软件质量保证与测试

Software Quality Assurance and Testing

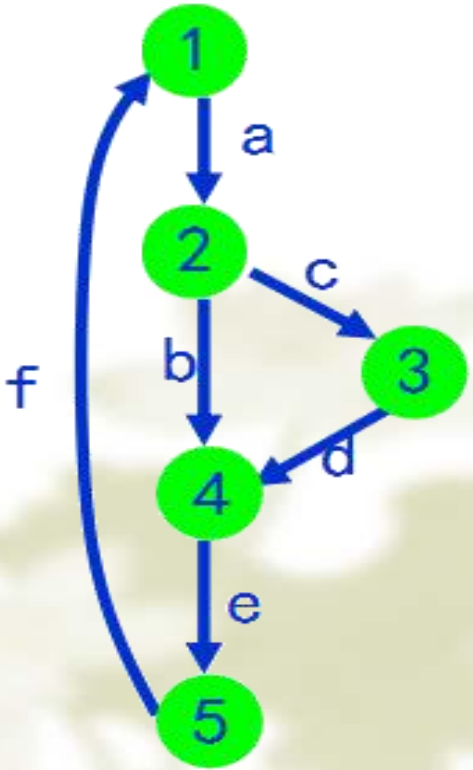
第 4 章 白盒测试

4.4 基本路径覆盖

金陵科技學院

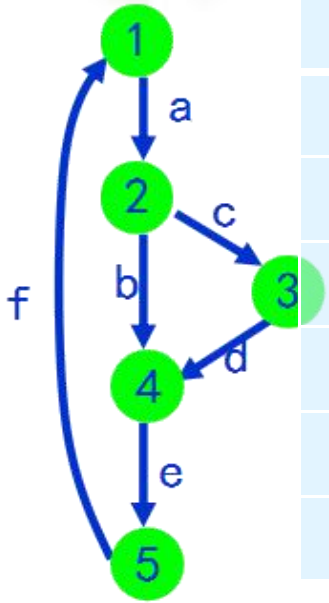
程序中的路径

在把程序抽象为有向图之后，我们把从程序入口到出口经过的各个节点的有序排列称为路径，可用路径表达式来表示这样的一条路径。路径表达式可以是节点序列，也可以是弧序列，例如有如图所示程序控制流图，其可能的程序执行路径如下表所示。



程序中的路径

需要注意的是，程序中存在循环时，如果执行循环的次数不同，那么对应的执行路径就不同，例如表中路径3和路径4就是由于程序执行循环次数不同而形成的不同的路径。



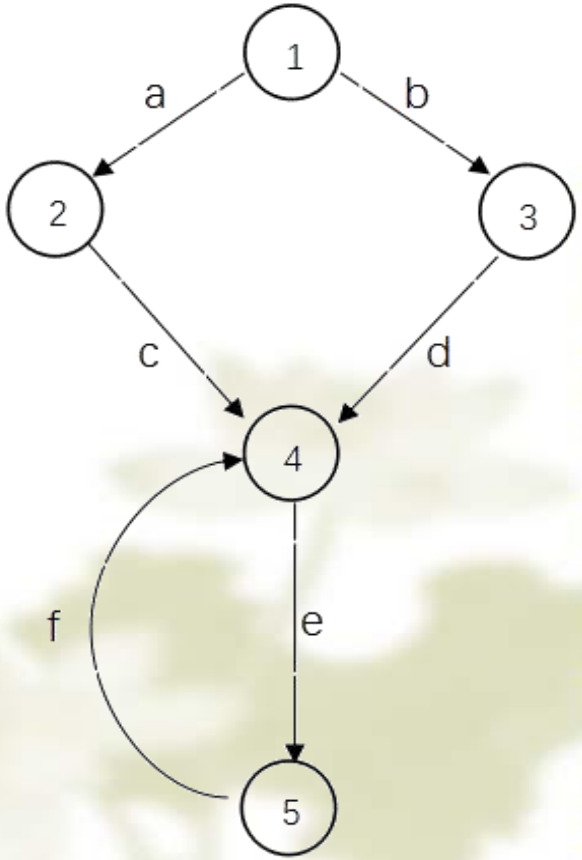
路径编号	弧序列表示	节点序列表示
1	acde	1-2-3-4-5
2	abe	1-2-4-5
3	abefabe	1-2-4-5-1-2-4-5
4	abefabefabe	1-2-4-5-1-2-4-5-1-2-4-5
5	abefacde	1-2-4-5-1-2-3-4-5
… …	… …	… …

程序中的路径

基本的路径表达，采用的是节点序列，或者是弧序列，例如表中路径1为acde，或者1-2-3-4-5。为增强路径表达能力，可以在路径表达式引入加法和乘方，加法可以表达IF分支结构，乘方可以表达循环结构。

设有程序控制流图如右图所示，则它所有可能的路径可表达为：

$(ac+bd)e(fe)^n$，n为循环的次数。



完全的路径覆盖一般不可能

程序中的一条IF语句就会形成两条路径。两条IF语句的串联就会有四条路径，在实际问题中，即使一个不太复杂的程序，其可能的路径都是一个庞大的数字，而如果程序中存在循环，则可能的路径就是天文数字。

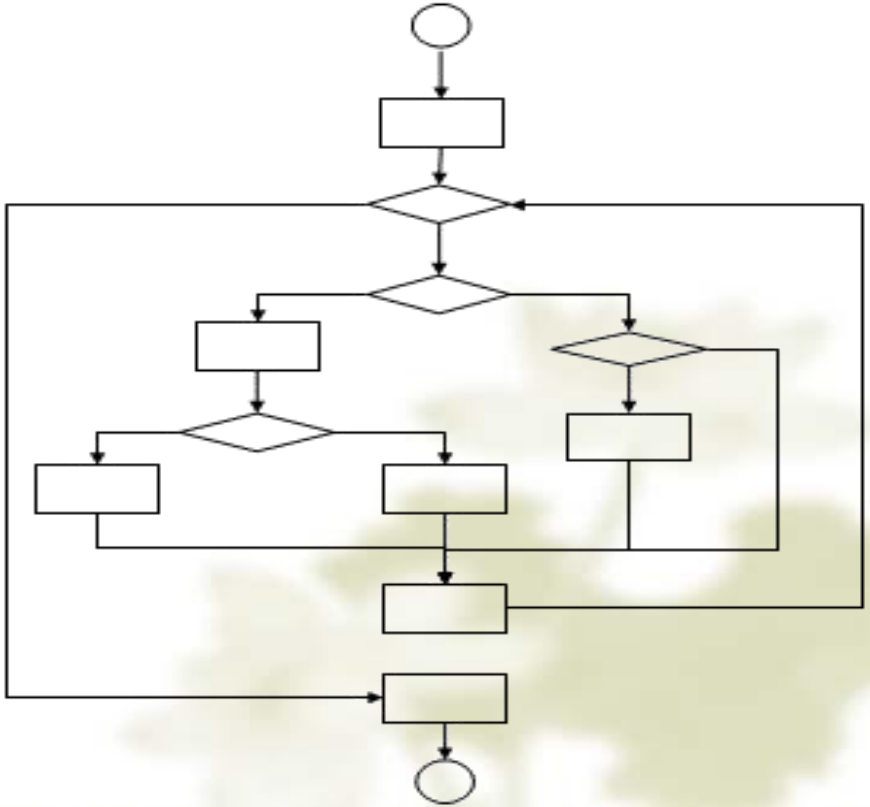
完全的路径覆盖一般不可能

设有如图所示的一个程序流程图，如果程序中循环的执行上限是10次，那么可能有多少条程序执行路径？

答案是，可能的程序执行路径条数 L：

$= 4^1+4^2+4^3+\cdots\cdots+4^{10}$

= 139 8100



完全的路径覆盖一般不可能

假设图中所有可能的路径都是可执行路径，某台计算机对该程序执行一次循环大约需要10微秒，那么把所有可能的路径都测试一遍，大约需要的时间为：

139 8100*10/100 0000 ≈ 14（秒）

假设该计算机一年365天每天24小时不停机，如果程序中循环的执行次数上限是20次，那么把所有可能的路径都测试一遍大约需要4072小时，如果循环次数上限是100次，则大约需要6.79*1047年。由此可见，对于实际的应用程序，做路径穷举测试是不可行的。

基本路径覆盖

既然无法对一个实际的应用程序进行路径的穷举测试，那么就只能选取部分路径来进行测试。

基本路径覆盖就是在程序控制流图的基础上，通过分析控制流图的环路复杂性，然后导出程序独立路径集合，再设计测试用例覆盖所有独立路径的一种动态白盒测试方法。由于基本路径覆盖把程序中的所有节点都覆盖到了，所以程序中的每一条可执行语句也至少会被执行一次，也就是说满足基本路径覆盖就一定是满足语句覆盖的。

基本路径覆盖

所谓独立路径是指，和其他的独立路径相比，至少有一个路径节点是新的，未被其他独立路径所包含。

从程序的环路复杂度可导出程序基本路径集合中的独立路径条数：

程序独立路径条数 = 程序的环路复杂度

这是确保程序中每个可执行语句至少执行一次所必须的测试用例数目的下界。

得出程序独立路径条数后，再根据控制流图，得出各条独立路径。所有独立路径组成独立路径集合，也就是基本路径集合。

基本路径覆盖

基本路径覆盖测试法的基本步骤如下：

1）画出程序控制流图

2）计算程序环路复杂性

3）确定独立路径集合

4）为每条独立路径设计测试用例

基本路径覆盖应当确保基本路径集中的每一条路径都能被执行到。一般是为每条独立路径设计一个测试用例，执行这个测试用例时，就能确保该独立路径会被执行。

借助于专门的工具软件，导出控制流图和确定基本路径的过程均可以自动完成。

基本路径覆盖

在基本路径测试中，称为图形矩阵的数据结构很有用。利用图形矩阵可以确定控制流图的环路复杂度，也就是基本路径集合中基本路径的条数。一个图形矩阵是一个方阵，其行/列都是控制流图中的节点，每行和每列依次对应到一个被标识的节点，矩阵元素对应到节点间的连接（即弧）。在图形矩阵中控制流图的每一个节点都用数字加以标识，每一条边都用字母加以标识。如果在控制流团中第i个结点到第j个结点有一个名为x的边相连接，则在对应的图形矩阵中第i行/第j列有一个非空的元素。

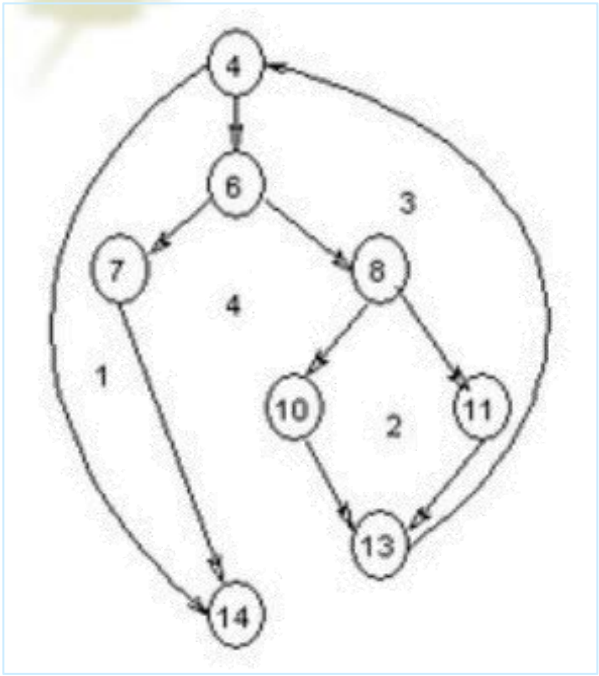
基本路径覆盖

对每个矩阵项加入连接权值，图形矩阵就可以用于在测试中评估程序的控制结构，连接权值为控制流提供了另外的信息。最简单的情况下，连接权值是1或0，分别表示存在连接或不存在连接。但是，连接权值可以被赋予更多有用的属性，如：**执行连接的概率；穿越连接的处理时间；穿越连接时所需的内存；穿越连接时所需的资源**等。

如果连接权值为1表示存在连接，那么图中有两个元素为1的行所代表的节点就一定是一个判定节点，通过计算图形矩阵中有两个元素为1的行的个数，就可以得出总的判定节点数，从而得出环路复杂度。这是确定环路复杂度的另一种方法。

基本路径覆盖

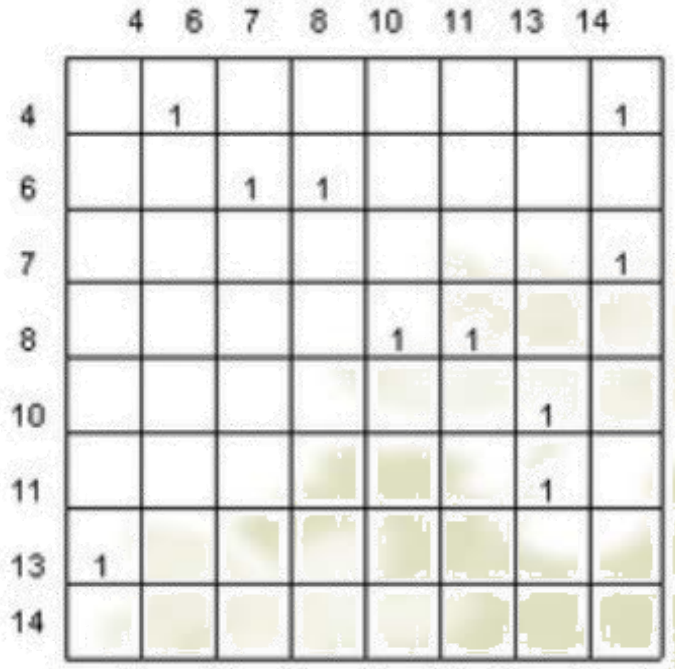
给出一个图形矩阵的实例如下：左边为程序控制流图，右边为对应的图形矩阵。



	4	6	7	8	10	11	13	14
4		1						1
6			1	1				
7								1
8					1	1		
10							1	
11							1	
13	1							
14								

基本路径覆盖

图形矩阵中，有两个元素为1的行有3个，所以判定节点数为3，从而得出环路复杂度为 3+1=4。



基本路径覆盖实例

下面我们来看一个基本路径覆盖的例子。

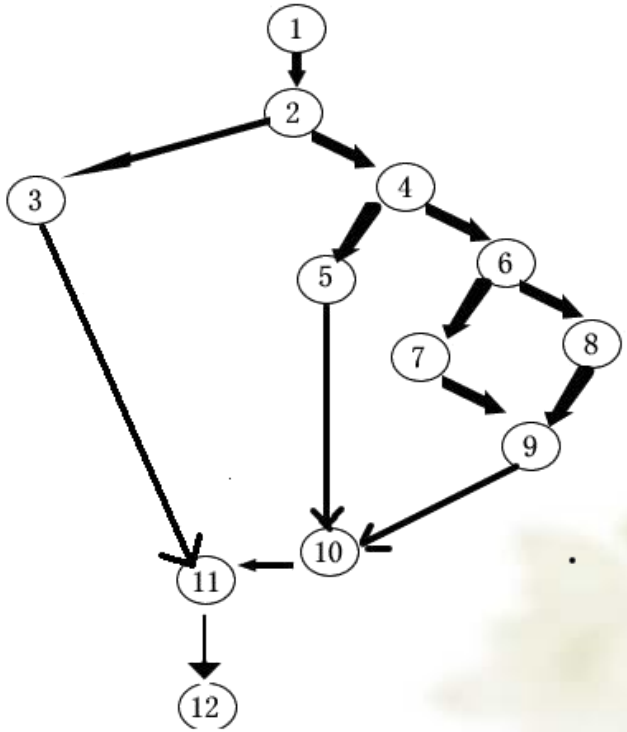
请针对程序段IsLeap，为变量year设计测试用例，满足基本路径覆盖，假设year的取值范围为1000——9999。

基本路径覆盖实例

```
int IsLeap(int year){
    if(year % 4 ==0){
        if(year % 100 ==0){
            if(year %400 != 0)
                leap=1;
            else  leap=0;
        }else
            leap=1;
    }else leap =0;
    return leap;
}
```

基本路径覆盖实例

(1)绘制出程序代码对应的控制流图，如下图所示。



基本路径覆盖实例

(2)计算环路复杂度V(G)。

V(G)=E-N+2=14-12+2=4； V(G)=判定点数+1=3+1=4

V(G)=区域数=4

(3)确定独立路径集合。

1-2-3-11-12

1-2-4-5-10-11-12

1-2-4-6-7-9-10-11-12

1-2-4-6-8-10-11-12

(4)设计测试用例。

针对各条独立路径设计的测试用例，如表所示。

基本路径覆盖实例

测试用例编号	测试数据	预期执行结果	测试路径
1	year=1001	leap=0	1-2-3-11-12
2	year=1004	leap=1	1-2-4-5-10-11-12
3	year=1100	leap=0	1-2-4-6-7-9-10-11-12
4	year=2000	leap=1	1-2-4-6-8-10-11-12

本节内容就讲到这里，谢谢，再见！

金陵科技學院