C 和类 F 满足 $D(X)\neq\Phi$,那么为 C 和 F 定义动态测试级。表 5 所示为图 6 中无环 EORD 的所有测试级。

表 5　图 3 中 EORD 的测试级

C. goal	C. all	C. type
{A}	{A,B,C,D,E}	S
{B}	{B}	S
{C}	{B,C,D,E}	S
{D}	{D}	S
{E}	{D,E}	S
{F}	{D,E,F,G}	S
{G}	{D,E,G}	S
{H}	{D,E,H,G}	S
{C}	{B,C,D,E,F,G}	Dy
{F}	{D,F,G,H}	Dy

这里先不考虑动态依赖边,所有的静态依赖边构成了一个无环的有向图[15]。对于有向无环图要找到其拓扑序列的步骤:(1)在有向图中选一个没有前驱的顶点并且输出;(2)从图中删除该顶点的所有以它作为尾的边。重复上述两步,直到全部顶点均已输出,或者当前图中不存在无前驱的顶点为止。然后再考虑动态依赖边,利用表 5 中动态依赖边的测试级,分配动态依赖的测试顺序。由于依赖关系的定义,若 A 依赖于 B,则先测试 B 再测试 A。故所得到的拓扑序列逆序即为测试顺序。

由上面所述的算法,得到图 3 的静态依赖测试级的拓扑序列为:(H,F,G,A,C,E,D,B)考虑动态依赖边后所得到的测试级测试顺序如图 4 所示。

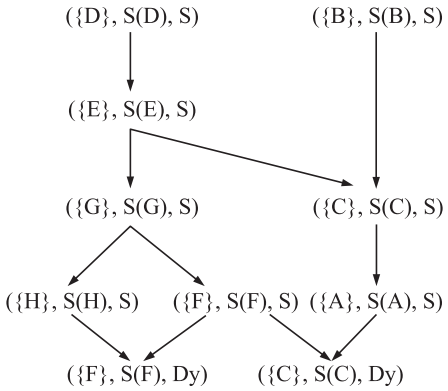

图 4　测试级测试顺序

3　实验仿真及结果分析

根据上述类测试顺序的算法设计并实现了一个工具 TLOG[16],该工具的输入信息是一个描述面向对象系统中类的关系的三元组列表。该列表可以手工输入也可以根据面向对象系统的统一建模语言设计文档中的 UML 类图获取。TLOG 主要有几个功能:(1)环路生成模块;(2)环路消除模块;(3)测试级排序模块。以 SD 空运物流进出口业务处理系统为实例验证本文方法的有效性。SD 系统中包含 10 个模块,详细信息如表 6。

SD 系统包含 1126 个环路(不考虑动态依赖),由于篇幅有限,只简单给出采用本方法打破静态依赖关系构成环路的过程,如表 7 所示。打破环路共删除 95 条边,实际需要构建 81 个测试桩。考虑动态依赖关系后,增加了 39 个动态依赖关系,环路数增加至 2283 个,表 8 给出了 SCC 中环路的打破过程。打破 EORD 中环路共删除 124 条边,实际需要构建 95 个测试桩。

本文就打破环路所需构造测试桩的数目,分别与文献[2]中 Kung 只考虑静态依赖关系的测试方法和文献[5-7]中引入 SCC 概念但没有用有向无环图概念的 Briand 方法进行比较,结果如图 5 所示。

实验结果证明:考虑类间的动态依赖关系后,实例中环路数目明显增多,Kung 方法由于没有考虑动态依赖,没有去除 EORD 中所有的环路,所需测试桩最少,但是测试不完整。本文方法虽然与 Briand 方法打破的环路数相同,但是本文方法所需测试桩少,且发现的接口错误数多。由此,本文改进的算法满足最小化测试桩的需求,并且打破环路多,发现错误多,提高了测试效率,减少了测试成本。

表 6 SD 系统的详细信息

系统名	类的数目	SCC 数目	Ag 边数目	As 边数目	I 边数目	Dy 边数目	静态边构成环路数	所有边构成的环路数
SD	44	25	106	150	62	39	1 126	2 283

表 7 打破静态依赖关系构成的环路过程

次序	处理的 SCC	SCC 中环路数目	删除的边
1	SCC{2,4,35,30,45,5,6,7,18,8,10,13,21,26,15,9,11,12,14,16,17,19,22,25,27,28,29,32,33,34,37,38,39,40,20,31,36,41,43,44}	1 426	43→30 37→26 6→10 32→27 43→44
2	SCC{2,4,35,45,5,6,7,18,8,10,13,21,26,15,9,11,12,14,16,17,19,22,25,27,28,29,32,33,34,37,38,39,40,20,31,36,41,43,44}	658	43→25 27→22 13→15
3	SCC{1,2,4,45,5,6,7,18,9,8,10,13,21,26,15,9,11,12,14,16,17,19,22,25,27,28,29,32,29,32,33,34,37,38,39,40,20,31,36,41,43,44}	284	1→18 11→15 19→34
…	…	…	…
25	SCC{2,44}	1	2→44
26	处理结束	0	

表 8 增加动态依赖关系后打破环路过程

次序	处理的 SCC	SCC 中环路数目	删除的边
1	SCC{2,3,4,35,30,45,5,6,7,9,18,8,10,13,21,26,15,9,11,12,14,16,17,19,20,22,25,27,28,29,32,33,34,37,38,39,40,20,31,36,41,43,44}	2 983	43→28 43→30 37→26 6→10 32→27 43→44
2	SCC{2,3,4,35,30,45,5,6,7,9,18,10,13,21,15,9,11,12,14,16,17,19,20,22,25,27,28,29,32,33,34,37,38,39,40,20,31,36,41,43,4}	982	26→3 43→25 27→22 13→15
3	SCC{1,2,3,4,35,30,45,5,7,9,18,8,10,13,21,15,9,11,12,14,16,17,19,20,22,25,27,28,29,32,34,37,38,39,40,20,31,36,41,43,44}	482	43→34 1→18 11→15 19→34
…	…	…	…
42	SCC{2,44}	1	2→44
43	处理结束	0	

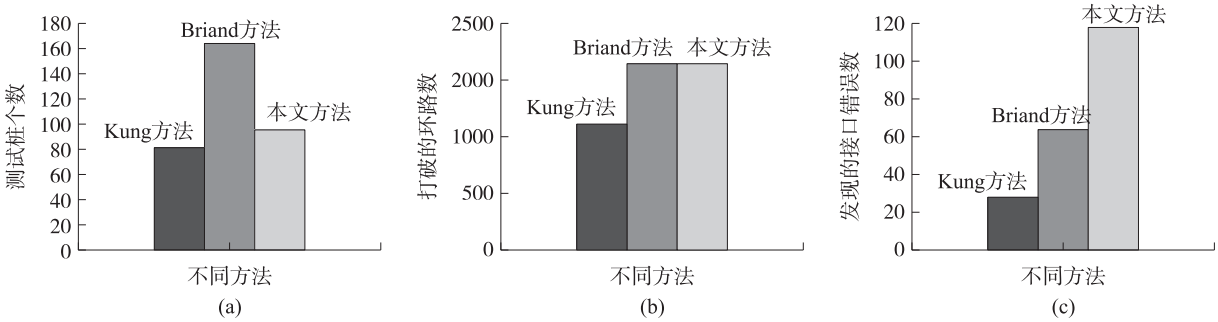

图 5 3 种方法的比较

4　结束语

在类间依赖关系构成环路的情况下,需要删除某些依赖关系以消除环路,同时建立测试桩。文中的算法首先分析 ORD 中类间的依赖关系,设定了边的删除规则去除环路,在此基础上运用有向无环图拓扑序列给出类的测试顺序。最后运用测试工具 TLOG 验证该方法较其他方法需要较少的测试桩,测试效率有明显的提高。本文的算法与 Kung 和 Briand 的算法相比考虑了动态依赖,并且使用有向无环图拓扑序列确定测试顺序,性能较优,只需要构造较少的测试桩,有效降低了测试成本。但是本文中也存在着不足之处:没有考虑抽象类的特点,实际上,抽象类会影响类间的依赖性,进而将影响类间测试顺序,所以抽象类的研究将是以后工作的重点。

参考文献:

[1] 王正山. 基于 ORG 的 OO 软件测试技术研究[D]. 合肥:合肥工业大学,2005.

[2] Kung D C, Gao J, Hsia P, et al. Class Firewall, Test Order, and Regression Testing of Object-Oriented Programs[J]. JOOP, 1995, 8(2):51-65.

[3] Le T Y, Jeron T, Jezequel J M, et al. Efficient Object-Oriented Integration and Regression Testing[J]. IEEE Transactions on Reliability, 2000, 49(1):12-25.

[4] Tarjan R. Depth-First Search and Linear Graph Algorithms[J]. SIAM Journal on Computing, 1972, 1(2):146-160.

[5] Briand L C, Labiche Y, Wang Y. Revisiting Strategies for Ordering Class Integration Testing in the Presence of Dependency Cycles [C]//Software Reliability Engineering, 2001. ISSRE 2001. Proceedings. 12th International Symposium on. IEEE, 2001:287-296.

[6] Briand L C, Feng J, Labiche Y. Using Genetic Algorithms and Coupling Measures to Devise Optimal Integration Test Orders[C]// Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering. ACM, 2002:43-50.

[7] Briand L C, Labiche Y, Wang Y. An Investigation of Graph-Based Class Integration Test Order Strategies[J]. IEEE Transactions on Software Engineering, 2003, 29(7):594-607.

[8] Tai K C, Daniels F J. Interclass Test Order for Object-Oriented Software[J]. Journal of Object-Oriented Programming, 1999, 12(4):18-25.

[9] 李都. 测试顺序选择策略研究[J]. 计算机工程与设计,2008,29(4):781-783.

[10] 张艳梅,姜淑娟,张红昌. 一种基于动态依赖关系的类集成测试方法[J]. 计算机学报,2011,34(6):1075-1089.

[11] 李小将,李佑禄,陈启安. 基于类的动态依赖关系的集成测试顺序分配策略[J]. 装备指挥技术学院学报,2005,16(1):93-97.

[12] Wang Z, Li B, Wang L, et al. Using Coupling Measure Technique and Random Iterative Algorithm for Inter-Class Integration Test Order Problem [C]//Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual. IEEE, 2010:329-334.

[13] Jiang S, Zhang Y, Yi D. Test Data Generation Approach for Basis Path Coverage[J]. ACM SIGSOFT Software Engineering Notes, 2012, 37(3):1-7.

[14] Labiche Y, Thevenod-Fosse P, Waeselynck H, et al. Testing Levels for Object-Oriented Software[C]//Proceedings of the 22nd International Conference on Software Engineering. ACM, 2000:136-145.

[15] 高剑,罗志增. 支持向量机在肌电信号模式识别中的应用[J]. 传感技术学报,2007,20(2):366-369.

[16] 关乐,褚金奎,王晓东,等. 系统级设计方法及其在力学特性集成测试中的应用[J]. 传感技术学报,2006,19(5):1313-1318.

陈建勋(1957-),男,博士,教授,CCF 高级会员,研究领域为软件工程、计算机图形学和 CAD 技术、基于计算机网络的应用技术,jxwh@ wust. edu. cn;

肖亦然(1988-),女,硕士研究生,研究方向为现代软件工程技术。