基于模型代数的基本路径集的自动生成

赵会群 卢 飞
(北方工业大学计算机学院 北京 100144)

摘 要 路径测试是一种根据路径设计测试用例的白盒测试技术,而基本路径测试是其中运用最广泛的一种路径测试方法。基本路径测试是在被测程序的控制流图的基础上导出基本的可执行的路径集合,因此程序控制流图是基本路径集自动生成的关键。考虑到依赖程序控制流图生成基本路径集的低效性,提出基于模型代数的基本路径集的自动生成方法。该方法通过分析被测程序,自动生成程序的模型代数表达式,并在模型代数表达式的基础上生成基本路径集。最后通过经典案例证明了该方法的有效性。

关键词 路径测试,白盒测试,基本路径集,模型代数
中图法分类号 TP312 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.04.025

Automatic Generation of Basis Path Set Based on Model Algebra

ZHAO Hui-qun LU Fei
(College of Computer, North China University of Technology, Beijing 100144, China)

Abstract Path testing is a white-box test method for designing test case. Basis path testing is one of the most widely used path test methods. The basic path test is the collection of the basic executable paths based on the control flow graph of the program, so the control flow graph is the key of automatic generation of the basic path set. In this paper, an automatic generation of basic path set method was proposed based on model algebra. The basic path set will be generated based on the model algebra expression which is automatically generated by analyzing the program. In the end, the classical cases show the efficiency of the method.

Keywords Path test, White-box test, Basis path set, Model algebra

随着计算机技术的快速发展和应用范围的不断扩大,很多行业都实现了计算机系统管理,如社交平台、新闻资讯、电子商务、办公娱乐、智能家居、智慧城市等。计算机软件作为计算机系统的灵魂,其规模和复杂程度也越来越高。为了保障计算机系统的可靠性,软件检测和维护的成本也越来越高[1]。

白盒测试,也称为结构测试或者逻辑驱动测试,是软件测试的一种重要方法,该方法的特点是定义严格,并以数学方法描述测试规约,所以其测试精确度很高[2]。为此,该方法多用于对软件可靠性有较高要求的应用领域,例如航天航空、军事国防等。其中白盒方法中的基本路径测试法是应用最广泛的测试方法之一。

基本路径测试能够经过所有的条件分支,保证了程序代码中每条可执行的语句至少执行了一次,具有较高的故障覆盖率[3]。传统的基本路径测试法,即 MaCable 法[4],首先要生成程序控制流图,通过分析程序控制流图的环形复杂度,确定程序的基本路径数量,再根据基本路径的图算法求得可执行的路径集合,从而设计测试用例。该方法自七十年代被提出以来已经被广泛采用。然而,随着被测程序的复杂性的增加,程序对应的控制流图也会相当复杂,基本路径生成算法也就相对复杂,从而测试用例的生成的代价也增长很快。

文献[5-8]针对基本路径的自动化生成均提出了相关算法,这些算法只是对 MaCable 法进行了优化和改进,并没有摆脱基本路径集的生成对程序控制流图的依赖性。文献[9]提出了自动化生成程序控制流图的算法,在一定程度上提高了基本路径集生成的自动化程度,但为此需要付出较高的代价。

文献[10]提出了一种运用遗传算法生成基本路径的方法。遗传算法是一种通过模拟自然进化过程搜索最优解的方法,近年来得到了广泛的应用,在某些领域也取得了十分丰硕的成果。其采用遗传算法进行路径生成时,首先需要确认一个表示程序路径的初始种群,然后运用遗传算子对种群进行处理,不断得到新的种群,并从中得到新的路径集合,通过适应度函数筛选求得最终所求路径。实际效果中,该算法会因为初始参数选择得不恰当(如初始的路径不可达,突变因子的初始值等)而对最终生成的程序路径集产生一定的影响。经过多次实验运行生成的基本路径集合并不理想,生成率较低。

本文提出一种基于模型代数的基本路径生成技术,该技术分析扫描待测程序,将其转换为模型代数表达式;然后对代数表达式进行分解处理,筛选出基本路径集,从而得到测试用

到稿日期:2015-11-30 返修日期:2016-02-19 本文受北方工业大学优势学科项目资助。
赵会群(1960-),男,博士,教授,主要研究方向为软件体系结构、软件测试、计算机网络、物联网、体育计算等;卢 飞(1989-),男,硕士生,主要研究方向为软件体系结构、软件测试,E-mail:820364030@qq.com。

例。本文第 1 节概述了模型代数；第 2 节提出了基于模型代数的基本路径集的生成算法；第 3 节结合软件测试中的经典案例证明了此方法的有效性；最后介绍了相关研究工作并给出了研究结论。

1　模型代数简介

模型代数是作者在文献[11]中提出的软件建模方法，该模型通过扩展进程代数，增加了适合描述软件模块之间调用关系的算子，从而对程序实现抽象和建模。利用代数方法建模程序结构，并利用数学方法化简和求解问题是一种新的尝试。为了使读者理解文中的方法和技术步骤，这里简单介绍模型代数。

结合软件系统特点，将进程代数中的算子进行扩展，把软件系统解释为组件连接运算的实现。模型代数描述了软件系统中组件与组件之间的关系。

下面定义出组件与组件之间的连接关系。

1)设 A,B 是论域 U 中的两个组件，组件 A 通过发送一个消息“激发”组件 B 来实现软件系统的功能，则称组件 A 激发组件 B。组件与组件之间的关系为顺序关系，记作 $A \oplus B$。

2)设 A,B 是论域 U 中的两个组件，组件 A 与组件 B 选择执行，则称组件 A 与组件 B 之间为选择关系，记作 $A+B$。

3)设 A,B 是论域 U 中的两个组件，组件 A 与组件 B 循环执行，则称组件 A 与组件 B 之间为循环关系，记作 $A \cdot B$。

4)设 A,B 是论域 U 中的两个组件，组件 A 通过“使用”组件 B 来实现功能需求，则称组件 A 与组件 B 进行了一次“使用”运算，记作 $A \otimes B$。

5)设 A,B 是论域 U 中的两个组件，组件 A 与组件 B 协同操作，记作 $A \Theta B$。

6)设 A,B 是论域 U 中的两个组件，组件 A 与组件 B 并行执行，记作 $A \parallel B$。

2　基本路径自动生成算法

基本路径集自动生成的流程图如图 1 所示。

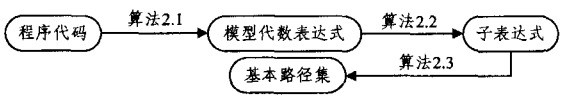

图 1　基本路径集自动生成的流程图

2.1　算法：生成模型代数表达式

输入：标记的程序伪代码。

输出：生成的代数模型表达式。

首先需要对伪代码进行标记划分。利用编译原理中划分基本块的方法构造结点。结点映射为软件系统的组件。

基本块：指程序中一个顺序执行的语句序列，其中只有一个入口语句和一个出口语句。

入口语句包括以下 3 种：1)程序的第一个语句；2)条件转移语句或无条件转移语句的转移目标语句；3)紧跟在条件转移语句后面的语句。

基本块划分步骤：1)求出四元式程序中各个基本块的入口语句；2)对每一入口语句，构造其所属的基本块，它是由该入口语句到下一入口语句(不包括下一入口语句)，或到一转移语句(包括该转移语句)，或到一停语句(包括该停语句)之间的语句序列组成的；3)凡未被纳入某一基本块的语句都是程序中控制流程无法到达的语句，因而也是不会被执行到的语句，可以把它们删除。

通过上述算法对代码进行基本块划分，接着扫描标记后的代码。

算法描述：

```
1. ReadPseudo_Code(Path); //读入基本块划分的伪代码
2. GenerateSuccessorMap();//构件结点前驱后继关系表
3. if (no descendant node)//递归出口，生成代数表达式
4.    return "n"+index;
5. else if (descendant node==1)//后继结点为一个
6.    generateSubExpression();//拼接表达式;
7. else //结点后继结点为多个
8.    for (every node in successorMap)
9.       generateSubExpression();//拼接表达式
```

2.2　算法：生成子表达式

输入：相应的模型代数表达式。

输出：生成子表达式。

读入模型代数表达式，生成代数表达式的子表达式。且子表达式与原表达式所表示的含义是相同的。

算法描述：

```
1. while (index<the size of alge_expr_list)
2.    item<-get value from alge_expr_list.
3.    switch (item) {
4.      case ")" or "}" or "]":
5.       while (if the top of basi_path_stack is not "(")
6.         if (! basi_path_stack. peek(). equals("+"))
7.           //入栈;
8.         else // 存入 temp 栈中
9.           pop the basi_path_stack; //弹出操作符
             //拼接以字符串形式存储的表达式;
10.          push(sb) to temp_stack.
11.          clear temp_list1;// 清空，用于下次}
12.        Endif
13.        push(sb) to temp_stack.
14.        pop the basi_path_stack;//弹出匹配的(符号
15.        if (! temp_stack is empty) {
16.            boolean mark=true;
17.              while(!basi_path_stack is empty&&mark==true)
18.            if (if the top of basi_path_stack is not "(")&& is not
               ("+")) {
19.              add to temp_list2;
20.            else
21.              mark=false;
22.            Endif
23.          Endif
24.        break;
25.     case "(" or "{" or "[":
26.        push(item) to basi_path_stack.
27.        break;
28.     case "+":
29.        push(item) to basi_path_stack.
30.        break;
31.     default:
```

```
32.     push(item) to basi_path_stack.
33.     break;
```

2.3 算法:基本路径集的选取

输入:生成的子表达式。

输出:基本路径集。

基本路径集中的每条路径具有下列 3 个特点:

1)每一条路径都是一条独立路径;

2)程序中所有的边都被访问;

3)程序中所有不属于该基本路径集的路径都可以由这个基本路径集中路径经过线性运算得到。

首先,利用正则表达式对子表达式进行切片,子表达式处理为结点连接关系的表达式。再通过下列算法从中进行筛选。

Joseph Poole[13] 提出了一种选取算法。

算法描述:

```
1. findBasisPathSet(node)
2.   if(node is a sink)
3.     print out this path as a solution;
4.   else if (node has not been visited)
5.     mark node as visited;
6.     label a default edge;
7.     findBasisPathSet(destination of default edge);
8.     For all other outgoing edges;
9.     findBasisPathSet(destination of edge);
10.  else
11.      findBasisPathSet(destination of default edge);
```

3 案例研究

案例:计算不超过 100 个数字的平均值,同时计算总和与有效数字的个数。

代码如下:

```
Procedure average
Type value [1:100] IS SCALAR ARRAY;
TYPE average . total. input. total. valid
        Minimum, maxImun, sum IS SCALAR
TYPE I IS INTEGER;
I=1;Sum=0;
1)  Total. input=total. valid=0;
DO WHILE value[i]<>-999 and total. input<100
4)  Increment total. input by 1;
5)  IF value[i]>=minimum AND value[i] <=maximum
7)  THEN increment total. valid by 1;
        Sum=Sum+valid[i]
ELSE skip
8)  ENDIF
Increment I by 1;
9)  ENDDO
IF total. valid>0
        11)THEN average=Sum/total. valid;
        12)ELSE average=-999;
13)  ENDIF
END average
```

代码中数字部分为经过基本块划分后的结点。图 2 为程序相对应的控制流图。

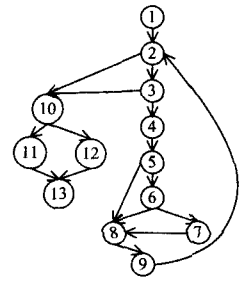

图 2 案例相对应的程序控制流图

首先,经过 2.1 节中的算法对伪代码进行处理,应用模型代数构建出其代数表达式如下(文中结点之间的关系处理为顺序运算):

n1⊕(n2⊕((n10⊕(n11⊕n13+n12⊕n13))+(n3⊕((n10⊕(n11⊕n13+n12⊕n13))+n4⊕(n5⊕((n8⊕n9⊕n2)+(n6⊕((n8⊕n9⊕n2)+((n7⊕n8⊕n9⊕n2)))))))

然后,利用 2.2 节的算法拆分复杂的模型代数表达式,即可重写为:

n1⊕n2⊕n10⊕n11⊕n13+
n1⊕n2⊕n10⊕n12⊕n13+
n1⊕n2⊕n3⊕n10⊕n11⊕n13+
n1⊕n2⊕n3⊕n10⊕n12⊕n13+
n1⊕n2⊕n3⊕n4⊕n5⊕n8⊕n9⊕n2+
n1⊕n2⊕n3⊕n4⊕n5⊕n6⊕n8⊕n9⊕n2+
n1⊕n2⊕n3⊕n4⊕n5⊕n6⊕n7⊕n8⊕n9⊕n2.

去掉括号拆分得到的子表达式实质上是程序的路径集合。用构建的模型代数表达式描述了程序的路径。

最后,通过 2.3 节算法即可从子表达式中筛选基本路径集。得到的基本路径集合如下所示。

path 1:1-2-10-11-13

path 2:1-2-10-12-13

path 3:1-2-3-10-11-13

path 4:1-2-3-4-5-8-9-2-…

path 5:1-2-3-4-5-6-8-9-2-…

path 6:1-2-3-4-5-6-7-8-9-2-…

扫描伪代码后,最终生成基本路径集。(程序运行耗费时间,单位是毫秒)程序运行截图如图 3 所示。

```
n1⊕n2⊕n10⊕n11⊕n13+n1⊕n2⊕n10⊕n12⊕n13+n1⊕n2⊕
n3⊕n10⊕n11⊕
去括号后的代数表达式:n1⊕n2⊕n10⊕n11⊕n13+n1⊕n2⊕
n10⊕n12⊕n13+
路径集合:->>>
[[1,2,10,11,13],[1,2,10,12,13],[1,2,3,10,11,13],[1,2,
3,
基本路径:->>>
[1,2,10,11,13]
[1,2,10,12,13]
[1,2,3,10,11,13]
[1,2,3,4,5,8,9]
[1,2,3,4,5,6,8,9]
[1,2,3,4,5,6,7,8,9]
耗费时间:5821418
```

图 3 Procedure average 案例实验结果

根据图的圈复杂度公式 $V(G)=e-n+2p$，其中，e 是 G 中的边数，n 是 G 中的节点数，p 是 G 中的组件数（$p=1$），Procedure average 的控制流程图的圈复杂度为 $17-13+2=6$。圈复杂度 $V(G)$ 的值提供了组成基本路径集的独立路径的上界。由此可以得出覆盖该程序的最少路径为 6 条。经过验证，正确生成了 Procedure average 的基本路径。

本文选取 5 个经典案例进行实验。

在相同实验环境下对以上 5 个经典案例分别运行 10 次并计算出平均运行时间。表 1 列出了案例的相关描述，该算法有效地生成了经典案例的基本路径集合。由于传统方法求基本路径集时需要人工绘制流程图，其时间无法明确度量，因此不具可比性。图 4 中，对算法进行纵向比较，可以看出，当代码量快速增大时，该算法求得基本路径集的时间消耗并未随之急剧增大，而是较平缓增长。

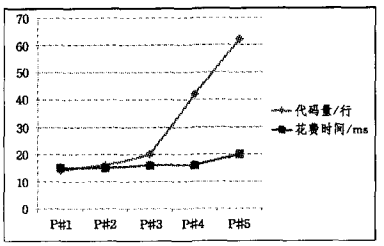

图 4　案例实验结果

表 1　案例 1—5 的实验结果

程序	代码量/行	传统(CFG)		代数模型(AM)		基本路径/条	程序描述	运行 10 次取平均值/μs
		结点/个	边/条	操作符/个	括号/个			
P#1	14	11	13	27	5	4	文献[10]中提供的案例，输入 a,b 值，经过处理后输出相应的 a,b 值	5398351
P#2	16	13	17	22	6	6	计算不超过 100 个数字的平均值，同时计算总和与有效数字的个数	5973898
P#3	20	13	17	23	11	6	任意输入 3 边的值，判断是否构成三角形以及该三角形的类型	8340201
P#4	42	17	19	19	3	4	通过一定的规则，计算佣金的问题	5705314
P#5	62	47	61	59	15	16	输入年月日，计算该日期后面的日期	15492726

结束语　基于模型代数的基本路径生成方法利用模型代数构建程序的代数表达式，导出基本路径集，从而设计测试用例。该方法提供了一种求基本路径集的全新思路，体现了模型代数的实际应用价值。该方法最终也有效地生成了待测程序的基本路径。

基于程序控制流图求基本路径集的方法需要建立在已知控制流图的情况下，每次都需要人工绘制控制流图，随着程序变得复杂，控制流图也会变得极为复杂，可操作性也会极大降低。本文中的方法与之相比更加高效，摒弃了绘制 CFG 图的复杂过程；与遗传算法相比，弥补了其在生成基本路径时的不稳定性。当然，实验对伪代码的识别有一定的规则要求，这造成代数表达式的生成存在一定的不足以及具有高圈复杂度的程序需要更加充分的测试等，这都是今后需要进一步研究及解决的问题。

参 考 文 献

[1] 宫云战，赵会群，赵瑞莲，等. 软件测试教程[M]. 北京：北京机械工业出版社.

[2] JORGENSEN P C. Software Testing A Craftsman's Approach (Second Edition)[M]. CRC Press, 2007.

[3] WANG G, JING X N, WANG Y J. The Application of Imroved McCabe Method in Basis Path Test[J]. Journal of Harbin University of Science and Technology, 2010, 15(1): 48-51. (in Chinese)
王冠，景小宁，王彦军. 基本路径测试中的 McCabe 算法改进与应用[J]. 哈尔滨理工大学学报，2010，15(1)：48-51.

[4] MCCABE, THOMAS J. Structural Testing: A Software Testing Methodology Using the Cyclomatic Complexity Metric[M]. National Bureau of Standards (Now NIST), Special Publication 500-99, Washington, D. C., 1982.

[5] ZHANG G M, LI X W, H C Y. Automatic Generationof Basis Path Set in Path Test[J]. Computer Engineering, 2007, 33(22): 195-197. (in Chinese)
张广梅，李晓维，韩丛英. 路径测试中基本路径集的自动生成[J]. 计算机工程，2007，33(22)：195-197.

[6] WU Q J, YANG X H, LU J C, et al. An Optimized Algorithm of Auto-generate Base Paths Set Base on Depth-first Search[J]. Journal of University of South China(Science and Technology), 2012, 26(3): 87-90. (in Chinese)
吴取劲，阳小华，鹿江春，等. 一种基于图深度优先搜索的基本路径集自动生成优化算法[J]. 南华大学学报(自然科学版)，2012，26(3)：87-90.

[7] YAN J, ZHANG J. Automatic Testing Based on Basis Paths [J]. Computer Science, 2004, 31(10): 62-64. (in Chinese)
严俊，张健. 基于基本路径的程序自动化测试[J]. 计算机科学，2004，31(10)：62-64.

[8] WANG M, CHEN S M, CHEN Y G. An Algorithm for Solving Basic Path Set[J]. Computer Applications and Software, 2014, 31(11): 11-14. (in Chinese)
王敏，陈少敏，陈亚光. 一种基本路径集求解算法[J]. 计算机应用与软件，2014，31(11)：11-14.

[9] XIE S X. Application Research of Program Graph Automatic Generation Based on Basic Path Test[J]. Journal of Tonghua Teachers College, 2009, 30(12): 32-35. (in Chinese)
解圣霞. 基于基本路径测试的程序图自动生成的应用研究[J]. 通化师范学院学报，2009，30(12)：32-35.

[10] GHIDUK A S. Automatic generation of basis test paths using variable length genetic algorithm[J]. Information Processing Letters, 2014, 114(6): 304-316.

[11] ZHAO H Q, SUN J. An Algebraic Model of Service Oriented Trustworthy Software Architecture[J]. Chinese Journal of Computers, 2010, 33(5): 890-899. (in Chinese)
赵会群，孙晶. 面向服务的可信软件体系结构代数模型[J]. 计算机学报，2010，33(5)：890-899.

[12] 陈火旺，刘春林. 程序设计语言编译原理(第三版)[M]. 长沙：国防工业出版社. 2000.

[13] POOLE J. A Method to Determine a Basis Set of Path to Perform[C]//Program Testing. NISTIR 5737, 1995.