面向对象 程序回归测试 数据流测试 (15)

计算机科学2000Vol. 27№. 8

43-46

面向对象程序回归测试策略研究

An Regression Testing Strategy for Testing Object-Oriented Programs

顾玉良 周淑秋

(首都师范大学计算机系 北京100037)

TP311.52

Abstract This paper puts forward an regression testing strategy for testing object-oriented programs based on data-flows. The basic idea is that by focusing on data aspect of the object system, relative data flows are combined to generate test path along which computation effect may be propagated to some suitable checkpoints. Errors are more likely detected based on such stuategy. Problem of limited controllability and observability in the tested system is more easily resolved. At the end, the basic method of regression test set generation is formed.

Keywords Object-oriented program, Regression testing, Data flow

1 引言

软件测试的基本过程是从单元测试、集成测试、有效性测试到系统测试。一般地说,回归测试属于软件维护的过程,当软件修改之后,回归测试用于确认修改的正确性,包括修改本身的正确性和未修改的部分未受到因修改而导致的不正确影响。关于回归测试的策略研究通常独立于常规测试,但在实际的过程中,在软件提交前的测试阶段,在不同层次的测试中发现错误,并提交修改之后,需要进行回归测试。

假设程序(被测对象)P,采用一组测试用例 T,其中包括初始的历史信息,执行 T 后发现错误,修改 P 成为 P'。回归测试的基本步骤包括:

1)根据前面测试产生的信息,确定错误的位置、错误的类型、错误的严重性级别等;

2)根据测试的总体目标、测试计划和错误信息,确定错误的影响范围,确定回归测试的策略;

3)对于 P',确定回归测试用例集 T',包括从 T 中选取的子集(不变的用例和修改的用例),以及因为设计规范的修改而需要增加的用例;

4)在 P'上执行 T';

5)确定对于 P'合适的用例集和测试历史信息。其中由于代码级和设计规范级的修改,需要从 T 中删除部分过时用例,合并 T、T',生成新的 T,用于以后的回归测试。

2 计算效果传播及检查

数据流测试策略基于程序变量的数据流分析而指导测试路径的选择。在文[1,2]中,作者研究了各种数据流覆盖及语句、分支覆盖的关系,表明数据流测试策略既考虑数据运算方面,又考虑程序的控制流程方面。但是,传统的数据流策略只基于数据定义和使用的覆盖,缺少对于错误发现能力的针对性和对于被测系统可控制性和可观察性的考虑。

考虑对象方法中如下程序片断,其中 z 是对象实例变量或共享变量:

(1)x=5;…y=x;…if(y>3)then…

(2)x=5;…y=x;…z=y;…

(3)x=5;…y=x;…O.output(y);

这里,假设每个对变量的定义-使用之间存在关于该变量的定义纯净的路径,(1)(2)(3)都包括传统数据流中两个相关的定义-使用对,在(1)中,对 x 的定义传播到对 y 的判定使用;在(2)中,对 x 的定义传播到对 z 的定义;在(3)中,对 x 的定义传播到对象 O,并且继续传播到系统的某个级别的输出。由此可见,一个变量的定义,通过辗转的使用和定义,可以影响到另一个变量的值,或者影响到路径的选择等,这种情形称为计算效果的传播。

Jeng[3]、Harrold[5]等研究了基于程序结构的错误分类。基于代码的错误包括计算错误和域错误两大类。域错误是由于路径对于输入域的划分不正确,可以是

顾玉良 博士,主要研究为软件工程。周淑秋 博士,主要研究领域为系统仿真。

在序列中多余或缺少某些子路径，也可以是因为不正确的判定引起，包括在条件表达式中的书写错误或程序运行时某些不正确的变量值导致条件表达式结果的错误，从而引起不正确的路径选择。计算错误导致程序（对象）状态和输出的错误，有时它可能导致域错误。

一般地，在所有可能的方法路径的序列中，当穿越错误代码位置时，存在特定的路径，使不正确的变量定义或使用沿该序列尽可能地传播到一个适宜于检查的位置，可以使发现错误的概率最大，从而尽量降低检查机制的漏检可能性。即在特定的控制和检查机制下，考虑基于相关数据流的组合问题，可以使测试路径对于错误是敏感的。

数据流策略对于发现代码错误具有较强的针对性。但由于测试过程中系统的可控制性和可观察性的限制[5]，测试可能难以组织。计算效果传播以数据为中心，为程序测试的路径选择和结果验证提供了一种基本的思路：以变量值的改变为中心，对可能的对象状态和全局状态的建立或修改，根据计算传播特征，组织相关的数据流，在合适的时机或位置确认这种修改对于系统的影响，从而可以较好地克服可控制性和可观察性的问题。结果检查主要包括以下几个方面：对象和系统的状态，控制流程的位置和在对象或系统的某个层次上的输出。

3 数据流变化的分类

在图1中，对于待测程序 P，其正确版本（一般情形下是未知的）包含正确的数据流集合 C，P 实际包含的数据流集合为 R。在对 P 进行测试后，修改 P 成为 P'，P' 包含的数据流集合 R' 由以下几部分组成：

·正确部分包括 R 中正确部分和因改正错误而增加的正确部分；

·错误部分包括 R 中未被更正的错误和由于修改而增加的错误 NE。

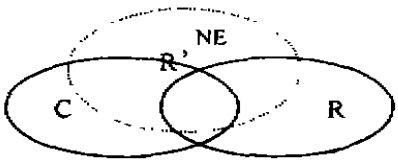


图1 正确数据流，未修改的
数据流和修改的数据流集合

数据流测试的基本目标是，通过对 R 的测试，尽量发现程序 P 中的错误。修改的版本 P' 包含 R'，我们希望 R' 与 C 的差异尽可能小，但 C 本身在一般情况下是不可预知的，只能通过 P' 中发现的错误尽可能少或在允许的范围之内，来推知这种结果。

基于数据流集合 C，考察数据流集合 R 和 R'，数据流发生的变化可以分如下几类：

1）数据流不变　某些代码编写错误可能没有直接改变数据流信息。如把 x=y+5；错写成 x=y*5；或 x=y+3；说明在图1所示的具有 C 集的程序仍然可能包含错误，这种错误可以通过设计测试数据和变量定义的计算效果传播发现。

2）数据流丢失　某些代码编写错误可能导致数据流信息的丢失。如 x=y+5；语句丢失，则关于 x 赋值导致的和 y 使用对应的计算效果传播序列，没有在 T 中直接考虑；

在测试之前，我们无法预知哪些语句丢失。如果某个计算效果传播的序列穿过语句的丢失点，并且该序列考虑了丢失语句中的变量的数据流组合，则有可能发现该错误。

3）数据流多余　某些代码编写错误可能导致多余的数据流信息。如果语句 x=y+5；是多余的，则在 T 中包含关于 x 赋值导致的和 y 使用对应的计算效果传播。对这种情形通常具有较好的错误发现能力。

4）数据流改变　如果把一个变量标识错写成另一个变量标识，则相应的数据流被改变。这种情形可以看成某些数据流信息多余和某些数据流信息丢失。如把 x=y+5；错写成 z=r+5；则 T 中包含关于 z 赋值导致的和 r 使用对应的计算效果传播，而关于 x 赋值导致的和 y 使用对应的计算效果传播，在 T 中没有直接考虑。

因而，可以通过实际存在的计算效果传播信息检查相应的错误。一般地说，R' 与 C 差异比 R 与 C 的差异小。对于基于数据流的回归测试策略，需要对（R'-R）部分进行测试。（R'-R）本身跨越修改部分、增加部分和不变部分的代码边界，而一个代码错误可能由若干个不同的数据流组合测试发现，因而，在回归测试中，一般地不需要对（R-R'）中的数据流进行重复的测试。

4 数据流组合策略

4.1 方法路径的分类

用 S 表示一个语句，DEF(S) 表示在 S 中赋值变量集合，DEF(c,S) 表示在非条件语句 S 中引用变量 c 的赋值变量集合，C_USE(S) 表示在非条件语句 S 中引用变量集合，P_USE(S) 表示在条件 S 中的引用变量集合。一条方法路径可以用从方法入口到方法出口的语句顺序 $S_1S_2\cdots S_n$ 表示。

对于一个实例变量或全局变量 x=DEF(S_i)，如果该定义中只引用常量，且 $S_i\cdots S_n$ 中不存在关于变量 x 的另一次定义，则该定义建立对象状态，并且其效果至

少持续到对象响应下一个消息，称该路径是对象（或系统）关于 x 状态创建的。

对于一个变量 a，一个实例变量或全局变量 x，如果 $x \in DEF(a,S_i)$，且 $S_i \cdots S_n$ 中不存在关于变量 x 的另一次定义，则该定义改变对象状态，并且其效果至少持续到对象响应下一个消息，称该路径是对象（或系统）关于 x 状态修改的。

对于一个实例变量或全局变量 x，如果 $x \in P\_USE(S_i)$ 或 $x \in C\_USE(S_i)$，且 $S_i \cdots S_l$ 中该使用之前不存在对 x 的定义，则它引用了对象本次响应消息前的状态（x 分量），称该路径是对象（或系统）关于 x 状态传播的，包括几种情形：

·情形1　通过消息中的使用传播至其它对象，从而影响该对象的状态或由该对象继续传播；

·情形2　传播至某个层次上用于结果检查，如检查用户界面上的输出；

·情形3　传播至该方法路径中的某个判定，从而影响判定结果，以决定路径选择；

·情形4　传播至某个实例变量或全局变量的定义，从而影响对象或系统的状态，创建或修改文件、数据库的数据等。

显然，一条方法路径可以同时具有以上三种特征。

4.2　数据流组合策略

为了在对象方法内部表示传播路径，采用 $DPU^+(x,S)$、$DCU^+(x,S)$ 和 $DCD^+(x,S)$ 分别表示 S 中定义变量 x，其计算效果传播到的条件语句集合、计算语句集合和定义语句集合。

任何对象类的测试包括从对象实例化开始到消亡的过程。对象测试序列是在测试过程中对应于从对象创建、活动至消亡的执行方法的路径序列，用 $mr_1,\cdots,mr_i,mr_{i+1},\cdots,mr_n$ 的形式表示。对象集测试序列是在测试过程中对应于从一组对象创建、活动至所有这些对象消亡的执行方法的路径序列。假设 $O_1,O_i,O_{i+1},\cdots,O_n$ 是若干对象，可以用 $O_1 \cdot mr_1,\cdots,O_i \cdot mr_i,O_{i+1} \cdot mr_{i+1},\cdots,O_n \cdot mr_n$ 的形式表示对象集测试序列。如果在对象集测试序列中独立地观察每个对象的活动，可以观察每个对象在其生存周期中所关心的数据流，在各自的响应序列上测试计算效果传播。于是，测试用例集可以按如下方法对相关的数据流进行组合：

1）对象内部的每个方法的数据流组合　对于每个方法，选择所有这样的路径，对于方法中涉及的每一个变量 $x \in DEF(S_i)$ 和任意一个 $S_i \in DPU^+(x,S_i)$ 或 $V_i \in DCU^+(x,S_i)$ 或 $S_i \in DCD^+(x,S_i)$，覆盖从 S_i 到 S_l 的计算传播路径。

2）对象内部方法之间的数据流组合　如果存在这样的对象测试序列，$mr_1,\cdots,mr_i,mr_{i+1},\cdots,mr_n$，如果满足下列任一个条件，则该对象测试序列对相关数据流组合：

·mr_i 是关于变量 x 的状态创建或修改路径，mr_{i+1}，是关于变量 x 的状态传播型路径（情形1、2、3、4）；

·mr_i 是关于变量 x 状态传播型路径（情形1、2、3），mr_{i+1} 是关于变量 x 的状态传播型路径；

·mr_i 是关于变量 x 状态传播型路径（情形4），且传播至对实例变量或全局变量 y 的定义，而 mr_{i+1} 关于变量 y 的状态传播型路径。

3）对象之间的数据流组合　$O_1,\cdots O_i,O_{i+1},\cdots,O_n$ 是若干对象，如果存在这样的对象集测试序列，$O_1 \cdot mr_1,\cdots,O_i \cdot mr_i,O_{i+1} \cdot mr_{i+1},\cdots,O_n \cdot mr_n$，如果满足下列任一个条件，则该对象集测试序列对相关数据流组合：

·$O_i \cdot mr_i$ 是关于全局变量 x 的状态创建或修改型路径，$O_{i+1} \cdot mr_{i+1}$ 是关于 x 的状态传播型路径（情形1、2、3、4）；

·$O_i \cdot mr_i$ 是关于全局变量 x 的状态传播型路径（情形1、2、3），$O_{i+1} \cdot mr_{i+1}$ 是关于 x 的状态传播型路径；

·$O_i \cdot mr_i$ 是关于全局变量 x 状态传播型路径（情形4），且传播至对全局变量 y 的定义，而 $O_{i+1} \cdot mr_{i+1}$ 关于变量 y 的状态传播型路径。

4.3　回归测试策略

以上的数据流组合策略对于测试和回归测试是相同的，从而利于测试生成和测试管理。文[8,9]讨论了在单元、集成和功能级的回归测试的一般问题。每次修改 P 成为 P′之后，需要对 P′进行回归测试。根据测试计划和实际进度，回归测试可以从修改单元的单元测试开始，也可以从包含该单元的某一级集成测试开始。设计回归测试用例集时需要考虑以下三类错误：

·以前的测试遗漏的错误；

·在修改过程中引入的新的错误；

·由于增加或删除功能而引入的新的错误。

在文[4,7]中，作者研究了选择回归测试的技术。本文则采用如下步骤，生成回归测试用例集 T′：

1）对于所用的修改，确定修改产生的影响范围，以及需要重新测试的部分。

对于程序修改的影响范围可以确定为所有穿过修改点的路径集 AIP（All Influenced Paths），在其中选择一个子集作为回归测试用例集 T′，程序修改可以从两个级别来看：

·语句级：更正错误的代码。错误的范围包括所有穿越更正代码部分的路径，实际需要回归测试的是一个子集；

·功能级：增加和删除功能。确定重新测试的范围，不但需要对增加的功能部分进行测试，还需要对增加

和删除部分与其余部分的交互进行测试。

2)在针对 P 的测试集 T 中，选择需要重新测试的子集作为 T'，并确定 T' 中需要修改的用例，进行修改。

在 T 中选择哪些穿越修改点的测试用例 T'，如果代码修改没有改变数据流，则选择这些用例；对于增加丢失的语句，如果增加了丢失的数据流，则确定生成用例覆盖之；对于删除语句，如果删除了多余的数据流，则确定相应的路径，从 T' 中删除；如果代码修改导致数据流改变，则作以上两项。

3)对于 T，增加测试用例。

由于代码修改引起的数据流变化，如果没有包含在 T 中，则按照数据流组合策略，从 AIP 中选择相应的用例，加入到 T' 中。

4)对于功能规范的修改，增加测试用例。

假设增加功能对应的代码部分为 AF，首先，针对 AF，按以上数据流组合策略建立测试集，加入到 T' 中，其次，需要考虑 AF 与其余部分之间的交互(路径)，加入到 T' 中。

假设删除功能对应的代码部分为 DF，需要考虑 DF 与其余部分之间的交互(路径)的修改。从 T 中选择测试用例，如果用例包含的数据流丢失，则删除该用例；否则，根据数据流测试策略，修改测试用例，加入到 T' 中。

结束语 针对面向对象软件特点，以数据为中心，计算效果传播比传统的数据流策略具有更好的测试路径选择能力和更强的代码错误发现能力，本文基于相关数据流组合测试策略，研究了面向对象程序回归测试问题。

在本文的数据流组合策略中，可以划分不同的组合的级别，但为了建立有效的分级，尚需要在功能和行为层面研究进行联合测试建模方法。

参 考 文 献

1 Ntafos S C. A Comparison of Some Structural Testing Strategies IEEE trans on Software Engineering, 1988, 14(6):868～874

2 Frankle P G, Weyuker E J. Provable Improvements on Branch Testing. IEEE trans. on Software Engineering, 1993,19(10):963～975

3 Jeng B. Integrating Data Flow and Domain Testing. In: Proc. of the 16th Intl. Conf. on Software Engineering. 1994. 154～162

4 Rothermel G, Harrold M J. Selecting Regression Tests for Object-Oriented Software. In: Proc. of the 16th Intl. Conf. on Software Engineering. 1994. 14～24

5 Harrold M J, McGregor J D. An Approach to Fault Modeling and Fault Seeding Using the Program Dependence Graph. J. System Software, 1997

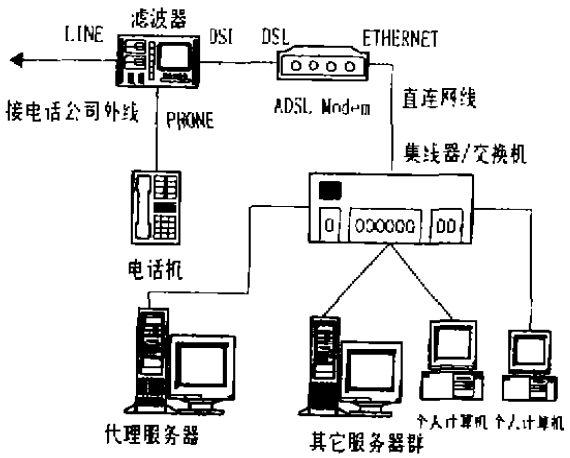
6 Binder R V. Design for Testability in Object-Oriented Systems.

7 Rothermel G, Harrold M J. A Safe, Efficient Algorithm for Regression Test Selection. In: Proc. of the 15th Intl. Conf. on Software Engineering. 1993. 358～367

8 Leung H K N, White L. Insights into regression testing. In: Proc. Conf. Software Maintenance. 1989. 60～69

9 Leung H K N, White L. A Study of regression testing and software regression at the integration level. In: Proc Conf. Software Maintenance. 1990. 290～301

(上接第17页)



(MSProxy、Wingate、Sygate 等)(WEB、NEWS、DNS、DB、EMAIL 等)

图3 学校局域网 ADSL 连接图

·提供了高速接入 Internet 网的途径，可达下行 9M/上行1.5Mbps 的传输速率。

·不需更改和添加线路，直接使用原有的双绞铜质电话线。经济高效，简单易用。

·总是处于联机状态，语音信号和数字信号可以并行传输，可同时"上网"和"打电话"。

·支持集中配置，消除延迟问题，不经过电话交换设备，上网不再需要缴纳电话费。

·充足的带宽为以后的宽带实时多媒体服务提供了传输的基础，并且每个用户都有特定的带宽，而不是和其他人共享。

参 考 文 献

1 Ford M, Lew H K. Internetworking Technologies Handbook. Cisco Press. Part: ADSL Technology, April 1998. 139～144